

EDUARDO FONTANA VIEIRA E SILVA

**SISTEMA DE INTEGRAÇÃO DE TECNOLOGIAS DE AGREGAÇÃO
DE MEDIÇÃO: UMA ABORDAGEM BASEADA EM MODELOS**

Dissertação apresentada ao Curso de Pós-Graduação em Computação Aplicada e Automação da Universidade Federal Fluminense, como requisito parcial para obtenção do Grau de Mestre. Área de Concentração: Processamento Paralelo e Distribuído.

Orientador:
Prof. Julius Leite, Ph.D.

NITERÓI
2004

EDUARDO FONTANA VIEIRA E SILVA

**SISTEMA DE INTEGRAÇÃO DE TECNOLOGIAS DE AGREGAÇÃO
DE MEDIÇÃO: UMA ABORDAGEM BASEADA EM MODELOS**

Dissertação apresentada ao Curso de Pós-Graduação em Computação Aplicada e Automação da Universidade Federal Fluminense, como requisito parcial para obtenção do Grau de Mestre. Área de Concentração: Processamento Paralelo e Distribuído.

Aprovada em 23/04/2004.

BANCA EXAMINADORA

Prof. Julius Cesar Barreto Leite, Ph.D. – Orientador
Universidade Federal Fluminense

Prof. Orlando Gomes Loques Filho, Ph.D.
Universidade Federal Fluminense

Prof. Gilberto Pires de Azevedo, D.Sc.
Universidade Federal do Rio de Janeiro/CEPEL

NITERÓI
2004

Dedico essa dissertação ao meu pai,
por sempre incentivar e investir em
meus estudos e sonhos.

AGRADECIMENTOS

Embora uma dissertação seja, pela sua finalidade acadêmica, um trabalho individual, há contribuições de naturezas diversas que não podem e nem devem deixar de ser realçadas. Por essa razão, desejo expressar os meus sinceros agradecimentos:

A DEUS, por estar presente em todos os momentos da minha vida e, especialmente, por ter me iluminado durante a realização deste trabalho e ser uma inesgotável fonte de inspiração nos momentos de dificuldade.

Ao meu pai, Baltazar dos Reis e Silva, um grande exemplo de vida no qual procuro me espelhar, pelo incentivo e confiança em todas minhas investidas. À Minha mãe, Dircey Vieira e Silva, pelo amor e força que me ajudam a alcançar meus objetivos. Aos meus irmãos, Ricardo, Jussara e Daniela, e minha cunhada, Regiane (mãe do meu afilhado(a)), pelo apoio e companheirismo. Eles, mesmo à distância, foram os grandes colaboradores por eu alcançar este sucesso.

Aos meus tios e tias, primos e primas, madrinha e padrinho e a minha avó que a distância me fez privar-me de muitos momentos agradáveis da companhia acolhedora dessas pessoas tão importantes em minha vida.

Um obrigado muito especial a minha querida Cris, pelos momentos juntos, pelo ombro, pelo carinho, pelas conversas, pela “leitura e sugestões no texto dessa dissertação” e por ter sido essa pessoa maravilhosa que apareceu em minha vida.

Ao meu orientador, Prof. Dr. Julius Leite, pela dedicação, confiança e incentivo em todos os momentos. Meu reconhecimento e gratidão pela paciência, compreensão, oportunidades e orientação.

A todos os professores do Instituto de Computação (IC) que participaram da minha formação no Mestrado, em especial ao Prof. Dr. Orlando Loques, pela co-orientação no

trabalho. Às secretárias do IC, Izabela e Ângela, por toda atenção e apoio necessário nas tarefas do dia a dia.

Ao amigo Sidney, um verdadeiro irmão de coração e um dos principais responsáveis por hoje eu estar aqui comemorando essa conquista.

Aos amigos de “Mouseville”: Vitor, Sidney, Jonivan, Áthila, André, Rone e Stênio; pela sincera amizade e pela convivência agradável. Cada um de vocês teve sua parcela de responsabilidade nesta conquista. Agradeço-lhes de uma forma muito especial pela força e o apoio nos momentos alegres e de desânimo.

A Ouro Preto e em especial a República Canaan “em sua alma”, pelo aprendizado, escola de vida, momentos de refúgio e alegria, onde nasceram grandes amizades, quase todas, senão todas, imortais. Quando digo “em sua alma”, falo das tradições, do lado acolhedor, responsável, alegre e festivo, mas principalmente falo dos muitos amigos conquistados, Interlace, Natal, Geada, Fiapo, Bidê, Love, Filó, Ki-Susto, Salsicha, Fester, Falamansa, Neném, Scooby, Nikita e a bixarada. Além de Perfil, Pinico, Ramal, Pantera, Cenourinha... E tantos outros de menor convivência, mas não menor importância.

Aos amigos conquistados Helson, Lílian e Helsinho, uma família e tanto que conheci aqui, obrigado pelos momentos de descontração.

Aos amigos que me ajudaram diretamente neste trabalho, Alexsandro, pelas muitas horas lado a lado, Sidney pelas muitas sugestões e Jonivan por diversos esclarecimentos.

A máfia ouropretana que invadiu Niterói e deixou esse lugar um tanto quanto familiar, o meu obrigado a Renatinha, Alexsandro, Glauco, Hébio, e companhia.

Agradeço também aos amigos Helder Mendes e José Viterbo, pelo companheirismo durante as disciplinas e trabalhos desenvolvidos e também pelos conselhos e sugestões.

Não poderia esquecer das grandes amigas, Ilza, Gabi e Idalmis pelos bons momentos que passamos abrandando a penúria da labuta por bons momentos juntos.

Sem dúvida, agradecer a todos que ajudaram a construir esta dissertação não é tarefa fácil. Ainda falta agradecer a muitas outras pessoas; Rodrigo, Márcia, Eyder, Erwin, enfim, a todas as pessoas que colaboraram direta ou indiretamente com esse trabalho, fica registrado meu muito obrigado. Pois essa conquista não é só minha, de certa forma todas essas pessoas tem um pedacinho de responsabilidade por ela, e eu não estou falando de terem feito uma linha de código que seja, ou de terem escrito parte desse texto por mim, estou falando dos momentos bons, que bons amigos compartilham e vivem. São muitas as vezes que a lembrança de que existe alguém que a gente gosta e que confia na gente e que está por aí pelo mundo afora é que nos dá força pra prosseguir, insistir e continuar. A amizade é algo que não é fácil de explicar, é um gostar por gostar, respeitando as diferenças, opiniões, opções religiosas e futebolísticas de cada um... Eu só gostaria de agradecer a todos vocês por serem meus amigos e por sempre mostrarem que ter amigos verdadeiros é a melhor riqueza que podemos ter na vida. Muito obrigado!

"Algo só é impossível até que alguém
duvide e acabe provando o contrário."

Albert Einstein

SUMÁRIO

<i>LISTA DE FIGURAS</i>	v
<i>LISTA DE TABELAS</i>	vii
<i>LISTA DE CÓDIGOS</i>	viii
<i>GLOSSÁRIO</i>	ix
<i>RESUMO</i>	xi
<i>ABSTRACT</i>	xii
<i>CAPÍTULO 1</i>	1
1.1 Dificuldades	2
1.2 Tendências de Integração	3
1.3 Medidores e Sistemas de Medição de Energia	4
1.4 Proposta	5
1.5 Estrutura da Dissertação	6
<i>CAPÍTULO 2</i>	8
2.1 Introdução	8
2.2 UCA (<i>Utility Communications Architecture</i>)	9
2.2.1 Introdução	9
2.2.2 Visão de Camadas (Perfis)	11
2.2.3 CASM (<i>Common Application Service Models</i>)	12
2.2.4 GOMSFE (<i>Generic Object Models Substations & Feeder Equipment</i>)	14
2.2.5 Exemplo de Utilização	17
2.2.6 Exemplo de Rede Integrada com UCA	18
2.3 DLMS/COSEM	19
2.3.1 Introdução	19
2.3.2 Arquitetura COSEM	20
2.3.3 Classes de Interface	22
2.3.4 Sistema de Identificação de Objetos	23
2.3.5 Modelo de Servidor COSEM	24
2.3.6 Associações	25
<i>CAPÍTULO 3</i>	26

3.1 Introdução	26
3.2 Heterogeneidade dos Sistemas e sua Integração	27
3.3 Padrões	29
3.3.1 A Norma NBR 14522	29
3.3.1 A Especificação DLMS/COSEM	30
3.4 Soluções Proprietárias	31
3.5 Segurança	33
<i>CAPÍTULO 4</i>	35
4.1 Introdução	35
4.2 Modelagem das Classes de Interface COSEM em Esquemas XSDs	37
4.3 Representação das Associações COSEM	40
4.3.1 Associação SN	41
4.3.2 Associação LN	42
4.4 Representação de um Medidor de Energia Genérico	43
4.4.1 Grandezas de Faturamento	44
4.4.2 Modelagem Adicional	48
4.5 Considerações Finais Sobre a Modelagem	50
4.6 Propostas de Segurança	51
4.6.1 Associações e Restrições de Acesso a Dados	52
4.6.2 Características de Segurança Herdadas dos Medidores	56
<i>CAPÍTULO 5</i>	58
5.1 Introdução	58
5.2 O Sistema Padronizado de Leitura e Parametrização de Medidores	59
5.3 Arquitetura Detalhada do Sistema	63
5.3.1 As Interfaces com o Usuário	64
5.4.2 O Sistema PLPM	65
5.4.3 O Simulador de Medidor	67
5.4.4 O Medidor Incorporado	67
5.4 Tipos de Usuários Definidos e Associações	68
5.4.1 Usuário Administrador 1	68
5.4.2 Usuário Administrador 2	69

5.5.3 Usuário Consumidor.....	70
5.6 Operação do Sistema	71
5.6.1 Processo de Parametrização do Medidor.....	72
5.6.2 Processo de Leitura do Medidor.....	74
5.7 Geração de <i>Logs</i> do Sistema.....	76
<i>CAPÍTULO 6</i>	78
6.1 Conclusões.....	78
6.2 Propostas de Trabalhos Futuros.....	79
6.3 Contribuições deste Trabalho	80
<i>REFERÊNCIAS BIBLIOGRÁFICAS</i>	83
<i>APÊNDICE A</i>	86
A.1 Introdução	86
A.2 XML	87
A.3 A XML é uma Linguagem de Marcação	89
A.4 XML é eXtensível	90
A.5 Estrutura de um Documento XML	91
A.6 Características Adicionais	92
A.7 Os Esquemas XML.....	93
A.7.1 <i>Document Type Definition</i>	94
A.7.2 <i>XML Schema Definition Language</i>	95
A.7.3 DTDs x XSDs.....	96
A.7.4 <i>XML Namespaces</i>	98
A.8 XSLT	99
A.9 <i>Java API for XML Processing (JAXP)</i>	100
A.10 XMLSpy	100
A.11 Potenciais Aplicações da Tecnologia XML	101
<i>APÊNDICE B</i>	104
B.1 Introdução	104
B.2 Questões no Nível IU/PLPM	107
B.2.1 Controle de Acesso à Aplicação	107
B.2.2 Comunicação Cliente-Servidor.....	108

B.2.2.1 Confiabilidade.....	108
B.2.2.2 Segurança.....	109
B.2.2.3 Comentários	110
B.3 Questões no Nível PLPM/URM	112
B.4 Conclusão.....	114

LISTA DE FIGURAS

Figura 2.1 – Integração através da arquitetura UCA.	9
Figura 2.2 – Perfil UCA.	11
Figura 2.3 – O modelo de servidor CASM.....	13
Figura 2.4 – Hierarquia de modelo de objetos.....	15
Figura 2.5 – Estruturação CASM / GOMSFE.....	16
Figura 2.6 – Definição de um produto UCA.	18
Figura 2.7 – Rede integrada UCA.	19
Figura 2.8 – Relação cliente-servidor em COSEM.	21
Figura 2.9 – Estrutura da camada de aplicação.	21
Figura 2.10 – Modelo de servidor em COSEM.....	24
Figura 4.1 – Um dos benefícios da modelagem de dispositivos é facilitar o reuso de definições comuns.	36
Figura 4.2 – Classe <i>Extended Register</i>	37
Figura 4.3 – Diagrama estrutural da classe de interface <i>Extended Register</i>	39
Figura 4.4 – Descrição da classe Associação SN (ver 4.6).	41
Figura 4.5 – Tipo lista de objetos da Associação SN.	41
Figura 4.6 – Tipo Associação SN.....	42
Figura 4.7 – Tipo lista de objetos da Associação LN.....	42
Figura 4.8 – Tipo Associação LN.....	43
Figura 4.9 – Medidor de energia genérico.....	44
Figura 4.10 – Representação das grandezas de faturamento de um medidor.....	45
Figura 4.11 – Representação das grandezas de faturamento do Canal 3.....	46
Figura 4.12 – Declaração da grandeza CN349 como do tipo <i>ExtendedRegister</i>	47
Figura 4.13 – Existe uma instância dessa classe para cada associação que um dispositivo pode suportar.	48
Figura 4.14 – Diagrama que representa as portas de comunicação do medidor.	49
Figura 4.15 – Diagrama que representa o armazenamento dos dias especiais em registros adicionais.	49
Figura 4.16 – Representação do Canal 3 no medidor SAGA1000.....	50

Figura 5.1 – Arquitetura geral do sistema.	60
Figura 5.2 – Tecnologias empregadas na aplicação.	64
Figura 5.3 – Interface inicial do usuário Administrador 1.	69
Figura 5.4 – Interface inicial do usuário Administrador 2.	70
Figura 5.5 – Interface inicial de acesso ao sistema.....	71
Figura 5.6 – Interface de identificação do medidor e usuário.	72
Figura 5.7 – Interface para configuração da data.	73
Figura 5.8 – Comando 29 no formato da Norma ABNT NBR 14522.	74
Figura 5.9 – Processo de Leitura do medidor.....	75
Figura 5.10 – Arquivo de <i>log</i> do sistema PLPM.	77
Figura 5.11 – Arquivo de <i>log</i> do simulador.	77
Figura 6.1 – Classe PO_NBR14522.....	81
Figura 6.2 – Classe Registro_alteracoes.....	82

LISTA DE TABELAS

Tabela 4.1 – Especificação dos campos da classe de interface Extended Register.....	38
---	----

LISTA DE CÓDIGOS

Código 4.1 – Código XSD da representação estrutural da Figura 4.3.....	40
Código 4.2 – Objeto CN103 em uma instância da classe <i>Register</i> em XML.	53
Código 4.3 – Associação LN em XML.	55
Código 4.4 – Associação SN em XML.	56
Código 4.5 – Grandeza 23 como do tipo <i>ExtendedRegister</i>	57
Código 4.6 – Registro das alterações.	57
Código 5.1 – Representação XML dos registradores que guardam os valores relativos a data no medidor.....	74

GLOSSÁRIO

ABRADEE	Associação Brasileira de Distribuidores de Energia Elétrica
ABNT	Associação Brasileira de Normas Técnicas
ACSE	Association Control Service Element
ANEEL	Agência Nacional de Energia Elétrica
ASE	Application Service Element
ASO	Application Service Object
CASM	Common Application Service Models
CF	Control Function
CORBA	Common Object Request Broker Architecture
COSEM	Companion Specification for Energy Metering
DCOM	Distributed Component Object Model
DLMS	Device Language Message Specification
DNP	Distributed Network Protocol
DTD	Document Type Definition
EEE	Empresas de Energia Elétrica
EPRI	Electric Power Research Institute
GOMSFE	Generic Object Models for Substation & Feeder Equipment
HTML	Hypertext Markup Language
IEC	International Electrotechnical Committee
IED	Intelligent Electronic Device
IEEE	Institute of Electrical and Electronics Engineers
IHM	Interface Homem Máquina
IP	Internet Protocol
ISO	International Standards Organization
IU	Interface de Usuário
JAXP	Java API for XML Processing
JDBC	Java Database Connectivity
JSP	Java Server Pages
LAN	Local Area Network

LDN	Logical Device Name
LN	Logical Name
MMS	Manufacturing Message Specification
OBIS	Object Identification System
ODBC	Open Database Connectivity
OSI	Open Systems Interconnection
PLPM	Padronizado de Leitura e Parametrização de Medidores
RMI	Remote Method Invocation
RTU	Remote Terminal Unit
SBO	Select Before Operate
SCADA	Supervisory Control and Data Acquisition
SGML	Standard Generalized Markup Language
SN	Short Name
SQL	Structured Query Language
TCP	Transmission Control Protocol
URI	Uniform Resource Identifier
URL	Universal Resource Locator
URM	Unidade Remota de Medição
W3C	World Wide Web Consortium
WAN	Wide Area Network
WWW	World Wide Web
XML	Extensible Markup Language
XSD	XML Schema Definition
XSL	Extensible Stylesheet Language
XSLT	Extensible Stylesheet Language Transformation

RESUMO

O suporte às atividades de uma empresa requer a colaboração entre os sistemas de computação que ela possui para compartilhar informações e processos. Assim, diversas organizações têm unido esforços no sentido de estabelecer padrões, arquiteturas e formatos de representação de dados que permitam a interoperabilidade entre os sistemas de uma empresa e mesmo entre empresas. No âmbito das Empresas de Energia Elétrica surgiram várias propostas nesse sentido. Alguns pontos em comum podem ser observados entre as diversas iniciativas. Um deles é que cada vez mais a integração de sistemas heterogêneos está sendo tratada como uma questão de “entendimento” entre esses sistemas. Assim, as propostas conduzem a uma abordagem que privilegia o entendimento da informação sendo compartilhada, através da utilização de modelos lógicos de informação. Um outro ponto tem sido a incorporação de conceitos do paradigma de software conhecido como Orientação a Objetos. Essa tecnologia permite a construção de modelos de dispositivos a partir da composição de blocos funcionais básicos. Esse trabalho apresenta uma proposta para integração de sistemas heterogêneos no contexto de Medidores de Energia Elétrica Eletrônicos. Com uma abordagem centrada na construção de modelos que descrevem o domínio do problema de medição, mostramos a possibilidade de se integrar diferentes tipos de medidores em um só sistema de configuração e leitura. O protótipo desenvolvido validou as técnicas e a modelagem empregadas.

Palavras-chave: Sistemas heterogêneos, medidores de energia elétrica, interoperabilidade.

ABSTRACT

The support to the activities of a company requests the collaboration among the several computation systems that it possesses. Thus, several organizations have been uniting efforts in the sense of establishing patterns, architectures and formats of representation of data that allow the interoperability among the systems of a company and even among companies. Some common characteristics can be observed among the several initiatives. One of them is that, more and more, the integration of heterogeneous systems is being treated as a question of "understanding" among those systems. These proposals lead to an approach that privileges the understanding of the information being shared, through the use of logical models of information. Another point has been the incorporation of concepts of the software paradigm known as Object Orientation. This technology allows the construction of models of devices through from the composition of basic functional blocks. This work presents a proposal for integration of heterogeneous devices in the context of Electronic Meters of Electric Energy. With an approach based in the construction of models that describe the domain of the measurement problem, we show the possibility of integrating different models of meters in one configuration and metering system. The developed prototype has validated the techniques and methodology employed.

Keywords – Heterogeneous systems, electric energy meters, interoperability.

CAPÍTULO 1

INTRODUÇÃO

Em tempos de desregulamentação do setor elétrico brasileiro é natural o surgimento de muitas soluções na área de automação e gerenciamento de energia elétrica. De um lado, estão milhares de consumidores procurando reduzir custos, mas sem perder em qualidade no que diz respeito à energia elétrica fornecida. Do outro lado estão dezenas de concessionárias de energia, preocupadas com a fiscalização da ANEEL (Agência Nacional de Energia Elétrica), e interessadas em conquistar os consumidores umas das outras, como num livre mercado.

Um dos principais problemas a ser resolvido pela maioria dessas concessionárias é a dificuldade de se adquirir informações de medidores eletrônicos instalados nos diversos pontos de distribuição, seja na indústria ou comércio, e, portanto, de manter o acompanhamento dos dados. Há muitos anos, um grupo de trabalho do ABRADDEE (Associação Brasileira de Distribuidores de Energia Elétrica), coordenado pela ABNT, e reunindo os principais fabricantes de medidores da época, criou o que hoje se chama de "protocolo ABNT", imposto por norma aos medidores eletrônicos instalados no país, e baseado em padrões internacionais. Mas o tempo passou, a tecnologia se desenvolveu muito rapidamente, e a situação que se configura no mercado é a seguinte:

a) Infelizmente, este protocolo tornou-se ultrapassado e já não é adequado aos medidores de uma nova geração.

b) Os vários fabricantes de medidores sempre buscaram soluções específicas a seus próprios produtos, dificultando ao máximo a integração com medidores de outras marcas.

Com isso algumas necessidades surgem, como interoperabilidade e compatibilidade. É necessário o desenvolvimento de interfaces para que os SGBDs concentradores de dados

se tornem compatíveis com quaisquer medidores. Mais ainda, estes dispositivos devem ter a capacidade de se comunicar com o sistema central usando um protocolo padrão.

Com interoperabilidade e compatibilidade estabelecidas torna-se mais fácil o gerenciamento, permitindo o acompanhamento e monitoração remota de metas de consumo de energia continuamente. O objetivo desse trabalho é propor um padrão para aquisição de dados de medidores de energia, bem como para os protocolos de comunicação envolvidos, utilizando tecnologias de computação de distribuição livre e gratuita (XML, Java, entre outras), assim como garantir segurança no acesso e manipulação dos dados, utilizando técnicas de autenticação, e na comunicação, utilizando técnicas de criptografia. Adicionalmente, é também seu objetivo construir um protótipo de aplicação de interface padrão para esses dispositivos para facilitar o controle e formatação dos dados.

1.1 Dificuldades

O ambiente computacional atual é composto por sistemas de *software* autônomos, distribuídos e heterogêneos. Esses sistemas têm sido construídos utilizando uma ampla diversidade de tecnologias computacionais, incluindo plataformas de *hardware*, sistemas operacionais, tecnologias de bancos de dados, formatos de representação de dados e linguagens de programação. Ou seja, a “heterogeneidade tecnológica” é uma característica predominante nesses sistemas.

Outra questão está relacionada à heterogeneidade de informação. O que ocorre é que muitas vezes cada sistema possui uma concepção particular do domínio de problema a ele associado. Diferentes concepções causam problemas de entendimento entre os sistemas.

Toda essa heterogeneidade dificulta o processo de integração. No contexto dos medidores de energia, um dos principais problemas está relacionado com as diferentes concepções que cada modelo tem do domínio do problema. Diferenças sintáticas e semânticas em relação às informações entre os diferentes modelos de medidor são comuns.

Somada a isso está a “necessidade” dos fabricantes em oferecer diferenciais em seus modelos, com o interesse de ganhar a preferência dos consumidores.

Uma tendência que vem mundialmente se configurando é a utilização dos protocolos da Internet (e mesmo essa própria rede) para o acesso aos equipamentos de medição e demais dispositivos eletrônicos inteligentes. Com a proposta de uma arquitetura baseada na Internet para construção de um sistema de integração, a questão da segurança dos dados tornou-se uma preocupação adicional no desenvolvimento deste trabalho. Assim, diversos níveis de segurança foram incorporados ao sistema de integração aqui descrito.

1.2 Tendências de Integração

A era dos sistemas de informação proprietários está terminando e uma nova era baseada em sistemas abertos, generalistas e interoperáveis está surgindo. Neste contexto, jamais a indústria de tecnologia da informação esteve tão orientada a padrões abertos e multiplataformas: TCP/IP tornou-se a rede universal; HTML, a “tradução” homem-máquina; Java, o código “língua franca” da indústria; e XML, o dado universal.

Assim, diversas organizações têm unido esforços no sentido de estabelecer padrões, arquiteturas e formatos de representação de dados que permitam a interoperabilidade entre os sistemas de uma empresa e mesmo entre empresas. No âmbito das Empresas de Energia Elétrica (EEE) surgiram várias propostas nesse sentido. A UCA (*Utility Communications Architecture*), desenvolvida pelo EPRI (*Electric Power Research Institute*), é uma proposta de arquitetura que tem como objetivo estabelecer padrões para integração das diversas áreas funcionais dentro das empresas do setor [UCA 97a]. Em particular, o modelo GOMSFE (*Generic Object Models Substations & Feeder Equipment*) [UCA 00] define os blocos de construção básicos para a composição de modelos lógicos de dispositivos. Outra proposta é a DLMS/COSEM (*Device Language Message Specification / Companion Specification for Energy Metering*) [DLMS 02a] que é uma iniciativa de um grupo de empresas europeias e visa especificar um medidor de energia interoperável, que possa ser utilizado em rede, através de diferentes formas de comunicação.

Alguns pontos em comum podem ser observados entre as diversas iniciativas. Um deles é que cada vez mais a integração de sistemas heterogêneos está sendo tratada como uma questão de “entendimento” entre esses sistemas. Assim, as propostas conduzem a uma abordagem que privilegia o entendimento da informação sendo compartilhada, através da utilização de modelos lógicos de informação. Esses mecanismos permitem que os sistemas possam entender corretamente os significados das informações sendo compartilhadas. Essa abordagem é chamada de Integração Semântica ou Integração Baseada em Modelos [Reyn 03].

Um outro ponto tem sido a incorporação de conceitos do paradigma de *software* conhecido como Orientação a Objetos. Essa tecnologia permite a construção de modelos de dispositivos a partir da composição de blocos funcionais básicos.

Uma tecnologia que vem se estabelecendo nesta e em muitas outras áreas da computação é a XML [W3C 98]. A XML surgiu como promessa e de fato vem se estabelecendo como uma ferramenta importante e amplamente usada na integração de sistemas. Os serviços *web*, por exemplo, que surgem como o futuro dos sistemas corporativos, usam XML como formato de representação de dados. A XML é um padrão de representação de dados de forma estruturada e em formato texto que permite, entre outras coisas, especificar a semântica dos dados.

1.3 Medidores e Sistemas de Medição de Energia

A carência de sistemas de automação e medição padronizados adequados ao novo modelo do setor elétrico é muito grande, e representa um desafio a ser vencido pelas concessionárias e demais agentes do setor elétrico brasileiro.

O problema é que a padronização quase sempre se limita à comunicação das informações, não havendo modelos comuns de dados. Neste caso, a integração destes sistemas e a utilização das informações trocadas dependem de algum tipo de “tradução”.

Atuando conjuntamente nas áreas de automação e medição, este trabalho, mostra uma opção para o desenvolvimento de um sistema de medição interoperável com diversos modelos de medidores de energia e que é também extensível. Esse desenvolvimento foi fundamentado em uma abordagem de integração de sistemas baseada em modelos que representam o domínio do problema.

Como ponto forte, o sistema desenvolvido possui compatibilidade com os medidores de energia mais populares no mercado e aderência aos padrões de segurança e formatação XML dos dados registrados.

1.4 Proposta

Esse trabalho não pretende ser final, seu foco é apresentar uma opção para construção de sistemas que possam integrar os diversos medidores de energia existentes. A estratégia passa pela definição de um modelo de medidor genérico que represente de uma forma unificada as diversas concepções do domínio do problema. Nesse sentido, esse trabalho busca o desenvolvimento de uma representação padrão (lógica) para medidores, configurável e expansível, capaz de integrar a definição de diversos medidores específicos.

Como visto nos sub-itens anteriores, as iniciativas e esforços para alcançar interoperabilidade levam ao paradigma da orientação a objetos. Isso se dá principalmente pelo fato desse paradigma permitir um alto grau de reaproveitamento de código. Nossa proposta é também baseada neste paradigma e sugere a construção dos modelos dos dispositivos a partir de blocos básicos de construção. Os blocos de construção de objetos são baseados nas classes de interface da iniciativa DLMS/COSEM, além de algumas classes adicionais, necessárias ao mapeamento de algumas idiossincrasias da Norma ABNT 14522. Grandezas e parâmetros de um medidor de energia foram então mapeados como objetos de uma das classes COSEM. É importante ressaltar que as classes COSEM foram escolhidas como os blocos básicos de construção do nosso modelo porque essa especificação está em estágio mais avançado em relação ao nosso contexto, visto que estamos trabalhando com medidores de energia. A proposta UCA/GOMSFEE não tinha

ainda no início desse trabalho *bricks* (blocos básicos de construção) específicos para um medidor de energia.

Os elementos do medidor genérico são mapeados, utilizando as classes de interface da especificação COSEM, em dados em formato XML. O modelo comum também agrega aos medidores características de segurança (restrições para acesso aos dados) propostas na especificação COSEM. Os documentos XML são as estruturas de dados internas do sistema.

A aplicação construída para prova dos conceitos estudados foi feita em uma arquitetura baseada na Internet, visando facilitar o acesso dos diversos interessados no contexto da medição de energia, como consumidores e concessionárias.

No desenvolvimento deste trabalho uma das preocupações principais está relacionada à segurança dos dados. Diversos níveis de segurança foram incorporados ao sistema.

1.5 Estrutura da Dissertação

O capítulo 2 apresenta uma breve revisão do estado da arte das propostas que surgem no processo de integração de sistemas heterogêneos, no contexto das empresas de energia elétrica.

O Capítulo 3 apresenta os diversos problemas que surgem no processo de integração de sistemas e, especialmente, no domínio de medição de energia.

O Capítulo 4 apresenta a nossa proposta; questões relacionadas à modelagem e aos requisitos de segurança empregados no sistema são abordados nesse capítulo.

O Capítulo 5 apresenta uma aplicação que implementa os conceitos de integração de sistemas baseada em modelos abordados no Capítulo 4.

O Capítulo 6 apresenta as conclusões e proposta de trabalhos futuros.

O Apêndice A apresenta diversas tecnologias relacionadas à linguagem XML, fundamentais para o desenvolvimento desse trabalho.

Finalmente, o Apêndice B apresenta uma breve discussão sobre diversas questões de segurança que foram incorporadas na aplicação.

CAPÍTULO 2

O ESTADO DA ARTE - UCA e DLMS/COSEM

Neste capítulo é apresentada uma breve revisão de algumas propostas do estado da arte do processo de integração de sistemas heterogêneos no contexto das empresas de energia elétrica (EEE). Uma delas é a arquitetura UCA que fornece um *framework* para a integração de sistemas dentro das empresas do setor. A outra é a DLMS/COSEM que é uma iniciativa de um grupo de empresas européias e visa especificar um medidor de energia interoperável, que troca informações baseadas em serviços.

2.1 Introdução

Na última década a indústria de equipamentos de medição procurou integrar importantes avanços tecnológicos na concepção de novos dispositivos eletrônicos inteligentes. Inicialmente, ainda influenciados pela área de aquisição de dados, os investimentos em pesquisa direcionavam-se para o desenvolvimento de sistemas de transporte de mensagens. Assim, arquiteturas e protocolos como MMS, DNP3 e Modbus chegaram ao mercado. Contudo, esses esforços não resolvem completamente um problema chave na área, que é a interoperabilidade. Isso significa não somente a possibilidade de utilização de diferentes interfaces para diferentes sistemas de comunicação, mas também a possibilidade de integração de dispositivos de diferentes fabricantes.

A incorporação de conceitos da tecnologia de *software* conhecida como Orientação a Objetos permitiu avanços consideráveis na questão da interoperabilidade. Com essa tecnologia é possível a descrição de um modelo de medidor padrão (dispositivo lógico) e, a partir daí, a derivação de dispositivos lógicos específicos para os equipamentos de diferentes fabricantes. Estes novos modelos devem trazer como diferencial o uso de um conjunto semântico único. Nesse sentido, estão sendo apresentadas, por organismos

internacionais de normatização, propostas de padronização para aplicação em empresas do setor elétrico.

Com o objetivo de compreender o estado da arte na tecnologia de integração de dados e medição e poder absorver os principais conceitos, os sub-itens abaixo apresentam as características fundamentais de dois esforços de padronização no contexto das EEE para uma nova geração de medidores de energia elétrica, a UCA e a DLMS/COSEM.

2.2 UCA (*Utility Communications Architecture*)

2.2.1 Introdução

A UCA está sendo desenvolvida no âmbito do EPRI (*Electric Power Research Institute*) com o envolvimento de setores da indústria [UCA 97a]. O objetivo é definir padrões para integração completa de empresas atuantes no setor, tomando como base padrões de comunicação internacionais já estabelecidos (Figura 2.1). As áreas funcionais enfocadas incluem: interface com o consumidor, distribuição, transmissão, centro de controle e sistemas de informação da empresa.

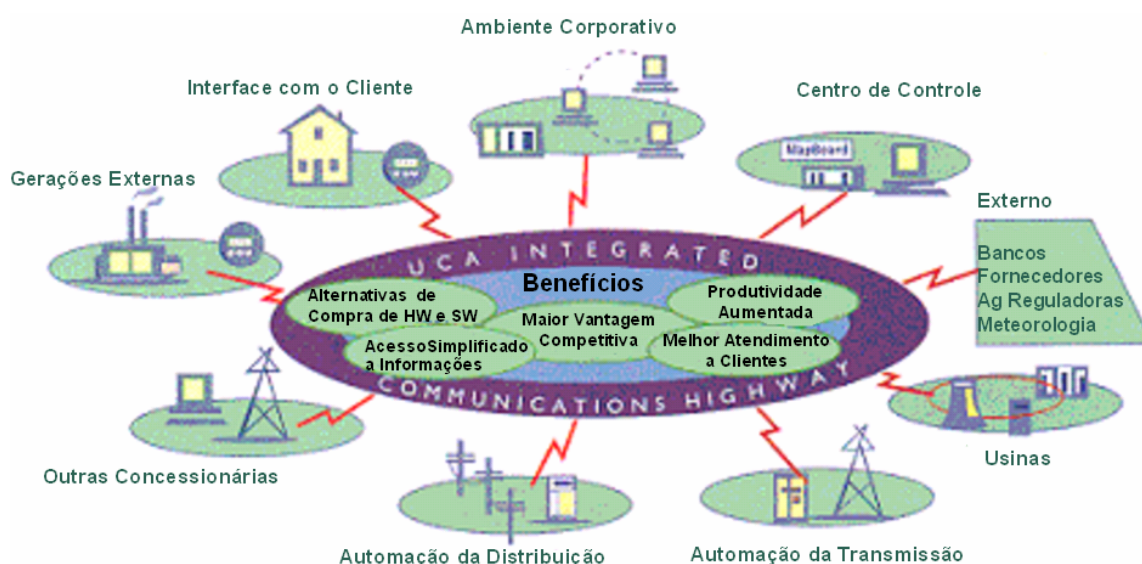


Figura 2.1 – Integração através da arquitetura UCA.

A UCA acompanha o estado da arte da tecnologia de computação usando a orientação a objetos na definição de modelos de dispositivos de interesse. Esses modelos definem formatos de dados comuns, identificadores e controles para dispositivos, tais como unidades de medição, *switches*, reguladores de voltagem, etc. Os modelos especificam os dados e o comportamento básico para a maioria das funções comuns do dispositivo específico, deixando em aberto a possibilidade de especializações introduzidas por diferentes fabricantes. Em princípio, os modelos são desenvolvidos através de um processo aberto à participação de fabricantes e empresas usuárias do setor. Um objetivo imediato da padronização seria alcançar a interoperabilidade facilitando a integração de dispositivos de diferentes fabricantes.

No contexto UCA, os modelos de objetos são definidos em termos de tipos padronizados de serviços. Estes serviços são definidos em termos abstratos e então mapeados para mensagens do protocolo básico da camada de aplicação. Os serviços básicos da camada de aplicação, da versão 2.0 de UCA, para aquisição de dados e funções de controle em todos os perfis são fornecidos pelo ISO/IEC 9506: MMS (*Manufacturing Message Specification*). Num nível intermediário, acima da MMS, são definidos serviços padronizados que objetivam isolar o efeito de possíveis revisões da camada de aplicação, sem incorrer em re-definições do modelo de objetos.

A UCA adota diversos protocolos estruturados de acordo com o modelo de referência OSI (*Open System Interconnection*) da ISO. O modelo de referência aloca as funções de comunicação para camadas definidas, as quais suportam uma variedade de padrões, de modo a permitir várias opções de custo e de desempenho. Cada usuário pode escolher dentre as opções de cada camada para definir os perfis adequados para sua aplicação. A UCA inclui dois perfis de 7 camadas, um usando o padrão OSI e outro usando o TCP/IP. A UCA também inclui um perfil de 3 camadas para uso sobre enlaces seriais em dispositivos de baixo custo.

2.2.2 Visão de Camadas (Perfis)

A proposta UCA teve sua origem como um *framework* para atender o conjunto de requisitos de comunicação que existe nas empresas do setor elétrico. A idéia surgiu da necessidade de permitir a comunicação entre os níveis de planejamento, SCADA, medição, proteção, e outros dispositivos de controle. Inicialmente, o processo se concentrou nos requisitos de comunicação entre as várias funções dentro de uma subestação. O objetivo era definir o perfil de subestação da "próxima geração", usando uma rede ponto-a-ponto de alta velocidade baseada em protocolos de comunicação existentes. O objetivo de interoperabilidade levou também à definição de nomes padrões para objetos e dados comumente usados. O processo resultou no perfil mostrado na Figura 2.2.

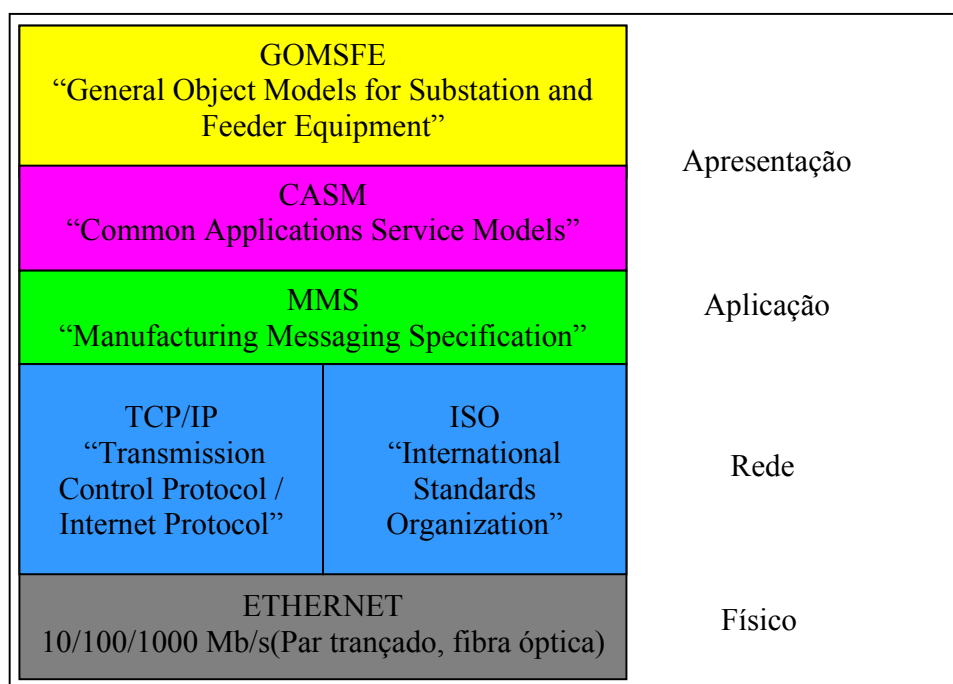


Figura 2.2 – Perfil UCA.

O perfil ilustrado usa a Ethernet para as camadas físicas e de enlace de dados. Embora o protocolo Ethernet original não seja determinístico, existem opções que amenizam esta característica permitindo sua utilização em aplicações de tempo real. Além disso, a tecnologia Ethernet oferece várias opções em termos de custo e desempenho. Para as camadas de rede, embora o objetivo original fosse permanecer dentro da esfera dos

padrões ISO (*International Standards Organization*), a popularidade da Internet ditou a inclusão de camadas de rede TCP/IP como um padrão *de facto*. Em 1999, o IEC (*International Electrotechnical Committee*) selecionou o TCP/IP como protocolo de rede mandatório para comunicação intra-subestação e inter-subestações, colocando as camadas ISO como opcionais. A adoção do TCP/IP abre caminho para a utilização de diversas outras tecnologias de comunicação no nível físico e para o estabelecimento de redes tipo Internet.

Para a Aplicação ou camada de serviço, foi escolhido o protocolo MMS (*Manufacturing Message Specification*), que disponibiliza um extenso conjunto de serviços para ler, escrever, definir e criar objetos de dados [MMS 90]. Duas outras camadas são definidas no modelo: CASM e GOMSFE.

2.2.3 CASM (*Common Application Service Models*)

O UCA/CASM [UCA 97b] fornece um conjunto comum de funções de comunicação (acesso a dados, relatórios, registros, funções de controle, etc.) que são encontradas na maioria dos dispositivos de campo. A definição dos serviços CASM visa alcançar:

- Isolação dos esforços de modelagem de detalhes de comunicação;
- Interoperabilidade de comunicação, não somente na sintaxe da mensagem, mas também na semântica dos dados trocados; e
- Redução de custos de integração e desenvolvimento através do uso de mecanismos comuns para acesso a dados e estabelecimento de comunicação.

Os serviços CASM podem ser mapeados em diversos padrões de comunicação existentes, contudo, no contexto UCA, foi escolhido o padrão MMS (ISO 9506). Isso permite que um fabricante implemente um *software* de aplicação em conformidade com a

UCA, independentemente das camadas inferiores de protocolo. Desta forma, as camadas inferiores podem ser escolhidas em função das necessidades operacionais do ambiente de instalação.

Os serviços CASM são definidos através de técnicas de modelagem de objetos. Dispositivos de campo podem incorporar os serviços do CASM pela definição por parte do fabricante das classes dos modelos de que seus objetos serão derivados. Por exemplo, um *Intelligent Electronic Device* (IED) que contenha um objeto de controle, que requeira um *commit* em duas etapas, por exemplo, deveria herdar os atributos e métodos associados à classe *Select Before Operate* (SBO) definidas no sub-modelo *Device Control Service* (ver em seguida) do CASM.

Em geral, os serviços CASM são definidos pela descrição dos procedimentos do servidor, que podem ser invocados por transações iniciadas por clientes. O Modelo de Servidor do CASM (Figura 2.3) consiste de vários sub-modelos:

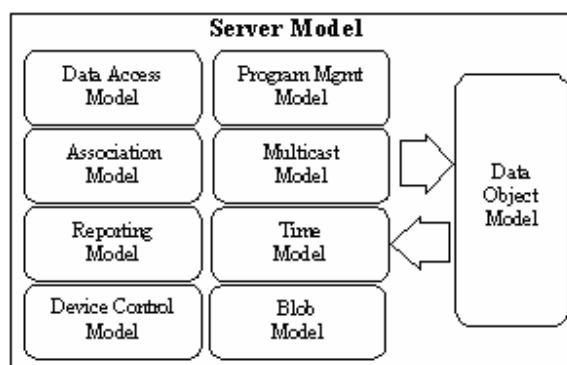


Figura 2.3 – O modelo de servidor CASM.

- **DataObject** – descreve as convenções de nomeação e construção dos dados do dispositivo;
- **Association** – descreve as construções e serviços para estabelecer e encerrar conexões entre um cliente e o Servidor;

- ***Data Access Service*** – define serviços utilizados para ler e/ou escrever em objetos de dados;
- ***Reporting Service*** - descreve as construções e serviços que o Servidor deverá suportar para monitorar condições e gerar relatórios de modo contínuo, baseado em parâmetros estabelecidos pelo cliente;
- ***Device Control Service*** - descreve as construções e serviços que devem ser suportados pelo Servidor para permitir que clientes controlem dispositivos conectados ao Servidor;
- ***Program Mgmt*** - gerencia os demais modelos dentro do servidor;
- ***Multicast Service*** - descreve as construções e serviços associados à operação em *multicast*;
- ***Time*** - descreve as construções e serviços que devem ser suportados pelo Servidor para permitir acesso e sincronização ao relógio do sistema implementado;
- ***Blob*** – fornece suporte para a transferência de dados arbitrariamente longos que não poderiam ser inseridos em uma única mensagem.

2.2.4 GOMSFE (*Generic Object Models Substations & Feeder Equipment*)

O documento GOMSFE [UCA 00] define uma biblioteca básica, etiquetas (*tags*) de informação, objetos comuns e modelos de objetos específicos padronizados (*bricks*) usados para a modelagem de subestação, controle ou aquisição de dados. Dentre os diversos itens disponíveis no modelo pode-se listar:

- Entradas/Saídas Genéricas (I/O);

- Funções de medição, transformação, interrupção e proteção;
- Modelo de objetos para RTU, transformador, interruptor e banco de capacitores.

O GOMSFE aplica o conceito de “*bricks*”, ou blocos básicos de construção, que são grupos reutilizáveis padronizados de objetos que representam uma função ou um dispositivo dentro de uma subestação. Os modelos de objetos são então definidos como grupos especializados de *bricks* associados que modelam dispositivos, funções ou aplicações de uma função específica ou aplicação no domínio do problema. É mostrada na Figura 2.4 a hierarquia utilizada na modelagem de objetos no GOMSFE, que são definidos como:

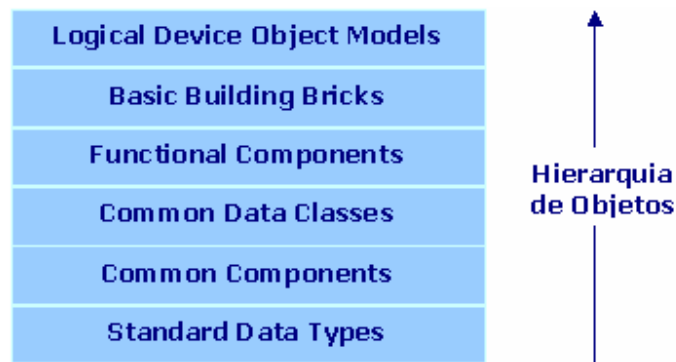


Figura 2.4 – Hierarquia de modelo de objetos.

- ***Standard Data Types*** definem a representação dos valores de classes e objetos. Determinam o formato, o número de bits e a faixa de valores possíveis;
- ***Common Components*** são abreviaturas que representam os componentes elementares usados na definição de classes e objetos. A lista de *Common Components* representa o conjunto mínimo que define as *Common Classes*. Um exemplo de *common component* é “b”, que representa um valor binário com *data type* BOOL;

- **Common Data Classes** são grupos de estruturas de componentes que são atributos do objeto. Os componentes são variáveis nomeadas associadas a dados representados pelas *Common Data Classes*;
- **Functional Components** são grupamentos que contêm os objetos de dados usados para definir o modelo do dispositivo;
- **Basic Building Bricks**, conforme citado anteriormente, são grupos reutilizáveis e padronizados de objetos de dados associados que tratam de uma função ou uso específico;
- **Logical Device Object Models** são grupos especializados de “bricks” associados que representam dispositivos, funções ou aplicações de uma função específica.

A relação entre o GOMSFE e o CASM está indicada na Figura 2.5, que mostra o CASM como fornecedor de serviços para utilização de dispositivos concebidos segundo o modelo GOMSFE. O CASM é responsável então pelo mapeamento dos objetos na camada de aplicação.

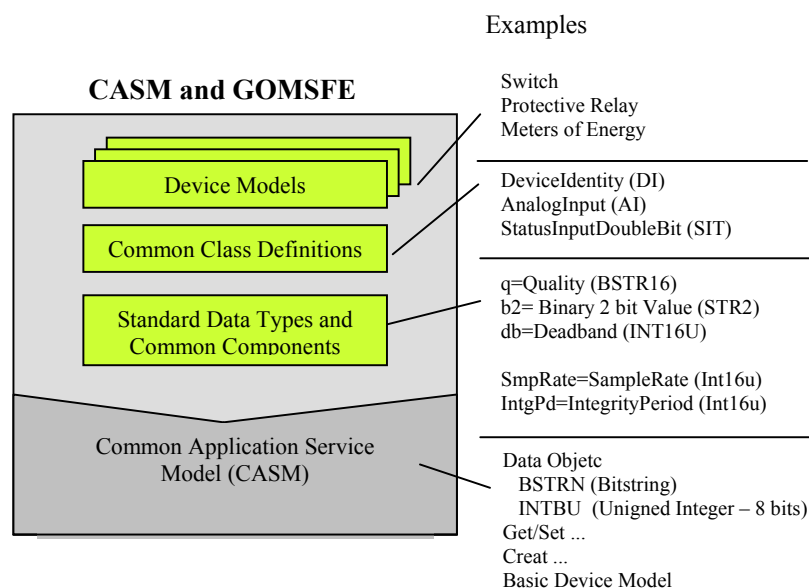


Figura 2.5 – Estruturação CASM / GOMSFE.

2.2.5 Exemplo de Utilização

A Figura 2.6 ilustra o processo de desenvolvimento de um componente segundo o padrão UCA. Numa etapa inicial, o fornecedor do dispositivo elabora uma Especificação de Produto, que define a funcionalidade que o dispositivo deve realizar de acordo com os requisitos da empresa. Com base na especificação inicial, os modelos padronizados apropriados da coleção genérica da UCA são selecionados. Se requerido, o fornecedor pode ainda adicionar ao conjunto especializações próprias, chegando a um modelo de dispositivo lógico específico chamado Modelo de Produto do fornecedor. É então feito o mapeamento dos objetos genéricos e serviços no CASM. Neste processo, o vendedor produz uma Definição de Aplicação e especifica completamente o *software*/protocolo de comunicação requerido para suportar o produto como um dispositivo compatível com o padrão UCA, independentemente dos protocolos das camadas de níveis inferiores. O fornecedor determina as camadas de protocolo mais baixas considerando o ambiente de operação onde será inserido o dispositivo (e.g., LAN de alta velocidade, rede WAN, etc.) e seleciona o perfil UCA apropriado a ser suportado. A especificação do perfil de comunicação, combinada com a definição da aplicação, gera o Projeto do Produto Final para implementação do *software* e disponibilização do produto.

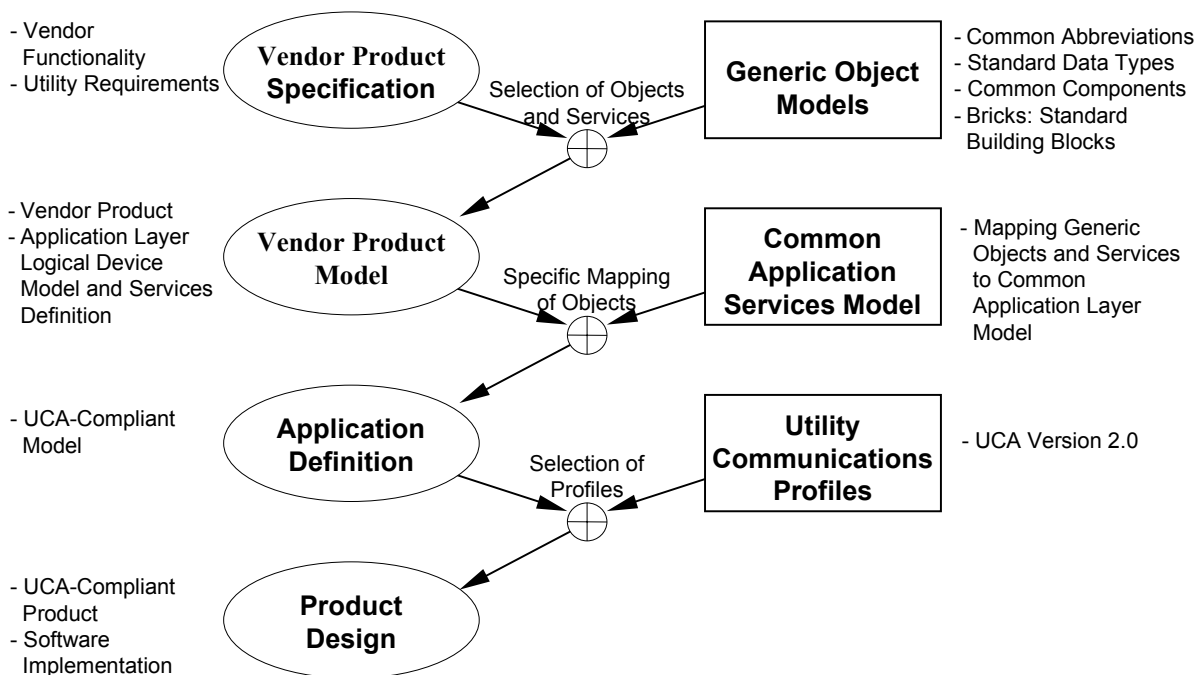


Figura 2.6 – Definição de um produto UCA.

2.2.6 Exemplo de Rede Integrada com UCA

O nível geral de integração alvo da arquitetura UCA é mostrado no diagrama apresentado na Figura 2.7. O diagrama mostra diversos níveis que poderiam ser integrados através de modelagens específicas e uma topologia adequada de rede. Como pode ser observado o CASM/GOMSFE é utilizado nos níveis de geração e de subestações de distribuição e transmissão. Uma opção da arquitetura é prevista para comunicação com dispositivos de consumidores. Exemplo de aplicações incluem leitura de medidores, cobrança eletrônica, monitoramento da qualidade de energia, detecção de interrupção, conexão/desconexão remota, e gerenciamento do lado da demanda. A modelagem UCA de dispositivos de consumidores utiliza serviços CASM, os quais podem ser usados sobre perfis de 7 ou 3 camadas, sobre uma variedade de meios de comunicação. A definição desta opção da arquitetura (*UCA Customer Interface Device Models*) encontra-se ainda em desenvolvimento, conforme citado em [UCA 97a].

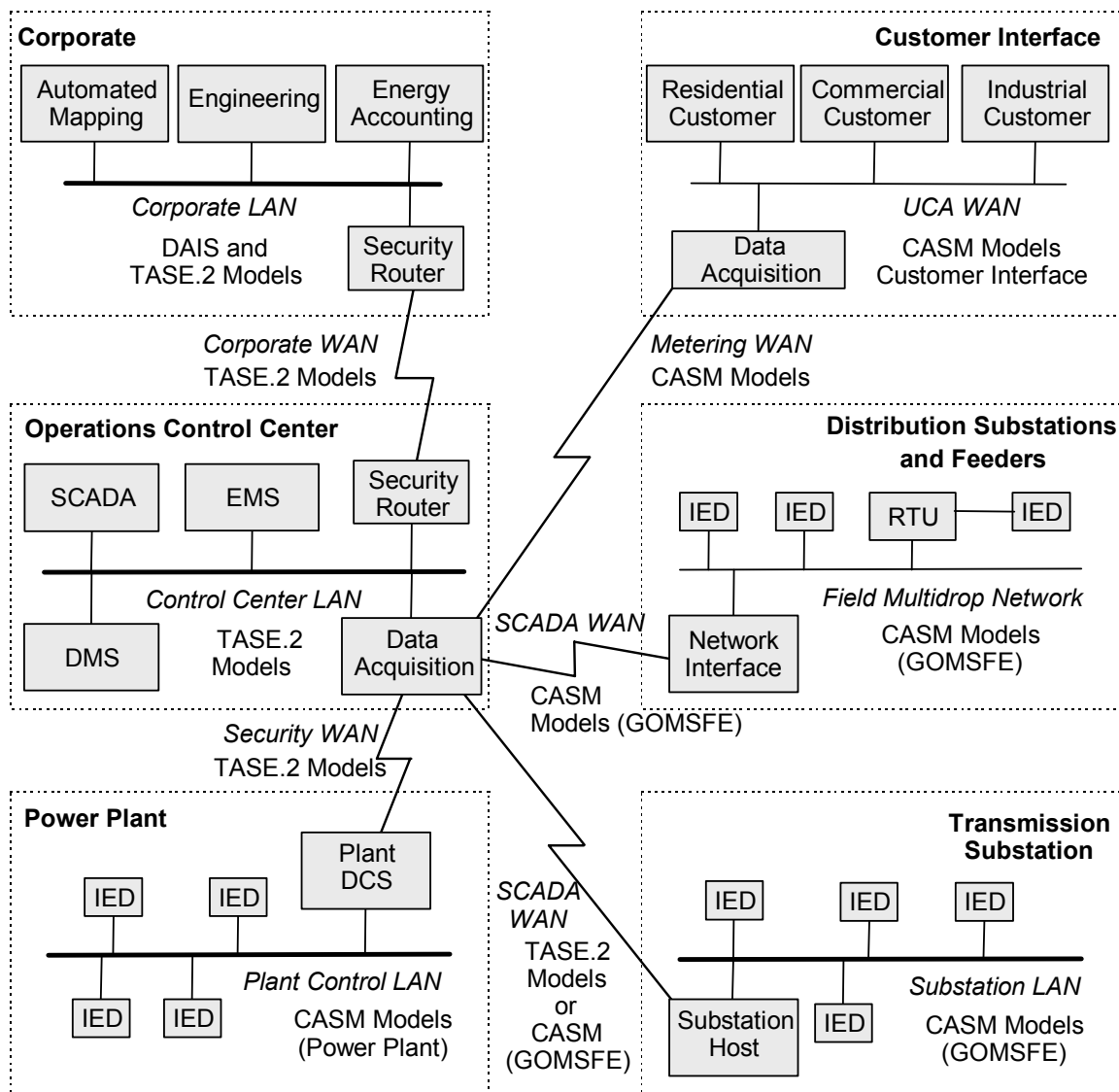


Figura 2.7 – Rede integrada UCA.

2.3 DLMS/COSEM

2.3.1 Introdução

DLMS/COSEM é uma iniciativa de um grupo de empresas europeias (e.g., Siemens e Schlumberger) com o objetivo de especificar um medidor interoperável, que possa ser utilizado em rede, através de diferentes formas de comunicação. DLMS significa *Device Language Message Specification* e representa um sistema de mensagens derivado do MMS.

COSEM significa *Companion Specification for Energy Metering* e trata da definição de um medidor como um dispositivo lógico, construído a partir de um grupo de blocos funcionais (os objetos).

COSEM é uma especificação adicional em relação ao padrão DLMS, ou seja, trata de uma extensão em relação a esse padrão (xDLMS). É importante ser notado que essa especificação não só define como os dados são transportados, mas, principalmente, especifica as funções de um medidor e os correspondentes métodos de acesso. Ou seja, permite que medidores sejam definidos de forma padrão, independentemente de modelo ou fabricante.

Na sequência, serão apresentados cinco pontos importantes, na ótica desse trabalho, para a compreensão da forma como a interoperabilidade é implementada através desse padrão: a arquitetura COSEM, as classes de interface, o sistema de nomeação de objetos, o modelo de servidor e o conceito de associação. É importante acrescentar que essa proposta é objeto de padronização pelo IEC.

2.3.2 Arquitetura COSEM

A estruturação de COSEM é feita em um modelo reduzido de três níveis (em relação ao modelo OSI/ISO de 7 níveis), sendo que somente o último nível, o de Aplicação, deve ter elementos específicos de suporte. Adicionalmente, essa estruturação torna transparentes os protocolos para transporte dos dados e o meio de comunicação utilizado, que tanto pode ser, por exemplo, uma WAN (*Wide Area Network*) quanto um par de fios trançados, entre outras possibilidades. COSEM trabalha com base no paradigma Cliente-Servidor, onde o dispositivo de medição representa o provedor de serviços. A Figura 2.8 representa esse modelo [DLMS 02a].

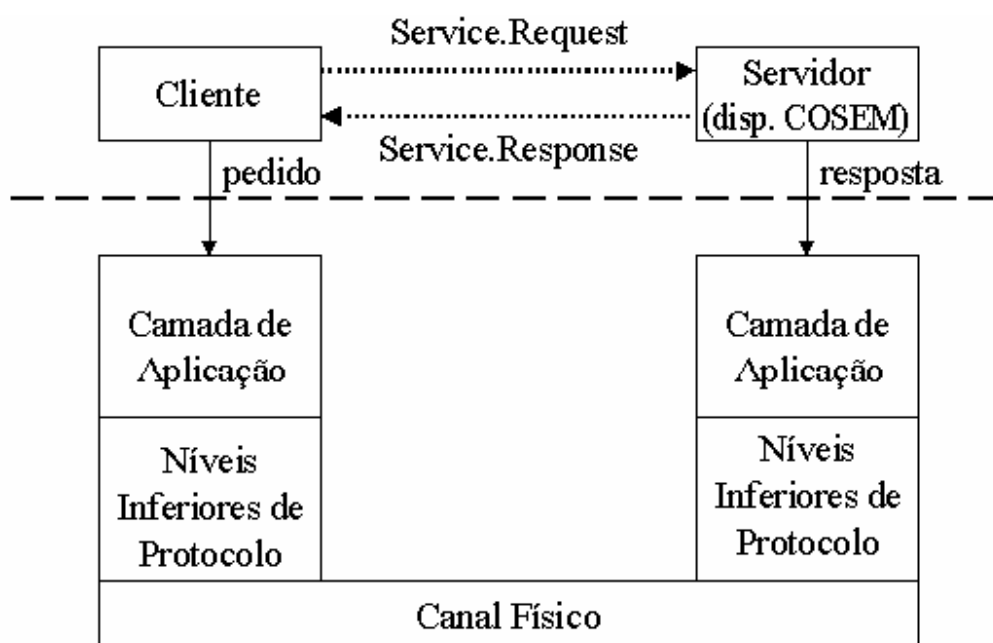


Figura 2.8 – Relação cliente-servidor em COSEM.

A parte dessa arquitetura a ser destacada é a camada de Aplicação. Essa camada em COSEM é especificada em termos da estrutura, dos serviços oferecidos aos processos da aplicação (tanto ao cliente quanto ao servidor) e dos protocolos. A Figura 2.9 identifica os principais elementos dessa camada.

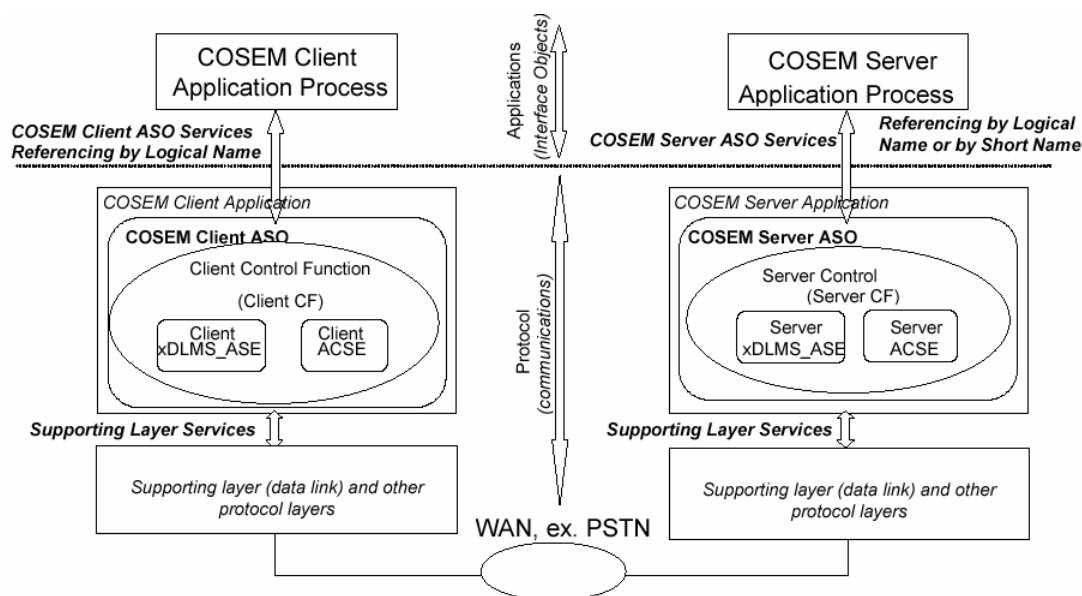


Figura 2.9 – Estrutura da camada de aplicação.

O principal componente de um cliente ou de um servidor no nível Aplicação é o ASO (*Application Service Object*), que fornece os serviços aos processos da aplicação e usa os serviços dos níveis inferiores de protocolo. Esse componente contém, minimamente, três elementos obrigatórios:

- ACSE (*Association Control Service Element*), que estabelece, mantém e libera associações, com ligação orientada a conexão;
- xDLMS_ASE (*Extended DLMS Application Service Element*), cujo papel é fornecer serviços de comunicação de dados entre equipamentos COSEM; e
- CF (*Control Function*), que especifica como os serviços do ASO invocam as primitivas do ACSE, do xDLMS_ASE e dos serviços da camada de suporte.

2.3.3 Classes de Interface

COSEM é uma especificação que permite a definição de medidores da mesma forma como esses são vistos pelos usuários dos sistemas de medição reais. As questões internas relativas à implementação não são especificadas no modelo. Um medidor é visto como um conjunto de blocos funcionais (e.g., registradores, relógio, escalonador), que podem ser acessados segundo regras estabelecidas. Em outras palavras, através da composição desses blocos funcionais, essa especificação permite a definição de um “medidor lógico”. Isso é feito através da técnica de modelagem por objetos, quer dizer, através da definição de classes de interface específicas ao domínio do problema (medição) e de suas instâncias (objetos).

O padrão já define um certo número de classes para suporte à medição de energia, através de uma biblioteca de classes. Para cada classe são definidos atributos (e.g., nome lógico, valor) e métodos (e.g., *reset*, *adjust_time*). Essas classes constituem os blocos básicos para a construção dos dispositivos lógicos, e foram criadas para mapear funções tradicionalmente existentes nos medidores de energia. Algumas das classes existentes nessa

biblioteca são *Register*, *Clock*, *Activity Calendar* e *Association* [DLMS 02b]. Por exemplo, um objeto da classe *Register* armazena um valor, cuja natureza é descrita através do atributo *logical name*, utilizando o padrão de identificação da especificação COSEM, o OBIS - *Object Identification System* [DLMS 02b].

2.3.4 Sistema de Identificação de Objetos

De forma a garantir a interoperabilidade, os nomes associados a todas as formas de dados, medidos ou processados (e.g., voltagem, potência ativa, média do período de tarifação) devem seguir a especificação OBIS. Isso é feito para garantir que diferentes fabricantes usem as mesmas regras para nomeação em seus produtos. Assim, um nome é uma combinação de seis grupos de valores, que descrevem de forma hierárquica o significado exato de cada item de dados. Simplificadamente, esses seis grupos representam [DLMS 02b]:

- **Grupo A:** define a característica do item a ser identificado, ou seja, se é um dado abstrato, ou se é um dado relacionado a sistemas específicos (eletricidade, gás, aquecimento ou água);
- **Grupo B:** define o número da entrada em um equipamento de medição multicanal;
- **Grupo C:** define o tipo de dado indicado no Grupo A. Por exemplo, se for um dado abstrato, esse campo poderá conter identificadores de contexto; se for um dado associado a um sistema elétrico, esse campo poderá indicar um fator de potência;
- **Grupo D:** identifica tipos resultantes do processamento de quantidades indicadas nos campos A e C. Por exemplo, pode identificar uma média de um período de tarifação;

- **Grupo E:** identifica processamentos adicionais em resultados de medidas, feitos a partir de dados identificados nos campos A até D. Por exemplo, poderia ser o harmônico 127 de uma grandeza identificada em A como elétrica, que em C é descrita como uma voltagem e que em D está descrita como valor instantâneo; e, finalmente
- **Grupo F:** define o armazenamento de dados para cada um dos diferentes períodos de tarifação.

2.3.5 Modelo de Servidor COSEM

Um dispositivo em COSEM é estruturado em três níveis: o dispositivo físico (e.g., um medidor), o dispositivo lógico (uma unidade funcional) e os objetos constitutivos [DLMS 02b]. Esses últimos representam os blocos funcionais a partir dos quais os dispositivos lógicos são construídos. A Figura 2.10, que indica a composição de um possível medidor multifunção, representa os conceitos acima referidos.

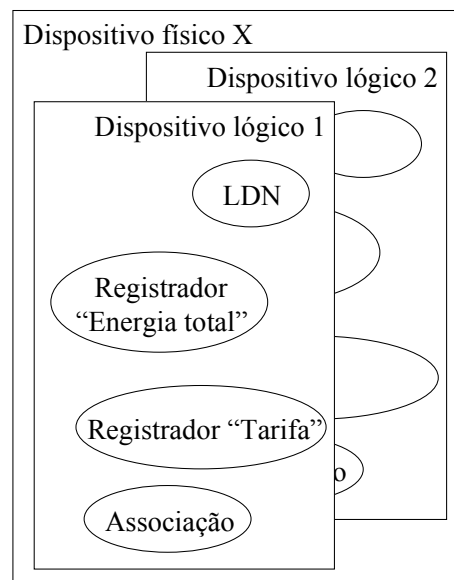


Figura 2.10 – Modelo de servidor em COSEM.

Na Figura 2.10, um dispositivo físico X é composto de dois dispositivos lógicos, 1 e 2. Cada um desses dispositivos lógicos é composto de um conjunto de objetos. No caso do

dispositivo 1, o objeto LDN identifica o dispositivo lógico, dois objetos do tipo Registrador guardam valores associados a parâmetros de medição, e o último objeto indicado mantém as informações relativas à associação. É importante notar que esse modelo é extremamente geral e flexível, possibilitando a composição de diferentes tipos de dispositivos.

2.3.6 Associações

Uma das classes de interface mais importantes, que define o relacionamento entre um usuário (cliente) e os objetos disponíveis no medidor, é a Associação [DLMS 02b]. Como indicado na apresentação da camada de Aplicação, de forma a acessar os objetos de um servidor (medidor), é necessária a criação de uma Associação. Durante a fase de conexão, quando a associação é estabelecida, um contexto é definido. Dependendo da associação criada, diferentes direitos de acesso podem ser oferecidos. Esses direitos dizem respeito aos objetos vistos pelo cliente, bem como às restrições que serão impostas nos acessos aos atributos e métodos desses objetos (e.g., atributo só disponível para leitura).

Os objetos visíveis em uma associação podem ser obtidos por um cliente através da leitura do atributo "*object_list*" do objeto "associação". Assim, diferentes clientes podem ter diferentes visões de um mesmo objeto. Além disso, como parte do contexto da associação, um procedimento de autenticação pode ser definido. Esse é um importante mecanismo de segurança embutido na arquitetura.

A especificação COSEM prevê dois tipos de autenticação, cuja utilização é dependente do grau de segurança já fornecido pelo sistema de comunicação. No primeiro caso, quando o sistema de comunicação já oferece mecanismos que evitem a "espionagem" dos dados trocados, um simples esquema de *password* é previsto. No segundo caso, quando não se espera nenhum suporte adicional por parte do sistema de comunicação, um protocolo de quatro fases é estabelecido entre cliente e servidor para implementar um mecanismo de autenticação baseado em chaves encriptadas.

Mais detalhes sobre as associações serão abordados nos capítulos seguintes.

CAPÍTULO 3

INTEGRAÇÃO DE SISTEMAS HETEROGÊNEOS

O suporte às atividades de uma empresa requer a colaboração entre os sistemas de computação que ela possui para compartilhar informações e processos disponíveis nesses sistemas. Uma diversidade de problemas dificulta a colaboração entre esses sistemas para fornecer o suporte requerido. Neste capítulo serão apresentados os diversos problemas encontrados pelas empresas para alcançar interoperabilidade entre sistemas. São enfatizados os problemas de integração que surgem no contexto dos medidores de energia.

3.1 Introdução

Nos dias atuais as empresas possuem seus processos de negócio automatizados por sistemas de computação. Porém, na maioria das vezes, os sistemas que compõem esse ambiente computacional são independentes uns dos outros, usam protocolos, linguagens, formato de arquivos e padrões de representação de dados diferentes, e muitas vezes são projetados e construídos por pessoas ou mesmo por empresas distintas. Essa heterogeneidade de sistemas dificulta bastante a integração e disponibilização das informações de maneira global devido ao fato deles não conseguirem se “comunicar”. Essa visão global e unificada dos dados é atualmente um requisito importante para propiciar a agilidade das atividades internas e o crescimento da competitividade da empresa.

Além do uso de diferentes tecnologias, existem outros problemas relacionados ao processo de integração de sistemas. Pode ser citada como exemplo a questão do entendimento, onde muitas vezes dois sistemas possuem as mesmas informações, mas as tratam com “nomes” diferentes.

Outras questões que devem ser consideradas surgem devido à arquitetura utilizada durante o processo de integração. Por exemplo, muitas vezes é utilizada a própria

arquitetura da Internet como meio de transmissão de dados entre os sistemas e, portanto, dados de caráter sigilosos estariam transitando livremente, e poderiam ser interceptados por *hackers*. Assim, questões de segurança devem ser consideradas com o objetivo de evitar esses crimes digitais.

Os sub-itens abaixo apresentam, resumidamente, os problemas que surgem no processo de integração de sistemas heterogêneos.

3.2 Heterogeneidade dos Sistemas e sua Integração

O ambiente computacional atual é composto por sistemas de *software* autônomos, distribuídos e heterogêneos. A integração desses sistemas apresenta diversas dificuldades, resultantes do fato de que eles não foram originalmente projetados para trabalhar em conjunto.

A heterogeneidade destas aplicações e sistemas é caracterizada pelas diferenças em suas implementações e podem ser agrupadas em [Silv 03]:

- Heterogeneidade tecnológica,
- Heterogeneidade semântica, e
- Heterogeneidade sintática.

A heterogeneidade tecnológica ocorre em função da aplicação de diferentes tecnologias, incluindo plataformas de *hardware*, sistemas operacionais, tecnologias de bancos de dados, formatos de representação de dados e linguagens de programação no desenvolvimento das aplicações.

A heterogeneidade sintática é devida à incompatibilidade das representações das informações em cada sistema: estruturas e tipos de dados escolhidos para um sistema podem ser completamente diferentes dos implementados em outro. Por exemplo, um sistema pode ter suas variáveis de condição representadas por uma variável booleana

armazenando *true* ou *false*; um outro sistema pode utilizar uma variável do tipo caracter com os valores “S” e “N” e, ainda, um terceiro sistema utilizar uma variável inteira com valores 0 e 1.

A heterogeneidade semântica está ligada aos conflitos que podem ocorrer resultantes de diferentes representações de conceitos em sistemas que devem interagir. De forma geral, esses conflitos de heterogeneidade semântica podem ser agrupados em [Reyn 03]:

- *Conflitos de Confusão*: ocorrem quando elementos de informação parecem ter o mesmo significado, mas não o tem;
- *Conflitos de Escala*: ocorrem quando diferentes sistemas de referência são utilizados para medir valores. Por exemplo, utilização de diferentes moedas;
- *Conflitos de Nomes*: ocorrem quando esquemas ou variáveis de nomes iguais possuem significados diferentes. Um problema freqüente neste sentido é a presença de homônimos e sinônimos.

Em [Reyn 03] é definida a interoperabilidade como sendo a habilidade de dois ou mais componentes de *software* cooperarem, a despeito de suas diferenças de linguagens de programação, interfaces, e plataformas de execução.

A integração de sistemas heterogêneos tem como objetivo permitir que estes sejam interoperáveis e que, deste modo, possam trocar informações e serviços, independentemente de suas diferenças tecnológicas, semântica ou sintática. Para permitir a troca de informações entre sistemas com problemas de heterogeneidade sintática e semântica, mecanismos de transformações estruturais e semânticas devem ser utilizados. A heterogeneidade tecnológica pode ser contornada pelo uso de mecanismos e protocolos que promovam a interligação dos sistemas.

3.3 Padrões

Um outro grande problema no contexto da integração de sistemas está relacionado a padrões. Um sistema quando criado obedecendo às regras e restrições de um padrão têm maior probabilidade de ser interoperável com outros sistemas que seguem o mesmo padrão. Mas, para isso, é muito importante que esses padrões sejam:

- Bem descritos, para que sejam implementados de maneira adequada;
- Abrangentes, para cobrirem as diversas funcionalidades necessárias ao sistema;
- Extensíveis, para permitir a inclusão de possibilidades “esquecidas” ou inexistentes durante a especificação do padrão.

Nos sub-itens 3.3.1 e 3.3.2 apresentaremos resumidamente uma exposição dos problemas e restrições existentes nos padrões e normas do contexto de medição de energia. Serão abordados a norma ABNT NBR 14522 [NBR 00] e o padrão DLMS/COSEM respectivamente.

3.3.1 A Norma NBR 14522

A Norma NBR 14522 (Intercâmbio de Informações para Sistemas de Medição de Energia Elétrica - Padronização) é a única norma que é imposta aos medidores de energia fabricados no Brasil, no que se refere à padronização de dados e troca de informação. Essa norma, criada em 1995, reúne os dados que formam os requisitos mínimos necessários para o propósito de tarifação de energia no contexto nacional. Ela define a estrutura de comandos e respostas que permitem o intercâmbio de informações (parametrização e leitura) entre medidor e leitor de dados.

Em relação aos três requisitos básicos, relacionados a padrões, citados acima, podemos dizer que essa norma é relativamente bem descrita, deixando a desejar no que diz

respeito aos diversos parâmetros e grandezas que podem existir em um medidor de energia; não é muito abrangente, pois, como dito acima, é uma norma que apresenta quase que somente os requisitos necessários à tarifação; e, é pouco extensível, pois são reservados apenas alguns comandos não implementados para futuras necessidades.

O principal problema com a NBR 14522 é justamente a pequena capacidade de expansão possível. Como exemplo, existem hoje no parque de medidores instalado nas EEE, medidores importados conhecidos como “Medidores de Fronteira”, que atualmente podem possuir até setenta e dois canais de dados. A NBR 14522 só prevê comandos para ler no máximo seis canais. Prever e permitir extensibilidade e a incorporação de informações adicionais, além de métodos para leitura dessas informações agregadas, torna-se um fator fundamental para permitir a construção de sistemas genéricos de comunicação com esses dispositivos.

É importante citar que, no contexto desse trabalho, foram estudados três medidores de energia elétrica de fabricação nacional, o “Elo 2180”, o “Spectrum SX” da Nansen, e o “Saga 1000” da ESB. Todos os três modelos dizem estar de acordo com a norma ABNT 14522. Mas o que ocorre na verdade é que eles usam essa norma somente como base. O Elo 2180, o mais simples dentre esses medidores, implementa apenas um conjunto de características dessa norma. O Saga 1000 é bastante fiel a norma, mas não cobre todas as suas características; por outro lado, acrescenta características próprias adicionais ao especificado pela norma. Já o Spectrum SX implementa muitas características da norma, desrespeita algumas e acrescenta uma diversidade de funcionalidades adicionais.

3.3.1 A Especificação DLMS/COSEM

O Padrão DLMS/COSEM (abordado no sub-item 2.3) define uma variedade de classes de interface que possibilitam o mapeamento das diversas funcionalidades de um medidor. É um padrão que permite o acréscimo de funções extras por parte dos fabricantes para os seus medidores específicos, sem prejudicar a interoperabilidade entre os diversos medidores que “falam” COSEM. Isso é devido à sua abordagem orientada a objetos.

O Padrão DLMS/COSEM é bem descrito e bastante abrangente, permitindo descrever através de suas classes quase todas as funções de um medidor (quase todos, pois algumas classes adicionais foram propostas nesse trabalho para o mapeamento de algumas idiossincrasias dos medidores nacionais, vide sub-item 6.3). É também extensível, permitindo a incorporação de funcionalidades extras.

O principal problema em se implantar o padrão COSEM esbarra na necessidade da implementação da camada de aplicação (como visto no sub-item 2.3.2) necessária para a comunicação entre cliente e servidor (o medidor). Um outro problema está relacionado à necessidade de operar com os medidores atuais, pois seria necessária a construção de *gateways* que permitissem a conversão de dados entre os vários modelos e os dispositivos lógicos COSEM. No futuro um medidor que seja construído para o *framework* COSEM poderá vir com as funcionalidades do dispositivo lógico já incorporadas.

3.4 Soluções Proprietárias

No contexto das EEE diversos dispositivos eletrônicos inteligentes são responsáveis pela automação e controle da distribuição, transmissão e medição de energia, como os relés de proteção e os medidores de energia. Esses dispositivos são fornecidos geralmente por diversos fabricantes e, muitas vezes, com vários modelos, e ainda com várias versões para cada modelo.

No projeto desses dispositivos, geralmente, existe a preocupação por parte dos fabricantes em seguir alguma norma, quando existe uma. Porém, quase sempre os fabricantes não seguem a norma completamente e, muitas vezes, fornecem funcionalidades além da norma. Isso ocorre porque as EEE necessitam cada vez mais de funções extras dos dispositivos. Por exemplo, funções que permitam avaliar a qualidade do fornecimento de energia, o que faz com que os fabricantes busquem soluções para atender a essas necessidades das EEE e mesmo tornar seus produtos mais atrativos.

Para cada dispositivo existem *softwares*, geralmente construídos pelos próprios fabricantes e específicos para cada modelo, que são capazes de comunicar e trocar informações com esse dispositivo. O problema com esses *softwares* é que, em muitos casos, eles são construídos usando protocolos e padrões de representação de dados proprietários em detrimento dos protocolos e padrões propostos pelas normas, além de geralmente possuírem código fechado.

No caso dos medidores de energia nacionais, o que ocorre é que os *softwares* usam as especificações da NBR 14522 para extrair os dados comuns, e contém funcionalidades extras para leitura dos dados adicionais não previstos na norma e implementados pelo dispositivo específico. Assim, por exemplo, o software Draco, produzido pela Nansen e que é próprio para o Spectrum SX, é capaz de trocar informações com o Saga 1000, mas somente aquelas definidas pela NBR 14522, e nada além daquilo especificado nessa norma.

Do ponto de vista dos clientes, nesse caso as EEE, as principais desvantagens do uso de sistemas proprietários são:

- Pouca liberdade de escolha: quando a empresa adquirir um modelo, ela deve seguir fiel a esse modelo, para poder usar os mesmos sistemas;
- Incompatibilidade entre os fabricantes: o sistema de um não se comunica com os dispositivos de outros;
- Código fechado dos sistemas: os fabricantes geralmente não abrem o código de seus sistemas ao cliente, o que permitiria, em muitos casos, a adaptação a outros dispositivos.

Do ponto de vista dos fabricantes, existem alguns motivos para continuar com essa política de desenvolvimento de sistemas proprietários, que só se comunicam com seus dispositivos, em detrimento de sistemas genéricos padronizados e com interface de serviços bem definida. Basicamente, o que ocorre é que permitir a interoperabilidade, ou seja,

permitir que seu sistema seja capaz de se comunicar com os dispositivos de outros fabricantes, permitiria que as EEE pudessem comprar aqueles dispositivos que estivessem com a melhor relação custo/benefício do mercado naquele momento. Ter o cliente “preso” aos seus produtos é muito mais vantajoso financeiramente.

3.5 Segurança

Um sistema computacional é dito seguro se ele atende a três requisitos básicos relacionados aos recursos que o compõem: confidencialidade, integridade e disponibilidade. A confidencialidade implica que a informação deva estar disponível somente para aqueles devidamente autorizados; a integridade, que a informação não deva ser destruída ou corrompida; e, a disponibilidade, que os serviços/recursos do sistema devam estar disponíveis sempre que forem necessários.

Alguns exemplos de violações de cada um desses requisitos no contexto de medidores de energia são:

- **Confidencialidade:** um intruso poderia obter acesso não autorizado a um medidor e fazer leituras de todas as informações do cliente para auferir vantagem estratégica para sua empresa ou negócio;
- **Integridade:** um intruso poderia ganhar acesso não autorizado ao sistema e mudar ajustes, de modo a obter benefícios financeiros para um determinado consumidor;
- **Disponibilidade:** O acesso ao medidor via interfaces de comunicação remota pode ser dificultado ou impedido por um invasor, simplesmente para causar desconforto e contratempo às empresas.

Uma política de segurança é um instrumento importante para proteger uma organização contra ameaças à segurança da informação que a ela pertence ou que está sob

sua responsabilidade. Uma ameaça à segurança é compreendida neste contexto como a violação de uma ou mais de suas três propriedades fundamentais.

Problemas relacionados à segurança podem surgir devido à arquitetura de um sistema computacional. Um computador ou dispositivo (um medidor, por exemplo) isolado, sem conexões físicas com outros dispositivos, tem chances menores de sofrer ataques do que outros que estejam conectados a uma rede, como a Internet, por exemplo. Isso porque em um computador isolado é necessária a presença física do invasor no local onde ele está localizado. Já no caso de um dispositivo ligado a outros dispositivos através de uma rede, e sem uma proteção adequada, um invasor que conheça os protocolos de comunicação e/ou as características dos dispositivos tem uma boa chance de conseguir atacá-los, roubando informação ou causando algum tipo de dano.

Com o intuito de oferecer maiores informações sobre os problemas relacionados à segurança, principalmente no contexto do ambiente computacional escolhido para o sistema que foi desenvolvido (Internet), o Apêndice B faz uma explanação sobre os diversos pontos vulneráveis da arquitetura e sobre vários mecanismos de segurança que podem ser implementados para minimizar os riscos de ataques. O apêndice também traz algumas soluções adotadas nesse contexto.

CAPÍTULO 4

INTEGRAÇÃO DE SISTEMAS BASEADA EM MODELOS

O capítulo 4 apresenta uma proposta de solução para a integração de sistemas heterogêneos no contexto de medição de energia. Ela é baseada na modelagem de um dispositivo genérico a partir de blocos básicos de construção. São expostas também as propostas para garantir segurança nos sistemas de medição de energia.

4.1 Introdução

O objetivo deste trabalho é desenvolver um sistema que permita integrar diversos modelos de medidores de energia no acervo das EEE. Uma estratégia para atingir este objetivo passa pela definição de um medidor genérico que possa representar, através de alguma especialização, os medidores já existentes. Deste modo, a elaboração de um modelo genérico de medidor tornou-se um elemento essencial para o desenvolvimento do projeto.

Um avanço fundamental na Engenharia de Software, que tem grande influência atualmente no projeto de sistemas, é o conceito de Orientação a Objetos (OO). Esta tecnologia envolve o uso de “objetos” e “classes” com o fim de oferecer uma melhor descrição e compreensão das entidades de um domínio de problemas particular. Em nosso caso específico, o domínio de problema é a descrição de um Medidor de Energia Elétrica. O uso desta abordagem permite a especificação de classes e objetos que definem, de forma abstrata, a semântica das funções de um medidor genérico.

Nesse sentido, o objetivo principal é definir blocos básicos de construção (componentes) reutilizáveis a serem empregados na especificação de modelos de medidores específicos de diferentes fabricantes. Um conceito fundamental da Orientação a Objetos é o de herança, que permite expressar semelhanças entre classes na estrutura do domínio do problema. Isto permite a definição de novas classes baseadas em classes já definidas. Com

a aplicação desse conceito, as funções e atributos comuns de um medidor genérico são definidos uma única vez, em uma classe abstrata. Desta forma, medidores específicos de cada fabricante podem ser obtidos como especializações ou implementações dessa classe. Instâncias correspondentes aos dispositivos físicos podem então ser representadas por objetos da classe associada (ver Figura 4.1).

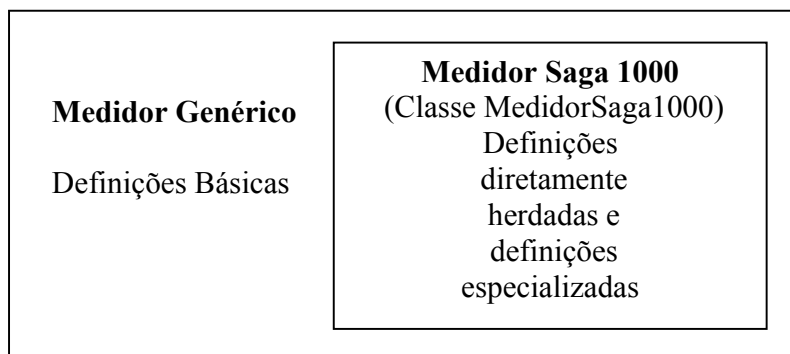


Figura 4.1 – Um dos benefícios da modelagem de dispositivos é facilitar o reuso de definições comuns.

Uma iniciativa nessa área, e ainda em estágio de desenvolvimento, é a especificação COSEM produzida pela Associação DLMS (ver Capítulo 2). O modelo que é proposto nesse trabalho usa como base as classes de interface definidas nessa especificação. No estágio atual dos esforços em direção à padronização do acesso a dispositivos de campo relacionados a empresas de energia elétrica, COSEM é a única que já define uma modelagem de medidores de energia elétrica, o que determinou nossa escolha. Assim, o modelo genérico foi baseado nessa especificação, procurando descrever através de suas classes as funções de medição definidas na Norma ABNT NBR 14522 [NBR 00]. Essa experiência nos permite, entre outras coisas, avaliar a capacidade de COSEM em representar adequadamente as funções de medição.

É importante ressaltar que não implementamos o padrão DLMS/COSEM, mas que usamos apenas suas classes de interface como base para gerar os nossos blocos básicos de construção. Essas classes são mapeadas em tipos de dados compostos, com base na estrutura de esquemas XSDs (Apêndice A, sub-item A.7.2). A composição dessas classes forma um esquema XSD, que representa o modelo lógico de um medidor de energia

genérico, e que pode ser especializado em XSDs de medidores específicos. Instâncias desses XSDs são os arquivos XML que representam um estado (um objeto) de um medidor específico. A tecnologia XML foi escolhida como forma de representação de dados devido as suas diversas vantagens mencionadas no Apêndice A.

Nos sub-itens seguintes, faremos uma exposição das linhas de modelagem seguidas nesse trabalho. Assim, estaremos apresentando somente algumas das classes utilizadas para definir um medidor, e não o projeto completo.

4.2 Modelagem das Classes de Interface COSEM em Esquemas XSDs

O primeiro passo na modelagem foi a definição das classes de interface COSEM como blocos básicos de construção nos esquemas XSD. COSEM define 24 classes de interface básicas [DLMS 02b]. Em nosso trabalho usamos apenas aquelas que seriam úteis para a modelagem dos medidores nacionais que seguem a norma NBR 14522 e são mais comuns no contexto das EEE. A notação utilizada para caracterizar uma classe de interface em COSEM define o nome da classe, seus atributos e seus métodos. Tomemos como exemplo a classe de interface *Extended Register* (ver Figura 4.2):

Extended Register		0..n	class_id = 4, Version = 0		
Attribute(s)		Data Type	Min.	Max.	Def.
1. logical_name	(static)	octet-string			
2. value	(dyn.)	instance specific			
3. scaler_unit	(static)	scal_unit_type			
4. status	(dyn.)	instance specific			
5. capture_time	(dyn.)	octet-string			

Figura 4.2 – Classe *Extended Register*.

Nesse exemplo, os diversos campos indicados estão abaixo descritos (na tabela 4.1 será mantida a nomenclatura em inglês, por compatibilidade com a documentação):

Class name	<i>Extended Register</i>	Descreve a classe
Cardinality	0..n	Especifica o número de instâncias dessa classe

		que pode existir em um dispositivo lógico.
<i>Class_id</i>	4	Identifica o código da classe (feito pela Associação DLMS)
<i>Version</i>	0	Código de versão da classe
<i>Attributes</i>	No exemplo, de <i>logical_name</i> a <i>capture_time</i> ; podem ser estáticos ou dinâmicos	Especifica os atributos que pertencem à classe; <i>logical_name</i> é sempre o primeiro atributo de uma classe e identifica a instância
<i>Data Type</i>	e.g., octet-string	Define o tipo de dados do atributo
<i>Min.</i>		Especifica se o atributo tem um valor mínimo
<i>Max.</i>		Especifica se o atributo tem um valor máximo
<i>Def.</i>		Especifica se o atributo tem um valor <i>default</i>

Tabela 4.1 – Especificação dos campos da classe de interface Extended Register.

Com o objetivo de facilitar a visualização da modelagem, as classes de interface são representadas por diagramas estruturais, construídos com o uso da ferramenta XMLSPY (Apêndice A, sub-item A.10). Por exemplo, a mesma classe *Extended Register* é vista na Figura 4.3 como o tipo complexo “*ExtendedRegister*”, com seus elementos constitutivos *class_id*, *version* e todos os seus atributos (*logical_name*, *value*, *scaler_unit*, *status*, e *capture_time*). Sendo essa notação recursiva, o atributo *scaler_unit* é representado pelo tipo complexo *scaler_unit_type*, que possui os elementos *scaler* e *unit*.

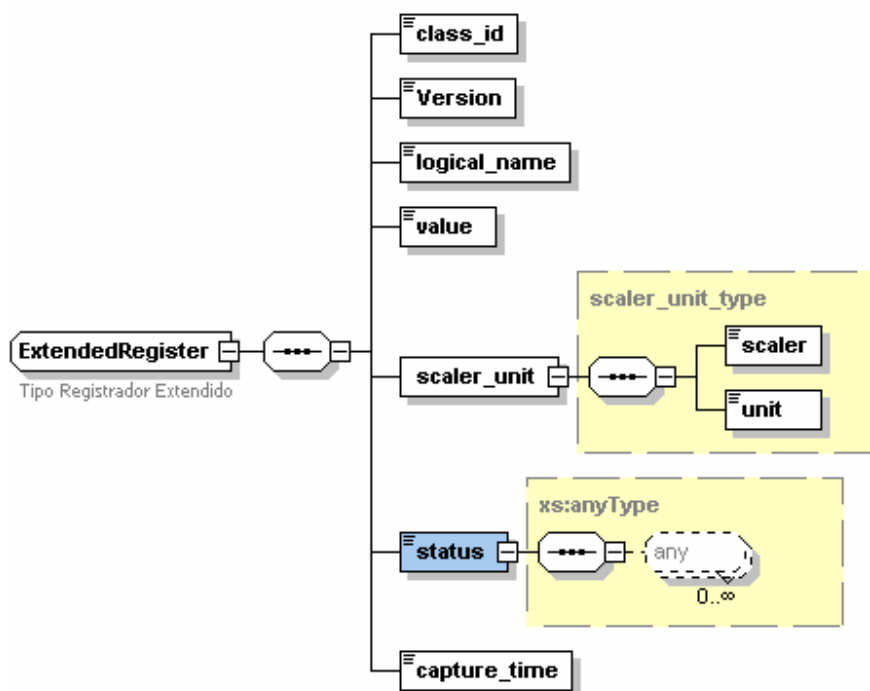


Figura 4.3 – Diagrama estrutural da classe de interface *Extended Register*.

O esquema XSD apresentado na Figura 4.3 pode ser visto em seu formato texto no Código 4.1. Como pode-se notar, este XSD mostra as restrições às quais um objeto do tipo *ExtendedRegister* e seus atributos devem obedecer. Assim, por exemplo, uma grandeza mapeada como uma instância dessa classe deve ter o atributo “*value*” (em negrito no código) do tipo inteiro longo (xs: *long*) e pode variar de 0 a 999999999.

```
<xs:complexType name="ExtendedRegister" class_id="4" version="0" >
  <xs:annotation>
    <xs:documentation>Tipo Registrador Extendido</xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="logical_name">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:whiteSpace value="preserve"/>
          <xs:minLength value="0"/>
          <xs:maxLength value="8"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:element>
    <xs:element name="value">
      <xs:simpleType>
        <xs:restriction base="xs:long">
          <xs:minInclusive value="0"/>
        </xs:restriction>
      </xs:simpleType>
    </xs:element>
    <xs:element name="scaler_unit">
      <xs:complexType base="scaler_unit_type">
        <xs:sequence>
          <xs:element name="scaler" type="xs:string"/>
          <xs:element name="unit" type="xs:string"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="status">
      <xs:complexType base="xs:anyType">
        <xs:sequence>
          <xs:element name="any" type="xs:anyType"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="capture_time" type="xs:long"/>
  </xs:sequence>
</xs:complexType>
```

```

        <xs:maxInclusive value="99999999"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:element>
  <xs:element name="scaler_unit" type="scaler_unit_type"/>
  <xs:element name="status" type="xs:anyType"/>
  <xs:element name="capture_time" type="xs:dateTime"/>
</xs:sequence>
</xs:complexType>

```

Código 4.1 – Código XSD da representação estrutural da Figura 4.3.

Assim, uma grandeza de um medidor que possui todos os atributos de um tipo *ExtendedRegister* é modelada como uma instância desse tipo complexo de dados. Com isso, ganha-se em clareza na modelagem, pois se modela apenas uma vez a classe e, para cada grandeza do tipo *ExtendedRegister*, basta associá-la a esse tipo (como veremos nos sub-itens posteriores).

4.3 Representação das Associações COSEM

Diversas outras classes COSEM foram mapeadas. Dentre elas, classes que representam as associações COSEM. Segundo a especificação COSEM para se acessar objetos no servidor (no caso, o medidor), uma associação de aplicação deve ser primeiro estabelecida. Esta associação caracteriza o contexto no qual as aplicações associadas irão se comunicar. Dois tipos de associação são definidos em COSEM, um para as chamadas associações de nome curto (*Short Name- SN*) e outro para as associações de nome lógico (*Logical Name - LN*). A diferença entre elas está: (i) na forma de endereçamento dos métodos e atributos dos objetos que elas referenciam, e (ii) nos tipos de restrições de acesso e segurança que elas oferecem. Mais detalhes sobre associações ver [DLMS 02b].

O modelo proposto também absorve da especificação COSEM o conceito de associação, dessa forma oferecemos um maior nível de segurança para os usuários do sistema. Assim, foram também mapeadas as classes de interface *Association SN* e *Association LN* como tipos complexos de dados *AssociationSN* e *AssociationLN* nos esquemas XSD.

4.3.1 Associação SN

Segundo a especificação COSEM, uma instância da classe de Associação SN (*Short Name*) mantém uma lista com objetos de um dispositivo lógico. Uma Associação SN é descrita como na tabela abaixo:

Association SN		0..n	class_id = 12, Version = 1		
Attribute(s)		Data Type	Min.	Max.	Def
1. logical_name	(static)	octet-string			
2. object_list	(static)	objlist_type			

Figura 4.4 – Descrição da classe Associação SN (ver 4.6).

A Figura 4.5 mostra o diagrama que representa o atributo *object_list*, que também é definido como um tipo complexo de dados. É nesse atributo que ficam armazenados uma referência a todos os objetos aos quais a associação faz referência, ou seja, um objeto que estiver na lista pode ser acessado de acordo com as restrições que estiverem estabelecidas pela associação.

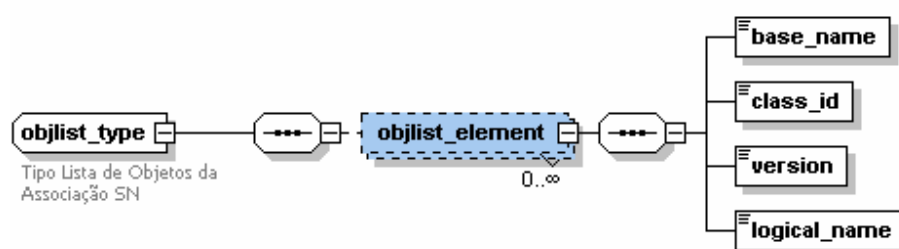


Figura 4.5 – Tipo lista de objetos da Associação SN.

Seguindo a mesma linha de modelagem mostrada para a classe de Interface *Extended Register*, na Figura 4.6, mostramos o diagrama estrutural XSD que representa uma Associação SN:

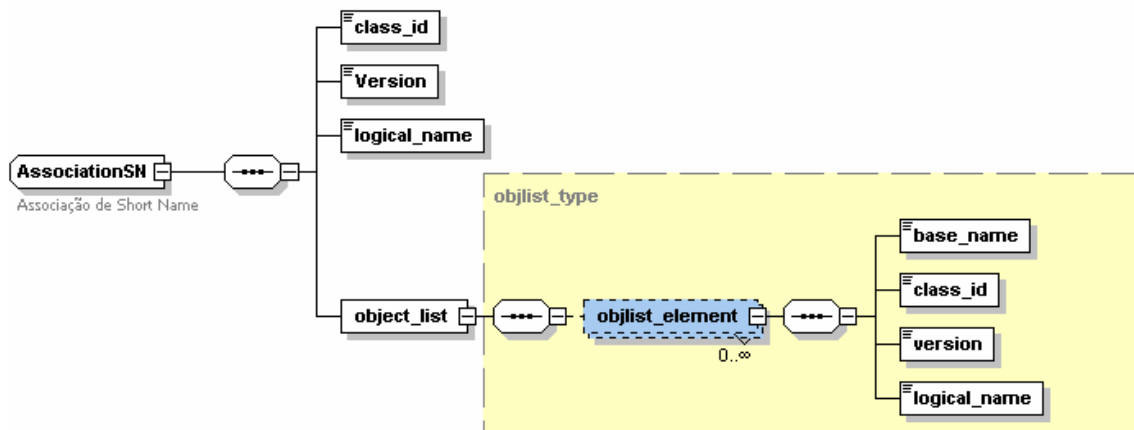


Figura 4.6 – Tipo Associação SN.

4.3.2 Associação LN

Segundo a especificação COSEM, uma instância da classe de Associação LN (*Logical Name*) mantém uma lista com todos os objetos de um dispositivo lógico, além dos direitos de acesso a cada um dos atributos e métodos desse objeto.

A representação utilizada é semelhante à mostrada anteriormente para a Associação SN. É interessante notar que o tipo lista de objetos (ver Figura 4.7), além de guardar os objetos, também guarda as restrições de acesso a cada método ou atributo, mantendo com isso um maior nível de segurança.

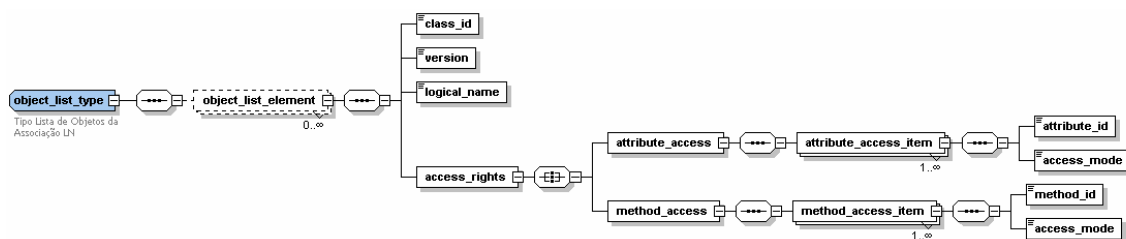


Figura 4.7 – Tipo lista de objetos da Associação LN.

Esse tipo de associação é mais complexo e exige do dispositivo que o utiliza um maior nível de capacidade de processamento. A Figura 4.8 apresenta o diagrama representativo para essa associação.

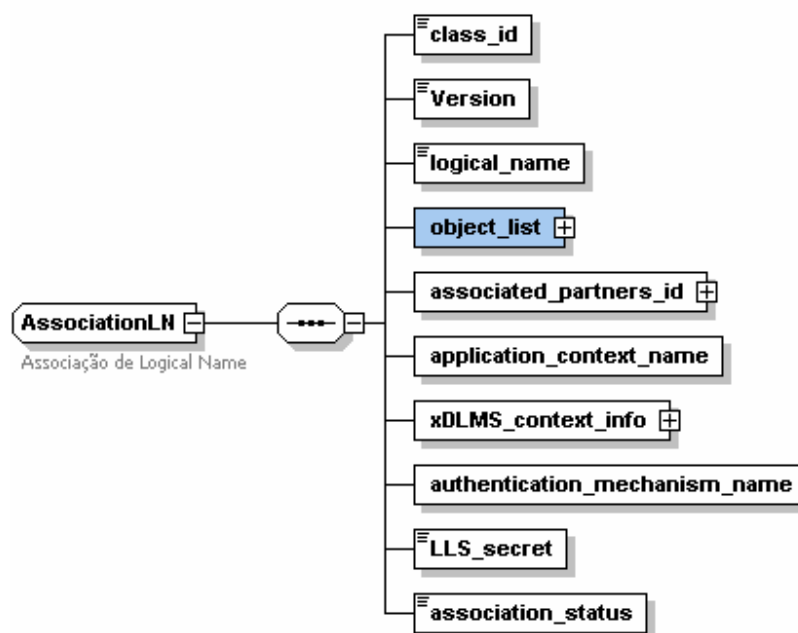


Figura 4.8 – Tipo Associação LN.

Aqui cabe mencionar que a associação LN apresenta alguns atributos que só fazem sentido no contexto de um *framework* COSEM, e que para o nosso caso são dispensáveis, porém foram mantidos para conformidade com a especificação.

Em subseções seguintes, no tópico que aborda as questões de segurança será demonstrado como funciona na prática um mapeamento dos objetos de um dispositivo lógico na associação.

4.4 Representação de um Medidor de Energia Genérico

Com base na norma nacional, e nos manuais dos medidores de interesse, identificamos as características gerais que um Medidor de Energia precisa possuir para atender às necessidades do mercado nacional. Visando sua validação, o modelo genérico proposto, que atua como um elemento unificador, foi utilizado para representar as definições de dados de três diferentes modelos de medidores: o Saga 1000 da ESB, o Spectrum SX da Nansen e o Elo2180 da Elo. A representação unificada resultou em um

medidor de energia genérico, que abrange o domínio do problema dos três modelos e a norma NBR 14522.

A estrutura do modelo é definida em sub-seções correspondentes às partes componentes de um medidor de energia genérico. Cada uma dessas partes é definida como um elemento composto de sub-elementos, esse refinamento prosseguindo até atingirmos um tipo básico como definido em COSEM. Um medidor, tipicamente, pode ser visto como indicado na Figura 4.9. A título de exemplo, ilustraremos a definição das grandezas de faturamento.

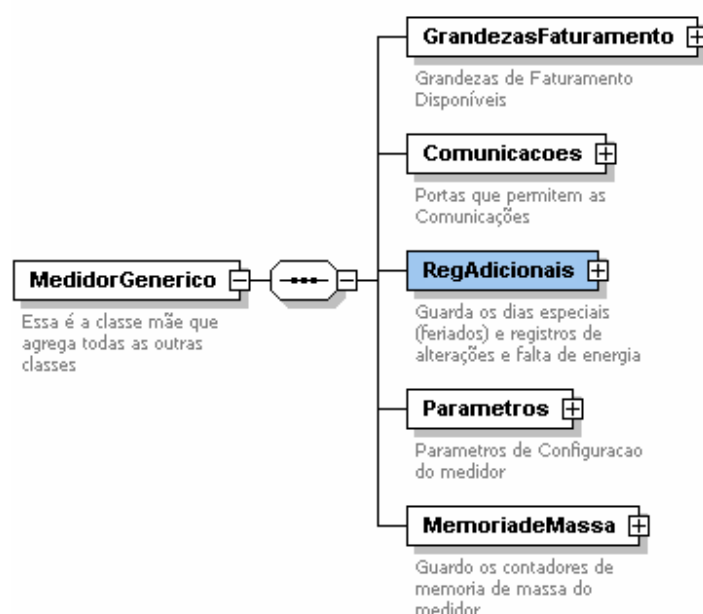


Figura 4.9 – Medidor de energia genérico.

4.4.1 Grandezas de Faturamento

O conjunto escolhido para compor cada subgrupo de grandezas de faturamento é amplo o bastante para possibilitar a representação dos medidores mais comuns e está de acordo com a norma NBR 14522 da ABNT. Referindo-se à Figura 4.10, IG representa as Informações Gerais de um medidor, CN1 as grandezas relativas ao Canal 1 (que tratam de energia ativa), CN3 as do Canal 3 (energia reativa capacitiva), etc. Da especificação COSEM, ASN e ALN representam, respectivamente, as Associações de *Short Name* e de

Logical Name, e LDN é o nome lógico do dispositivo. Podemos notar que no diagrama estão indicadas as duas possibilidades de associação, o que não seria o caso para a especialização de um medidor específico. Contudo, como se trata do modelo genérico, de onde os modelos específicos serão obtidos, as duas possibilidades estão presentes.

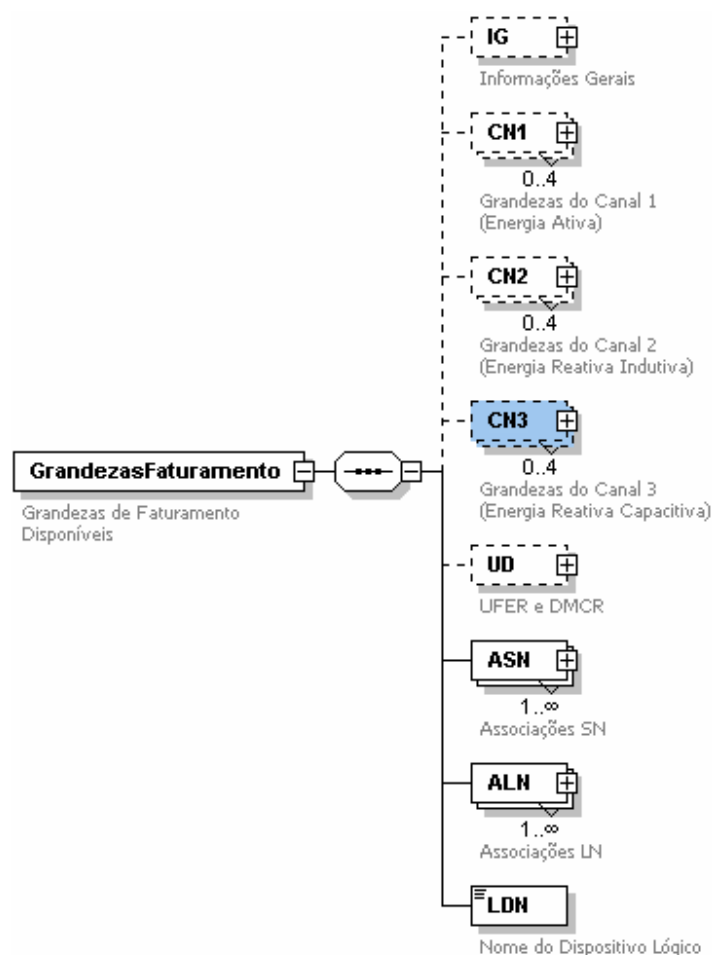


Figura 4.10 – Representação das grandezas de faturamento de um medidor.

Cada um dos elementos que modelam as grandezas de faturamento é composto por elementos filhos, que representam as grandezas específicas. Um exemplo, como mostrado na Figura 4.11, apresenta uma representação hierárquica dos elementos que compõem o Canal 3. Em um medidor de energia o Canal 3 contabiliza as informações necessárias ao faturamento relativas a energia reativa capacitiva. Essas informações ficam armazenadas nos registradores totalizadores geral, em horário de ponta, reservado e fora de ponta.

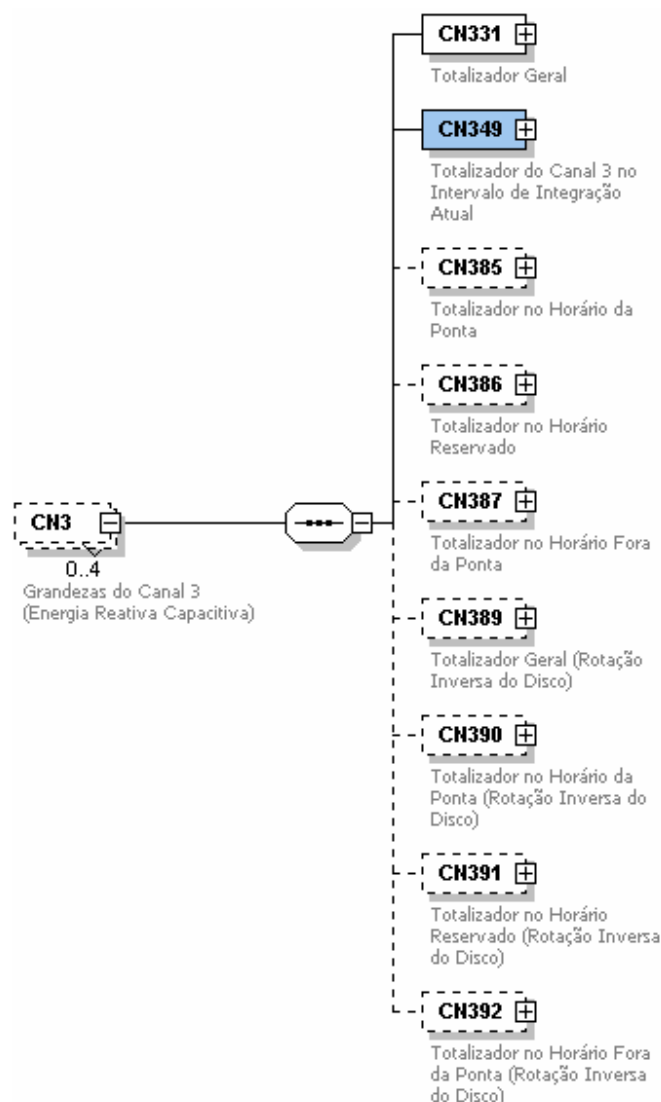


Figura 4.11 – Representação das grandezas de faturamento do Canal 3.

Adicionalmente, cada uma das grandezas terá um tipo de acordo com suas necessidades e seguindo a recomendação da especificação COSEM. Por exemplo, a grandeza CN349 (Totalizador do Canal 3 no Intervalo de Integração Atual), em destaque na Figura 4.11, é do tipo Extended Register. A Figura 4.12 mostra a associação da grandeza ao seu tipo. E assim para cada uma das outras grandezas associa-se uma das classes de interface COSEM que melhor mapeie seus atributos. Segundo a nomenclatura que adotamos, “CN349” diz respeito ao Canal (CN3) e ao código de referência (49) da grandeza nas normas ABNT.

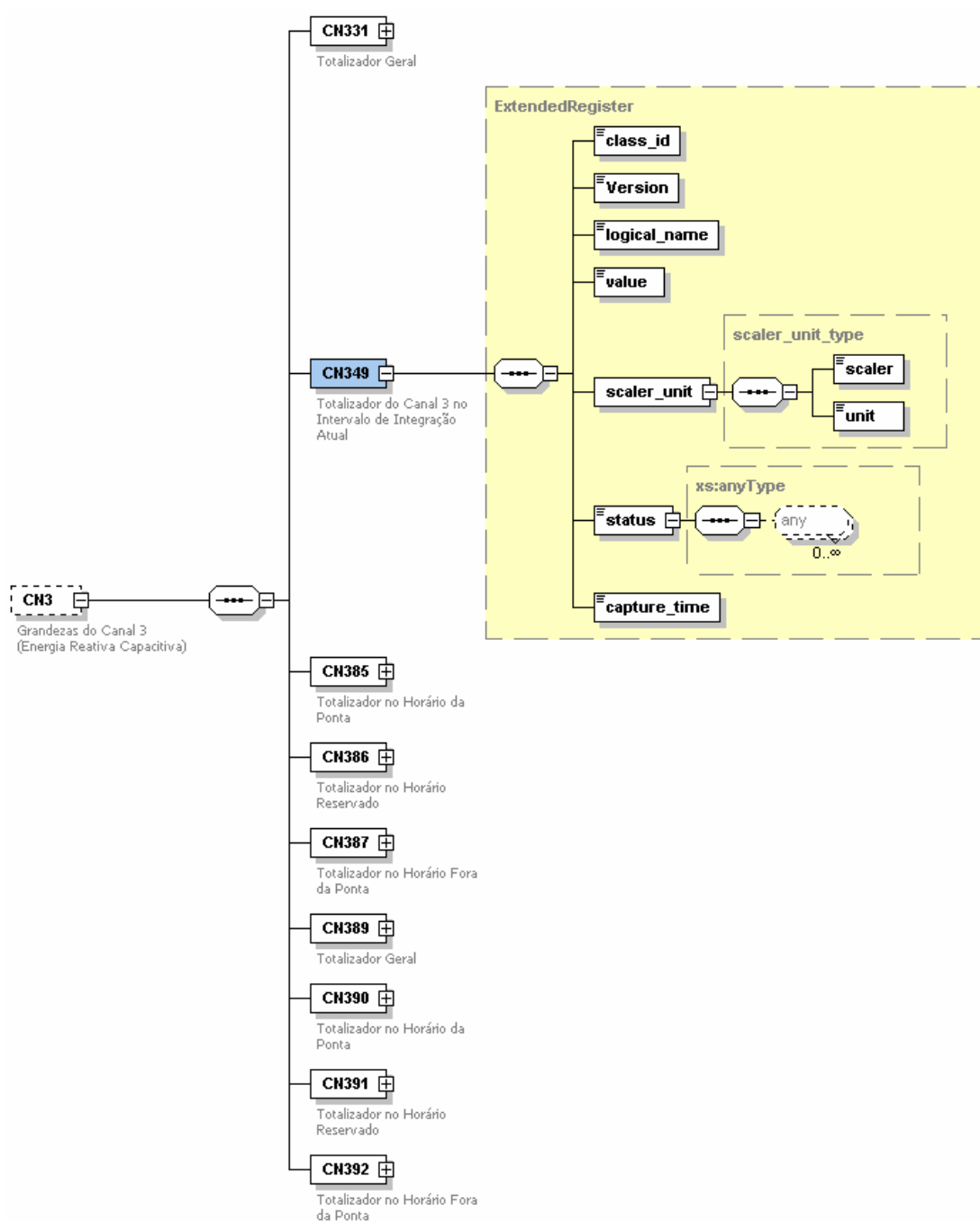


Figura 4.12 – Declaração da grandeza CN349 como do tipo *ExtendedRegister*.

Finalmente, se a associação utilizada no dispositivo lógico é do tipo *AssociationSN*, haverá uma instância dessa classe de interface para cada associação que o dispositivo possa manter (Figura 4.13).

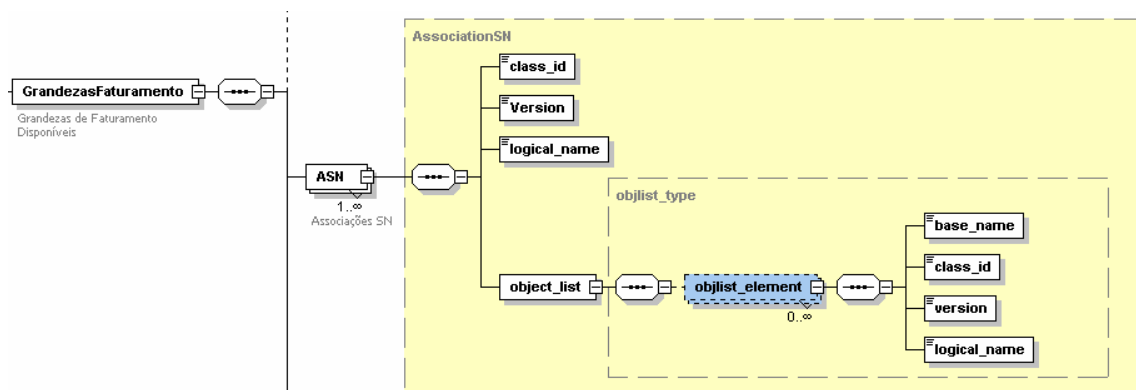


Figura 4.13 – Existe uma instância dessa classe para cada associação que um dispositivo pode suportar.

4.4.2 Modelagem Adicional

Ainda compondo um medidor genérico, quatro outros blocos de informação são especificados. O primeiro é relacionado às interfaces de comunicação disponíveis, ou seja, um bloco identificando as características básicas das portas de comunicação. O segundo é relacionado aos registros adicionais, como, por exemplo, os dias especiais (feriados), que são armazenados para o caso de tarifação diferenciada. O terceiro bloco armazena parâmetros de configuração do medidor. E o quarto bloco armazena dados relativos a memória de massa. As linhas gerais da modelagem seguem o que já foi apresentado no item de grandezas de faturamento. As Figuras 4.14 e 4.15 apresentam um exemplo dessa modelagem.

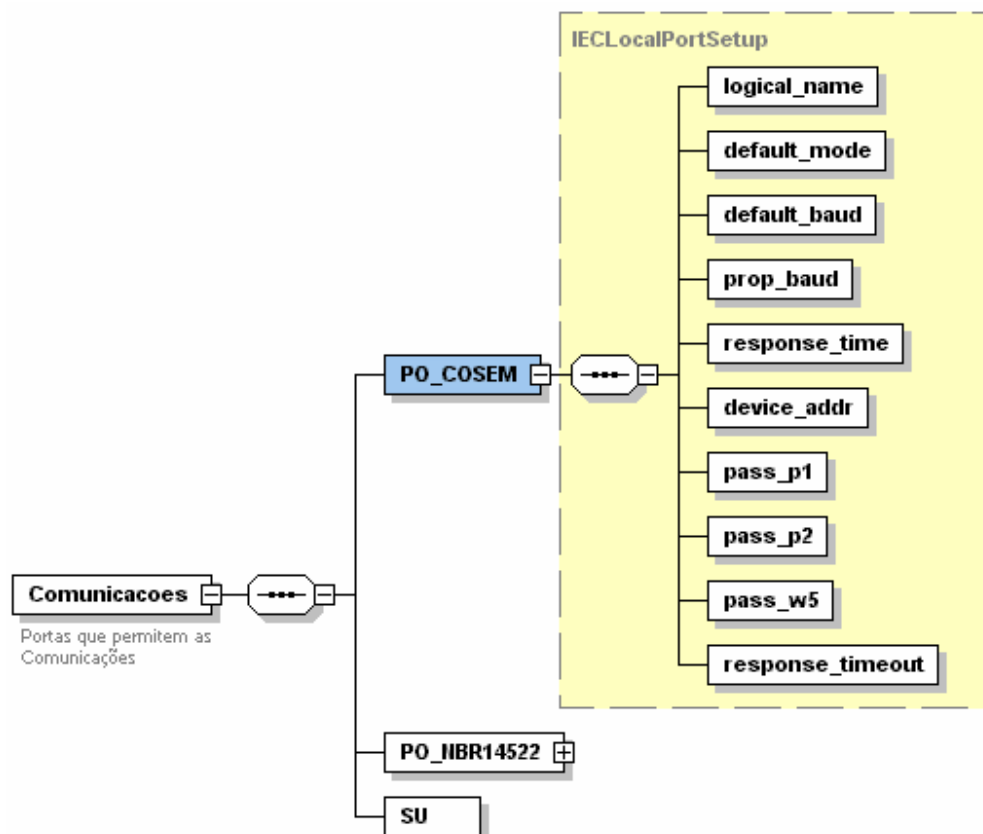


Figura 4.14 – Diagrama que representa as portas de comunicação do medidor.

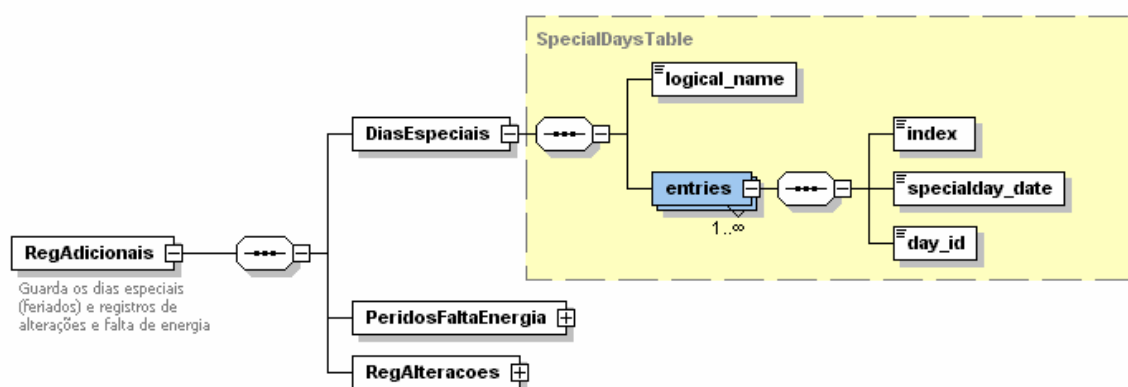


Figura 4.15 – Diagrama que representa o armazenamento dos dias especiais em registros adicionais.

4.5 Considerações Finais Sobre a Modelagem

Foram apresentadas as linhas básicas da modelagem que foi empregada na definição de um medidor genérico. Um modelo genérico, em princípio, abrange toda a possível variedade de medidores, com os seus dados, grandezas e parâmetros. Um ponto importante foi a utilização da tecnologia XML para a modelagem, o que permite a fácil adaptação da representação genérica para qualquer medidor específico. Então, com esse modelo baseado em XSD, descrevemos as funcionalidades e também especificamos os formatos das grandezas e parâmetros dos medidores. A partir daí, para cada medidor específico, pode ser gerado um XSD que especializa o XSD do medidor genérico, e que inclui apenas as peculiaridades do medidor específico.

Como exemplo, um XSD que modela o medidor Saga1000, possui no canal 3 apenas as grandezas que existem no dispositivo real, como mostra a Figura 4.16:

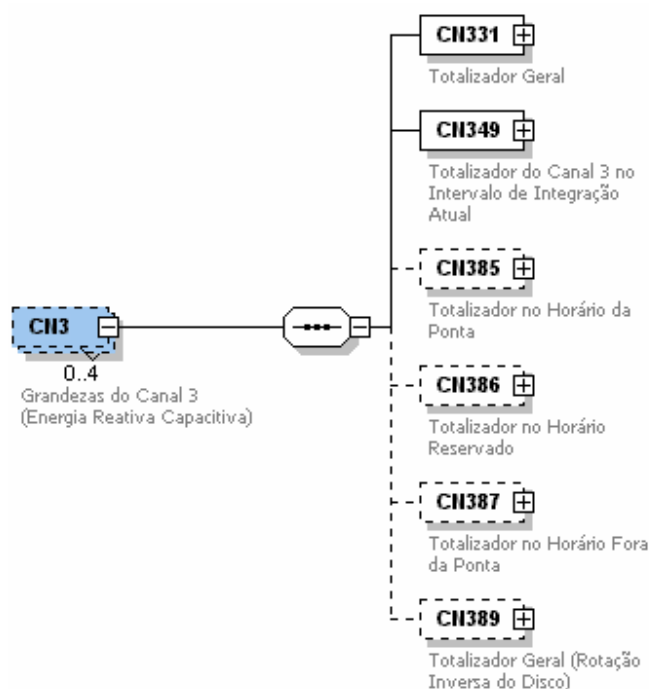


Figura 4.16 – Representação do Canal 3 no medidor SAGA1000.

O XSD de um dispositivo específico define então o formato dos documentos XML que representam objetos (instâncias do XSD) que contém os dados do medidor em um determinado instante. Ou seja, um sistema que deseja realizar uma leitura dos dados de um medidor, solicita e espera receber um documento XML em conformidade com esse XSD, para que seja capaz de processar os dados corretamente. Em adição, considerando o fato de que uma empresa possui diversos medidores de um determinado modelo, o XSD correspondente serve para representar toda a gama de medidores desse modelo.

Um ponto importante a ser destacado é que na modelagem, e como veremos, também na implementação do sistema, houve uma preocupação na adoção de padrões e em soluções de *software* livres e gratuitos, e ainda a independência de plataforma de execução. Como vimos o COSEM foi usado para modelagem, sem, no entanto, desrespeitar a especificação das normas ABNT. E utilizamos as tecnologias XML como forma de representação e modelagem dos dados.

4.6 Propostas de Segurança

A questão da segurança é um ponto de extrema importância para se manter a integridade dos dados em um sistema computacional, principalmente quando o acesso ocorre via *web*, e havendo ainda múltiplos agentes envolvidos. No caso dos medidores de energia e sistemas de medição, os agentes são as empresas de energia elétrica, os consumidores e as entidades reguladoras.

Para garantir a integridade dos dados, propomos uma política de segurança em diversos níveis. Um primeiro nível é baseado na própria arquitetura (detalhada no capítulo 5) da Internet que adotamos, nos seus protocolos e mecanismos de autenticação padronizados. Um outro nível de segurança é baseado na geração de *logs* em diversos pontos do sistema. Um terceiro nível baseado nas associações estabelecidas com os dispositivos lógicos. Adicionalmente, foram também mantidas as tradicionais características de segurança já implementadas na área de medição, como o registro das alterações realizados no medidor (subseção 4.6.2).

No que diz respeito ao primeiro nível de segurança foram adotadas soluções disponíveis livremente, gratuitas e comprovadamente eficazes para troca de informações. Essas tecnologias são a SSL (*Secure Socket Layer*), o conceito de *roles e realms*, e a autenticação através de certificados digitais [ASF 03]. Essas tecnologias são melhores detalhadas no Apêndice B.

No nível de geração de *logs*, são armazenadas informações de acesso ao sistema e ao medidor feitas pelos usuários que interagem com o sistema. Por exemplo, se um usuário muda algum parâmetro no medidor, serão armazenados, a identificação do usuário, o parâmetro que foi alterado, a data, a hora, e o número IP da máquina de onde a alteração foi feita. A geração de *logs* é importante e pode ser usada para a realização de auditorias no caso de suspeita de fraudes. Mais detalhes sobre os *logs* serão abordados no capítulo 5.

O outro nível está associado diretamente ao dispositivo lógico e refere-se ao conceito de associações. Quando um dispositivo fornece um dado em XML ao sistema, ele fornece também as restrições de acesso a esse dado de acordo com o usuário que o está requisitando. No sub-item seguinte detalharemos melhor esse conceito.

4.6.1 Associações e Restrições de Acesso a Dados

As associações constituem um dos mecanismos usados para implementação dos requisitos de segurança no padrão COSEM. Através das associações é possível estabelecer um nível de segurança relativo às restrições de acesso aos dados no medidor. Essa segurança é oferecida em dois níveis:

- No nível de usuários - onde para cada classe de usuários um tipo diferenciado de acesso é fornecido;
- No nível de objetos e de seus atributos e métodos – onde a associação fornece o tipo de acesso a uma grandeza específica mapeada em um objeto.

Aqui detalharemos primeiramente a Associação LN. O XSD que define o formato dos dados XML dessa associação foi visto na Figura 4.8. Em seguida detalharemos a Associação SN.

Para entender o mapeamento, tomemos como exemplo um objeto da classe *Register* da especificação COSEM (a classe *Register* também foi usada na modelagem). A representação em XML de um objeto registrador que guarda dados de energia ativa total (no modelo CN103, do Canal 1, do conjunto de Grandezas de Faturamento), é fornecida no Código 4.2.

```
<CN103>
  <class_id>3</class_id>
  <version>0</version>
  <logical_name>CN103</logical_name>
  <value>123456</value>
  <scaler_unit>
    <scaler>0</scaler>
    <unit>30</unit>
  </scaler_unit>
</CN103>
```

Código 4.2 – Objeto CN103 em uma instância da classe *Register* em XML.

O trecho de código XML que representa uma instância de uma Associação LN, e que está em conformidade com o XSD da Figura 4.8, é mostrado no Código 4.3. A associação indica que para um usuário Consumidor, o sistema permitirá somente o acesso de leitura aos atributos 3, 4 e 5, e não permitirá o acesso aos atributos 1 e 2 do objeto CN103. Podemos observar (no trecho em negrito no código) que existem cinco *tags* do tipo *<attribute_access_item>* que mapeiam os direitos de acesso de cada um dos cinco atributos da classe *Register* (*class_id*, *version*, *logical_name*, *value*, *scaler_unit*). Fazendo então essa correspondência de valores, um usuário Consumidor tem acesso de leitura aos atributos *logical_name*, *value* e *scaler_unit*, e não tem acesso aos atributos *class_id* e *version*. Assim, para cada um das grandezas do medidor um *object_list_element* é criado e as restrições de acesso são definidas para cada um dos usuários.


```

<ALN>
  <class_id>15</class_id>
  <Version>0</Version>
  <logical_name>Consumidor</logical_name>
  <object_list>
    <object_list_element>
      <class_id>3</class_id>
      <version>0</version>
      <logical_name>CN103</logical_name>
      <access_rights>
        <attribute_access>
          <attribute_access_item>
            <attribute_id>1</attribute_id>
            <access_mode>no_access</access_mode>
          </attribute_access_item>
          <attribute_access_item>
            <attribute_id>2</attribute_id>
            <access_mode>no_access</access_mode>
          </attribute_access_item>
          <attribute_access_item>
            <attribute_id>3</attribute_id>
            <access_mode>read_only</access_mode>
          </attribute_access_item>
          <attribute_access_item>
            <attribute_id>4</attribute_id>
            <access_mode>read_only</access_mode>
          </attribute_access_item>
          <attribute_access_item>
            <attribute_id>5</attribute_id>
            <access_mode>read_only</access_mode>
          </attribute_access_item>
        </attribute_access>
        <method_access>
          <method_access_item>
            <method_id>1</method_id>
            <access_mode>1</access_mode>
          </method_access_item>
        </method_access>
      </access_rights>
    </object_list_element>
    <object_list_element>
      <class_id>3</class_id>
      <version>0</version>
      <logical_name>CN104</logical_name>
      <access_rights>
        <attribute_access>
          <attribute_access_item>
            <attribute_id>1</attribute_id>
            <access_mode>no_access</access_mode>
          </attribute_access_item>
          <attribute_access_item>
            <attribute_id>2</attribute_id>
            <access_mode>no_access</access_mode>
          </attribute_access_item>
          <attribute_access_item>
            <attribute_id>3</attribute_id>
            <access_mode>no_access</access_mode>
          </attribute_access_item>
        </attribute_access>
      </access_rights>
    </object_list_element>
  </object_list>

```

```

        </attribute_access_item>
        <attribute_access_item>
            <attribute_id>4</attribute_id>
            <access_mode>no_access</access_mode>
        </attribute_access_item>
        <attribute_access_item>
            <attribute_id>5</attribute_id>
            <access_mode>no_access</access_mode>
        </attribute_access_item>
    </attribute_access>
    <method_access>
        <method_access_item>
            <method_id>1</method_id>
            <access_mode>1</access_mode>
        </method_access_item>
    </method_access>
</access_rights>
</object_list_element>
</object_list>

<associated_partners_id>
    <client_SAP>xxxxxxx</client_SAP>
    <server_SAP>xxxxxxx</server_SAP>
</associated_partners_id>
<application_context_name>xxxxxxx</application_context_name>
<xDLMS_context_info>
    <conformance>xxxxxxx</conformance>
    <max_receive_pdu_size>xxxxxxx</max_receive_pdu_size>
    <max_send_pdu_size>xxxxxxx</max_send_pdu_size>
    <dlms_version_number>xxxxxxx</dlms_version_number>
    <quality_of_service>xxxxxxx</quality_of_service>
    <cyphering_info>xxxxxxx</cyphering_info>
</xDLMS_context_info>
<authentication_mechanism_name>LLS</authentication_mechanism_name>
<LLS_secret>54321</LLS_secret>
<association_status>xxxxxxx</association_status>

</ALN>

```

Código 4.3 – Associação LN em XML.

A Associação SN (*Short Name*) oferece restrições de acesso mais limitadas que a associação LN, não restringindo o acesso em nível de atributos como a LN, e sim no nível dos objetos. O trecho XML representado no código 4.4 mostra uma instância dessa associação. O conceito é bastante simples: se existe o objeto na lista, o usuário (nesse caso, o Consumidor) tem acesso total a todos os seus atributos.

```

<ASN>
    <class_id>12</class_id>
    <Version>1</Version>
    <logical_name>Consumidor</logical_name>

```

```

<object_list>
  <objlist_element>
    <base_name>10</base_name>
    <class_id>3</class_id>
    <version>0</version>
    <logical_name>CN103</logical_name>
  </objlist_element>
  <objlist_element>
    <base_name>20</base_name>
    <class_id>3</class_id>
    <version>0</version>
    <logical_name>CN104</logical_name>
  </objlist_element>
</object_list>
</ASN>

```

Código 4.4 – Associação SN em XML.

Finalmente, um aspecto interessante sobre as questões de segurança é que, quanto maior o nível de segurança que se deseja oferecer a um sistema, mais oneroso se torna o seu desenvolvimento e processamento. Aqui não é diferente, como visto nas associações SN e LN. A manutenção das estruturas dessas associações torna os arquivos, que guardam os documentos XML, relativamente grandes.

4.6.2 Características de Segurança Herdadas dos Medidores

As características de segurança implementadas no medidor são também mapeadas e trazidas para o modelo. Por exemplo, a grandeza de código 23 que guarda o Número de Reposições de Demanda (faturas) realizadas no medidor é representada por um objeto do tipo *ExtendedRegister*. As alterações (parametrização do medidor) também são registradas em um elemento <RegAlteracoes> do grupo de registros adicionais. É possível verificar, então, se houve violação da segurança por uma leitura ou parametrização não autorizada, através de uma auditoria onde os dados da leitura anterior registrada têm que ser concordantes com os dados da leitura atual.

O trecho XML do código 4.5 representa então a grandeza IG23, e o valor do número de reposições de demanda pode ser obtido no atributo <value>. Outro atributo relevante para essa grandeza é o atributo *capture_time*, que guarda o momento em que a última operação de reposição de demanda foi realizada.

```

<IG23>
  <class_id>4</class_id>
  <Version>0</Version>
  <logical_name>IG23</logical_name>
  <value>10</value>
  <scaler_unit>
    <scaler>0</scaler>
    <unit>0</unit>
  </scaler_unit>
  <status>0</status>
  <capture_time>2001-12-17T09:30:47</capture_time>
</IG23>

```

Código 4.5 – Grandeza 23 como do tipo *ExtendedRegister*.

Finalmente, o trecho XML do Código 4.5 traz o registro das alterações feitas no medidor. Em negrito no código, “67” representa o código da Alteração, “123456” o número de série da leitora de onde foi feita a alteração, “2004-03-08T14:04:20” a data e a hora que a alteração foi realizada, “admin1” é o usuário que realizou a alteração, e “200.20.15.240” o IP da máquina de onde a operação foi realizada.

```

<RegAlteracoes>
  <Alteracao index="1">
    <codigo>67</codigo>
    <num_serie_leitor>123456</num_serie_leitor>
    <capture_time>2004-03-08T14:04:20</capture_time>
    <usuario>admin1</usuario>
    <ip>200.20.15.240</ip>
  </Alteracao>
  <Alteracao index="2">
    <codigo>29</codigo>
    <num_serie_leitor>123456</num_serie_leitor>
    <capture_time>2004-03-09T13:14:19</capture_time>
    <usuario>admin1</usuario>
    <ip>200.20.15.240</ip>
  </Alteracao>
</RegAlteracoes>

```

Código 4.6 – Registro das alterações.

CAPÍTULO 5

O SISTEMA DE INTEGRAÇÃO

Este capítulo apresenta uma aplicação da abordagem de integração de sistemas baseada em modelos discutida no Capítulo 4. Toda a aplicação foi desenvolvida usando protocolos não proprietários e tecnologias abertas.

5.1 Introdução

Com base no que foi discutido anteriormente no Capítulo 3, é constatável a necessidade das EEE de um sistema não proprietário padronizado de medição de energia que seja capaz de interagir com o parque de medidores atuais e que seja extensível aos futuros medidores. Para a validação e prova dos conceitos estudados e propostos, foi desenvolvido o sistema PLPM (Padronizado de Leitura e Parametrização de Medidores). Centrado na abordagem de integração de sistemas baseada em modelos, o sistema PLPM fornece uma interface comum padronizada de leitura e parametrização de dados nos diferentes modelos de medidor.

Para a construção da aplicação, assumimos um cenário distribuído, baseado em ambiente WWW multiplataforma, com interface acessível via navegadores comuns. Esta opção permite que diversas arquiteturas de *hardware* e *software* sejam suportadas, evitando o comprometimento com sistemas proprietários. Além de propiciar independência de plataformas específicas, o ambiente escolhido facilita a disponibilização do sistema para múltiplos usuários fisicamente dispersos. Nas seções que seguem, apresentamos e detalhamos a arquitetura da aplicação desenvolvida. O objetivo é explicar algumas decisões tomadas no projeto, e também mostrar como as diversas tecnologias empregadas foram integradas.

5.2 O Sistema Padronizado de Leitura e Parametrização de Medidores

A solução adotada para o desenvolvimento do Sistema Padronizado de Leitura e Parametrização de Medidores de Energia (PLPM) foi a integração dos dados através de um modelo comum de informação (como visto no Capítulo 4) que representa o domínio do problema, e que pode ser particularizado para representar dispositivos similares de diferentes fabricantes.

O Sistema desenvolvido é capaz de manipular os dados de medidores, e de entender o modelo, suas restrições e refinamentos de acordo com cada um dos XSDs específicos que representam cada medidor particular, suas restrições de acesso de acordo com cada usuário e cada grandeza e/ou parâmetro que se deseja ler/alterar.

A utilização de um cenário distribuído baseado na *web* para a construção da aplicação tornou-se conveniente por esta apresentar um ambiente suportado por diversas arquiteturas de *hardware* e *software*. A arquitetura utilizada também permite que o sistema seja acessado e operado remotamente por múltiplos usuários. Isso traz flexibilidade e transparência, na medida em que permite a aquisição dos dados, por todos os agentes envolvidos, gerador, vendedor, comprador, auditor, etc. A Figura 5.1 mostra a arquitetura da aplicação:

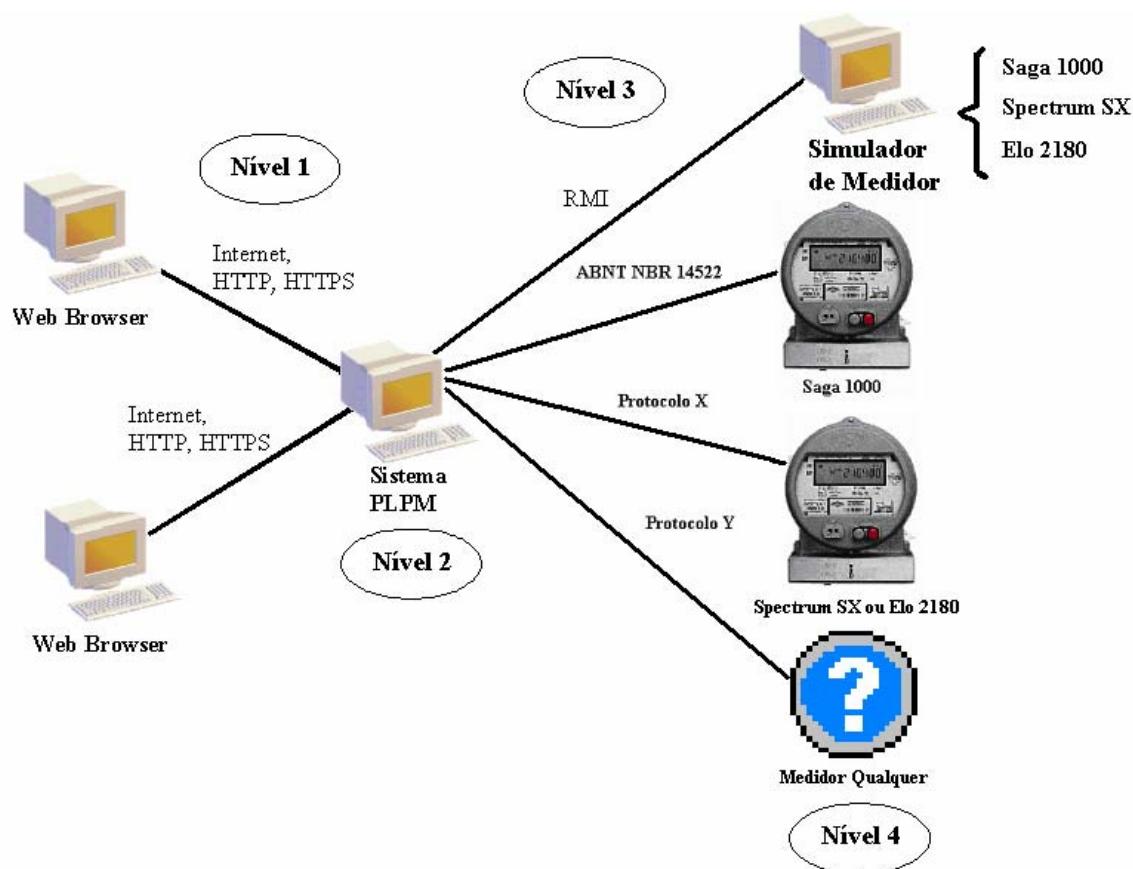


Figura 5.1 – Arquitetura geral do sistema.

As Interfaces de Usuários (IUs) do protótipo são descritas por páginas HTML, contendo *applets* para a exibição dos dados das leituras realizadas no medidor e para validação de parâmetros a serem carregados no medidor. As IUs são visualizadas em navegadores *web* e trocam dados codificados em HTML e XML com o Sistema através de protocolos padrões. O Sistema PLPM atua como um servidor *web*, com sub-funções específicas para ler, processar e armazenar os dados, e repassar parâmetros ao Medidor de Energia.

Em uma primeira fase do projeto, para possibilitar a realização de testes, foi necessária a construção de um simulador de medidor. Basicamente, o que esse simulador faz é retornar ao sistema PLPM, quando solicitado, um arquivo de leitura em formato XML. Esse arquivo é obtido pelo mapeamento de arquivos K7 (arquivos de leitura gerados por medidores de energia reais) no formato XML. O simulador é capaz de simular qualquer

modelo que possa ser especializado a partir do modelo genérico. Ele respeita as restrições dos modelos específicos representados pelos seus respectivos XSDs.

O protocolo de comunicação entre o sistema PLPM e o simulador de medidor é o RMI (*Remote Method Invocation*). A escolha desse protocolo deve-se ao fato de utilizarmos a linguagem Java no desenvolvimento. Qualquer outro protocolo de comunicação poderia ser usado, desde que este seja capaz de transferir arquivos no formato XML.

Na segunda fase, incorporamos um medidor real ao sistema. O modelo escolhido foi o medidor Saga 1000 da ESB. Como vimos no capítulo 3, um grande problema na área é que os medidores usualmente utilizam protocolos proprietários para a troca de dados, ou ainda algum outro dentre diversos padrões, como o Modbus. Ou seja, não há uma padronização entre os fabricantes além daquela indicada pela norma NBR 14522. Esse último protocolo, embora insatisfatório para as demandas atuais, é o único em comum para aquisição de dados de medidores nacionais. Dessa forma, em nosso protótipo, esse foi o protocolo escolhido para comunicação entre o PLPM e o medidor.

É importante notar que o sistema PLPM é projetado para trabalhar com um “medidor do futuro”. Para os nossos propósitos, caracterizamos como um medidor do futuro, aquele que, quando solicitado pelo sistema, devolve uma leitura de dados e/ou parâmetros no formato de um arquivo XML, em conformidade com o modelo XSD que o descreve. Para adaptar o sistema PLPM aos medidores atuais é necessário o uso de linguagens de transformação, como XSLT, e *parsers* como JAXP. Essas linguagens (detalhes no Apêndice A) permitem transformar os dados do formato que os medidores fornecem para o XML e vice-versa.

Para ler dados, ou alterar os parâmetros de configuração de um medidor, o usuário precisa acessar a IU através de um navegador. No primeiro instante, é exigida a autenticação do usuário através de um formulário HTML. Se a autenticação for realizada com sucesso, a IU apresenta a interface de leitura e manipulação de dados. O usuário

precisa então se identificar para estabelecer seu nível de permissão de acesso aos dados e indicar de qual medidor deseja ter informações. A interface faz uma consulta ao sistema para identificar o usuário e o modelo do medidor e sua configuração padrão, de forma transparente para o usuário.

No contexto de leitura, o PLPM solicita os dados ao medidor. Se for o simulador, ele fornece um documento XML gerado automaticamente através do mapeamento de arquivos K7. A aplicação recebe os dados que são interpretados e exibidos para leitura nas IUs. No caso de ser o Medidor real o processo é semelhante, porém o mapeamento dos Arquivos K7, que são obtidos através de leituras via protocolo ABNT, nos documentos XML são feitas pelo próprio PLPM.

No contexto de parametrização, a aplicação permite que parâmetros de configuração de medidores de energia sejam inseridos pelos usuários e transformados para as mesmas representações XML, o que permite que sejam manipulados e armazenados de maneira padronizada. Quando for o medidor real, existe ainda o mapeamento dos parâmetros do documento XML para estruturas (definidas no protocolo ABNT) de comandos de parametrização que são enviados ao medidor.

O Sistema então atua como o servidor da aplicação, ficando responsável por receber as requisições de consulta e atualização, e repassá-las aos medidores. Ele mantém um banco de dados que armazena leituras e configurações dos medidores e possui módulos para tratamento dos dados manipulados.

Ainda na Figura 5.1, podemos observar que os medidores reais Spectrum SX e Elo 2180 estão incorporados na arquitetura, e a comunicação entre eles e o PLPM é feita através de um Protocolo X qualquer. Apesar de não termos trabalhado diretamente com medidores reais desses modelos, a sua incorporação é bastante simplificada, pois é necessária apenas a conversão dos dados obtidos através do Protocolo X para as representações XML e vice-versa. Um outro medidor qualquer também pode ser inserido, já que o protótipo utiliza um modelo padronizado como mencionado anteriormente, e,

portanto, a inclusão de novos tipos de medidores que obedecem a esse modelo torna-se bastante simplificada.

No que se refere aos requisitos de segurança incorporadas, na Figura 5.1 podemos observar algumas indicações de níveis em que dividimos o sistema:

- **Nível 1:** Características de segurança convencionais de transporte de dados criptografados adotadas na internet e disponíveis livremente, HTTPS e SSL. Processo de autenticação de usuários através de *login* e *password* com a tecnologia de *realms* e *roles* e certificados digitais;
- **Nível 2:** Geração de logs, com registro de todas as interações entre usuários e sistema;
- **Nível 3:** Associações – baseadas nos conceitos da especificação COSEM;
- **Nível 4:** Características tradicionalmente adotadas pelos medidores como o registro das interações entre um usuário e o medidor.

Deve ser acrescentado que no desenvolvimento da aplicação foi mantido um esquema de *log* no simulador de medidor, que é uma maneira adicional de implementar características de segurança.

Uma abordagem maior sobre os requisitos de segurança incorporados foi feita no Capítulo 4, e ainda dedicamos o Apêndice B inteiramente a propósito semelhante.

5.3 Arquitetura Detalhada do Sistema

Conhecendo melhor a arquitetura e o funcionamento básico da aplicação, podemos agora detalhar as tecnologias empregadas em seus diversos “nós”.

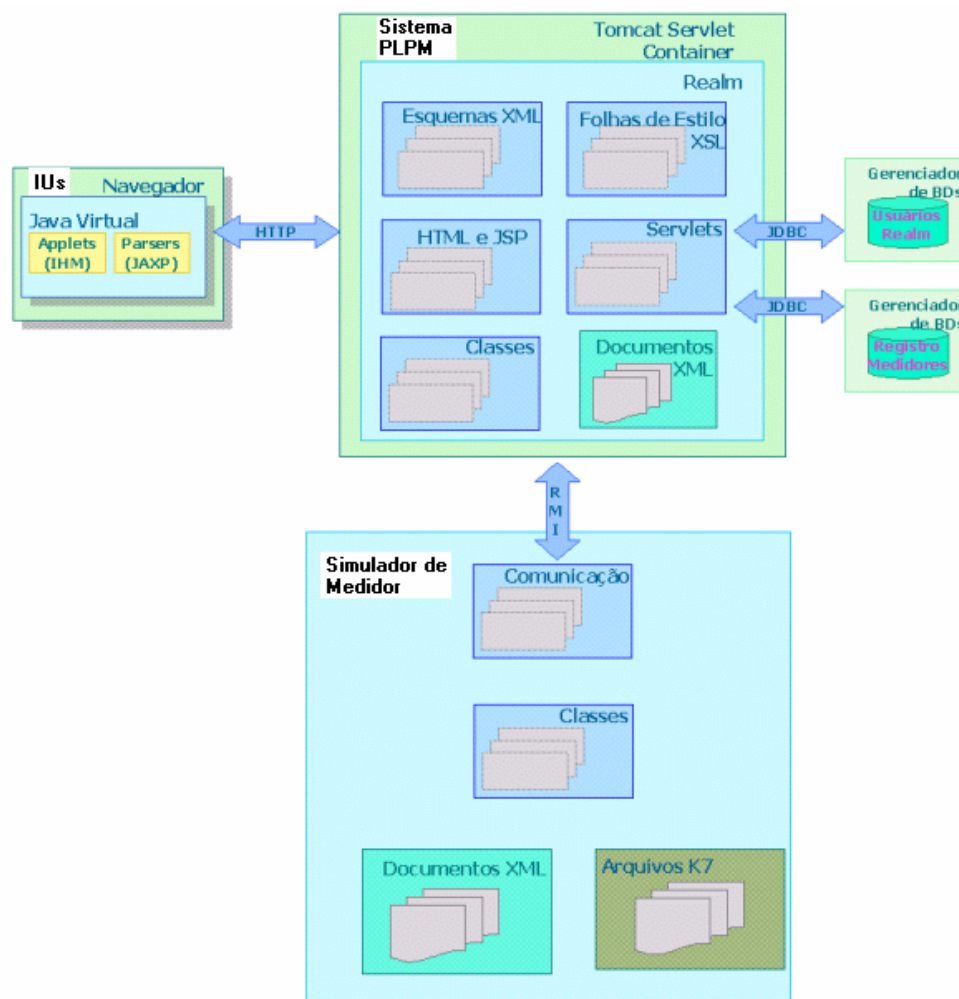


Figura 5.2 – Tecnologias empregadas na aplicação.

A Figura 5.2 mostra a arquitetura detalhada do sistema. Podemos observar na figura as tecnologias que compõem os três grandes blocos do sistema: a Interface do Usuário, o Sistema PLPM e o Simulador de Medidor. É interessante notar que em toda a arquitetura procuramos adotar os padrões abertos da indústria de TI, como Java, ODBC, MySQL, TCP/IP e XML, entre outros.

5.3.1 As Interfaces com o Usuário

As IUs são constituídas de navegadores *web* com capacidade de executar *applets* Java. Esses navegadores podem ser, por exemplo, o Internet Explorer 5 ou superior, o

Netscape Navigator 6 ou superior, entre outros, com o *plugin* do ambiente de execução Java 1.4.1.02.

Os *applets* são códigos de aplicativos Java executados no contexto de um navegador e que agregam a eles funcionalidades não suportadas pelas tradicionais páginas HTML. Especificamente, em nosso protótipo, implementamos controles gráficos da interface com o usuário, instanciamos processadores XML (*parsers*) JAXP para a validação de dados e estabelecemos conexões seguras com o servidor através das funcionalidades oferecidas por esta tecnologia.

Uma grande vantagem em se trabalhar *applets* num sistema do tipo cliente-servidor é que a versão mais atual do *software* cliente estará sempre disponível ao acessar o sistema, não precisando ser reinstalado como em outros tipos de *softwares* clientes.

Através das IUs um usuário pode acionar o sistema, se identificar e, após devidamente autenticado, pode interagir com qualquer medidor dentre os cadastrados na base de dados de medidores do sistema. Identificado o medidor desejado é possível, através das IUs, visualizar dados de uma leitura e, dependendo de suas permissões, parametrizar o medidor.

5.4.2 O Sistema PLPM

Para a implementação do servidor (PLPM) foi utilizada a tecnologia *servlet*. Um *servlet* é um componente *web* baseado na tecnologia Java, gerenciado por um aplicativo conhecido como *servlet container*, que dinamicamente gera conteúdo. Assim como outros componentes baseados em Java, os *servlets* são classes Java independentes de plataforma que podem ser carregadas e executadas em servidores *web* compatíveis com Java, de forma a personalizar o comportamento destes servidores no tratamento das requisições recebidas. O *servlet container* utilizado na implementação deste protótipo é o Apache Tomcat 4.0.5.

O *servlet container* do Apache Tomcat possui implementado internamente a possibilidade de se trabalhar com a tecnologia de *realms*. Uma base de dados *realm* contém os nomes dos usuários autorizados, suas senhas e seus níveis de acesso (*roles*) aos serviços dentro do servidor, e estas informações são utilizadas no momento da autenticação. O conceito de *realms*, que é melhor abordado no Apêndice B, foi utilizado como uma opção sobrepunção de segurança, mas criamos apenas um tipo de usuário com permissões totais de acesso dentro do *servlet container*. Essa decisão foi tomada devido ao fato do significado dos *roles* (papéis) atribuídos a um usuário nessa tecnologia terem uma certa semelhança com o conceito das associações, onde permissões de acesso são estabelecidas de acordo com o tipo de usuário. Portanto, considerando que o conceito de associação tem uma maior relevância dentro do contexto dos sistemas de medição de energia, preferimos trabalhar em mais profundidade as associações.

Dentro do *servlet container* e sobre as restrições de um *realm* existem diversos componentes que fornecem as funcionalidades do PLPM, entre eles os próprios *servlets* que fazem a comunicação com as IUs, as classes Java que implementam o núcleo do sistema manipulando dados e acessam as bases de dados, as classes de comunicação, as páginas HTML e JSP (Java Server Pages) que contém os *applets* que são enviados às IUs e as tecnologias que envolvem XML.

Duas bases de dados auxiliam o PLPM e o acesso a elas é feito via JDBC (Java Database Connectivity). A primeira é a base de dados de *realms* que contém os usuários com permissão de acesso ao sistema. A segunda base de dados contém todos os medidores que o sistema reconhece. Através de uma consulta a essa segunda base de dados é possível identificar o modelo e o fabricante do medidor através do número de série do equipamento. Com o modelo identificado o sistema passa a tratar os dados como especificado no XSD que o descreve.

Em um contexto de leitura, após identificado o medidor, o sistema PLPM é responsável por receber as requisições dos usuários, acionar o medidor e solicitar os dados requisitados. O medidor gera um arquivo de leitura e envia de volta ao sistema. O sistema

recebe o arquivo que corresponde a uma leitura, identifica as restrições que o modelo possui de acordo com o XSD que o descreve e, então, envia os dados ao cliente de acordo com suas permissões identificadas nas associações (o mapeamento das restrições de acesso aos dados é um processo simples e foi explicado no Capítulo 4).

5.4.3 O Simulador de Medidor

O simulador de medidor foi desenvolvido numa primeira fase do projeto para possibilitar a realização de testes. Para isso foi gerada uma carga de dados a partir de arquivos K7 de leitura de dados de medidores reais. O simulador é capaz de simular qualquer um dos três medidores estudados. O que ele faz é mapear um arquivo K7 em arquivos XML descritos pelo XSD do medidor que se deseja simular. Todo esse processo é feito no momento em que o sistema, acionado por um usuário, solicita ao simulador de medidor uma leitura. Ele é estruturado com base em diversos arquivos K7, arquivos XML no formato dos três medidores, classes que fazem o mapeamento entre os arquivos e classes que fazem a comunicação com o sistema PLPM.

5.4.4 O Medidor Incorporado

Numa segunda etapa do projeto incorporamos um medidor de Energia real. O medidor escolhido foi o Saga 1000 da ESB. O que determinou a nossa escolha foi o alto percentual desse modelo no acervo das EEE no Brasil. Com a incorporação de um medidor real foi possível avaliar a capacidade de integração do sistema. Os resultados foram bastante satisfatórios. O que é feito é a transformação interna de dados, obtidos por leituras no medidor através do protocolo ABNT 14522, nas estruturas dos arquivos XML que o sistema manipula e vice-versa. Na figura 5.2, para o caso do medidor real, podemos ignorar na arquitetura o simulador de medidor e o protocolo RMI e considerar o medidor real comunicando com o PLPM através do protocolo ABNT NBR 14522.

5.4 Tipos de Usuários Definidos e Associações

Em caráter de avaliação de conceitos, para analisarmos o emprego das associações, definimos três tipos de usuários, dois administradores com permissões distintas e um consumidor com permissões restritas. As restrições de acesso aos dados para cada usuário são descritas de acordo com associações no medidor. Esses usuários são:

- Administrador 1 – tem direitos de leitura sobre todos os dados e parâmetros, e permissão para parametrizar o medidor;
- Administrador 2 - tem direitos de leitura sobre todos os dados e parâmetros, mas não tem permissão para parametrizar o medidor;
- Consumidor – tem direito de leitura de alguns dados de faturamento para controle interno.

5.4.1 Usuário Administrador 1

Um usuário Administrador 1 tem os seguintes direitos de acesso sobre um medidor:

- Leitura de grandezas relativas ao consumo de energia;
- Leitura de parâmetros do medidor;
- Configuração de parâmetros do medidor.

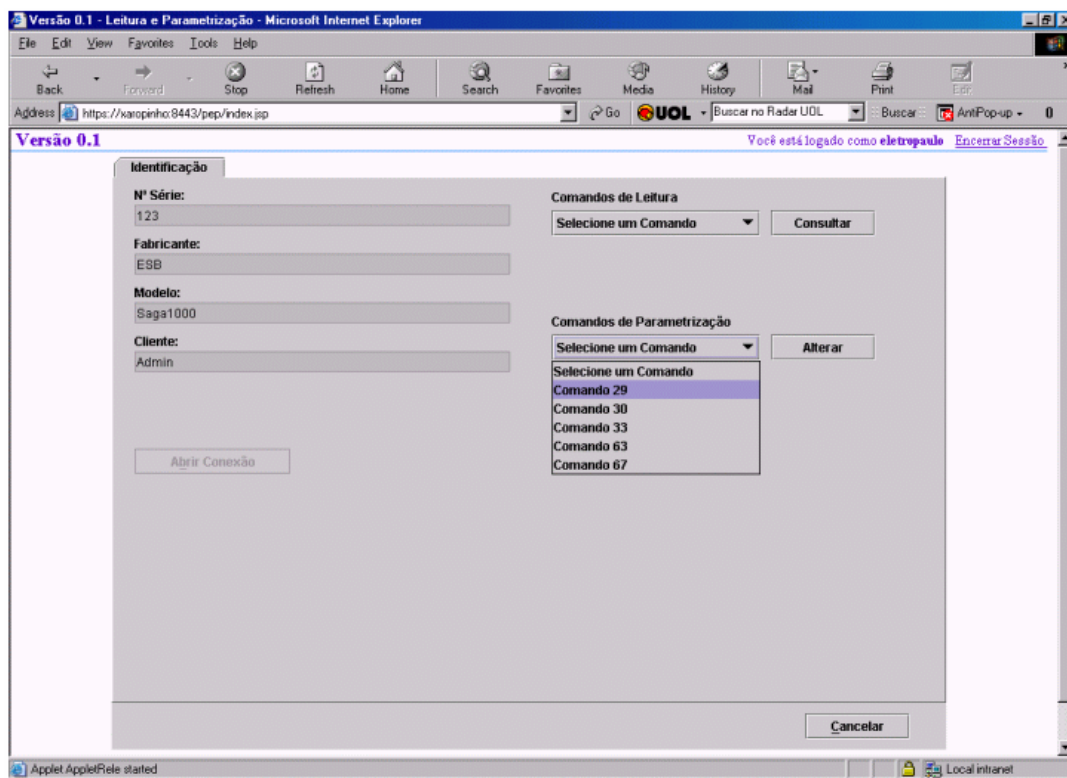


Figura 5.3 – Interface inicial do usuário Administrador 1.

A interface inicial de acesso ao sistema para um usuário do tipo Administrador 1 pode ser observada na Figura 5.3.

5.4.2 Usuário Administrador 2

Um usuário Administrador 2 tem os seguintes direitos de acesso sobre um medidor:

- Leitura de grandezas relativas ao consumo de energia;
- Leitura de parâmetros do medidor.

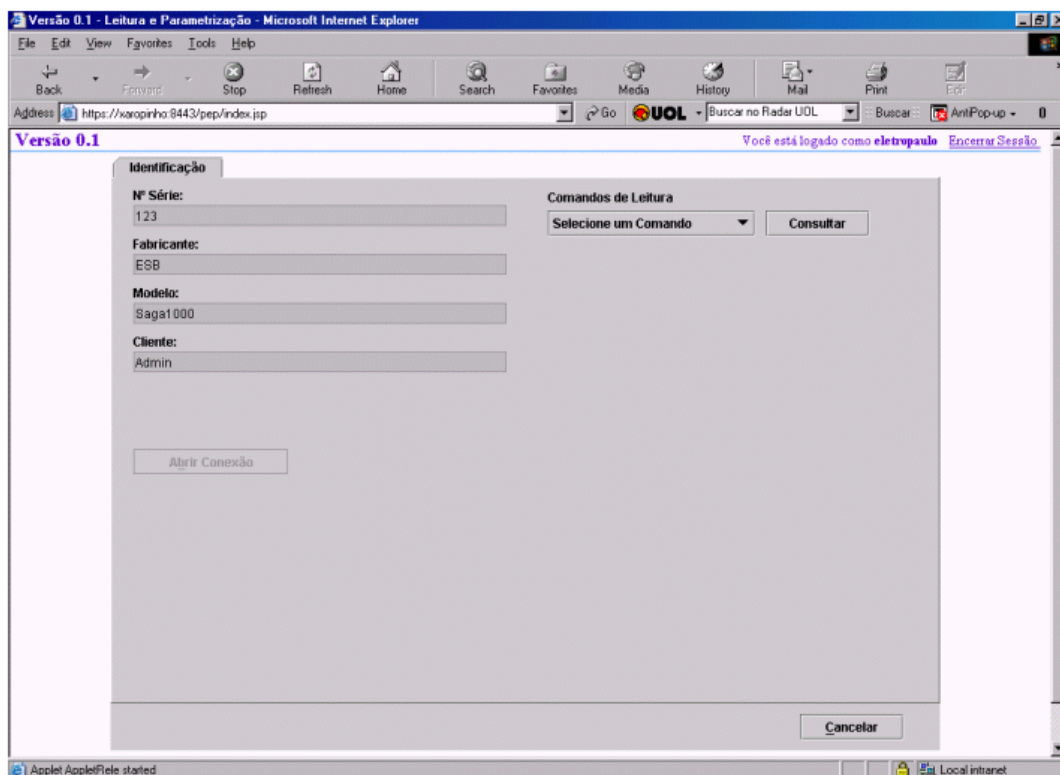


Figura 5.4 – Interface inicial do usuário Administrador 2.

A interface inicial de acesso ao sistema para um usuário do tipo Administrador 2 pode ser vista na Figura 5.4. Como pode ser observado nessa interface não existe a caixa de seleção para parametrizar o medidor como na interface da Figura 5.3. Isso se deve ao fato de um usuário do tipo Administrador 2 não possuir, segundo estabelecido nas associações, permissões de parametrizar o medidor.

5.5.3 Usuário Consumidor

Um usuário Consumidor tem os seguintes direitos de acesso sobre um medidor:

- Leitura de algumas grandezas relativas ao consumo de energia.

A interface inicial para um usuário do tipo Consumidor é idêntica ao usuário Administrador 2. As diferenças de permissões entre eles serão vistas na próxima seção.

5.6 Operação do Sistema

Para poder operar o sistema é necessário que o usuário se identifique. Esse processo é constituído de *login* e senha. A Figura 5.5 mostra a interface inicial onde devem ser digitados os dados de identificação do usuário.

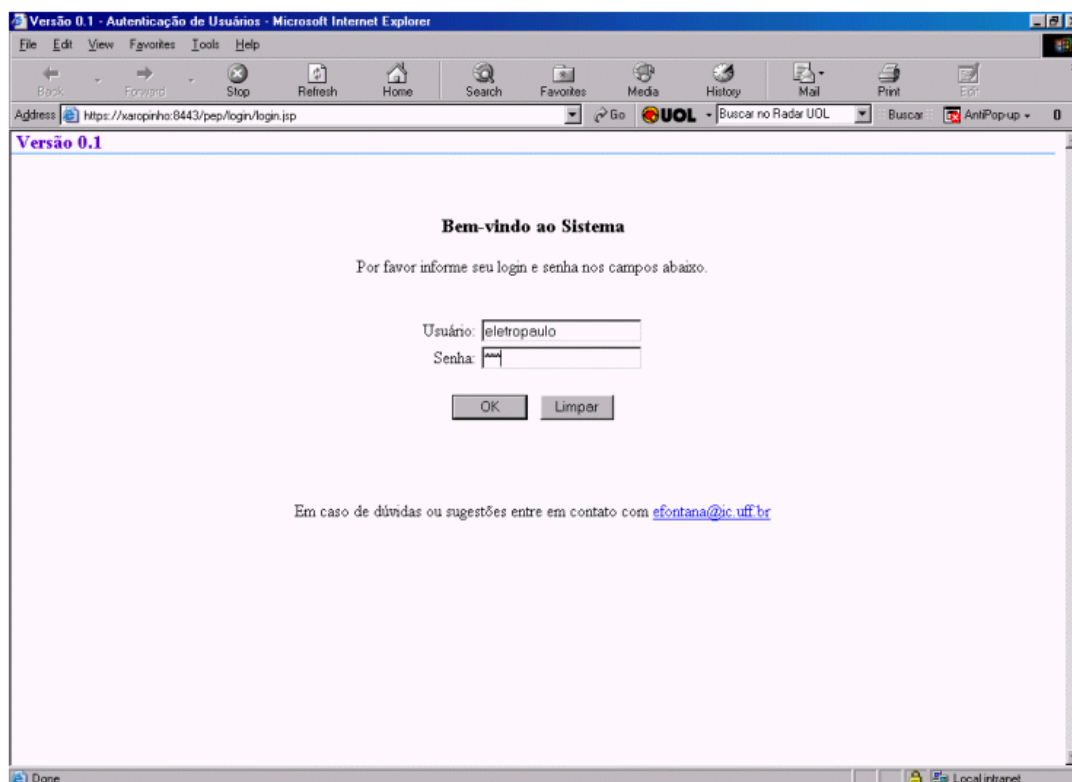


Figura 5.5 – Interface inicial de acesso ao sistema.

Ao submeter os dados através do botão OK, telas com mensagens de aviso de que se está entrando num ambiente seguro e dando informações sobre o certificado digital são lançadas.

Após o processo de *logon* é exibida a interface inicial do sistema (Figura 5.6). A próxima etapa é a identificação do medidor através do seu número de série e do usuário, para se estabelecer o nível de permissão nas associações.

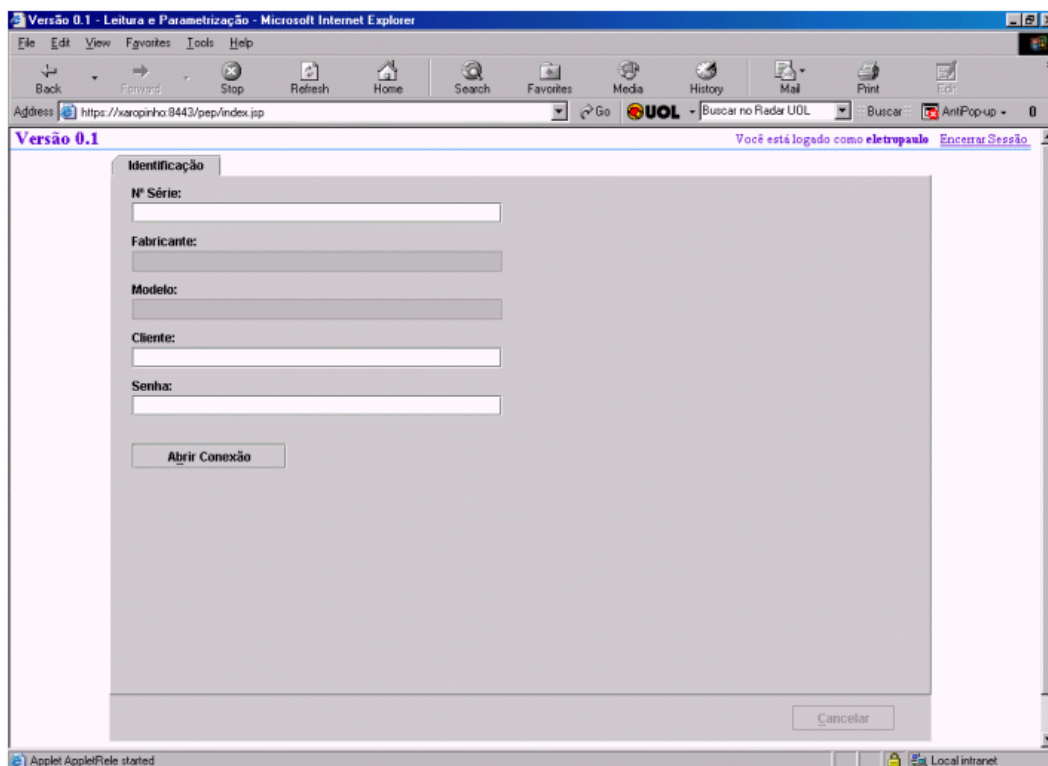


Figura 5.6 – Interface de identificação do medidor e usuário.

A operação do sistema é relativamente simples, e é constituída de apenas dois processos básicos, o de leitura e o de parametrização. É importante frisar que o único usuário que tem direito à operação de parametrização é o usuário Administrador 1.

5.6.1 Processo de Parametrização do Medidor

Para testar o processo de parametrização do medidor foram simulados cinco dos comandos mais usados pelos operadores no processo de configuração de um medidor. A numeração corresponde ao código do comando na norma ABNT NBR 14522:

- 29 – Alteração da data
- 30 – Alteração da hora
- 33 – Alteração das constantes de multiplicação
- 63 – Alteração da condição da reposição de demanda automática
- 67 – Alteração da tarifa de reativos

Para parametrizar o medidor através do sistema a primeira tela é a mostrada na Figura 5.3. O usuário seleciona um dos comandos de parametrização que deseja e pressiona o botão “Alterar”. Em seguida aparece a tela que contém o parâmetro que se deseja alterar com a indicação de seu valor atual no medidor. A Figura 5.7 mostra a tela de parametrização para o comando 29 (alteração da data).

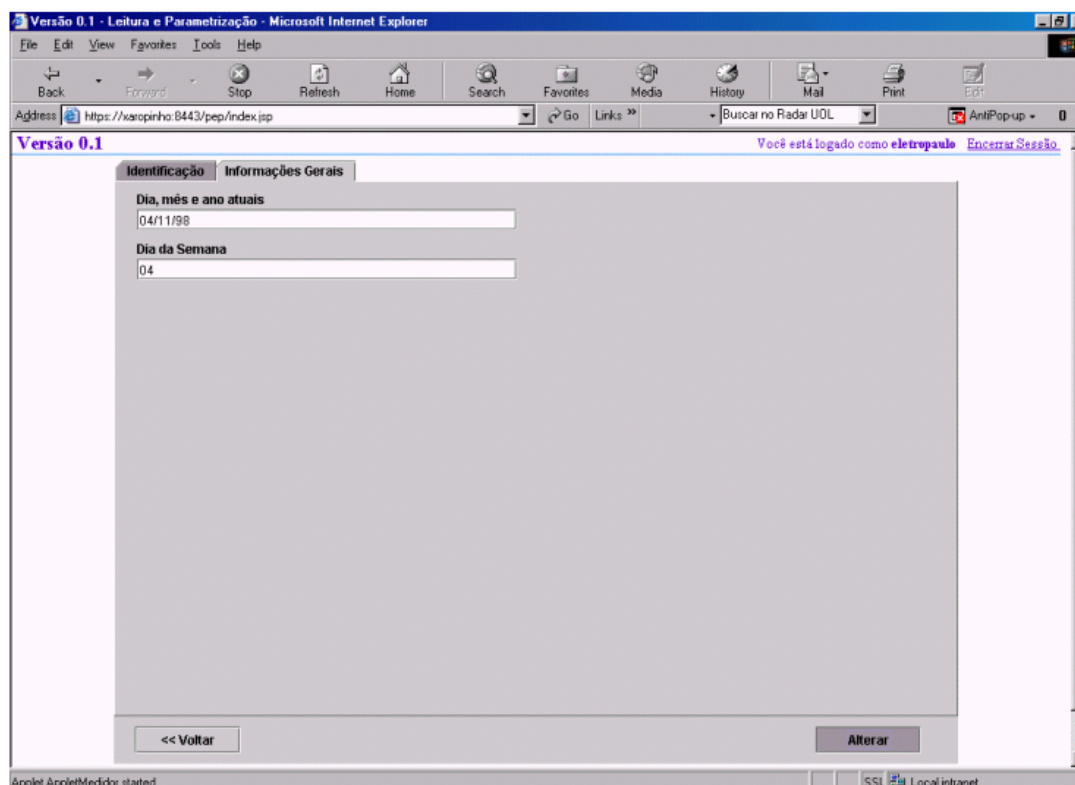


Figura 5.7 – Interface para configuração da data.

O usuário então digita o novo valor e pressiona o botão “Alterar”. A partir daí todo o processo é automatizado e realizado pelo Sistema PLPM. O novo valor é enviado ao medidor e alterado na representação XML que registra os parâmetros no medidor. O Código 5.1 mostra a data alterada na estrutura do documento XML.

```
<IG01>
  <class_id>1</class_id>
  <Version>0</Version>
  <logical_name>IG01</logical_name>
  <value>04/11/98</value>
</IG01>
<IG012>
```

```

<class_id>1</class_id>
<Version>0</Version>
<logical_name>IG012</logical_name>
<value>04</value>
</IG012>

```

Código 5.1 – Representação XML dos registradores que guardam os valores relativos a data no medidor.

Para realizar a adaptação ao medidor real, uma transformação é aplicada ao código XML para obter o formato de comando de parametrização que o medidor entenda. A Figura 5.8 mostra uma representação dos dados, do comando 29 para o formato previsto na Norma ABNT NBR 14522, que é capaz de configurar os medidores de energia atuais, obtidos através de uma dessas transformações.

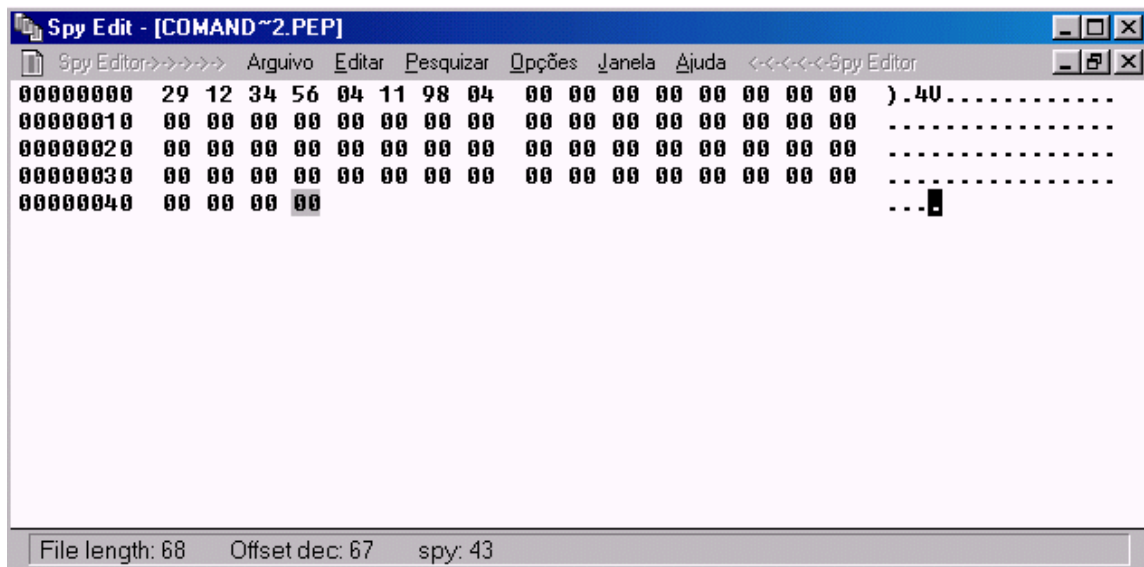


Figura 5.8 – Comando 29 no formato da Norma ABNT NBR 14522.

5.6.2 Processo de Leitura do Medidor

Em um processo de leitura em nosso sistema a interface é adaptada automaticamente, de forma transparente ao usuário, ao medidor específico. Isso é possível através da identificação das restrições de um medidor através de seu XSD. Para testar o processo de Leitura do medidor foram simulados dois comandos básicos:

- 24 – Leitura de registradores relativa à última reposição de demanda
- Informações gerais

Para fazer uma leitura no medidor através do sistema a primeira tela é a mostrada na Figura 5.4. O usuário seleciona um dos comandos de leitura que deseja e pressiona o botão “Consultar”. Em seguida, aparece a tela que contém os dados com a leitura solicitada e de acordo com as permissões de acesso do usuário. A Figura 5.9 ilustra os dados relativos ao Canal 1 de um medidor Saga 1000, de uma leitura realizada por um usuário Consumidor.

Figura 5.9 – Processo de Leitura do medidor.

Como podemos notar, a Figura 5.9 possui as grandezas listadas em três maneiras. Essa foi a forma que achamos conveniente para ilustrar as restrições que um medidor possui em relação ao modelo genérico, e também as restrições de um usuário em relação às associações. As grandezas com o nome em negrito, e os campos em branco e com valores preenchidos, como as quatro primeiras no exemplo, representam aquelas que existem no

medidor e que o usuário tem acesso. As grandezas com os nomes em cinza representam as grandezas que existem no modelo genérico, mas não são implementadas nesse medidor específico. As outras grandezas com o nome em negrito, porém com os campos preenchidos com o texto “Sem acesso”, representam as grandezas que o medidor possui, mas que o usuário atual não tem permissão de acesso. É importante lembrar que essas permissões são definidas nas associações.

5.7 Geração de *Logs* do Sistema

Todos os aspectos de segurança propostos no Capítulo 4 foram implementados no sistema. Já nos referimos a vários deles no decorrer do texto. A geração de *logs* foi um aspecto adicional de segurança implementado no sistema.

A geração de arquivos de *log* já foi vista no passado como uma forma de gerar grande sobrecarga e consumir recursos de armazenamento. Mas com o barateamento dos recursos de *hardware* e aumento do poder de processamento esse aspecto de segurança tornou-se um requisito fundamental para qualquer grande sistema de *software*. O registro de todas as interações de usuários com o sistema, ou de outros fatores relevantes, facilita em muitos casos a triagem da origem de um problema.

Na nossa aplicação são gerados *logs* no Sistema PLPM e no simulador de medidor de todas as interações com os usuários. No sistema, são registrados em arquivo texto o código que representa o tipo de acesso ao medidor, a data e a hora, o usuário que fez o acesso, o IP da máquina de onde o acesso foi feito e o medidor com o qual o usuário está interagindo. A Figura 5.10 mostra um exemplo desse arquivo de *log*.



Figura 5.10 – Arquivo de *log* do sistema PLPM.

No simulador, são registrados o código que representa o tipo de acesso, a data e a hora, o usuário que fez o acesso e o IP da máquina de onde o acesso foi feito. A Figura 5.11 mostra um exemplo desse arquivo de *log*.



Figura 5.11 – Arquivo de *log* do simulador.

CAPÍTULO 6

CONCLUSÕES E TRABALHOS FUTUROS

6.1 Conclusões

O objetivo principal desta dissertação foi demonstrar, em termos conceituais e práticos, os benefícios e vantagens da abordagem de utilização de modelos para a integração de sistemas heterogêneos, principalmente no contexto de medidores de energia elétrica. Nessa abordagem, a interoperabilidade entre sistemas é tratada em um nível mais alto de abstração, no nível de representações de domínio.

O uso de modelos como estratégia de integração mostrou-se uma boa alternativa no processo de integração de sistemas. Essa abordagem permite que a solução esteja centrada nas funcionalidades e comportamento dos sistemas, separando-a dos detalhes das tecnologias utilizadas na implementação.

Para comprovar a viabilidade da integração de sistemas com uma abordagem baseada em modelos foi construído um sistema que permitiu integrar, através de uma interface unificada, diferentes medidores com concepções distintas do domínio do problema.

Durante o desenvolvimento deste trabalho várias questões foram levantadas. O conhecimento adquirido nos permite avaliar que durante um processo de integração de sistemas algumas decisões tomadas podem fazer a diferença entre o sucesso e o fracasso. Assim é muito importante considerar:

- **O uso de padrões:** Esta tendência verifica-se em várias (muitas) áreas. O uso de padrões barateia os custos de desenvolvimento e manutenção, facilita a

aceitação de produtos, permite a interoperabilidade entre fabricantes diversos, entre outras vantagens.

- **O uso da tecnologia de orientação a objetos:** Permite a representação da entidade a ser modelada como uma classe abstrata, da qual podem ser obtidas implementações concretas. No caso, permite a definição de um componente lógico (medidor) que pode ser especializado para cada fabricante ou modelo particular. Isso permite um tratamento único para qualquer que seja o medidor. Além disso, essa representação permite a extensão para a adição de novas funções em relação ao modelo básico, o que é uma vantagem competitiva para cada fabricante em particular.
- **A importância da segurança:** Com a utilização em rede dos novos dispositivos eletrônicos inteligentes torna-se possível o acesso indevido (não autorizado ou ferindo restrições de acesso) a esses dispositivos. Dessa forma, segurança torna-se um conceito fundamental a ser adotado em qualquer plataforma. Conceitos como o de Associação, conforme empregado na proposta DLMS, tornam-se fundamentais para o bom funcionamento do sistema. Nesse caso, cada dispositivo pode ser apresentado de formas diversas para diferentes usuários (ex., administradores, cliente), em função das restrições de acesso que se desejar aplicar.
- **O uso de padrões da Internet:** O uso dessa tecnologia, além das vantagens associadas à padronização acima referidas, permite o acesso às funções de um dispositivo de qualquer lugar e sempre com a mesma interface de usuário (ou uma simplificação, no caso de computadores portáteis).

6.2 Propostas de Trabalhos Futuros

Conforme dito anteriormente, este trabalho não pretendeu ser definitivo, mas sim contribuir na investigação das soluções adequadas à integração de sistemas. Como

continuação deste, propomos a seguir a investigação e o desenvolvimento de alguns temas que se evidenciaram ao longo da realização do projeto.

O primeiro passo na evolução do sistema seria a inclusão de novos modelos de medidores com a nomenclatura similar. Isto permitiria a integração das configurações dos novos equipamentos que utilizarem este padrão ao sistema PLPM atual. Esta implementação poderia ser realizada com a inclusão de novos esquemas com o modelo de dados descrito no padrão.

Um outro passo seria a incorporação de novas interfaces ao sistema. Isso poderia ser realizado, por exemplo, através do fornecimento de leituras de dados no formato de documentos XML através de *Web Services* (tecnologia recente que vem se destacando no processo de integração de sistemas), ficando o processamento dos dados sob responsabilidade do cliente.

Outra questão seria a criação de uma interface gráfica para a elaboração de novos modelos. A exemplo das facilidades apresentadas pelo *software* XMLSpy, poderia ser criado um *software* que permitisse a elaboração de novos modelos de medidores com recursos do tipo "corte", "cópia" e "cola". Os blocos básicos (componentes e classes COSEM mapeadas em XSD) desenvolvidos poderiam ser disponibilizados em uma biblioteca com elementos gráficos, facilitando o uso dos usuários não experimentados em XML.

6.3 Contribuições deste Trabalho

Este trabalho apresentou uma proposta de unificação de sistemas de leitura e parametrização de medidores de energia que poderá servir de base para o desenvolvimento de uma solução de integração de sistemas. A abordagem utilizada poderá servir de exemplo na integração de outros sistemas em outros domínios.

Foram também apresentadas as propostas de padrões de integração sendo desenvolvidas para a integração de sistemas em empresas do setor elétrico. Esta compilação poderá ser usada como ponto de partida para o conhecimento destas propostas por estudantes e profissionais da área que queiram se informar sobre as tendências de padrões para esta área.

A implementação do protótipo permitiu verificar de modo prático os benefícios da utilização de XML na troca e armazenamento de informações e na modelagem de equipamentos.

Foi possível avaliar e demonstrar a capacidade das classes COSEM em representar as diversas funções do domínio de medição. COSEM surge como uma proposta concreta de padronização para os sistemas de medição que deve ser considerada nos futuros projetos de integração dentro das empresas.

Finalmente, como um adicional ao Padrão COSEM, e também como uma contribuição desse trabalho, duas classes inexistentes em COSEM, que se fizeram necessárias para o mapeamento de algumas características dos medidores nacionais, foram criadas. A primeira está relacionada com a porta ótica de comunicação com o medidor. Foi necessária a criação de uma classe para registrar os dados de configuração dessa porta de comunicação (Figura 6.1).

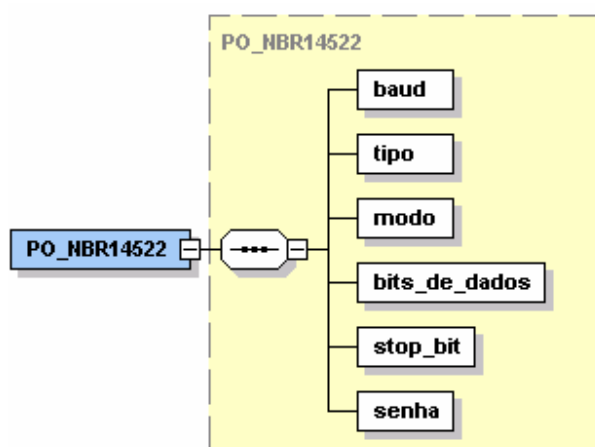


Figura 6.1 – Classe PO_NBR14522.

A segunda classe criada está relacionada com o registro das alterações nos dispositivos (Figura 6.2). Ela atende aos dispositivos atuais que registram o código da alteração, o número de série do leitor e o momento da operação. Ela também já prevê as futuras mudanças para o cenário do PLPM, ou outro sistema com arquitetura semelhante, com o registro do usuário e do número IP da máquina de acesso ao medidor.

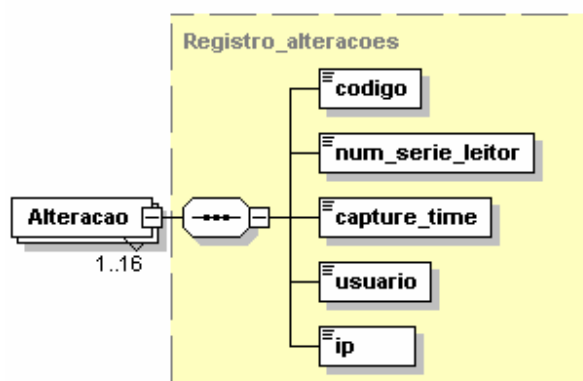


Figura 6.2 – Classe Registro_alteracoes.

REFERÊNCIAS BIBLIOGRÁFICAS

[Ahme 01] K. Ahmed, S. Ancha, et al., “*Professional Java XML*”, Wrox Press Ltd., 2001.

[Alt 04] Altova, “*XMLSPY*”. Disponível em <http://www.xmlspy.com>, página consultada em Fevereiro de 2004.

[ASF 03] Apache Software Foundation, “*The Tomcat 4 Servlet/JSP Container - Documentation Index*”, <http://jakarta.apache.org/tomcat/tomcat-4.0-doc/index.html>.

[Birb 01] M. Birbeck, J. Diamond, et al., “*Professional XML – 2nd Edition*”, Wrox Press Ltd., 2001.

[DLMS 02a] DLMS UA 1000-2, “*COSEM Three Layer Connection Oriented Architecture*”, 3a. ed., DLMS User Association, 2002.

[DLMS 02b] DLMS UA 1000-1, “*COSEM Identification System and Interface Objects*”, 5a. ed., DLMS User Association, 2002.

[ISO 86] ISO, “*Information Processing Text and Office Systems Standard Generalized Markup Language (SGML)*”, ISO 8879, 1986.

[MMS 90] IEC, “*Manufacturing Message Specification (MMS) - Service Definition / Protocol Definition*”, International Standard ISO/IEC 9506-1 / 9506-2, 1990.

[NBR 00] Associação Brasileira de Normas Técnicas, “Intercâmbio de informações para sistemas de medição de energia elétrica”, NBR 14522, Rio de Janeiro, 1995.

[NSTF 97] National Security Telecommunications Advisory Committee Information Assurance Task Force, “*Electric Power Risk Assessment*”, Março 1997, http://www.ncs.gov/n5_hp/Reports/EPRA/electric.html.

[Oman 00] P. Oman, E. O. Schweitzer III e D. Frincke, “*Concerns About Intrusions into Remotely Accessible Substation Controllers and Scada Systems*”, <http://www.selinc.com/techpprs/6111.pdf>.

[Reyn 03] E.M.D. Reynoso, “Integração de Sistemas Baseada em Modelos: Uma Aplicação no Setor Elétrico”, Dissertação de Mestrado, PGCC – IC/UFF, 2003.

[Schw 01] K. Schwarz, IEEE UCA and IEC 6185, “*Seamless Communication from Powerplants to Customer Interfaces*”, SCC, Alemanha, www.nettedautomation.com/download/UCA_Seamless_2001-06.pdf.

[Silv 03] C. M. P. Silva, “Integração de Sistemas de Automação: Uma Proposta Para a Unificação de Sistemas de Configuração de Relés de Proteção”, Dissertação de Mestrado, PGCC – IC/UFF, 2003, página 15.

[Stal 99] W. Stallings, “*Cryptography and Network Security*”, Prentice Hall, Nova York, EUA, 1999.

[Sun 03] Sun, “*Java API for XML Processing (JAXP)*”, <http://java.sun.com/xml/jaxp/index.html>

[UCA 00] EPRI, “*Generic Object Models for Substation & Feeder Equipment (GOMSFE)*”, Versão 0.91, Fevereiro 2000.

[UCA 97a] EPRI, “*Introduction to UCA Version 2.0*”, Editorial Draft 1.0, 1997.

[UCA 97b] EPRI, “*Common Application Service Models (CASM) and Mapping to MMS*”, Editorial Draft 1.4, 1997.

[W3C 98] W3C, “*Extensible Markup Language (XML) 1.0 (Second Edition)*”, W3C, 1998.

[W3C 99a] W3C, “*Namespaces in XML*”, W3C Recommendation, 1999. <http://www.w3.org/TR/Rec-xml-names>.

[W3C 99b] W3C, “*eXtensible Stylesheet Language Transformation (XSLT) Versin 1.0*”, W3C, 1999, <http://www.w3.org/TR/xslt>.

[W3C 00] W3C, “*Simple Object Access Protocol (SOAP) 1.1*”, W3C Note, 2000. <http://www.w3.org/TR/2000/NOTE-SOAP-20000508>.

[W3C 01] W3C, “*XML Schema Part 0: Primer*”, W3C, 2001, <http://www.w3.org/TR/xmlschema-0/>.

[Win 99] D. Winer, “*XML-RPC Specification*”, 1999. <http://www.xmlrpc.com/spec>.

APÊNDICE A

XML – EXTENSIBLE MARKUP LANGUAGE

Este Apêndice aborda diversas tecnologias e soluções abertas utilizadas durante o desenvolvimento desse trabalho. Essas tecnologias são centradas na linguagem XML, uma grande aposta do mercado atual de tecnologias da informação. A XML é uma linguagem para descrição de dados padronizada que reúne em um único documento os dados e sua semântica.

A.1 Introdução

A rápida evolução da Internet tornou a *World Wide Web* um amplo repositório de informações dos mais diversos domínios. Além disso, também possibilitou o intercâmbio de informações entre organizações de forma muito simples, sem a necessidade de se ter uma conexão física direta entre elas. Porém, as informações disponíveis através da Internet apresentam dificuldades para uma utilização eficiente e eficaz. A principal dificuldade deve-se ao fato de que as organizações que provêm estas informações utilizam diferentes maneiras para estruturá-las, isto é, diferentes formatos de representação. Quando a informação é transferida para outras organizações, a falta de um formato padrão de representação causa problemas, tais como incompatibilidade entre o sistema gerador da informação e o sistema receptor dessa informação. Assim, as diferenças na estrutura e significado das informações constituem um obstáculo para a interoperabilidade entre os sistemas das organizações que desejam intercambiar informação.

Existem várias formas para resolver este problema. Uma delas é desenvolver programas de extração e manipulação, cujo principal objetivo seja permitir a transferência de informações entre organizações. A principal desvantagem desta abordagem é que o custo de desenvolvimento e manutenção destes programas é muito alto. Assim, a adoção de uma linguagem comum parece ser o caminho mais lógico para resolver este problema.

Desta forma, a linguagem XML (*eXtensible Markup Language*) surge como um mecanismo para satisfazer a necessidade de se ter um formato universal de representação de dados que permita o intercâmbio de informação entre sistemas heterogêneos. Com a linguagem XML é possível representar quaisquer tipos de dados estruturados ou semi-estruturados, dentre os quais podemos enumerar: transações comerciais, catálogos de livros, relatórios financeiros, técnicos ou estatísticos, dados de tipo gráfico, anúncios publicitários, entre outros.

Um fator determinante no sucesso da tecnologia XML é que, além de fornecer um formato universal para representar dados, permite descrevê-los. Ou seja, os dados são autodescritíveis e podem oferecer as informações (como devem ser interpretados, e o que deve ser feito com eles) que os usuários ou aplicações necessitam para processar corretamente estes dados. Considerando que a identificação dos dados dá alguma compreensão do que eles significam, XML é às vezes descrita como um mecanismo para especificar a semântica dos dados. XML é baseada em texto, é fácil de entender e de escrever. Além disso, a tecnologia XML fornece uma ampla disponibilidade de ferramentas de *software* que facilitam o rápido desenvolvimento de aplicações, tornando ainda mais atrativa sua utilização.

A.2 XML

O termo XML (*eXtensible Markup Language*) é usado para fazer referência à linguagem utilizada para descrever dados e a estrutura de documentos e para referenciar um conjunto de tecnologias relacionadas. Assim, como a maioria das linguagens, ela tem um conjunto de regras e convenções que definem que elementos são válidos em um documento XML, e como esses elementos podem ser combinados para formar um documento válido.

XML é uma linguagem para especificação de documentos contendo informação estruturada. Documentos com informação estruturada possuem conteúdo (palavras, imagens, etc.) e alguma indicação de que função este conteúdo representa. Quase todos os

documentos têm alguma estrutura. Uma linguagem de marcação é um mecanismo para identificar estruturas em um documento. A especificação XML define uma forma padrão para adicionar marcação a documentos.

A XML é uma linguagem de marcação assim como HTML, e foi desenvolvida em resposta à necessidade de uma generalização da linguagem HTML. Cabe mencionar que tanto XML quanto HTML são derivadas da mesma linguagem “mãe”, SGML (*Standard Generalized Markup Language*). Em HTML, contudo, existe um número fixo de *tags* e a semântica de cada *tag* é sempre a mesma. XML não especifica nenhuma semântica nem um conjunto fixo de *tags*. Na realidade, XML é uma metalinguagem para descrever linguagens de marcação. Em outras palavras, XML fornece um aparato para definir *tags* e o relacionamento estrutural entre eles. A partir do momento que não há um conjunto pré-definido de *tags*, não pode haver nenhuma semântica antecipadamente expressada. Toda a semântica de um documento XML será definida pelas aplicações que o processam ou por folhas de estilo.

A SGML, que foi concebida por volta de 1960-1970, deu vida a um perfil/subconjunto chamado XML, publicado como uma recomendação W3C em 1998. Dependendo do ponto de vista e necessidades, as diferenças entre SGML e XML são irrelevantes ou imensas. A SGML é mais personalizável, mais flexível e mais abrangente, ao custo de ser mais difícil de implementar. Hoje em dia a SGML não é, na maioria das vezes, a linguagem de marcação escolhida no início de um projeto em nível empresarial, mas muitas aplicações que usam SGML, implementadas antes de 1998, ainda estão executando de forma produtiva.

Um objetivo específico da linguagem XML é manter o mesmo poder de descrição da linguagem SGML, sendo removida a maior parte de sua complexidade. A especificação SGML [ISO 86] tem quase 400 páginas, enquanto que a especificação XML [W3C 98] não tem mais de 40 páginas. Por mais de uma década, SGML foi a linguagem de marcação padrão mais usada para manutenção de repositórios de documentação estruturada, de forma independente do fornecedor. Mas a SGML não é muito adequada para o propósito de troca

de informação através da *web*, isso porque ela é complexa e de difícil aprendizagem o que fará com que seja apenas utilizada por comunidades que possuam um elevado nível técnico e, conseqüentemente, não tenha uma grande aceitação por parte dos usuários normais. A XML é definida como um *perfil* de aplicação da SGML. Definir XML como um perfil de aplicação de SGML significa que qualquer sistema em conformidade com o SGML é capaz de ler documentos XML. No entanto, usar e entender documentos XML não requer um sistema que seja capaz de entender a completa generalidade e complexidade de SGML. Pode se dizer, a grosso modo, que a XML é uma forma restrita de SGML.

A linguagem XML é uma metalinguagem, ou seja, a XML possui a habilidade de definir arbitrariamente outras linguagens de marcação. Explica-se: a XML não tem um conjunto fixo de marcações, não têm um vocabulário específico (*labels* para elementos e atributos) e uma sintaxe declarada (gramática definindo a hierarquia e outras características). As marcações podem ser definidas segundo as necessidades específicas de cada usuário ou aplicação. Isto permite que a XML possa representar praticamente qualquer tipo de informação. Assim, a principal característica desta linguagem é sua extensibilidade. Exemplos de linguagens baseadas em XML são a WML (*Wireless Markup Language*) e o ebXML (*Electronic Business XML Language*).

A.3 A XML é uma Linguagem de Marcação

A XML é uma linguagem de marcação baseada em texto, isto é, uma linguagem que contém elementos especiais (também chamadas de marcações, rótulos, etiquetas ou *tags*) utilizados para descrever a estrutura e conteúdo de diferentes tipos de documentos, a serem utilizados principalmente por aplicações computacionais. As marcações fornecem significado sobre o conteúdo dos documentos. Desta forma, documentos escritos em uma linguagem de marcação consistem de texto, que representa o conteúdo do documento, e marcações que adicionam estrutura e informação sobre esse conteúdo. Exemplos de linguagens de marcação são o HTML, RTF (*Rich Text Format*) e *Latex*.

Para ilustrar o conceito de marcações, a seguir é apresentado um exemplo muito simples. Seja a frase “Meu nome é Eduardo e tenho 25 anos”. Um ser humano ao ler esta sentença saberia que “Eduardo” é um nome de uma pessoa e que “25” é a sua idade. Porém, os programas computacionais não são seres humanos e não têm como saber isto. Assim, utilizamos os metadados para dar significado aos dados. A seguir apresentamos a informação que desejamos passar com a sintaxe da linguagem XML:

```
< Pessoa >
  < nome >Eduardo< / nome >
  < idade >25< / idade >
< / Pessoa >
```

Exemplo A.1 – Linguagem de marcação.

A inserção das marcações `< Pessoa >`, `< nome >`, `< idade >` permite que um programa de computador possa entender que “Eduardo” é o nome de uma pessoa e que 25 é sua idade.

A.4 XML é eXtensível

XML é uma linguagem que permite aos projetistas de *software* definir seus próprios elementos de acordo com as necessidades do sistema, diferentemente de HTML, onde o universo de elementos disponíveis e *tags* é finito e fixo. Essa é a natureza extensível de XML: é possível definir, estender e refinar o conteúdo e a estrutura dos documentos de acordo com os requisitos de uma aplicação específica ou domínio de problema.

Tomemos como base o exemplo A.1, para este exemplo três *tags* básicas foram criadas `< Pessoa >`, `< nome >` e `< idade >` para passar a informação que desejávamos. Mas suponhamos que em outra ocasião fosse necessário a um sistema outras informações de uma pessoa, além de nome e idade, como o sexo e o estado civil, por exemplo. Para adicionar informações ao documento XML basta adicionar elementos extras, `< sexo >` e `< estado_civil >`.

```

< Pessoa>
  < nome>Eduardo</ nome>
  < idade>25</ idade>
  < sexo>M</ sexo>
  < estado_civil>solteiro< estado_civil>
</ Pessoa>

```

Exemplo A.2 – XML é extensível.

Com uma linguagem que tem a característica de ser extensível, podemos adequar as informações de acordo com a necessidade dos sistemas.

A.5 Estrutura de um Documento XML

Um documento XML é basicamente formado de elementos, atributos e dados, além de algumas *tags* especiais, como os comentários da linguagem, por exemplo. Existe sempre um elemento raiz e todos os outros elementos estão aninhados dentro desse elemento raiz. O elemento raiz possui elementos internos e pode ter atributos. Os elementos internos ao elemento raiz são chamados elementos filhos. Os elementos filhos podem ter atributos ou mesmo outros elementos filhos, e assim por diante. É esta possibilidade de uma *tag* conter outras *tags* que dá a XML sua habilidade para representar estruturas de dados hierárquicas. Vejamos o exemplo A.3 que representa uma forma de estruturar mensagens de e-mail através de documentos XML.

```

< message>
  < to>julius@ic.uff.br</ to>
  < from>efontana@ic.uff.br </ from>
  < subject>Apêndice A - XML </ subject>
  < text>Esse é o Apêndice A...</ text>
</ message>

```

Exemplo A.3 – Estrutura de um documento XML.

As *tags*, nesse exemplo, identificam a mensagem como um todo, os endereços do remetente e destinatário, o assunto e o texto da mensagem. Para cada *tag* inicial, como exemplo a *tag* <to>, existe uma *tag* final correspondente </to>. Os dados entre as *tags* inicial e final correspondente definem um elemento de dado XML. Note também que o conteúdo da *tag* <to> está inteiramente contido dentro do escopo das *tags* do elemento raiz <message></message>.

As *tags* também podem conter atributos que são informações adicionais incluídas como parte da própria *tag*. O exemplo A.4 mostra uma estrutura de mensagem de e-mail ilustrando o uso de atributos:

```
<message to="julius@ic.uff.br" from="efontana@ic.uff.br">  
  <subject>Apêndice A - XML </subject>  
  <text>Esse é o Apêndice A...</text>  
</message>
```

Exemplo A.4 – Estrutura de um documento XML com atributos.

A.6 Características Adicionais

Embora os documentos XML não tenham sido criados primariamente para serem lidos por usuários, a especificação XML claramente expressa como um de seus objetivos que "os documentos XML deverão ser legíveis por seres humanos e razoavelmente claros" [W3C 98]. Esta característica de legibilidade contribuiu para a adoção de XML. XML permite a comunicação entre pessoas e computadores e garante abertura, transparência e independência de plataforma comparada a um formato binário, por exemplo.

O tratamento de alguns conceitos relacionados à estrutura dos documentos XML pode ter um impacto significativo no projeto e desempenho de aplicações baseadas em XML. Esses conceitos dizem que os documentos XML devem ser:

1. Bem-formados - Um documento XML precisa ser bem formado para ser analisado. Um documento XML bem formado é conforme com as regras da sintaxe XML e suas restrições, tais como:

- O documento deve conter exatamente um elemento raiz e todos os outros elementos são filhos desse elemento raiz;
- Todas as *tags* devem ser balanceadas; isto é, cada elemento deve ter uma *tag* de início e uma de fim;

- Elementos podem ser aninhados, mas eles não podem ser sobrepostos;
- Todos os valores dos atributos devem estar entre aspas.

2. Válidos - De acordo com a especificação XML um documento é considerado válido se ele tem uma declaração do DTD (veja subseção A.7.1) associado e concorda com as restrições expressadas nesse DTD. Para ser válido, um documento XML deve reunir os seguintes critérios:

- Ser bem formado;
- Referenciar um esquema baseado em DTD que seja acessível usando a *tag* de declaração do tipo do documento;
- Estar Conforme com o DTD referenciado.

Com o surgimento de novas linguagens de esquemas, a noção de validade é estendida além da especificação inicial a outras linguagens de esquemas não baseadas em DTDs, tais como XSDs.

A.7 Os Esquemas XML

O objetivo principal das *tags* XML é fornecer informação do significado dos dados nos documentos XML. Mas, por mais que as *tags* ajudem a descrever os documentos XML, elas não são suficientes, mesmo quando cuidadosamente escolhidas, para fazer um documento completamente auto-descritivo. Além do mais, como visto na seção A.2, na linguagem XML não existe um conjunto pré-definido de *tags* para ser utilizado. As *tags* são então definidas na criação do esquema do documento para cada aplicação específica. Os esquemas definem as restrições que os documentos XML e seus conteúdos devem seguir, de acordo com o modelo de dados necessário para suprir as regras de negócio da aplicação

específica. Os esquemas ajudam a melhorar a capacidade de descrição dos documentos XML a partir do momento que eles definem a sua exata estrutura.

Ao iniciar um projeto que usa a XML o primeiro passo é definir o esquema. Os esquemas definem um vocabulário, uma sintaxe e uma semântica. O vocabulário define que marcações podem ser utilizadas na construção dos documentos XML, a sintaxe define a hierarquia das marcações e a semântica define o significado das marcações.

Os esquemas podem ser utilizados por programas que precisem fazer validações na estrutura do documento, geralmente programas que envolvem a inserção ou atualização de informação numa base de dados, ou ainda ser utilizado como documento de referência a outros usuários que queiram utilizar a mesma linguagem para compartilhar de informação.

As linguagens de esquema recomendadas pela W3C são as *Document Type Definitions* (DTDs) (ver subseção A.7.1) e a *W3C XML Schema Definition* (XSD) (ver subseção A.7.2) [W3C 01], mas outras linguagens são possíveis. Um esquema XML define os elementos que podem aparecer em um documento e os atributos que podem ser associados a um elemento. Um esquema também define a estrutura de um documento e a hierarquia entre os elementos: quais elementos são filhos de outros, a seqüência em que os elementos filhos devem aparecer e o número de filhos que um elemento pode ou deve ter. Um esquema também pode definir valores *default* para atributos. A XSD permite ainda definir os tipos de dados dos elementos e atributos dos documentos XML (veja uma comparação entre DTDs e XSDs no sub-item A.7.3).

A.7.1 Document Type Definition

A linguagem de esquema DTD (*Document Type Definition*) é uma recomendação W3C que permite descrever classes de documentos XML. Uma DTD:

- Define qual a estrutura de um documento;

- Especifica quais os elementos disponíveis para formar um documento XML;
- Especifica, para cada elemento, quais são seus possíveis atributos, qual o seu escopo, que sub-elementos possuem e em que ordem.

O Exemplo A.5 mostra a estrutura de uma DTD (em negrito no código) e de um documento XML que é conforme com ela. A DTD diz que o documento para ser conforme com ela deve possuir um elemento raiz, chamado no exemplo <LIVRORECEITAS>, e que esse elemento deve ter somente um tipo de elemento filho, no caso, <RECEITA>. O elemento <RECEITA> pode aparecer uma ou mais vezes e possui os elementos filhos <TITULO> e <INGREDIENTE>. O elemento <TITULO> não possui elementos filhos, deve aparecer uma única vez e como o primeiro elemento filho. O elemento <INGREDIENTE> pode aparecer uma ou mais vezes e não possui elementos filhos. A DTD diz ainda que o elemento <RECEITA> deve possuir o atributo ORIGEM obrigatório.

```

<!DOCTYPE LIVRORECEITAS [
<!ELEMENT LIVRORECEITAS      (RECEITA*)>
<!ELEMENT RECEITA            (TITULO,INGREDIENTE*)>
<!ELEMENT TITULO              (#PCDATA)>
<!ELEMENT INGREDIENTE        (#PCDATA)>
<!ATTLIST RECEITA ORIGEM CDATA #IMPLIED>
]>

<LIVRORECEITAS>
  <RECEITA ORIGEM="Brasil">
    <TITULO> Bolo </TITULO>
    <INGREDIENTE> 500g de farinha </INGREDIENTE>
    <INGREDIENTE> 200g de açúcar </INGREDIENTE>
    <INGREDIENTE> 300g de manteiga </INGREDIENTE>
  </RECEITA>
</LIVRORECEITAS>

```

Exemplo A.5 – Uma DTD e um documento XML conforme com ela.

A.7.2 XML *Schema Definition Language*

A XSD é a atual linguagem de definição de esquemas mais usada para o propósito de especificar o conteúdo de documentos XML. Para ilustrar as XSDs vejamos o exemplo A.6, que é um esquema que define a estrutura e o conteúdo das *tags* do documento do exemplo A.1:

```

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:element name="pessoa">
    <xs:annotation>
      <xs:documentation>Dados que descrevem uma
pessoa</xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:sequence>
        <xs:element name="nome">
          <xs:annotation>
            <xs:documentation>Nome da pessoa</xs:documentation>
          </xs:annotation>
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:length value="50"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
        <xs:element name="idade">
          <xs:annotation>
            <xs:documentation>Idade</xs:documentation>
          </xs:annotation>
          <xs:simpleType>
            <xs:restriction base="xs:int">
              <xs:minInclusive value="0"/>
              <xs:maxInclusive value="150"/>
              <xs:totalDigits value="3"/>
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>

```

Exemplo A.6 – XSDs definindo documentos.

O Esquema diz que um documento que descreve uma pessoa deve possuir um elemento raiz <pessoa>, que esse elemento deve ter uma sequência de dois elementos filhos obrigatórios, <nome> e <idade>. O dado do elemento <nome> deve ser do tipo *string* de tamanho 50 caracteres. O dado do elemento <idade> deve ser do tipo inteiro, com no máximo 3 dígitos e deve variar de 0 a 150.

A.7.3 DTDs x XSDs

Originalmente, a sintaxe DTD, que é parte central da especificação XML 1.0 e se tornou uma recomendação em 1998, era a única linguagem permitida para a criação de esquemas de domínios específicos. No entanto, com o crescimento da adoção de XML, tornou-se claro que a sintaxe da DTD tinha algumas limitações. As limitações da DTD são:

- Ela tem uma sintaxe diferente da XML. Isto significa que os projetistas devem entender duas linguagens diferentes e utilizar dois tipos de *parsers*;
- Ela não suporta *namespaces* (ver subseção a seguir). Com os *namespaces* é possível validar elementos específicos dentro do documento XML;
- Ela não pode expressar os tipos dos dados, especialmente para o conteúdo dos elementos. Por exemplo, com uma DTD não é possível dizer que um elemento `<preco>` deve conter um número, muito menos que este número deve ser maior que zero, com precisão de duas casas decimais e um símbolo de moeda como R\$.

Para suprir esta deficiência, A W3C definiu a XSD (XML Schema Definition language), que se tornou recomendação da W3C em 2001. A XSD é uma tentativa para resolver todos esses problemas através da definição de uma sintaxe baseada em XML que inclui:

- Ela própria é uma aplicação baseada em XML - um XSD pode ser escrito e manipulado exatamente como um documento XML;
- Permite expressar os tipos de dados que inclui verificação de faixa de valores;
- Validação por *namespaces* baseadas em URI (Uniform Resource Identifier); com isso a XSD permite o projeto de esquemas modulares e permite a composição de definições de esquemas XSDs;

Entretanto, as XSDs não são uma solução definitiva. Em particular, os XML Schema não substituem as DTDs: eles podem ser utilizados em conjunto no mesmo documento. Uma DTD é mais fácil de ser escrita do que um esquema em XML Schema correspondente e, ainda, a DTD é a maneira mais prática para descrever documentos narrativos.

A.7.4 XML *Namespaces*

Os XML *namespaces* [W3C 99a] são uma recomendação W3C cuja idéia é podermos criar um documento em que uma parte obedeça a um esquema e outras partes obedeçam a outros. Assim, podemos criar novas estruturas de documentos combinando partes de outras já existentes. Vejamos o Exemplo A.7:

```
<LIVRORECEITAS xmlns="http://www.ic.uff.br/XSDs/receitas.xsd">
  <RECEITA>
    <TITULO> Omelete </TITULO>
    <INGREDIENTE> 2 ovos </INGREDIENTE>
    <INGREDIENTE> 100g de mussarela </INGREDIENTE>
    <INGREDIENTE> 100g de presunto </INGREDIENTE>
  </RECEITA>
  <RECEITA xmlns:marc="http://www.ic.uff.br/XSDs/marca.xsd">
    <TITULO> Bolo </TITULO>
    <INGREDIENTE> 500g de farinha </INGREDIENTE>
    <marc:MARCA> Dona Benta </art:MARCA>
    <INGREDIENTE> 200g de açúcar </INGREDIENTE>
    <marc:MARCA> União </art:MARCA>
    <INGREDIENTE> 300g de manteiga </INGREDIENTE>
    <marc:MARCA> Itambé </art:MARCA>
  </RECEITA>
</LIVRORECEITAS>
```

Exemplo A.7 – Uso de *namespaces* em XML.

O Exemplo A.7 mostra um XML que representa um livro de receitas. A primeira receita contida no escopo da *tag* <RECEITA> contém três *tags* <INGREDIENTE>, que contém informação sobre os ingredientes para se fazer uma omelete. O significado da *tag* <INGREDIENTE> vem do esquema especificado pelo atributo *xmlns* da *tag* raiz (mãe de todas as outras) <LIVRORECEITAS>, ou seja, "http://www.ic.uff.br/XSDs/receitas.xsd". Em seguida, criamos uma nova receita. Agora, porém, se deseja também colocar para cada ingrediente uma marca desejável. Contudo, o XSD "receitas.xsd" não especifica a

semântica para uma *tag* em que possamos fornecer uma marca. Assim, como não previmos nenhum elemento para isso, importamos uma semântica através da especificação de um *namespace* que é feita através do atributo *xmlns* da *tag* onde desejamos fazer valer o novo esquema. No exemplo, "http://www.ic.uff.br/XSDs/marca.xsd" é o esquema especificado como o *namespace marc* através do atributo *xmlns* da segunda *tag* <RECEITA>, que contém um significado para a *tag* <MARCA>, e possivelmente para outras *tags*. A partir deste momento todos os elementos do esquema marca.xsd podem ser usados desde que prefixados com "marc" <marc:MARCA>.

É importante ressaltar que a especificação de um *namespace* é válida dentro do escopo da *tag* onde ele foi definido. Assim só podemos usar a *tag* <MARCA> dentro do escopo da segunda *tag* <RECEITA>. Se criarmos uma terceira receita através de uma terceira *tag* <RECEITA>, e não especificarmos para ela um novo *namespace*, o XSD que volta a valer é o especificado pela sua *tag* mãe <LIVRORECEITAS>.

Os *namespaces* são úteis também para resolver conflitos de nomes. Por exemplo, vamos supor que em um determinado documento XML desejamos passar a informação de um endereço através de uma *tag* <ENDERECO>. Essa *tag*, dependendo do contexto pode ser entendida como um endereço eletrônico, como uma URL ou um *email*, ou ainda como um endereço residencial. Assim, através do uso de *namespaces*, podemos definir qual o esquema a ser utilizado para cada *tag* <ENDERECO> contida em um documento XML. Por exemplo, poderíamos definir dois *namespaces eletr* e *resid* que se refeririam a dois esquemas *eletronico.xsd* e *residencial.xsd* e, então, essas *tags* passariam a ser <eletr:ENDERECO> e <resid:ENDERECO>.

A.8 XSLT

Muitas vezes é necessário transformar um documento XML para permitir o intercâmbio de dados e a comunicação com outras aplicações do sistema e também com sistemas externos. Quando necessário os dados em XML são transformados para o formato utilizado por estas aplicações, tais como outros formatos de XML, HTML, SGML, texto,

entre outros. A ferramenta de transformação utilizada é o XSLT [W3C 99b] (*eXtensible Stylesheet Language Transformation*).

A XSLT [W3C 99b] é uma linguagem que, de acordo com a primeira frase de sua especificação, foi projetada principalmente para transformar um documento XML em outro. A XSLT é usada para transformar dados utilizando folhas de estilos que descrevem o mapeamento do formato de origem para o formato requerido.

A.9 Java API for XML Processing (JAXP)

Um passo necessário para que as marcações XML possam ser acessadas pelas aplicações é fazer a tradução destas marcações XML para um formato que a linguagem de programação utilizada possa entender. Este processo é chamado de tradução (*parsing*). Um outro passo importante é a validação dos documentos XML. Este processo é responsável por garantir que os dados expressos no documento XML concordam com o modelo de dados definido através de um esquema (DTD ou XML *Schema*) [Birb 01].

A JAXP (*Java API for XML Processing*) é uma API Java que permite aplicações analisarem e transformarem documentos XML usando um recurso que é independente de uma implementação particular de processamento XML. A razão para existir JAXP é facilitar o uso de XML por aplicações desenvolvidas para a plataforma Java. Ela fornece uma interface para que documentos XML e aplicativos Java possam conversar. Ela define um pacote Java que fornece funcionalidades básicas para ler, manipular, validar e gerar arquivos XML. Para maiores informações, consultar [Sun 03].

A.10 XMLSpy

O XMLSpy [Alt 04] é uma ferramenta para a manipulação de documentos XML que suporta a modelagem, edição, depuração e validação de qualquer tecnologia baseada em XML, incluindo esquemas XML, XSL, XSLT, entre outras. Ele fornece um ambiente gráfico de desenvolvimento que facilita o entendimento e a edição de modelos em XML. A

Figura A.1 mostra uma representação gráfica construída com o XMLSpy de um XML Schema para o exemplo A.2.

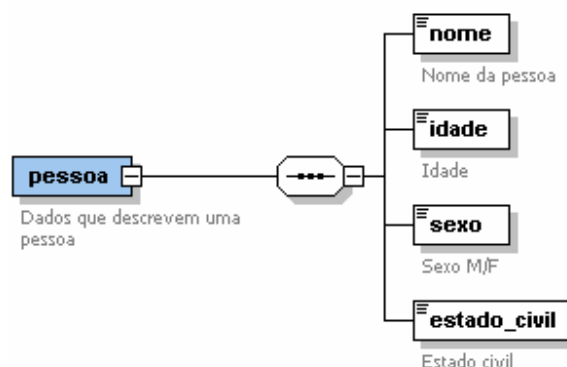


Figura A.1 – Representação gráfica do XSD construído no XMLSpy.

A.11 Potenciais Aplicações da Tecnologia XML

Como mencionado, a linguagem XML pode ser utilizada em um amplo espectro de domínios de aplicação. Nesse sentido, podemos identificar os seguintes domínios de aplicação:

- **Especificação de protocolos de comunicação baseados em XML.**
 - Vários protocolos de comunicação, como CORBA ou DCOM, realizam as mesmas tarefas, mas de uma forma incompatível. Isto é, para que haja comunicação entre eles é necessário um mecanismo de intermediação. Estes mecanismos de intermediação, também chamados de interfaces, são construídos para cada par de protocolos incompatíveis. XML permitiria utilizar um formato de intermediação comum. Exemplos de protocolos baseados em XML são o XML-RPC [Win 99] e o SOAP (*Simple Object Access Protocol*) [W3C 00].
- **Construção de mecanismos de busca (*search engines*) mais eficientes.**

- Um mecanismo de busca poderia pesquisar nas marcações, em vez de pesquisar o documento inteiro. O XQuery é uma proposta de linguagem de consulta em documentos XML.
- **Gerenciamento de conteúdo *web*.**
 - A linguagem XML permite que um mesmo documento XML seja visualizado ou interpretado de diferentes formas por diferentes usuários ou aplicações. Mecanismos de transformação XML (XSL, XSLT) permitem a conversão de dados XML de uma forma para outra, utilizando um conjunto de regras que são aplicadas aos dados na primeira forma.
- **Intercâmbio de informação**
 - A XML pode facilitar o intercâmbio de informação entre aplicações heterogêneas, através da definição de um formato comum de representação de transações e mensagens. Isto é fundamental para possibilitar o conceito de *eBusiness* (*electronic Business*), ou seja, o comércio eletrônico através da *web*. O *framework ebXML* (*electronic business XML*) visa definir um conjunto padrão de formatos de transações comerciais a serem trocadas pelas organizações que fazem comércio entre elas.
- **Integração de Aplicações.**
 - A capacidade da XML de fornecer um formato padrão de intercâmbio de dados pode simplificar o desenvolvimento de interfaces padronizadas para a interconexão de aplicações.
- **Sistemas de Arquivos.**

- Os sistemas de arquivos consistem de listas hierárquicas, compostas de pastas e arquivos. Qualquer pasta pode conter arquivos ou outras pastas. Um conceito equivalente ao conteúdo de elementos XML, que pode conter dados ou outros elementos. Porém, um sistema de arquivos trabalha sobre um modelo de atributos fixos, onde cada tipo de arquivo tem atributos específicos, tais como nome, tamanho, tipo, data de criação, data da última modificação, data do último acesso. Este modelo de atributos fixos é bastante rápido, mas é também limitado em relação à meta-informação que pode acompanhar um arquivo. Com um sistema de arquivos XML é possível eliminar essas restrições.

Embora a linguagem XML possa ser utilizada em um amplo espectro de domínios de aplicação, ela apresenta também alguns problemas. O maior deles é a falta de limitações para o significado de um nome de marcação. Por exemplo, uma marcação para um usuário pode identificar a localização de um cliente e para outro usuário a localização de uma área geográfica. Para uma efetiva utilização, os usuários que utilizam a XML como formato padrão de intercâmbio de dados devem concordar também em definir um vocabulário compartilhado de marcações.

APÊNDICE B

ASPECTOS ADICIONAIS DE SEGURANÇA UTILIZADOS

B.1 Introdução

Um estudo recente da entidade norte-americana NSTAC (*National Security Telecommunications Advisory Committee*) alertou que os sistemas de energia elétrica atuais são altamente vulneráveis à intrusão. Em particular, o estudo demonstrou que as subestações são os elementos mais vulneráveis na rede elétrica, principalmente devido ao fato de seus dispositivos de acesso remoto serem quase que totalmente desprotegidos [NSTF 97]. Os autores do relatório concluíram que ataques remotos a dispositivos em subestações poderiam causar grandes interrupções no fornecimento de energia e até mesmo provocar seu colapso total.

Questões de segurança também se aplicam no caso dos medidores de energia, que se encontram localizados fisicamente nas instalações das empresas usuárias, potencialmente mais vulneráveis que as subestações de distribuição de energia. No caso dos medidores, um intruso poderia ganhar acesso ao sistema e mudar ajustes, de modo a obter benefícios financeiros para um determinado consumidor. Uma outra possibilidade, seria o intruso usar as informações do cliente para auferir vantagem estratégica para sua empresa ou negócio.

O objetivo deste Apêndice é discutir algumas questões de segurança relacionadas a sistemas de energia, incluindo unidades remotas de processamento usadas em sistemas de automação e controle do setor elétrico. A discussão é centrada na arquitetura, baseada em Internet, proposta para o sistema desenvolvido neste projeto. Contudo, a análise e os resultados apresentados podem ser aplicados em outros sistemas semelhantes do setor elétrico, atuais e a serem futuramente desenvolvidos.

Na arquitetura proposta podem ser diferenciados três elementos (Figura B.1):

- Sistema Padronizado de Leitura e Parametrização de Medidores (PLPM), que age como um servidor e encapsula os *softwares*, metadados e dados relativos à aplicação, no caso, a integração de diversos tipos de medidores e suas respectivas instâncias.
- Interfaces de Usuários (IU), que agem como clientes do sistema PLPM, nossa aplicação de interesse.
- Unidades Remotas de Medição (URM), unidades que incluem *hardware* e *software* necessários para implementar funções específicas. Neste caso, dispositivos medidores multifunção que incluem funções de leitura de registros e ajustes de parâmetros.

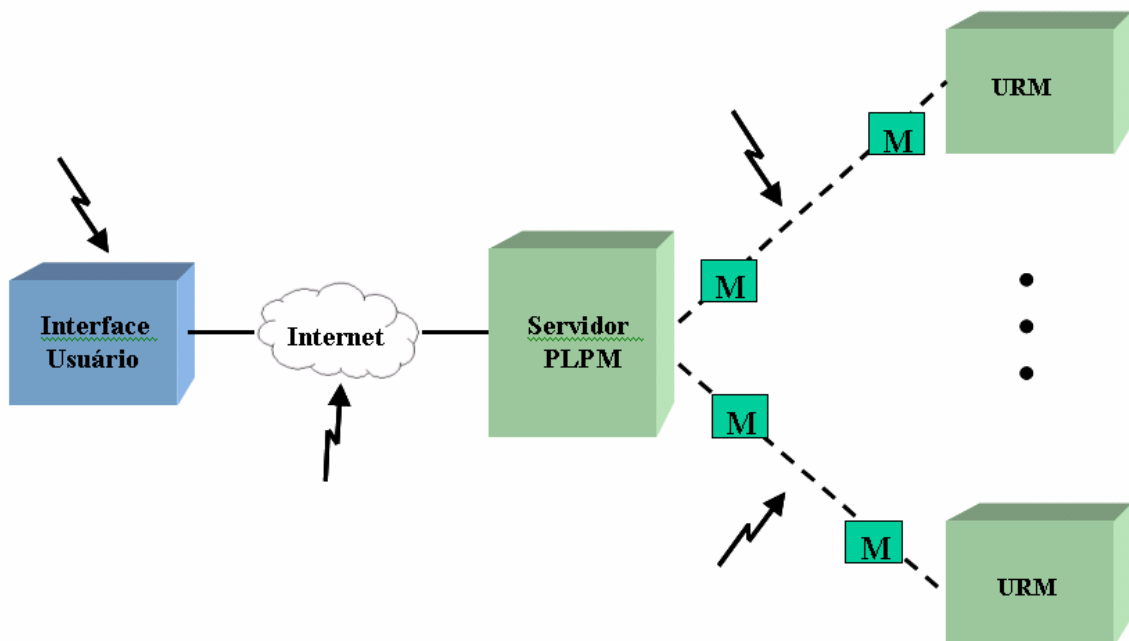


Figura B.1 – Arquitetura proposta.

Na Figura B.1, as setas representam os principais pontos vulneráveis a ataques na arquitetura em questão. Além dos elos de comunicação, a segurança inclui o aspecto de controle de acesso às interfaces de usuário. Adicionalmente, deve-se realizar esse mesmo

tipo de controle no acesso ao sistema PLPM e às unidades remotas, ou seja, nesta aplicação, os dispositivos medidores. No sistema a comunicação entre os usuários (representados pelas IUs) e o sistema PLPM será baseada em ambiente Internet. Desta forma, a segurança pode ser obtida tomando como base a tecnologia de uso corrente neste ambiente. Notar que os mecanismos de segurança da própria aplicação são adicionais aos que estão sendo aqui apresentados.

Embora no futuro a comunicação entre o PLPM e as URM's possa ser baseada na tecnologia Internet, atualmente, nos sistemas de interesse do projeto, ela é baseada em comunicação via *modem*, sobre linhas discadas, utilizando protocolos especializados da área de controle (e.g., Modbus) ou proprietários. Desta forma, neste nível de comunicação consideraremos basicamente o último cenário (ver seção B.3).

Ameaças à segurança por meio de ataques diretos às interfaces de usuário embutidas nos medidores requerem o acesso físico às instalações dos consumidores. Uma vez obtido o acesso físico ao medidor, o controle de acesso às suas funcionalidades, utilizando senhas, via a interface (porta) serial remota, é praticamente idêntico ao exigido para controlar o acesso a um usuário remoto, ponto a ser discutido na subseção B.2.1. Além da porta para acesso remoto, o padrão nacional [NBR 00] prevê duas portas adicionais: (i) uma porta ótica para a programação e acesso aos registros do medidor; e, (ii) uma porta serial extra, do tipo somente-leitura, para acesso do usuário. A restrição somente-leitura para a porta de acesso do usuário elimina a possibilidade de intrusão. Quanto à porta ótica, com acesso controlado por senha, o padrão nacional prevê, como proteção adicional, um registro de memorização das alterações efetuadas, possibilitando a identificação de intrusões através de um processo de auditoria externa.

B.2 Questões no Nível IU/PLPM

O propósito de disponibilizar a interface da aplicação em ambiente distribuído impõe a necessidade de um programa cliente para acesso ao servidor da aplicação PLPM. Na forma tradicional, é necessário instalar o código executável do cliente nas máquinas adequadas. No caso da aplicação precisar ser alterada, seria necessário redistribuir e reinstalar a aplicação, tornando onerosa a manutenção da aplicação. Por outro lado, a tecnologia dos navegadores (*browsers*), usados em ambientes *web*, permite que a versão mais atual da aplicação seja disponibilizada ao cliente (e.g., via *applets*) a cada acesso realizado através de um navegador, sem nenhum custo adicional de distribuição. Isto introduz preocupações ligadas à segurança no lado do cliente e no servidor. Por exemplo, é necessário impedir que um cliente impostor, usando um código malicioso, ganhe acesso ao servidor e realize operações indevidas. Ou seja, é preciso manter o controle de quais usuários estão habilitados a acessar a aplicação e especificar quais recursos da aplicação poderão ser disponibilizados para um usuário específico. Além do controle de acesso, é necessário garantir que os dados e comandos transmitidos entre os usuários e o servidor da aplicação não serão acessados ou adulterados por entidades não autorizadas. Neste sentido, torna-se necessário o emprego de técnicas que garantam a segurança no acesso e na transferência de informação.

B.2.1 Controle de Acesso à Aplicação

O controle de acesso no nível de aplicação é baseado na autenticação de usuários, usando o tradicional mecanismo *logon*/senha, que pode apresentar um alto nível de proteção, desde que usado adequadamente. Por exemplo, uma *password* de 6 dígitos, selecionados num conjunto de 90 caracteres, se não fosse usada nenhuma política limitante de tentativas, levaria 7.361 anos para ser quebrada usando-se um *modem* de 38.400 bps [Oman 00].

Para a implementação do controle de acesso baseado em senhas ao servidor, foi utilizado a tecnologia Realm embutida no servidor Tomcat (que inclui um *servlet*

container), uma solução baseada na plataforma Java com qualidade comercial e de código aberto [ASF 03]. Um domínio Realm é composto por uma base de dados de nomes de usuários e senhas que identificam usuários válidos de uma aplicação web. Para cada usuário desta base de dados pode ser atribuída uma lista de valores conhecidos como papéis (*roles*). Através destes papéis é possível verificar se um determinado usuário tem permissão para executar determinadas funções da aplicação, por exemplo, ler e modificar ou somente ler os ajustes e os registros de um medidor. Uma vez autenticado através de sua senha, o usuário e seus papéis associados são mantidos até que a sessão ultrapasse o tempo limite especificado ou o usuário explicitamente encerre a sessão (*logout*). Deve ser notado que a aplicação construída possui mecanismos de proteção adicionais, através do conceito de associações, como definido no padrão COSEM.

B.2.2 Comunicação Cliente-Servidor

Como visto, a segurança na transmissão de dados no sistema PLPM proposto foi tratada em dois contextos. Primeiramente, foi garantido o sigilo e a integridade das transferências entre as interfaces dos usuários (IU - clientes), por onde os usuários do sistema fornecem a senha de acesso e acessam os dispositivos medidores, e o servidor da aplicação, onde estes dados são processados e armazenados. O outro contexto é constituído pela transmissão dos dados entre o servidor da aplicação PLPM e as unidades remotas de medição e será tratado na seção 3.

B.2.2.1 Confiabilidade

A *web* (*World Wide Web* ou WWW) define uma forma básica de construção e troca de documentos sobre a Internet baseada na especificação de um protocolo de requisição/resposta conhecido como HTTP (*Hypertext Transfer Protocol*) e uma linguagem para a construção de documentos, a HTML (*Hypertext Markup Language*). Em seu desenvolvimento, a *web* foi sendo utilizada como base para um sistema de informação distribuído, baseado no modelo cliente-servidor, devido ao protocolo HTTP. O protocolo

HTTP é um protocolo baseado em conexões TCP/IP constituído de dois tipos de mensagens: requisição (*request*) HTTP e resposta (*response*) HTTP.

O TCP/IP é um modelo de rede onde são utilizados os protocolos TCP (*Transmission Control Protocol*) e IP (*Internet Protocol*). O protocolo IP é responsável pelo encaminhamento dos dados até o destino. É um protocolo sem conexão e sem detecção de erros, por isso, pouco confiável. Para realizar o encaminhamento, ele recebe o datagrama do TCP e insere seu cabeçalho. Neste cabeçalho há as informações sobre os endereços de origem e destino. Estas informações são usadas pelos roteadores intermediários, entre a origem e o destino, para determinar a rota do datagrama. O protocolo TCP é definido como sendo um protocolo orientado a conexão, que realiza o tratamento de erros e perda de comunicação, para uso sobre o IP. Desta forma, ele provê confiabilidade (no sentido da recuperação de erros de transmissão) na transferência de dados entre dois pontos terminais, localizados em dois *hosts* diferentes, como é o caso do cliente (navegador) e do servidor *web*, os quais utilizam o protocolo HTTP, que opera sobre o TCP. Entretanto, o TCP é um protocolo desprovido de mecanismos de segurança, ou seja, é vulnerável à ação de agentes não autorizados para o acesso aos dados transmitidos.

B.2.2.2 Segurança

A SSL (*Secure Socket Layer*) é uma das tecnologias de segurança mais usuais atualmente para a transmissão de dados em redes baseadas em TCP/IP. Seu objetivo é prover um canal seguro, isto é, com privacidade, garantia de autenticidade dos pares e garantia de integridade das mensagens transmitidas [Stal 99]. Devido a estas características, este protocolo foi usado para implementar a segurança na transmissão de dados entre os clientes e o servidor do protótipo do sistema. A SSL é uma tecnologia que permite aplicações baseadas no modelo cliente-servidor estabelecerem comunicação através de uma conexão TCP/IP segura. Isto significa que os dados enviados são criptografados por um destes agentes (cliente ou servidor), em seguida são transmitidos e então decifrados pelo agente receptor antes de serem processados. Este processo é válido nos dois sentidos, ou seja, tanto o servidor quanto o cliente fazem a criptografia dos dados antes do envio. Os

principais servidores HTTP e navegadores *web* disponíveis atualmente têm suporte nativo à tecnologia SSL, facilitando a sua utilização.

Além da criptografia, outro aspecto importante do protocolo SSL é a autenticação. Durante a tentativa inicial de comunicação com um servidor através de uma conexão segura, o cliente recebe um conjunto de credenciais do servidor na forma de um "Certificado Digital". Esta é a garantia que o cliente tem de que está tentando estabelecer uma conexão com o servidor original, evitando assim que servidores impostores acessem os dados da aplicação.

Deste modo, o SSL implementa a segurança na transmissão de dados através de três serviços:

- **Autenticação:** permite que as partes envolvidas na comunicação sejam seguramente identificadas, evitando que entidades impostoras venham assumir o papel de alguma entidade do sistema para ter acesso às informações protegidas.
- **Criptografia:** garante a privacidade das informações transmitidas através de algoritmos de criptografia que “embaralham” os dados transmitidos de forma que somente entidades confiáveis tenham a capacidade de “desembaralhar” estes dados.
- **Integridade:** protegem a informação transmitida contra erros e modificações indevidas no conteúdo.

B.2.2.3 Comentários

A tecnologia SSL vem se mostrando eficiente na implementação de canais seguros sobre a pilha de protocolos TCP/IP. Este fato pode ser observado pela sua ampla utilização na realização de transações confidenciais na Internet, como operações financeiras, por

exemplo. No desenvolvimento do sistema utilizamos a plataforma Java, que disponibiliza todos os recursos necessários para a implementação de aplicações que utilizam canais seguros SSL. Uma aplicação cliente-servidor baseada em HTTP pode utilizar o protocolo de segurança SSL de forma transparente para o usuário.

A utilização de certificados digitais é um outro ponto importante a se considerar. Como discutido anteriormente, para o funcionamento adequado em condições reais, o protocolo SSL deve adquirir um Certificado Digital emitido por uma Autoridade de Certificação, garantindo a autenticidade do certificado e, conseqüentemente, do servidor acessado. Na nossa aplicação, simulamos a emissão de um certificado através de uma ferramenta (*keytool*) disponível na plataforma Java. Esta solução poderia ser usada como uma solução interna para um sistema fechado, como o usado no âmbito de uma empresa. Contudo, ela pode ser modificada para operar com Certificados Digitais emitidos por Autoridades de Certificação conhecidas. Esta extensão seria essencial para prover a segurança necessária em um ambiente aberto como a Internet.

De forma geral, a segurança implementada no protótipo segue o esquema da Figura B.2. Como descrito, o estabelecimento de um canal seguro garante a ocultação das informações transmitidas, o que inclui a senha de acesso ao sistema na etapa de *login*.

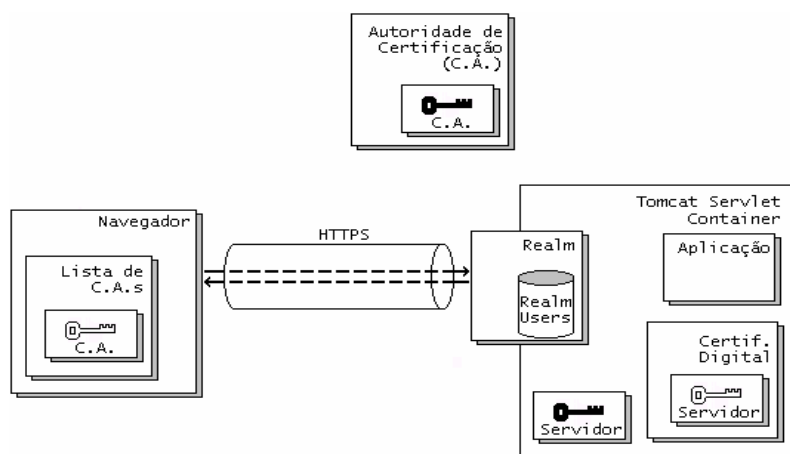


Figura B.2 – Segurança no protótipo.

B.3 Questões no Nível PLPM/URM

Atualmente, a comunicação neste nível é realizada através de linhas telefônicas com a utilização de *modems* para o estabelecimento de ligações ponto a ponto entre os dispositivos participantes. Com a evolução da tecnologia, a tendência é conectar os equipamentos através de redes IP, possibilitando a comunicação através de topologias mais flexíveis e eficientes. Neste sentido, diversos esforços de padronização da tecnologia da Internet estão sendo desenvolvidos por entidades tais como o EPRI e o IEC, os quais deverão ser incorporados em novos produtos e sistemas do setor elétrico [Stal 99]. Em relação à segurança, sendo o contexto similar ao da Internet, a solução dos problemas passa pela utilização das técnicas já descritas anteriormente. Contudo, algumas adaptações podem ser consideradas para o ambiente específico, tal como a utilização de *Intranets*, ou da tecnologia *Virtual Private Networks* (VPN), ou mesmo a criação de redes fisicamente independentes [Oman 00].

Tendo em vista o exposto, nesta seção a discussão será restrita aos sistemas de medição atuais, como os encontrados no acervo das EEE. Nos sistemas em questão, a segurança é obtida através da utilização de senhas para o acesso à interface do usuário, tanto no equipamento remoto (medidor) quanto no computador hospedeiro do *software* utilizado para os ajustes. Desta forma, utilizando-se as recomendações adequadas para a confecção e utilização de senhas estes modos de acesso estão razoavelmente protegidos. O elo fraco do sistema é a própria linha telefônica, usada para comunicação com os dispositivos, que permite a intrusos capturarem a senha de acesso através de um processo simples de monitoração do tráfego da informação. Obtendo a senha, um intruso tendo acesso ao *software*, ou aos formatos das mensagens, e conhecendo o número telefônico associado ao dispositivo remoto, poderia obter o controle do mesmo e alterar os ajustes do medidor conforme desejasse. As soluções para este problema partem do pressuposto de ser inviável alterar substancialmente os equipamentos existentes. A seguir elas são listadas e discutidas:

- Contratar com a empresa telefônica restrições de acesso ao número telefônico associado à URM, que só poderia ser acessada a partir de um conjunto de números pré-determinados. Essa técnica oferece proteção equivalente a uma linha dedicada (*leased line*) virtual, mas a mesma pode ser violada se o intruso se conectasse diretamente à linha física de comunicação.
- Utilização de *modem* com capacidade de *callback*, e restrição do número a ser chamado no retorno no próprio *modem*. Ainda assim, um intruso tecnicamente competente poderia simular o comportamento do sistema telefônico e ganhar acesso ao dispositivo remoto. Além disso, essa solução, bem como anterior, impediria que dispositivos remotos sejam acessados de números arbitrários, o que poderia ser necessário em situações imprevistas, e.g., caso de manutenção de emergência feita em campo.
- Utilização de *modem* com capacidade de criptografia (ver, p.ex., <http://csrc.nist.gov/cryptval/140-1/140sp/140sp061.pdf>), o que parece ser uma solução com custo viável para o problema, oferecendo segurança similar à oferecida pelas técnicas de criptografia usadas na Internet, sem requerer alteração interna nos equipamentos existentes. Contudo, neste caso, teria que ser usado um esquema especial (portador humano) para a distribuição da chave usada para comunicação com o dispositivo remoto.

Vale lembrar que a última solução também poderia ser implantada através de modificações no *software* das partes envolvidas de modo a adicionar o suporte à criptografia. Esse suporte pode ser facilmente implementado no lado do sistema PLPM desenvolvido. Contudo, no lado das unidades remotas, além de requerer a utilização de recursos ociosos nem sempre nelas existentes, esta opção requer conhecimentos da arquitetura interna das mesmas, informação de difícil fornecimento pelos fabricantes de equipamentos.

B.4 Conclusão

Na arquitetura do protótipo identificamos dois níveis onde foi necessária a utilização de mecanismos de segurança. No nível IU-PLPM, a segurança é obtida através do controle de acesso à interface por senha e através de canais seguros de comunicação implementados com o mecanismo SSL. No nível PLPM-URM foi constatado que os sistemas legados menosprezam a questão da segurança, a qual é obtida exclusivamente via controle de acesso baseado em senhas. Desta forma, a solução a ser adotada para esses sistemas depende do nível de segurança extra que se desejar obter, podendo usar as técnicas enumeradas na seção 3, individualmente ou em conjunto (deve ser lembrado que o conceito de Associação deverá ser implementado nos sistemas da nova geração acima desse nível de protocolo). A nosso ver, a utilização de *modems* com capacidade de criptografia proveria uma solução razoável para o problema, sem requerer modificações substanciais nos sistemas já existentes. Com a evolução da tecnologia, a provisão de segurança no nível PLPM-URM poderia ser obtida utilizando-se mecanismos similares aos usados no contexto da Internet, como os descritos no nível IU-PLPM. Contudo, algumas modificações teriam que ser feitas para adaptar as técnicas para utilização em dispositivos remotos sem grande capacidade de processamento.

Vale lembrar que a questão da segurança é de suma importância nos futuros sistemas onde praticamente todos os dispositivos estarão conectados e poderão se comunicar através de uma rede de comunicação. A solução dessa questão requer esforços visando a padronização dos mecanismos e protocolos necessários para permitir a interoperabilidade dos equipamentos de diferentes fabricantes. Neste sentido, torna-se importante acompanhar o desenvolvimento dos padrões nesta área de interesse, dentre eles o UCA (*Utility Communication Architecture*) [UCA 97a] e o COSEM [DLMS 02b]. Seria também benéfica a organização de grupos de trabalho reunindo técnicos da agência reguladora, dos fabricantes de equipamento, das universidades e das concessionárias do setor visando uma participação nacional neste esforço.