UNIVERSIDADE DO VALE DO RIO DOS SINOS
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
PROGRAMA INTERDISCIPLINAR DE PÓS-GRADUAÇÃO
EM COMPUTAÇÃO APLICADA
ÁREA DE CONCENTRAÇÃO MODELAGEM E SIMULAÇÃO

**Ricardo de Andrade Kratz**

# Fábrica de adequação de conteúdo de ensino para Objetos de Aprendizagem Reutilizáveis (RLOs) respeitando a norma SCORM

São Leopoldo
2006

UNIVERSIDADE DO VALE DO RIO DOS SINOS
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
PROGRAMA INTERDISCIPLINAR DE PÓS-GRADUAÇÃO
EM COMPUTAÇÃO APLICADA
ÁREA DE CONCENTRAÇÃO MODELAGEM E SIMULAÇÃO

**Ricardo de Andrade Kratz**

# Fábrica de adequação de conteúdo de ensino para Objetos de Aprendizagem Reutilizáveis (RLOs) respeitando a norma SCORM

Dissertação a ser avaliada como requisito parcial para a obtenção do título de Mestre em Computação Aplicada

Orientador:
Prof. Dr. Sérgio Crespo Coelho da Silva Pinto

São Leopoldo
2006

Ficha catalográfica elaborada pela Biblioteca da
Universidade do Vale do Rio dos Sinos

K91f Kratz, Ricardo de Andrade
Fábrica de adequação de conteúdo de ensino para Objetos de Aprendizagem Reutilizáveis (RLOs) respeitando a norma SCORM / por Ricardo de Andrade Kratz. - 2006
125 f. : il. ; 29cm.

Dissertação (mestrado) - Universidade do Vale do Rio dos Sinos, Programa de Pós-Graduação em Computação Aplicada, 2006.
"Orientação: Prof. Dr. Sérgio Crespo Coelho da Silva Pinto, Ciências Exatas e Tecnológicas".

1. Engenharia de software. 2. Educação à distância. 3. Objetos de aprendizagem. I. Título..

CDU 004.41:37.018.43

Catalogação na Publicação:
Bibliotecária Eliete Mari Doncato Brasil - CRB 10/1184

Ricardo de Andrade Kratz

# "FÁBRICA DE ADEQUAÇÃO DE CONTEÚDO DE ENSINO PARA OBJETOS DE APRENDIZAGEM REUTILIZÁVEIS (RLOS) RESPEITANDO A NORMA SCORM"

Dissertação apresentada à Universidade do Vale do Rio dos Sinos (UNISINOS) como requisito parcial para obtenção do título de **mestre** em Computação Aplicada

Aprovado em janeiro de 2006

BANCA EXAMINADORA

---

Prof. Dr. Sérgio Crespo Coelho da Silva Pinto

---

Prof. Dr. Arthur Tórgo Gómez

---

Prof[a]. Dr[a]. Lúcia Giraffa

# Dedicatória

À minha esposa (Viviane) pelo carinho, amor e suporte nesta jornada.
Aos meus pais (Christina e Fernando) pelo exemplo de vida,
às minhas irmãs e amigos pelo afeto e companheirismo.

# Agradecimentos

Gostaria de ser capaz de agradecer a todas as pessoas que me ajudaram e contribuíram para o meu crescimento pessoal e para a realização desta difícil etapa. Porém, diante da impossibilidade de relacioná-los aqui, gostaria de prestar meus agradecimentos àqueles que, desde o início, confiaram no meu esforço e dedicação, e que de variadas formas contribuíram para a realização deste trabalho.

Agradeço em especial a minha esposa Viviane que abdicou de várias coisas para me acompanhar com muito amor e dedicação. Também agradeço aos meus Pais, Fernando e Christina, pelos exemplos de honradez e perseverança. Sem eles não seria possível ter sequer iniciado esta jornada.

Às minhas irmãs, Lúcia e Marta, que mesmo de longe sempre acreditaram no meu potencial.

Ao Programa Interdisciplinar de Pós-Graduação em Computação Aplicada da Unisinos, através de seus professores, colaboradores e colegas de mestrado, pela oportunidade única de aprendizado.

Ao professor Sérgio Crespo Coelho da Silva Pinto, por ser meu orientador, pela sua amizade e inestimável auxilio no desenvolvimento desta dissertação.

Aos professores Arthur Tórgo Gómez e Lúcia Giraffa por participarem da minha banca. A professora Rosa Maria Vicari que não pode participar da banca examinadora.

À CAPES pelo apoio financeiro para a realização desta pesquisa.

"Feliz é aquele que transfere o que sabe e aprende o que ensina."
(Cora Coralina)

# Resumo

Para promover a reutilização de conteúdos de aprendizagem é necessário promover a normalização desses conteúdos para que possam funcionar corretamente em qualquer sistema de *e-learning*. A normalização permite: uma fácil reutilização; a portabilidade dos conteúdos criados; a padronização dos processos de criação; e a gestão dos conteúdos de aprendizagem. Porém, normalizar um ambiente de Educação a Distância já existente não é uma tarefa trivial e necessita de uma adequação tanto dos conteúdos de aprendizagem quanto do sistema de *e-learning*, pois toda a arquitetura é alterada, sendo necessário separar o conteúdo da estrutura. Esta pesquisa propõe e implementa uma arquitetura denominada Fábrica de Adequação capaz de promover a interoperabilidade dos conteúdos educacionais e promover a adequação de recursos educacionais para Objetos de Aprendizagem Reutilizáveis, respeitando a norma SCORM, utilizando de tecnologias de Agentes de *Software*, Meta-dado (SCORM) e *Web Services*, os quais dispõem de mecanismo de autonomia, interação, invocação, acesso e reuso de serviços. A solução utiliza Agentes de Software baseados em *Web Services*, os quais permitem uma eficiente descrição de serviços através de suas tecnologias padrões como a WSDL, SOAP e UDDI (todas baseadas em XML). Trata-se do desenvolvimento de um conjunto de serviços para a manipulação e adequação de recursos educacionais. Como forma de testar e validar os serviços desenvolvidos, foram realizados estudos de casos, onde foi possível avaliar a Fábrica de Adequação e comprovar que a arquitetura é capaz de realizar o objetivo proposto.

**Palavras-chaves:** Engenharia de *Software*, Educação à Distância, Objetos de Aprendizagem, Agentes de *Software* e *Web Services*.

# Abstract

To promote learning object reusability it is necessary to provide the normalization of its contents so that they can work appropriately in any e-learning system. The normalization allows easy reusability, portability of created contents, standardization of processes of creation, and management of learning objects. However, to normalize an existing E-learning environment it is not a trivial task and it requires adequacy either of the learning contents and the e-learning system, once the whole architecture is modified, being necessary though to separate the content from the structure. This research proposes and implements an architecture called Adequacy Factory which is capable of promoting the interoperability of educational contents and the adequacy of educational resources for Learning Object Reusability, respecting the SCORM norm by using technologies such as Software Agents, Meta-Data (SCORM), and Web Services which make use of autonomy mechanism, interaction, invocation, access, and reuse of services. The solution uses Software Agents based on Web Services which allow efficient service description through its standard technologies such as the WSDL, SOAP and UDDI (all based on XML). This is about the development of a set of services for the manipulation and adequacy of educational resources. As a way to test and validate the developed services, study cases have been carried out, and from them it was possible to evaluate the Adequacy Factory and to prove that the architecture is capable of reaching the proposed goal.

**Keywords:** Software Engineer, e-learning, Learning Objects, Software Agents and Web Services.

# Lista de Figuras


2.1 Modelo de Ciclo para criação de um curso a distância via *Web*, adaptado do modelo de Margaret Discoll (Discoll, 1998). . . . . 11

3.1 Classificação de Agentes, adaptado de (Franklin and Graesser, 1996). . . . . 17

4.1 Arquitetura dos *Web Services* (Ferris and Farrell, 2003). . . . . 21
4.2 Camadas conceituais de *Web Services* (Rheinheimer, 2004). . . . . 22
4.3 Estrutura básica de um documento XML. . . . . 23
4.4 Exemplo de uma hierarquia em XML. . . . . 24
4.5 Atributo em um documento XML. . . . . 24
4.6 Documento WSDL HelloService.wsdl (Souza, 2004). . . . . 25
4.7 UDDI utilizado para descobrir um *Web Service* (Souza, 2004). . . . . 27

5.1 Árvore completa do LOM V1.0 (LOM, 2005). . . . . 31
5.2 Objeto de Aprendizagem (Bettio, 2003). . . . . 33
5.3 Divisão de um Objeto de Aprendizagem (Bettio, 2003). . . . . 33
5.4 Exemplo de Metadado de um Objeto de Aprendizagem (Bettio, 2003). . . . . 35
5.5 *Package Interchange File* (PIF) (IMS, 2005). . . . . 35
5.6 Exemplo de SCO, adaptado de (SCORM, 2005). . . . . 37
5.7 Diagrama da seção *Organizations* (SCORM, 2005). . . . . 38
5.8 *Run-Time Environment* de uma plataforma de *e-learning*, adaptado de (SCORM, 2005). . . . . 39
5.9 Grafo de referência entre as especificações do IEEE, IMS, AICC, SCORM, ARIADNE e Dublin Core, adaptado de (Pereira et al., 2003). . . . . 43

6.1 Arquitetura de solução para *e-learning* da Cisco (Kelly and Barritt, 1998). . . . . 45
6.2 Estrutura Interna do RLO (Barritt, 2001). . . . . 46
6.3 Estrutura interna de um RIO (Barritt, 2001). . . . . 47
6.4 Modelo da Cisco para níveis de agregação em conteúdos de aprendizado, adaptado de (Kelly and Barritt, 1998). . . . . 47
6.5 Estrutura dinâmica de navegação, adaptado de (Barbeira and Santos, 2002). . . . . 49
6.6 Arquitetura e Modelo de Dados (Barbeira and Santos, 2002). . . . . 50
6.7 Telas e funcionamento do SCORMisizer. . . . . 53
6.8 Telas e funcionamento do SCORAggregator. . . . . 54
6.9 Processo de produção dos módulos e objetos de aprendizagem (Nascimento and Morgado, 2004). . . . . 56

7.1 Ambientes de EAD tradicionais. . . . . 60
7.2 Ambientes de EAD com repositório compartilhado. . . . . 61
7.3 Arquitetura geral da Fábrica de Adequação (ACEAS) (Kratz et al., 2005). . . . . 62
7.4 Serviços da Fábrica de Adequação. . . . . 62
7.5 Visão geral da Arquitetura (vista pelo LMS). . . . . 63

7.6 Diagrama de Caso de Uso. . . . . 64
7.7 Diagrama de Seqüência Geral da Fábrica. . . . . 64
7.8 Diagrama de Classes. . . . . 65
7.9 Classe Proxy de acesso ao *Web service*. . . . . 66
7.10 Tela principal da Fábrica de Adequação. . . . . 67
7.11 Tela cadastro de novos repositórios. . . . . 68
7.12 Adequação de Recuso Educacional (URL). . . . . 69
7.13 Importação de Arquivo de Metadado. . . . . 69
7.14 Editor de Metadado (XML *Parse*). . . . . 70
7.15 Editor de Estruturação de Agregação. . . . . 71
7.16 Manifesto de um *Content Aggregation*. . . . . 72
7.17 Objeto de Aprendizagem com mecanismo de comunicação. . . . . 73
7.18 Código do "sco.htm". . . . . 73
7.19 Pacote Scorm (sco.zip). . . . . 74
7.20 Estrutura da etiqueta da Fábrica de Adequação. . . . . 75
7.21 Descrição do *Web method* criaEtiqueta. . . . . 75
7.22 Fonte do *Web method* criaEtiqueta. . . . . 76
7.23 Mecanismo de Pesquisa de Objeto de Aprendizagem. . . . . 77
7.24 Resultado da pesquisa por objeto de Aprendizagem. . . . . 78

8.1 Tipos de Mídias do TelEduc. . . . . 82
8.2 Tipos de Mídias do AVA. . . . . 82
8.3 Interface do LMS. . . . . 83
8.4 API *Adapter*. . . . . 84
8.5 Curso de Álgebra do Sistema de *E-learning*. . . . . 85
8.6 Capítulo 01 do Curso de Álgebra. . . . . 85
8.7 Exemplo de teste do *Conformance Test Suite* em um Objeto de Aprendizagem. . . . 87

A.1 Exemplo de metadado de um SCO "metadado.xml". . . . . 101

B.1 Exemplo de manifesto de um SCO "imsmanifest.xml". . . . . 105

C.1 Código de comunicação do SCO com o LMS "SCORMGenericLogic.js". . . . . 108

# Lista de Tabelas

2.1 Principais tecnologias utilizadas em EAD. . . . . 9

3.1 Conjunto de formas e características de agentes (OMG, 2000) (FIPA, 2005). . . . . 16

5.1 Código dos erros mais freqüentes (SCORM, 2005) . . . . . 40
5.2 CMI Data Model (SCORM, 2005) (CMI, 2005) . . . . . 41

6.1 Tabela comparativa. . . . . 57

8.1 Resultados da validação dos Objetos de Aprendizagem. . . . . 87

# Lista de Abreviaturas

ACEAS – Ambiente de Construção de Etiquetas de Adaptação SCORM
ADL – *Advanced Distributed Learning*
ADOdb – Base de Dados
AICC – *Aviation Industry CBT Committee* API – *Application Program Interface*
ARIADNE – *Alliance of Remote Institute of Electrical and Distribution Networks for Europe*
AVA – Ambiente Virtual de Aprendizagem
B2B – *Business to Business*
CBT – *Computer Based Training*
CMI – *Computer Managed Instructions*
CEdMA – *Computer Education Management Association*
DOM – *Document Object Model*
DTD – *Document Type Definition*
EAD – Educação à Distância
EDUCOM – *Educação com Computadores*
EIA – *Enterprize Application Integration*
EML – *Educational Modelling Language*
FIPA – *Foundation for Intelligent Physical Agents*
FIPA-ACL – *FIPA - Agent Communication Language*
FTP – *File Transfer Protocol*
FTS – Formação Tecnológica e de Serviços
GD – General *Design*
HTML – *Hyper Text Markup Language*
HTTP – *HyperText Transfer Protocol*
IE – Informática na Educação
IEEE – *Institute of Electrical and Electronics Engineers*
ILSG – *Internet Learning Solutions Group*
IIS – *Internet Information Services*
IMS – *Instructional Management Systems*
IVEN – *International Virtual Education Network*
LCMS – *Learning Content Management Systems*
LMS – *Learning Management System*
LO – *Learning Objects* (Objetos de Aprendizagem)
LOM – *Learning Object Metadata*
LTSC – *Learning Technology Standards Committee*
Nied – Núcleo de Informática Aplicada à Educação
NLII – *National Learning Infrastructure Initiative*
MEC – Ministério da Educação
PIF – *Package Interchange File*
PT Inovação – Portugal Telecom Inovação
QoS – Qualidade de Serviço
QTI – *Question & Test Interoperability*

RLO – *Reusable Learning Objects* (Objetos de Aprendizagem Reutilizáveis)
RIO – Objetos de Informação Reutilizáveis
RIVED – Rede Internacional Virtual de Educação
RPC – *Remote Procedure Call*
SCO – *Sharable Content Objects*
SCORM – *Sharable Content Object Reference Model*
SEB – Secretaria de Educação Básica
SEED – Secretaria de Educação à Distância
SGML – *Standard General Markup Language*
SIAC – Sistemas de Instrução assistida por Computador
SOAP – *Simple Object Access Protocol*
STI – Sistemas Tutores Inteligentes
TI – Tecnologia da Informação
UDDI – *Universal Distribution Discovery and Integration*
UNICAMP – Universidade Estadual de Campinas
UNISINOS – Universidade do Vale do Rio dos Sinos
UML – *Unified Modeling Language*
URL – *Unified Modelling Language*
W3C – *World Wide Web Consortium*
WSDL – *Web Services Description Language*
WWW – *World Wide Web*
XML – *Extensible Markup Language*
XSD – *XML Schema Definition*

# Sumário

**Lista de Figuras** **i**

**Lista de Tabelas** **iii**

**Lista de Abreviaturas** **iv**

**1 Introdução** **1**
1.1 Motivação e Relevância . . . 2
1.2 Objetivos . . . 3
1.3 Contribuições do Trabalho . . . 4
1.4 Organização do Trabalho . . . 4

**2 Educação a Distância** **5**
2.1 Introdução . . . 5
2.2 Definições . . . 6
2.3 História da Educação a Distância . . . 7
2.4 Tecnologias para Educação a Distância . . . 8
2.5 Ensino a Distância através da *Web* . . . 10
2.5.1 O desenvolvimento de cursos . . . 10
2.5.2 *E-learning* . . . 12
2.6 Conclusão . . . 13

**3 Agentes *Software*** **14**
3.1 O que são Agentes de *Software*? . . . 14
3.2 Taxonomia dos Agentes . . . 15
3.3 *Foundation for Intelligent Physical Agents* (FIPA) . . . 17
3.3.1 Modelo de Gerenciamento de Agentes . . . 18
3.3.2 Ontologias para comunicação entre agentes . . . 18
3.4 Agentes na interoperabilidade de Conteúdos Educacionais . . . 18
3.5 Conclusão . . . 19

**4 *Web Services*** **20**
4.1 Introdução . . . 20
4.2 O que são *Web Services*? . . . 20
4.2.1 Arquitetura de *Web Services* . . . 21
4.2.2 Camadas dos *Web Services* . . . 22
4.3 Tecnologias Padrões Utilizadas nas Arquiteturas *Web Services* . . . 22
4.3.1 *eXtensible Markup Language* (XML) . . . 22
4.3.1.1 Declaração . . . 23
4.3.1.2 Elementos . . . 23
4.3.1.3 Comentários . . . 23

4.3.1.4 Hierarquia . . . 23
4.3.1.5 Atributos . . . 23
4.3.1.6 Elementos Vazios . . . 24
4.3.2 *Web Service Description Language* (WSDL) . . . 24
4.3.3 *Simple Object Access Protocol* (SOAP) . . . 25
4.3.4 *Universal Description, Discovery and Integration* (UDDI) . . . 26
4.4 Conclusão . . . 27

**5 Objetos de Aprendizagem Reutilizáveis (RLOs) 28**
5.1 Introdução . . . 28
5.2 O que são Objetos de Aprendizagem? . . . 29
5.3 Metadados Educacionais . . . 29
5.3.1 *Learning Object Metadata* (LOM) . . . 30
5.3.1.1 Objetivos . . . 30
5.3.1.2 Estrutura Básica dos Metadados . . . 30
5.3.2 IMS *Global Learning Consortium, Inc* . . . 32
5.3.2.1 Armazenamento . . . 34
5.3.2.2 IMS *Content Packing* . . . 35
5.3.3 *Sharable Content Object Reference Model* (SCORM) . . . 36
5.3.3.1 *Content Aggregation Model* . . . 36
5.3.3.2 *Run-Time Environment* . . . 38
5.3.3.3 *Data Model* . . . 40
5.4 Conclusão . . . 42

**6 Trabalhos Relacionados 44**
6.1 Introdução . . . 44
6.2 A Proposta da Cisco Systems para Objetos de Aprendizagem Reutilizáveis (RLOs) . . . 45
6.2.1 Criação de Conteúdo de Aprendizagem . . . 46
6.3 Portugal Telecom Inovação e o FORMARE . . . 47
6.3.1 Adequação da Plataforma Formare . . . 48
6.3.1.1 Disponibilizar conteúdos . . . 49
6.3.1.2 Modelo de Dados . . . 49
6.3.1.3 *Application Program Interface* (API) . . . 50
6.3.1.4 Gestão de Conteúdo . . . 50
6.3.1.5 Relatórios . . . 51
6.4 Click2learn ToolBook . . . 51
6.4.1 Neuron da plataforma ToolBook . . . 51
6.4.2 Ferramenta do ToolBook . . . 52
6.4.2.1 SCORMisizer . . . 52
6.4.2.2 SCORMAggregator . . . 52
6.5 Rede Internacional Virtual de Educação (RIVED) . . . 54
6.5.1 Especificação . . . 55
6.5.2 Fábrica Virtual . . . 55
6.5.2.1 Desenvolvimento dos Módulos de Aprendizagem . . . 56
6.6 Conclusão e considerações sobre os trabalhos relacionados . . . 57

**7 Fábrica de Adequação: ambiente de adequação de recurso educacional para Objetos de Aprendizagem SCORM 59**
7.1 Introdução . . . 59
7.2 Arquitetura de um Repositório de Conteúdo de Aprendizagem Tradicional . . . 60
7.3 Visão geral da arquitetura proposta ACEAS (Ambiente de Construção de Etiquetas de Adaptação SCORM) . . . 61

7.3.1 Diagramas da Fábrica de Adequação . . . 63
7.4 Camada Interface . . . 66
7.5 Camada de Serviços . . . 67
7.5.1 SCORM *Single Item Package* . . . 67
7.5.1.1 Metadado . . . 68
7.5.1.2 Manifesto . . . 70
7.5.2 SCORM *Package Aggregator* . . . 70
7.5.2.1 Metadado . . . 71
7.5.2.2 Manifesto . . . 72
7.5.3 Mecanismo de Comunicação . . . 72
7.5.3.1 Funcionamento . . . 73
7.5.4 Pacote SCORM . . . 74
7.6 Camada de Persistência . . . 74
7.7 Mecanismo de Pesquisa de Objetos de Aprendizagem . . . 76
7.8 Múltiplos Contextos . . . 77
7.9 Conclusão . . . 78

**8 Estudo de Caso** **80**
8.1 Introdução . . . 80
8.2 Primeiro Estudo de Caso . . . 81
8.3 Segundo Estudo de Caso . . . 82
8.4 Terceiro Estudo de Caso . . . 83
8.4.1 Protótipo de LMS . . . 83
8.4.2 Acessando um Objeto de Aprendizagem da Fábrica de Adequação . . . 84
8.5 ADL SCORM *Conformance Test Suite* . . . 86
8.5.1 *Sharable Content Object (SCO) Run-Time Environment (RTE) Conformance Utility Test* . . . 86
8.5.2 *Metadata Conformance Utility Test* . . . 86
8.5.3 *Manifest Utility Test* . . . 86
8.5.4 Resultados . . . 86
8.6 Conclusão . . . 88

**9 Conclusões e Trabalhos Futuros** **89**
9.1 Trabalhos Futuros . . . 90

**Referências** **92**

***Web*-Referências** **97**

**A Arquivo exemplo de Metadado "metadata.xml"** **99**

**B Arquivo exemplo de Manifesto "imsmanifest.xml"** **102**

**C Código Genérico JavaScrit de Comunicação "SCORMGenericLogic.js"** **106**

# CAPÍTULO 1

# Introdução

Este capítulo apresenta uma visão geral do trabalho a ser realizado, destacando-se a motivação para a realização do mesmo, bem como o objetivo a ser alcançado.

O ensino a distância (EAD) através da *Web* tem se tornado cada vez mais parte integrante do contexto mundial de ensino. Atualmente, governos, universidades e empresas estão em busca de recursos e de conhecimentos para utilizar esta nova forma de ensino com o objetivo de minimizar problemas de custos e de distância entre os participantes do evento educacional.

Porém, adequar os meios tradicionais de ensino ao ensino a distância é uma tarefa árdua e tem sido tema de estudo para muitos pesquisadores da área educacional e de tecnologia. A tecnologia caminha a passos largos, mostrando que ferramentas e recursos não irão faltar para o desenvolvimento de cursos a distância. Em conseqüência, o uso desenfreado e desorganizado da tecnologia pode ter um resultado pedagógico catastrófico (Gonzáles 2000) (Zaina, 2002).

Neste intuito, no início da década de 90, vários grupos isolados iniciaram trabalhos e estudos com o conceito de normalizar a apresentação de conteúdos educacionais na *Web*. O Grupo *Learning Object Metadata* (LOM) (WG12, 2005) do Instituto Nacional de Ciência e Tecnologia e a CEdMA (*Computer Education Management Association*) (CEdMA, 2005) investiram seus esforços no estudo de objetos de aprendizagem (LOs), incluindo modularidade, orientação para armazenamento em banco de dados e o uso de elementos de marcação que evoluíram para os metadados.

Não existe um consenso sobre a definição de objetos de aprendizagem (LO). Para o *Learning Technology Standard Comitee* (LTSC) (IEEE, 2005), LOs podem ser considerados como qualquer entidade, digital ou não, que possa ser usada, reusada e referenciada durante alguma forma de aprendizagem suportada por tecnologia. Ainda podem ser definidos como "pequenos" recursos educacionais que podem ser organizados, utilizados e reutilizados em cursos a distância assistidos pela *Web* (Jacobsen, 2004).

Para a reutilização, um objeto de aprendizagem precisa ser modular, interoperável e ter a capacidade de ser descoberto. Muitas organizações e grupos de pesquisas vêm trabalhando no sentido de alcançar essas características e também no sentido de aprimorar a eficiência e eficácia desses objetos. A maioria dos esforços concentra-se na definição de padrões. Iniciativas como o *Learning Technology Standard Comitee* (LTSC) do *Institute of Electrical and Electronics Engineers* (IEEE) [IEEE, 2005], a *Alliance of Remote Institute of Electrical and Distribution Networks for Europe* (ARIADNE)

(ARIADNE, 2005), o *Instructional Management Systems* (IMS) *Global Learning Consortium* (IMS, 2005), a Canadiam Core (CanCore, 2005) e a *Advanced Distributed Learning initiative* (ADL, 2005) têm contribuído significativamente na definição de padrões para objetos de aprendizagem e de metadados educacionais.

No entanto, mesmo com o surgimento de estudos de padrões, diversos ambientes de educação a distância (AVA, TelEduc, e-ProInfo, Blackboard, entre outros) (AVA,2005)(TelEduc, 2005)(eProInfo, 2005)(Blackboard, 2005) não adotam nenhum padrão ou especificação para apresentação de seus recursos educacionais. Segundo Barbeira (Barbeira, 2002), adequar um ambiente de *e-learning* para algum padrão é uma tarefa árdua e envolve uma equipe multidisciplinar. Desta maneira, os recursos educacionais existentes neste ambiente não podem ser reutilizados de forma sistemática.

Assim, este trabalho busca aproveitar da tecnologia de Agentes de *Software* e *Web Services* para promover a interoperabilidade de conteúdos educacionais e a adequação de recursos educacionais já existentes. O resultado dessa busca é uma abordagem que auxilia a normalização (adequação) de recursos educacionais para que possam ser reapoveitados em novos cursos *on-line*.

## *1.1* Motivação e Relevância

De acordo com Brusilovsky (Brusilovsky, 1998), as vantagens de um sistema de ensino baseado na "*Web*" são claras: independência de sala de aula e independência de plataforma. Um aplicativo instalado e suportado em um único local pode ser usado por milhares de estudantes por todo o mundo, bastando que eles tenham um computador com acesso à Internet. A criação de sistemas de ensino que englobem funções inteligentes, como por exemplo a de um tutor que utiliza uma base de conhecimento do aluno para tomada de decisões, de avaliação, de qual conteúdo aplicar, e outras decisões relativas a um professor, se torna um trabalho extremamente complexo.

A constante evolução da Engenharia de Computação e das tecnologias para a *Web* permitiu a expansão das alternativas para suportar o processo de ensino e aprendizagem, principalmente a educação assistida pelo computador (educação a distância), a qual tomou um novo rumo fazendo uso dessas tecnologias. Os ambientes educacionais para *Web* (*e-learning*) necessitam se adaptar às tecnologias emergentes, de forma a fazer o melhor uso e alcançar o máximo de benefícios possíveis.

Normalmente, a produção e o desenvolvimento dos conteúdos educacionais digitais somados com a definição de um curso é um processo dispendioso (Rosember, 2002), podendo-se concluir que a adaptação de um curso tradicional para o meio de Educação a Distância é uma tarefa complexa e onerosa. Porém, não reapoveitar esses conteúdos colabora ainda mais para a elevação dos gastos que envolvem essa produção (Kratz et al., 2005).

Logo, existe na comunidade de Informática na Educação (IE) o objetivo de facilitar o desenvolvimento de cursos *Web*, ou aplicações voltadas à educação a distância que podem trazer inúmeros benefícios, como por exemplo o reuso. A reutilização e a interoperabilidade de serviços e recursos educacionais para a construção de novas aplicações/soluções reduz de forma acentuada os custos de implementação.

Albuquerque (Albuquerque, 2005) ressalta a utilização de conteúdos didáticos estruturados e mais organizados. Para isso, o autor defende a adoção de especificações no aprendizado na *web* (*e-learning*). Especificações são guias e sugestões para implementação de um determinado produto, elas são ferramentas utilizadas por desenvolvedores para tomar decisões.

Neste contexto, surgem as especificações internacionais voltadas a *e-learning*, dentre elas se destaca a norma *Sharable Content Object Reference Model* (SCORM), que é considerada a especificação mais completa do mercado (Gomes et al., 2004). Sendo um dos principais padrões utilizados em *e-learning* no mundo (Barbeira and Santos, 2002)(Pereira et al., 2003)(Kratz et al., 2005), o SCORM pretende unificar as normas para a criação de conteúdos educativos, baseando-se para isso, no trabalho já realizado quer pelo Aviation Industry CBT Committee (AICC) (AICC, 2005), quer pelo *Instructional Management Systems* (IMS) (IMS, 2005).

Segundo Aroyo e Kommers (Aroyo and Kommers, 1999), agentes podem influenciar diferentes aspectos em sistemas educacionais. Eles fornecem novos paradigmas educacionais, apóiam teorias e podem ser de muito auxílio, tanto para aprendizes quanto para docentes na tarefa de aprendizado auxiliado por computador. A aplicação de agentes no setor educacional se dá principalmente na forma de assistentes pessoais, guias para usuários, sistemas alternativos de ajuda, arquiteturas dinâmicas de sistemas distribuídos, mediadores homem-máquina e, neste trabalho, na interoperabilidade de conteúdos educacionais.

Outra tecnologia que ganha destaque mundialmente são os *Web Services*, que são componentes de *software* que independem de implementação ou de plataforma e podem ser descritos, publicados e invocados sobre uma rede através de mensagens padrão *eXtensible Markup Language* (XML) (Newcomer, 2002)(Rheinheimer, 2004). Os *Web Services* apresentam uma estrutura que possibilita a comunicação entre aplicações, onde o serviço pode ser invocado remotamente, ou ser utilizado para compor um novo serviço, e são fundamentais na interoperabilidade de sistemas heterogêneos independentemente da plataforma que executam ou que foram desenvolvidos.

Com base no que foi explicado, o que motiva esta pesquisa é a busca por uma solução que permita a interoperabilidade de conteúdos educacionais e o reaproveitamento de recursos educacionais já existentes em novas soluções.

## 1.2 Objetivos

O objetivo geral deste trabalho é promover a interoperabilidade dos conteúdos educacionais, usando tecnologias de Agentes de *Software* e *Web Services* para desenvolver um modelo que promova a adequação de recursos educacionais para Objetos de Aprendizagem Reutilizáveis respeitando a norma SCORM.

Na implementação desse modelo será dado o nome de "Fábrica de Adequação", que permite normalizar recursos educacionais de forma semi-automática.

A fim de alcançar o objetivo geral, foram definidos os seguintes objetivos específicos:

- Estudar aspectos da área de Agentes de *Software*, mostrando seus conceitos e características;
- Estudar o funcionamento e aspectos dos *Web Services*, pesquisando seus conceitos, arquitetura e protocolos;
- Estudar os principais aspectos da área de Objetos de Aprendizagem Reutilizáveis, mostrando seus conceitos, funcionalidades e tecnologias;

- Propor e implementar um modelo de adequação de recursos educacionais que suporte a interoperabilidade de objetos de aprendizagem. Esse modelo deve levar em consideração as especificações definidas pela norma SCORM;
- Implementar uma arquitetura básica em Microsoft .NET ASP que permita a adequação e a interoperabilidade de recursos educacionais;
- Avaliar e validar as propostas efetuadas.

## 1.3 Contribuições do Trabalho

Ao final deste trabalho, formulamos uma proposta de uma abordagem que se espera que auxilie a interoperabilidade de objetos de aprendizagem através de um modelo de Agentes *Web Services*.

Buscou-se através de revisão bibliográfica a base conceitual do trabalho. E, modelo-se e implementou-se uma solução de adequação de recursos computacionais para a norma SCORM.

Outro ponto disponibilizado é a criação de múltiplos contextos para o mesmo objeto de aprendizagem, pontencializando a sua reusabilidade e evolução pedagógica. Desta maneira, busca-se auxiliar a implementação de ambientes de aprendizagem a partir da possibilidade de se reaproveitar recursos educacionais já existentes.

## 1.4 Organização do Trabalho

No Capítulo 2, são apresentados conceitos de Educação a Distância (EAD), sua história e as tecnologias utilizadas. E, enfatiza-se a educação assistida pela *Web* (*e-learning*).

O Capítulo 3 apresenta aspectos da área de Agentes de *Software*, destacando a sua aplicabilidade na interoperabilidade de conteúdos educacionais.

O Capítulo 4 mostra os *Web Services* como solução para integração de aplicações/sistemas.

No Capítulo 5, foram analisados alguns trabalhos relacionados que colaboraram com o que foi definido nesta pesquisa.

O Capitulo 6 apresenta o modelo de adequação e interoperabildade de conteúdos educacionais, bem como, sua implementação (Fábrica de Adequação) explicando-se a arquitetura proposta e respectivas funcionalidades.

O Capítulo 7 apresenta os estudos de casos realizados para validar e avaliar tanto o modelo quanto a implementação.

Por fim, no Capítulo 8, são apresentadas as considerações finais, tais como conclusões, dificuldades encontradas e trabalhos futuros.

# CAPÍTULO 2

# Educação a Distância

Nos últimos anos, a Educação a Distância (EAD) tem se tornado um tópico de destaque na educação. Vem crescendo o número de conferências, simpósios, projetos e publicações relacionadas ao assunto. Com o advento da *Web*, a prática da EAD sofreu grandes mudanças nas últimas décadas. Tem-se buscado, além de inovações tecnológicas, inovações pedagógicas que tornem este processo de ensino/aprendizagem algo mais do que uma simples cópia do modelo tradicional (Lucena e Fuks, 2000)(Simonson et al., 2000).
Neste Capítulo, são apresentados conceitos de Educação a Distância, sua história, as tecnologias utilizadas e um pouco mais sobre EAD baseado na *Web*, auxiliando no entendimento deste trabalho.

## *2.1* Introdução

Com a proliferação dos computadores somada à popularização da Internet, incrementou-se a forma e a maneira de comunicação entre seus usuários, formando um leque de novos serviços e aplicações, através dessa rede heterogênea de ambientes, sistemas e plataformas. Vários serviços e sistemas de comércio eletrônico, bancários, entretenimento, negócios, aplicações educacionais, serviços governamentais, dentre outros, estão em franca expansão. A Informática não surgiu para revolucionar apenas a comunicação entre empresa, governos e pessoas, mas também para auxiliar o professor no processo de ensino/aprendizagem de forma eficaz e efetiva (Kratz et al., 2005).

A utilização da tecnologia como apoio ao ensino presencial tem sido amplamente adotada principalmente dentro do contexto da *Web*. Adequar os meios tradicionais de ensino ao ensino *on-line* é uma tarefa árdua e tem sido tema de estudo para vários pesquisadores da área educacional e de tecnologia. Existe um grande desenvolvimento de infra-estruturas tecnológicas que servem como suporte ao ensino a distância, porém, o uso indiscriminado e desorganizado da tecnologia pode ter um resultado catastrófico; é preciso associar a tecnologia às metodologias pedagógicas, adaptando-as à realidade virtual, para assim se obter sucesso no processo de aprendizagem (McCormick, 2004)(Zaina et al., 2004).

Este Capítulo apresenta uma visão geral sobre educação a distância, principalmente com o advento da Internet e o surgimento dos Objetos de Aprendizagem.

## 2.2 Definições

Os primeiros conceitos de Educação a Distância (EAD) estabelecem uma comparação com a Educação Presencial onde o professor é a figura principal do processo de ensino/aprendizagem. Mesmo que esse comportamento não seja de todo incorreto, existem entendimentos mais amplos e modernos da EAD. Dentro desses entendimentos e definições de Educação a Distância encontrados na literatura, se destacaram as seguintes definições:

- Definição de Otto Peters (1973)
  - "Educação/Ensino a Distância é um método racional de partilhar conhecimento, habilidades e atitudes, através da aplicação da divisão do trabalho e de princípios organizacionais, tanto quanto pelo uso extensivo de meios de comunicação, especialmente para o propósito de reproduzir materiais técnicos de alta qualidade, os quais tornam possível instruir um grande número de estudantes ao mesmo tempo, enquanto esses materiais durarem. É uma forma industrializada de ensinar e aprender" (Nunes, 1994).
- Börje Holmberg (1977)
  - "O termo 'educação a distância' esconde-se sob várias formas de estudo, nos vários níveis que não estão sob a contínua e imediata supervisão de tutores presentes com seus alunos, nas salas de leitura ou no mesmo local. A educação a distância beneficia-se do planejamento, direção e instrução da organização do ensino" (Holmberg, 1985).
- Desmond Keegan (1980)
  - "O ensino a distância é o tipo de método de instrução em que as condutas docentes acontecem à parte das discentes, de tal maneira que a comunicação entre o professor e o aluno se possa realizar mediante textos impressos, por meios eletrônicos, mecânicos ou por outras técnicas" (Keegan, 1991).
- Jorge Goñi e Hugo Fuks (2002)
  - "A definição mais abrangente de Educação a Distância inclui todas as formas de ensino/aprendizagem nas quais aprendizes e/ou instrutores se comunicam de algum modo, além de reuniões presenciais em sala de aula e interagem" (Goñi and Fuks, 2002).
- Eva Kaplan-Leiserson (2005)
  - "Educação a Distância é o processo de aprendizagem sem contato pessoal regular com um instrutor ou com outros colegas em sala de aula presencial, enquanto ensino a distância significa o processo de ensino sem contato pessoal regular em regime presencial e onde o estudante e o professor estão separados pelo tempo, localização e/ou ambos. A Educação alcança locais remotos via meios assíncronos e síncronos, incluindo correspondência escrita, texto, gráficos, áudios e vídeo-áudios, videocassete, CD-ROM, educação *on-line*, vídeo conferência, TV interativa" (Kaplan-Leiserson, 2005).

Segundo a definição de Moore (1996), "a Educação a Distância é uma aprendizagem planejada que normalmente ocorre em um local diferente do tradicional e como resultado requer projeto de curso e técnicas instrucionais especiais, métodos especiais de comunicação eletrônica ou outra tecnologia, bem como sistemas organizacionais e administrativos especiais" (Moore and Kearsley, 1996).

Para o autor, há seis elementos que são essenciais para uma definição clara de EAD:

1. separação entre estudante (aprendiz) e professor (instrutor);
2. influência de uma organização ou instituição educacional, principalmente no planejamento e preparação dos materiais de aprendizado;
3. uso de meios técnicos - mídia;
4. providências para comunicação em duas vias;
5. possibilidade de seminários (presenciais) ocasionais;
6. participação na forma mais industrial de educação.

Ao se observar as várias definições da literatura, é possível concluir que a Educação a Distância possui um grande número de características que levam a uma mesma idéia: a transposição da barreira de distância e de tempo presentes entre a relação do professor (instrutor) com o aluno (aprendiz) através do uso de mecanismos de comunicação.

Assim, a educação a distância (EAD) se caracteriza pelo processo de ensino-aprendizagem, no qual o professor e o aprendiz estão separados geograficamente e/ou temporalmente durante a maior parte da aprendizagem. Essa separação gera a necessidade de mediação - utilização de uma ou mais mídias - para a comunicação entre os atores no cenário educacional. A EAD moderna baseia-se na interatividade, onde o aprendiz tem que interagir com seu educador e outros aprendizes, independentemente da distância. Essa interação pode ser individual - entre o aluno e a informação disponibilizada em livros, programas de computador e laboratórios experimentais - e social com um educador e/ou outros aprendizes. A interação social é síncrona - em tempo real, ao vivo e conversacional - ou assíncrona - retardada ou adiantada para antes ou depois da interação individual com o conteúdo educacional que é objeto de estudo.

## 2.3 História da Educação a Distância

A Educação a Distância é mais antiga do que parece, uns dos primeiros cursos foi ministrado por correspondência na Inglaterra no final do século XIX, sendo Sir Issac Pitman, da empresa *Correspondence Colleges*, o seu principal criador. Esse primeiro curso a distância teve a intenção de fornecer formação a um grupo de pessoas que, por motivos geográficos, de comodidade, econômicos ou sociais, não podiam freqüentar aos tradicionais locais de ensino (Sá, 2000).

No ano de 1881, foi criado por William Rainey Harper, primeiro reitor e fundador da Universidade de Chicago, um curso de Hebreu por correspondência. Outras iniciativas foram os cursos a distância oferecidos pela *Queen's College* do Canadá, em 1889, registrando uma grande procura,

graças principalmente ao seu baixo custo e às grandes distâncias que separam os centros urbanos daquele país (Loyolla and Prates, 1999).

No Brasil, a EAD surgiu no século XX com escolas por correspondência de caráter profissionalizante, utilizando de revista e jornais em seus cursos técnicos (radiotécnica, datilografia e outros) (Rheinheimer, 2002).

De 1960 a 1970, o principal mecanismo de EAD foi o rádio, onde estações de rádio ministravam cursos à população. A partir da década de 1970, se popularizou o uso da televisão, a chamada "Televisão Educativa" para preparação aos exames supletivos.

Nos anos 80, o computador começou a ser usado como tecnologia educacional. O Governo Federal criou a Comissão Especial de Educação, com o objetivo de definir normas e diretrizes para regulamentar a área de informação. O EDUCOM (Educação com Computadores) foi um projeto piloto que visava o desenvolvimento de pesquisas e disseminação do uso dos computadores no processo de ensino.

Na década de 90, com a transmissão via satélite em canal aberto, possibilitou-se a disseminação de curso de aperfeiçoamento e capacitação de docentes do ensino fundamental. Na metade da década, foi criada a Secretaria de Educação a Distância do Ministério da Educação (SEED/MEC). No final da década, a Internet teve seu maior desenvolvimento, possibilitando um encurtamento das distâncias entre os seus participantes e permitindo uma melhor interatividade, disseminando os programas e cursos de EAD (Rheinheimer, 2002).

O uso da Informática na EAD pode oferecer melhorias na qualidade de ensino em larga escala e ao mesmo tempo possibilitar a redução dos custos. A interatividade pode ser incorporada neste processo com o emprego da hipermídia interativa, a qual agrega textos, imagens, vídeos, animações, fotos e som, criando um ambiente de ensino novo, atraente e multissensorial (Salvador, 1995).

Efetivamente com o *World Wide Web* (WWW), se atinge o ponto mais alto no desenvolvimento do ensino a distância. Esse ambiente visual com amplas potencialidades possibilita à utilização de inúmeras tecnologias de apoio à educação a distância.

## *2.4* Tecnologias para Educação a Distância

A Educação a Distância iniciou como ensino profissionalizante, evoluindo das novas tecnologias de comunicação e informação. A Tabela 2.1 mostra a evolução no tempo das principais tecnologias (Rheinheimer, 2002)(Simonson et al., 2000).

**Tabela 2.1:** Principais tecnologias utilizadas em EAD.

| **Tecnologia** | **Descrição** |
|---|---|
| Material impresso | Considerado um meio instrucional básico, trata-se da criação de material educacional impresso de um determinado assunto destinado a um público específico. |
| Correspondência | Mecanismo de distribuição e interação de conteúdo e exercícios educacionais que utilizam os correios como meio. Normalmente são enviadas lições junto com formulários de avaliação do tema estudado. Após a realização do estudo, o aluno envia, pelo correio, os formulários e exercícios resolvidos. |
| Telefone | Trata-se de uma tecnologia que permite comunicação bilateral, possibilitando uma comunicação entre os participantes, com alto grau de controle tanto do emissor quanto do receptor. São aulas dadas via telefone, permitindo a comunicação direta e instantânea entre os participantes. |
| Fax | Tecnologia que utiliza troca de mensagens escritas por meio do Fax. A grande vantagem é possibilitar o registro escrito do que é transmitido, com custos mais reduzidos do que o uso do telefone. |
| Rádio | Primeira tecnologia que assume um papel ativo no processo de instrução. Seu funcionamento é feito pela transmissão de informações por estímulo auditivo. Possibilita uma relação unidirecional entre o emissor (rádio) e o receptor. Os alunos recebem programas de rádio didáticos, enviados por um professor de uma estação emissora. |
| Televisão | Tecnologia que introduz recursos visuais e movimentos. Trata-se de um mecanismo mais sofisticado onde as matérias são preparadas para a transmissão aos alunos. A transmissão pode ser feita por circuitos abertos ou fechados. Um bom exemplo do uso desta tecnologia são os programas criados pela Fundação Roberto Marinho: o Telecurso 2000 e o canal Futura. Trata-se de um meio unidirecional, devendo ser utilizado com outras tecnologias complementares. |
| Áudio-cassete | Meio complementar de ensino que utiliza recursos auditivos onde são criadas fitas de cassete com conteúdos educacionais. Possibilita ao usuário controlar o tipo e a duração das mensagens, bem como repeti-las quando necessário. Trata-se de uma tecnologia relativamente barata, podendo ser usada individualmente ou em grupo. |

| Tecnologia | Descrição |
|---|---|
| Videocassete | Tecnologia que utiliza recursos visuais e auditivos onde são preparados aulas e lições em fitas de videocassete. Apresenta várias vantagens como: qualidade das imagens e sons, possibilidade de repetições instantâneas e o uso do material individualmente ou em grupos. |
| Satélite | Tecnologia que utiliza comunicação via satélite para transmissão de imagens de televisão. Possibilita a criação de vários canais e cursos de cunho educacionais. Outra novidade que essa tecnologia proporciona é a realização de videoconferências, distribuição de vídeos promocionais, educativos ou de treinamento, acesso à Internet, entre outros. |
| Computador | Uma das tecnologias mais complexas e com mais recursos para permitir a Educação a Distância, permitindo tanto a criação de conteúdos áudio-visuais, como de sistema de ensino que possibilita uma interatividade do aluno com o conteúdo. Inicialmente, eram usadas produções de multimídia na forma de CD-ROM. Evoluiu para modelos como: Sistemas de Instrução assistida por Computador (SIAC) e Sistemas Tutores Inteligentes (STI). Esses modelos estabelecem um paradigma na interação entre o estudante e o sistema de ensino. |
| Internet | Trata-se de uma tecnologia auxiliar aos computadores que permite criar um meio de comunicação entre os alunos e as instituições de ensino. Conhecida como WWW (*World Wide Web*), permite a transmissão de todo tipo de mídia de ponto a ponto do planeta. Possibilitou o surgimento da multimídia interativa. Possibilita o surgimento de comunidades virtuais de ensino, bem como permitiu uma revolução da forma que a EAD era conhecida. |

## 2.5 Ensino a Distância através da *Web*

Nesta Seção, será feito um estudo sobre o ensino a distância através da *Web*, mostrando sua importância no contexto mundial de ensino e as etapas que envolvem o desenvolvimento deste ambiente de ensino.

### 2.5.1 O desenvolvimento de cursos

A melhor maneira do desenvolvimento de curso através da *Web* é um ponto que tem sido amplamente discutido. A maneira de disponibilizar curso deve ser adaptada aos objetivos principais do curso em si, adequando aos recursos tecnologias disponíveis para atender às necessidades do seu criador.

A usabilidade[1] é a definição do perfil dos usuários que utilizarão um ambiente de EAD, neste caso, o sistema educacional tem o objetivo de facilitar a aprendizagem (Nielsen, 1999). Mas, a Internet apresenta mais desafios do que apenas definir os perfis dos usuários; é necessário definir como serão repassadas as informações e conteúdos. A Figura 2.1 mostra o ciclo de desenvolvimento de um curso a distância através da *Web*.

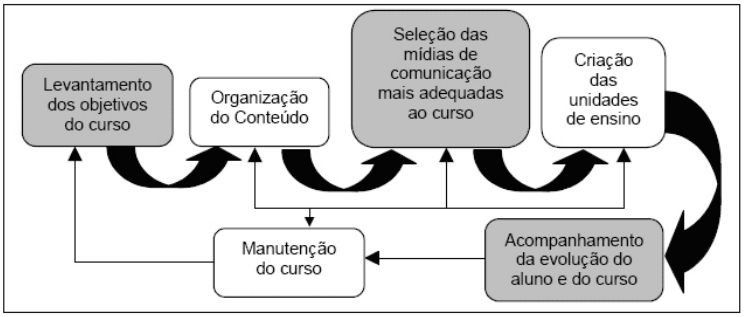


**Figura 2.1:** Modelo de Ciclo para criação de um curso a distância via *Web*, adaptado do modelo de Margaret Discoll (Discoll, 1998).

Segundo o modelo de Margaret Discoll (Discoll, 1998), a criação de um curso a distância segue as seguintes etapas:

- Levantamento dos objetivos do curso
  - Determinar qual o objetivo que o curso pretende alcançar, ou seja, que resultado final o criador do curso (professor) pretende dos seus usuários (alunos);
  - Definido o objetivo, o professor deve pensar no propósito do curso: curso profissional, de atualização, de formação, dentre outros. É importante observar que cada propósito apresenta um enfoque diferente, permitindo ao seu criador escolher a melhor forma de apresentar e disponibilizar os conteúdos educacionais;
- Organização do curso
  - Definir a organização do curso. Nesta etapa é que o criador do curso (professor) define qual a melhor maneira de estruturar o conteúdo e de como acompanhar a evolução do usuário (aluno) dentro do curso, sendo necessária para a próxima etapa a definição dos tipos de mídias que serão utilizadas;
- Seleção das mídias de comunicação mais adequadas ao curso
  - Nesta etapa, é definida qual a melhor maneira de apresentar o conteúdo aos usuários finais (alunos). Na escolha da mídia, deve ser levado em conta vários pontos: a quantidade e qualidade da banda disponível, os navegadores (*browsers*), o sistema operacional do usuário, dentre outros fatores;

[1] Área do conhecimento que estuda o desenvolvimento de interface de sistema como o objetivo de minimizar o tempo necessário de aprendizagem para uso do sistema.

  - Uma solução normalmente utilizada em curso via *Web* é o uso de hipertexto e hipermídia, os quais possibilitam a criação de uma estrutura hierárquica entre as unidades de ensino, permitindo ao aluno escolher o caminho que deseja seguir durante o processo de aprendizagem. Porém, outros tipos de mídias podem ser usadas, podendo ser de vários formatos e características como: texto, imagens, som, páginas HTML (*Hyper Text Markup Language*), animações, vídeos e qualquer formato *Web*;

- Criação das unidades de ensino

  - Desenvolver os conteúdos educacionais, normalmente desenvolvidos no formato HTML, porém, é possível usar recursos adicionais de vídeo, som e imagens para criar um material de melhor qualidade;
  - No intuito de permitir uma maior interatividade com os alunos, se faz necessário o uso de linguagens de programação (javascript, asp, php, dentre outras) que permitem um controle dos objetos educacionais. Essa prática requer um conhecimento de Informática sofisticado, o que nem sempre se constitui em uma realidade entre os professores, docentes e profissionais da educação;

- Acompanhamento da evolução do aluno e do curso

  - Desenvolver a página principal do curso, mecanismo de segurança, de acesso ao curso, de comunicação e correspondência, além de formas de acompanhamento dos desempenhos dos alunos e do curso em si. Novamente se faz necessário um conhecimento avançado em programação para desenvolver e disponibilizar esses mecanismos;
  - As formas mais utilizadas para permitir a interação entre os alunos e os professores são: fórum, lista de discussão, sala de bate-papo (*chat*), dentre outros. Esses mecanismos possibilitam um melhor aproveitamento do curso, uma troca de experiências e conhecimentos, além de permitir a identificação de falhas na aprendizagem e corrigi-las. Outra forma de avaliar os alunos e os cursos é a realização de testes, avaliações e exercícios;

- Manutenção do curso

  - Corrigir falhas e oferecer novos recursos aos alunos. Manter um curso de EAD sempre em constante evolução é de vital importância para o seu sucesso e sobrevivência no mercado;
  - Mudanças e correções são inevitáveis, algumas delas são simples, porém, mudanças estruturais podem comprometer todo um trabalho de desenvolvimento. Ter ferramenta que auxilie a realização dessas manutenções é tão importante como quaisquer outras.

### 2.5.2 *E-learning*

*E-learning* é uma estratégia habilitada para a *Web*, que oferece um amplo conjunto de soluções, que estimulam o crescimento de comunidades de conhecimento/aprendizado, para aumentar o desempenho das instituições. Usando a tecnologia de *e-learning*, a aprendizagem não ocorre somente a partir de instrução digital, mas, principalmente, a partir do acesso a informações bem estruturadas, correspondendo a uma forma inovadora para o aprendizado (Rosenberg, 2002).

Segundo Carol Fallon and Sharon Brown (Fallon and Brown, 2002), *e-learning* é qualquer aprendizagem, treinamento ou educação que é auxiliado pelo uso de tecnologias (bem conhecidas) promovidas por computadores, baseada em tecnologias da Internet.

*E-learning* pode ser classificada em duas categorias: assíncronas e síncronas. Este trabalho se baseia em ambientes de *e-learning* assíncronos.

*E-learning* Síncronos usa um modelo de ensino que simula uma classe de aula usando tecnologias da Internet. É chamada de "síncrono" porque todos os participantes estão presentes ao mesmo tempo. Existe uma variedade de *softwares* projetados especificamente para este propósito, oferecendo conteúdos em tempo real, interações em sala de bate-papo, vídeo-conferências, *whiteboards* (Fallon and Brown, 2002).

*E-learning* Assíncronos é a versão *Web* para a versão do *Computer Based Training* (CBT), que é tipicamente oferecida por CD-ROM ou em redes de computadores locais. No caso de *e-learning*, os conteúdos educacionais são entregues por um Servidor *Web* e entregues por demanda em estações de trabalho. O curso pode ser acessado na casa dos alunos, em qualquer momento e podem ser revistos quantas vezes o aluno julgar necessário (Fallon and Brown, 2002).

Um dos avanços tecnológicos mais promissores na área de *e-learning* é a criação de soluções baseadas em objetos educacionais, onde cada um corresponde ao menor bloco de instrução ou informação, elaborada de forma independente (Rosenberg, 2002).

Utilizando objetos educacionais, é possível criar bibliotecas de conhecimento, permitindo que diferentes cursos utilizem um mesmo objeto. Outra vantagem está na personalização do aprendizado a partir da seleção e configuração daqueles objetos que atendam e auxiliem o aprendiz na construção e apropriação do próprio saber. Um objeto educacional poderá ser: um programa em Java, um arquivo de som, uma imagem, um filme ou um programa de simulação, dentre outros.

Já existe a catalogação de objetos educacionais usando vários padrões como o *Instructional Management Systems* (IMS) (IMS, 2005), *Learning Object Metadata* (LOM) (LOM, 2005) e *Sharable Content Object Reference Model* (SCORM) (SCORM, 2005) (padrões que serão vistos posteriormente) que incluem especificações para a interoperabilidade, permitindo reuso. Essa padronização contribuirá na produção de recursos educacionais.

Usando a tecnologia de objetos educacionais, pode-se obter uma significante redução nos custos de produção de cursos a distância. É neste sentido, que o presente trabalho busca facilitar a adequação de recursos educacionais existentes para essa tecnologia.

## 2.6 Conclusão

No processo de ensino e aprendizagem, a modalidade de educação a distância (EAD) tem tido papel fundamental nestes últimos anos e vem se desenvolvendo conforme o barateamento das diversas formas de mídia e a sua correspondente aceitação no mercado (vídeo, fita cassete, programas de rádio e televisão, CD, computador, etc).

Com a popularização da Internet, esta passou a ser um dos veículos mais importantes para o ensino. Muitos projetos de EAD passaram a considerar a Internet como meio preferido de realização. Como foi visto neste Capítulo, os objetos de aprendizagem são um dos avanços tecnológicos mais promissores na área de *e-learning*, trata-se de um novo paradigma que redefine a estruturação dos ambientes de EAD.

# CAPÍTULO 3

# Agentes *Software*

Atualmente, os Agentes de *Software* são uma das áreas de maior atividade em pesquisa e desenvolvimento dentro da Ciência da Computação (Gomes, 2005). Uma das características dos agentes é a autonomia (ou semi-autonomia), ou seja, a capacidade de agir e de interagir com outros agentes de acordo com um objetivo.
Ainda não existe um conceito unânime e uniforme para o termo "agente". Porém, a idéia de agente teve origem com John McCarthy, na metade da década de 50, e o termo foi criado por Oliver G. Selfridge poucos anos antes, quando ambos estavam no *Massachusetts Institute of Technology*. Eles tinham em mente um sistema que, quando definido um objetivo, pudesse resolver os detalhes das operações computacionais apropriadas e pudesse pedir e receber conselhos, oferecidos em termos humanos, quando estivesse em dificuldade. Um agente poderia ser um "*soft robot*" vivendo e realizando suas tarefas dentro do mundo dos computadores (Bradshaw, 1997).
Este Capítulo apresenta uma conceituação dos agentes de *software* mostrando suas funcionalidades e sua aplicabilidade, principalmente na interoperabilidade de conteúdos educacionais.

## *3.1* O que são Agentes de *Software*?

Segundo os pesquisadores Frankin e Graesser (Ferris and Farrell, 2003), um agente é um sistema que é capaz de sentir e agir no ambiente do qual faz parte, seguindo sua própria agenda e agindo de forma a atendê-la.

Outra definição de agente de *software*, segundo Bradshaw (Bradshaw, 1997), é "...Uma entidade de *software* que funciona de maneira contínua e autônoma em um ambiente particular... com a habilidade de conduzir atividades de forma flexível e inteligente, sensível a mudanças no ambiente... Em condições favoráveis, um agente que trabalhe continuamente... estaria capacitado a aprender a partir de suas experiências. Além disso, é esperado que um agente que habita um determinado ambiente juntamente com outros agentes seja capaz de se comunicar com eles, e, talvez, se deslocar de uma local a outro ao fazer isso".

Para Gomes (Gomes, 2005), "os agentes se caracterizam como uma extensão dos objetos, porque proporcionam uma abstração maior do que estes, atuando como sistemas baseados em conhecimento, pois trabalham com informações".

O *Green Paper* (OMG, 2000) do *Object Management Group* (OMG) apresenta uma relação de características que um agente de *software* pode apresentar, das quais se destacam:

- autonomia: onde um agente autônomo tem a capacidade de agir sem interferências externas de usuários ou de outros sistemas;
- interatividade: é quando os agentes são capazes de se comunicar com outros agentes e com o ambiente onde se encontram;
- mobilidade: é a capacidade do agente de se mover de um ambiente para outro;
- pró-atividade: um agente pró-ativo é aquele capaz de tomar decisões, sem ter sido solicitado para tal, mas que atenda algum objetivo ou meta do mesmo, ou seja, o agente não reage apenas ao ambiente, ele tem propósitos e age orientado a metas;
- racionalidade: os agentes racionais são capazes de escolher uma ação a ser executada baseados em suas metas (agenda) e no seu conhecimento de que uma ação particular vai levá-los mais próximos à conclusão de seu plano de metas.

## 3.2 Taxonomia dos Agentes

Os agentes podem ser classificados conforme os tipos existentes e suas diferentes classes através da apresentação de uma taxonomia[1] de agentes. Segundo Nwana (Nwana, 1996), há sete categorias de agentes: colaborativos, de interface, móveis, de informação, reativos, híbridos e "inteligentes" (*smart agents*).

Um agente pode ser classificado como um agente híbrido, ou seja, aquele que é constituído da combinação de dois ou mais tipos de agente. Por exemplo, um agente que implementa tanto o comportamento reativo, quanto o deliberativo, poderia apresentar respostas mais rápidas às mudanças no seu ambiente, aplicando o comportamento reativo, enquanto outros problemas orientados a metas e que não exijam tal agilidade seriam tratados pelo comportamento deliberativo.

Os agentes também podem ser enquadrados em um dos seguintes tipos:

- agentes de informação: são responsáveis pelo gerenciamento, manipulação e ordenação de informação de diferentes fontes. Os agentes são caracterizados pelo que fazem e não pelo que são;
- agentes móveis: são processos de *software* que possuem a capacidade de se locomover em uma rede de computadores, mas a capacidade de se movimentar por si só não caracteriza um agente;
- agentes reativos: são os agentes que possuem a capacidade de reagir a estímulos do ambiente em que se encontram, mas não possuem modelos internos do mesmo;

[1] Segundo do Dicionário Webster: O termo taxonomia refere-se ao estudo dos princípios de classificação científica - http://www.m-w.com/dictionary.htm (visitado em 07/02/2005).

- agentes de interface: são capazes de interagir e colaborar com o usuário em um ambiente de trabalho (execução), desempenhando o papel de assistentes pessoais;
- agentes colaborativos: enfatizam a autonomia e a cooperação com outros agentes para executar tarefas para seus donos, sendo capazes de colaborar com outros agentes para realizar determinadas tarefas;
- agentes inteligentes (*smart agents*): na verdade este tipo de agente não é definido pelo autor, mas representam apenas o desejo atual dos pesquisadores. São agentes capazes de apresentar todas as características citadas relativas à autonomia, aprendizagem e cooperação.

Segundo o *Object Management Group* (OMG, 2000) e a *Foundation for Intelligent Physical Agents* (FIPA) (FIPA, 2005), os agentes encontrados nos sistemas de Tecnologia da Informação (TI) possuem requisitos especiais, que precisam executar como *software*, robôs ou uma combinação deste. Desta maneira, a OMG define um conjunto de formas e características de agentes em TI que são apresentadas na Tabela 3.1.

**Tabela 3.1:** Conjunto de formas e características de agentes (OMG, 2000) (FIPA, 2005).

| **Tipo** | **Descrição** |
|---|---|
| Agentes de *Software* | São entidades de *software* autônomas que interagem com o seu ambiente, ou seja, são agentes implementados por *software*. |
| Agentes Autônomos | A autonomia tanto para o OMG quanto para a FIPA é uma característica necessária em um agente, que possibilita ao mesmo agir sem interferências externas de usuários ou de outros sistemas. |
| Agentes de Interação | A interação entre os agentes, o seu ambiente e entre outras entidades também pode ser medida em escalas. A interação entre os agentes pode ocorrer por chamadas simples de métodos ou através da modificação e percepção de alterações no ambiente. |
| Agentes Adaptativos | Pode-se considerar um agente como adaptativo se ele for capaz de reagir a estímulos de outros agentes ou do ambiente. |
| Agentes Móveis | Os agentes móveis são aqueles capazes de se mover de um ambiente para outro. A mobilidade para o OMG, apesar de ser uma característica importante, não é uma condição necessária para definir um agente. |
| Agentes de Coordenação | Trata-se dos agentes responsáveis pela coordenação de um grupo (sociedade) de agentes, similares às sociedades humanas em que existe a necessidade de coordenar as ações individuais para que algum objetivo seja alcançado. |
| Agentes Inteligentes | Um agente inteligente é aquele que tem o seu estado formalizado por conhecimento, sejam crenças, metas, desejos, intenções, planos etc. e é capaz de agir no ambiente baseado neste conhecimento. |
| Agentes Encapsuladores | Os agentes encapsuladores são uma forma de permitir a interação entre agentes e sistemas que não foram desenvolvidos utilizando este paradigma. |

Os agentes de *software* podem ser visualizadas em diferentes áreas de aplicação. Porém, os agentes (Tabela 3.1) facilitadores, gerentes ou com função de *brokering*, são na verdade papéis que estão sendo desempenhados em um determinado momento ou sistema (OMG, 2000).

Outra classificação de agentes é, conforme Franklin e Graesser (Franklin and Graesser, 1996), segundo suas propriedades, levando em consideração uma única propriedade ou, então, a combinação de algumas delas.

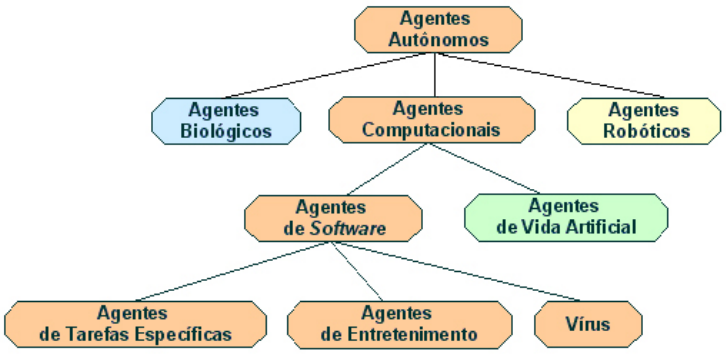


**Figura 3.1:** Classificação de Agentes, adaptado de (Franklin and Graesser, 1996).

Na Figura 3.1, Franklin e Graesser mostram a taxonomia de agentes como aquela que cobre muitos dos exemplos encontrados na literatura. Nessa classificação inicial, os autores sugerem que os Agentes sejam categorizados por estruturas de controle, ambientes (banco de dados, sistema de arquivos, rede, Internet), linguagem em que são escritos e aplicações (Franklin and Graesser, 1996).

## 3.3 *Foundation for Intelligent Physical Agents* (FIPA)

A *Foundation for Intelligent Physical Agents* é uma fundação internacional sem fins lucrativos voltada para o desenvolvimento de padrões (especificações) que possibilitem a implementação e interoperabilidade de agentes (FIPA, 2005). As especificações produzidas pela FIPA são divididas em:

- Aplicações de Sistemas Multiagentes: os quais são exemplos de domínio, definição de ontologias e descrição de serviços onde é possível utilizar agentes FIPA;
- Arquitetura Abstrata: trabalha com entidades abstratas fundamentais para a construção de serviços e de um ambiente de agentes. Ela define quais são as características arquiteturais que um sistema de agentes ou de multiagentes deve ter;
- Comunicação entre Agentes: nas especificações da FIPA a comunicação foi dividia em:
  - Atos comunicativos: os quais definem os atos de comunicação padrão da linguagem FIPA-ACL (FIPA - *Agent Communication Language*);
  - Linguagem de Interação: responsável pela definição da seqüência lógica de troca de mensagens, especificando atos comunicativos para cada uma delas;

  - Linguagem de Conteúdo: define as diferentes formas de representar (codificar) a informação passada através de uma mensagem ACL;

- Gerenciamento de Agentes: define o ambiente onde os agentes FIPA existem e operam. Ele é responsável por estabelecer o modelo lógico de referência para a criação, registro, localização, comunicação, migração e desativação dos agentes;

- Transporte de Mensagens entre Agentes: especifica a forma como as mensagens são transportadas e representadas através de diferentes protocolos de rede.

### 3.3.1 Modelo de Gerenciamento de Agentes

O modelo de gerenciamento de agentes cria o ambiente onde um agente existe (reside) e opera (executa). Ele estabelece um modelo lógico de referência para criação, registro, localização, comunicação, migração e desativação de agentes na plataforma FIPA.

### 3.3.2 Ontologias para comunicação entre agentes

O Modelo de comunicação FIPA é baseado na hipótese de que dois agentes, que vão se comunicar (conversar), conheçam e apliquem o mesmo significado para todos os símbolos utilizados no conteúdo das mensagens. Isso é realizado através da asseguração de que os agentes conheçam a mesma ontologia.

## *3.4* Agentes na interoperabilidade de Conteúdos Educacionais

Agentes que trabalham com busca e tratamento de informação são uma opção para a implementação de serviços apoiados por padrões de interoperabilidade de produtos e conteúdos educacionais (Lucena et al., 2001), podendo ser relacionados como:

- agentes responsáveis pela procura por informação que saibam como localizar os dados, como obter acesso ou mesmo negociar os modos de acesso, e que tenham informações sobre a fonte;

- agentes responsáveis pela filtragem de informação que possuam modelos de seus usuários, propiciando uma busca mais efetiva e personalizada;

- agentes mediadores, são aqueles que auxiliam a negociação de diferentes agentes interessados em obter informações ou conteúdos educacionais;

- agentes responsáveis por monitorar dados em diferentes fontes de informação, sejam elas servidoras de conteúdos educacionais ou não, apoiando de alguma forma a mobilidade dos aprendizes, professores ou projetistas/analistas.

Com o aumento da oferta da quantidade de dados disponíveis, a utilização de agentes para busca de informações já vem sendo considerada uma solução para a interoperabilidade de conteúdos educacionais e pode impulsionar a utilização destes (Klusch, 1999).

Segundo Aroyo e Kommers (Aroyo and Kommers, 1999), agentes podem influenciar diferentes aspectos em sistemas educacionais. Eles fornecem novos paradigmas educacionais, apóiam teorias, e

podem ser de muito auxílio tanto a aprendizes quanto a docentes na tarefa de aprendizado auxiliado por computador. A aplicação de agentes no setor educacional se dá principalmente na forma de assistentes pessoais, guias para usuários, sistemas alternativos de ajuda, arquiteturas dinâmicas de sistemas distribuídos, mediadores humano-sistema e na interoperabilidade de conteúdos educacionais.

Em particular, esses serviços precisam atender a uma série de requisitos, como por exemplo, personalização, adaptação, suporte para mobilidade do usuário, suporte para usuários enquanto lidam com novas tecnologias, efetividade, suporte a informação, entre outros. Agentes surgem para fornecer soluções para estes requisitos de um modo mais eficiente em comparação com outras tecnologias existentes (Aroyo and Kommers, 1999).

## 3.5 Conclusão

O termo agente tem sido empregado de diversas formas no universo da computação, cujas definições nem sempre se convergem. No entanto, há um consenso em relação às características fundamentais de um agente. Assim, segundo Ferris e Farrel (Ferris and Farrell, 2003), as características de reatividade, autonomia, pró-atividade e continuidade temporal constituem elementos básicos para a definição de um agente.

No contexto da educação, os Agentes de *Software* têm sido cada vez mais explorados desde a recuperação de informação no ambiente *Web*. Neste trabalho, o conceito de agente é inserido como assistentes que, executando um conjunto mínimo de tarefas individualmente, possibilitam a interoperabilidade de conteúdos educacionais.

Esta pesquisa utiliza a visão de agentes segundo Bradshaw (Bradshaw, 1997) e Lucena (Lucena et al., 2001), onde os agentes possuem uma determinada autonomia, mas podem operar segundo solicitações.

# CAPÍTULO 4

# *Web Services*

*Web Services* utilizam tecnologias baseadas em *eXtensible Markup Language* (XML), que apresentam uma estrutura arquitetural que possibilita a comunicação entre aplicações, onde é possível a utilização de serviços sem que haja a necessidade de saber qual plataforma ou linguagem de programação foram utilizadas na sua construção.
Este Capítulo apresenta uma conceituação de *Web Services*, bem como os componentes de uma estrutura básica (Registro de Serviços, Solicitante de Serviço e Provedor de Serviço). Também estarão sendo apresentadas algumas tecnologias padrões para construção de *Web Services*, tais como: *eXtensible Markup Language* (XML) (Santos, 2002), *Web Services Description Language* (WSDL) (Newcomer, 2002), *Simple Object Access Protocol* (SOAP) (Walsh, 2002) e *Universal Distribution Discovery and Integration* (UDDI) (Rheinheimer, 2004). Essas tecnologias permitem invocar um serviço sem a necessidade de conhecer a plataforma ou linguagem de programação usada na sua construção.

## *4.1* Introdução

Os *Web Services* surgiram no intuito de substituírem as tradicionais estratégias "*Enterprise Application Integration*" (EAI), cuja tradução seria "Integração de Aplicações Corporativa". Inicialmente, eram usados exclusivamente para designar a tentativa de uma empresa de interligar suas aplicações internas de negócios, de forma que os dados poderiam ser compartilhados. Recentemente, a sua aplicabilidade foi expandida para, também, englobar a união de dados e processos com parceiros comerciais - *Business to Business* (B2B) (Jagiello and Júnior, 2003). Eles apresentam uma estrutura que possibilita a comunicação entre aplicações, onde o serviço pode ser invocado remotamente, ou ser utilizado para compor um novo serviço.

## *4.2* O que são *Web Services*?

Um *Web Service* corresponde a qualquer serviço disponível através da Internet que utiliza um sistema padrão (*standard*) de mensagens baseado em XML, e que não é dependente de qualquer sistema operacional ou linguagem de programação. Um *Web Service* descreve a si próprio através de

uma gramática XML e pode ser descoberto através de um mecanismo de pesquisa simples (Cerami, 2002).

*Web Services* são componentes de *software* que independem de implementação ou de plataforma e podem ser descritos, publicados e invocados sobre uma rede através de mensagens padrão XML (Newcomer, 2002)(Rheinheimer, 2004).

A *World Wide Web Consortium* (W3C) define um *Web Service* como: "uma aplicação de *software* identificado por um *Unified Modelling Language* (URL), cujas interfaces e ligações são capazes de ser definidas, descritas e descobertas como artefatos XML. Um serviço *Web* suporta interações diretas com outros Agentes de *software* usando mensagens baseadas em XML, trocadas via protocolos baseados na Internet" (W3C, 2005).

Os *Web Services* combinam os melhores aspectos do desenvolvimento baseado em componentes na *Web*. Assim como componentes de *software*, os *Web Services* representam uma funcionalidade *black-box* que podem ser reutilizados sem a preocupação com a linguagem utilizada no seu desenvolvimento (Graham et al., 2002).

### 4.2.1 Arquitetura de *Web Services*

A arquitetura de *Web Services*, apresentada na Figura 4.1, é baseada em um Provedor de serviços, um Solicitante de serviços e um Registro de serviços. O Provedor de serviços é responsável por disponibilizar os serviços e armazenar sua descrição, através da WSDL, a qual contém detalhes de interface, operações dos serviços e mensagens de entrada e saída para as operações. Após o recebimento da descrição de um serviço, o Provedor publica a descrição do mesmo, em WSDL, em um Registro de serviços. O Solicitante de serviços é uma aplicação que invoca uma interação com o serviço, a qual pode ser um navegador *Web* ou outra aplicação qualquer como um outro *Web Service*. O Registro de serviços é o local onde os Servidores publicam seus serviços e onde os Solicitantes procuram por serviços (Ferris and Farrell, 2003).

Na arquitetura de *Web Services*, a descrição de um serviço cobre todos os detalhes necessários para garantir sua utilização, incluindo o formato das mensagens, protocolos de transporte e localização.

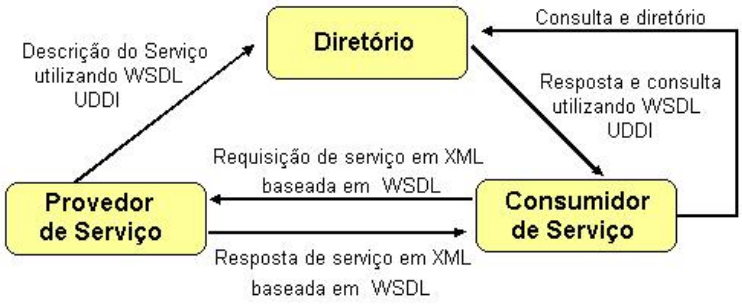


**Figura 4.1:** Arquitetura dos *Web Services* (Ferris and Farrell, 2003).

O Provedor de serviço é a plataforma acessada na solicitação do serviço. Trata-se da entidade que cria o *Web Service*, sendo responsável por realizar sua descrição em algum formato padrão e publicar os detalhes em um registro de serviço central.

O Servidor de Registro tem papel importante dentro dos *Web Services*, vez que é o local onde os Provedores publicam as descrições dos serviços. Uma descrição de serviço, por sua vez, torna possível descobrir onde está um *Web Service* e como invocá-lo. O seu funcionamento é iniciado quando o Provedor cria uma descrição que detalha a interface do serviço com suas operações e as mensagens de entrada e saída para cada operação. Uma descrição de ligação é então criada, mostrando como enviar cada mensagem para o endereço onde o *Web Service* está localizado. Já o Solicitante de serviço é uma aplicação que invoca ou inicia uma interação com um serviço.

### 4.2.2 Camadas dos *Web Services*

Os *Web Services* estão baseados em um conjunto de protocolos e linguagens padrões da *Web*, dentre elas se destacam o *HyperText Transfer Protocol* (HTTP), o *Simple Object Access Protocol* (SOAP), a *Web Service Description Language* (WSDL) e o *Universal Description, Discovery and Integration* (UDDI), sendo a linguagem XML a base dos três últimos padrões. Assim, a Figura 4.2 apresenta as camadas conceituais de *Web Services*.


**Figura 4.2:** Camadas conceituais de *Web Services* (Rheinheimer, 2004).

## *4.3* Tecnologias Padrões Utilizadas nas Arquiteturas *Web Services*

Anteriormente, foram mencionadas tecnologias padrões para a estruturação de *Web Services*, dentre elas estão a XML, WSDL, SOAP e UDDI, que estão abordadas de forma mais completa nas subseções que seguem.

### 4.3.1 *eXtensible Markup Language* (XML)

*eXtensible Markup Language* (XML) foi criada em 1996 pelo W3C, liderado por Jon Bosak, da Sun Microsystems, o qual desejava trazer para a *Web* o máximo de funcionalidade da linguagem *Standard General Markup Language* (SGML). No ano de 1997, foi realizada a primeira conferência sobre XML, em San Diego, CA (EUA), em que foi apresentado um artigo histórico de Jon Bosak intitulado "XML, Java, *and the future of the Web*". O artigo apresentava as principais características da linguagem: os autores de documentos XML poderiam definir novas marcações e atributos; a estrutura do documento poderia ser aninhada em qualquer nível de complexidade; qualquer documento XML poderia conter uma descrição opcional de sua gramática para uso de aplicações que necessitam executar uma validação estrutural (Santos, 2002).

*Markup Language* ou Linguagem de Marcação são linguagens onde a sintaxe é baseada em marcas (*tags*). As linguagens de marcação possibilitam que sejam simples e de fácil entendimento (ao ser humano), não existe um padrão do uso de *tags*. A Figura 4.3 mostra um exemplo de estrutura básica de um documento XML (Filho, 2005).

```
<?xml version="1.0"?>
<curso>
    <nome>Introdução em XML</nome>
    <descricao>Curso de introdutório na linguagem XML.</descricao>
    <nota>10</nota>
</curso>
```

**Figura 4.3:** Estrutura básica de um documento XML.

#### 4.3.1.1 Declaração

O uso de declaração da *tag* é importante, pois mostra que se trata de um documento XML e a versão em que foi escrita, podendo também declarar outros atributos que podem ser importantes na leitura do documento.

#### 4.3.1.2 Elementos

Elementos iniciam-se com '<' e terminam com '>'. Outra característica da linguagem XML é que, assim como os Sistemas Unix/Linux, o XML diferencia letras maiúsculas de minúsculas, ou seja: <curso> é diferente de <Curso>

#### 4.3.1.3 Comentários

Os comentários podem conter qualquer dado ou informação, sendo que o uso de comentários auxilia no entendimento do documento, principalmente para a sua transmissão. O comentário inicia com '<!- -' e termina com '- ->'.

#### 4.3.1.4 Hierarquia

A hierarquia da linguagem XML é baseada na semântica ou na estrutura lógica de documentos. A Figura 4.4 mostra a representação gráfica e seu correspondente em XML de uma hierarquia.

Toda hierarquia possui uma raiz, que é o início da hierarquia e geralmente uma abstração. Na Figura 4.4, a raiz é bebidas.

#### 4.3.1.5 Atributos

*Tags* XML podem conter atributos, porém, diferentes de elementos, não possuem sub-atributos ou outros elementos. Na declaração de atributos, devem ser dados seu nome e valor, podendo ter mais de um atributo a uma única *tag*, como pode ser visto na Figura 4.5.


**Figura 4.4:** Exemplo de uma hierarquia em XML.

```
<?xml version="1.0"?>

<tagxml-1 valor="1"> OK </tagxml-1>
<tagxml-2 valor="1" valor2=="1"> OK </tagxml-1>

<artigos>
   <xml>
     <nome idioma="pt-br"> Introdução ao XML </nome>
     <nome idioma="ing"> XML Introduction </nome>
     <nome idioma="es"> Introduccion de XML </nome>
     <nome idioma="al"> XML Einfuhtung </nome>
     <nome idioma="it"> Introduzine di XML </nome>
   </xml>
</artigos>
```

**Figura 4.5:** Atributo em um documento XML.

#### 4.3.1.6 Elementos Vazios

Na linguagem XML, os elementos vazios possuem uma sintaxe modificada. Os elementos vazios são utilizados para marcar uma determinada ação, onde são representados no inicio '<' e no final '\>' como no exemplo: '<vazio\>'.

### 4.3.2 *Web Service Description Language* (WSDL)

O *Web Service Description Language* (WSDL) (Newcomer, 2002) consiste em uma linguagem padrão XML para a descrição de interfaces de serviços, visando oferecer um formato padrão para representar os tipos de dados passados nas mensagens, interface descrevendo as funções disponíveis, informações sobre o protocolo de transporte a ser utilizado e informações sobre endereços dos serviços na rede. A sua utilização com *Web Services* acontece da seguinte maneira: para cada *Web Service* há um arquivo WSDL, escrito em XML, cuja função é descrever as operações que o *Web Service* realiza.

A WSDL é dividida em três elementos principais, os quais são: definição de tipo de dado, operações abstratas e protocolos de ligação. Cada elemento pode ser especificado em diferentes documentos XML e ser importado em diferentes combinações, tornando a descrição final de um *Web Service*, ou definido em um único documento XML. A Figura 4.6 apresenta um exemplo de um documento WSDL (HelloService.wsdl) (Cerami, 2002). Este código descreve um serviço com apenas uma função pública, a função sayHello. Essa função é o conhecido programa "Alô mundo"

que ao receber um parâmetro (*string*) retorna uma *string* de resposta. Por exemplo, se o parâmetro for "*world*", o serviço retorna "*Hello, world*!".

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="HelloService"
  targetNamespace="http://www.ecerami.com/wsdl/HelloService.wsdl"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:tns="http://www.ecerami.com/wsdl/HelloService.wsdl"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <message name="SayHelloRequest">
     <part name="firstName" type="xsd:string"/>
  </message>

  <message name="SayHelloResponse">
     <part name="greeting" type="xsd:string"/>
  </message>
  <portType name="Hello_PortType">
     <operation name="sayHello">
       <input message="tns:SayHelloRequest"/>
       <output message="tns:SayHelloResponse"/>
     </operation>
  </portType>
  <binding name="Hello_Binding" type="tns:Hello_PortType">
     <soap:binding style="rpc"
        transport="http://schemas.xmlsoap.org/soap/http"/>
      <operation name="sayHello">
       <soap:operation soapAction="sayHello"/>
       <input>
         <soap:body
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:examples:helloservice"
            use="encoded"/>
       </input>
       <output>
         <soap:body
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:examples:helloservice"
            use="encoded"/>
       </output>
     </operation>
  </binding>
  <service name="Hello Service">
     <documentation>WSDL File for HelloService</documentation>
     <port binding="tns:Hello_Binding" name="Hello_Port">
       <soap:address
          location="http://localhost:8080/soap/servlet/rpcrouter"/>
     </port>
  </service>
</definitions>
```

**Figura 4.6:** Documento WSDL HelloService.wsdl (Souza, 2004).

### 4.3.3 *Simple Object Access Protocol* (SOAP)

O *Simple Object Access Protocol* (SOAP) (Walsh, 2002) (Bray et al., 2000), de uma maneira geral, define o formato que as mensagens transportadas na rede devem ter para encaminhar requisições aos serviços *Web*. O SOAP é um protocolo de chamada de objetos remotos, desenvolvido inicialmente

pela Microsoft, que se tornou um padrão do *World Web Web Consortium* (W3C). Atualmente, a sua funcionalidade se destaca na utilização com *Web Services*. A comunicação é realizada por troca de mensagens transmitidas em formato XML, incluindo parâmetros usados na chamada, bem como os dados de resultados. Isso significa que a mensagens podem ser utilizadas e compreendidas por várias plataformas de *hardware*, sistemas operacionais, linguagens de programação ou *hardware* de rede.

Um pacote SOAP é um documento XML específico e possui as seguintes partes (Newcomer, 2002)(Hansen, 2003):

- envelope: responsável por definir o início e o fim das mensagens, que poderá tratá-las, sendo que o tratamento é obrigatório ou opcional;
- cabeçalho: local que possui os atributos opcionais das mensagens;
- corpo: contém os dados em XML;
- anexo: consiste em um ou mais documentos anexados à mensagem principal;
- *Remote Procedure Call* (RPC): define como o modelo RPC irá interagir com o SOAP;
- codificação: responsável por definir como representar dados simples e complexos a serem transmitidos nas mensagens.

### 4.3.4 *Universal Description, Discovery and Integration* (UDDI)

O *Universal Description, Discovery and Integration* (UDDI) é um padrão que começou a ser desenvolvido em 2000 pela IBM, Microsoft e Ariba. O Objetivo do projeto UDDI é criar um padrão para a descoberta de serviços. O UDDI permite localizar um serviço, sendo uma especificação técnica para descrever, descobrir e integrar *Web Services*. Ele é constituído de duas partes: uma especificação técnica para construir e distribuir *Web Services*, a qual permite que as informações sejam armazenadas em um formato XML específico; e o UDDI *Business Registry* ou UDDI *cloud services*, que é uma implementação operacional completa da especificação UDDI (Rheinheimer, 2004).

Os dados UDDI podem ser separados em três categorias principais:

- *white page*: na categoria páginas brancas são incluídas informações gerais sobre uma empresa específica, como o nome, a descrição, o contato, o endereço e os números de telefone;
- *yellow pages*: na categoria páginas amarelas são incluídos dados de classificação gerais da empresa ou serviço oferecido, como o ramo ou a destinação;
- *green pages*: na categoria páginas verdes contém as informações técnicas sobre *Web Services*.

Um registro UDDI contém informações categorizadas sobre negócios, serviços que eles oferecem e associações desses serviços com especificações dos *Web Services*. A Figura 4.7 ilustra como é possível registrar as informações de um *Web Service* com o registro UDDI.

Inicialmente, foi gerado um documento WSDL (arquivo) para descrever o *Web Service* com suporte do Processador SOAP (1) e utiliza-se a *Application Program Interface* (API) UDDI para

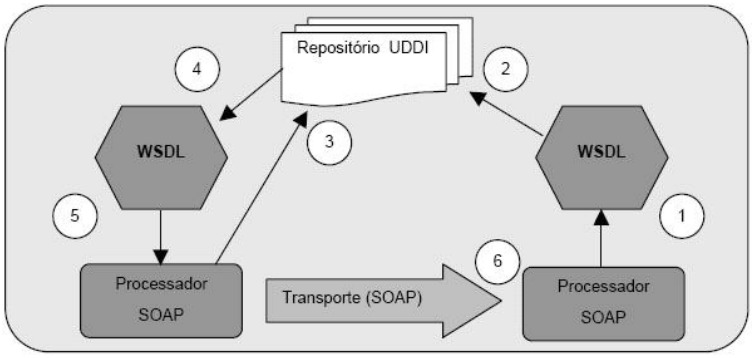


**Figura 4.7:** UDDI utilizado para descobrir um *Web Service* (Souza, 2004).

registrar as informações no repositório (2). Os dados são transmitidos juntamente com as informações sobre contato e o registro possui uma entrada (URL que aponta para o Servidor SOAP) com a localização do WSDL, quando outro processador SOAP requisita o registro (3) para obter o WSDL (4). Após, o cliente gera a mensagem apropriada (5) para enviar a uma operação específica através de determinado protocolo (6). O protocolo utilizado nesse exemplo foi SOAP sobre HTTP.

## *4.4* Conclusão

Este Capítulo apresentou as principais abordagens de arquitetura de *Web Services*, bem como as tecnologias padrões que estão sendo utilizadas. As diferentes abordagens de arquitetura demonstram que padrões como SOAP, UDDI e WSDL representam uma forma de permitir a comunicação de *Web Services* construídos em diferentes plataformas.

Outra característica dos *Web Services* é a promoção da interoperabilidade de aplicações heterogêneas, no caso desta pesquisa, promover a interoperabilidade de recursos educacionais.

# CAPÍTULO 5

# Objetos de Aprendizagem Reutilizáveis (RLOs)

Um Objeto de Aprendizagem caracteriza-se por ser uma entidade de material educacional reutilizável (Gomes, 2005), o qual é definido como qualquer entidade digital ou física (não digital) que possa ser usada para a aprendizagem, educação ou treinamento (IEEE, 2002).
Este Capítulo tem como objetivo familiarizar o leitor com a abordagem de objetos de aprendizagem reutilizáveis, suas tecnologias e principais padrões.

## *5.1* Introdução

Os Objetos de Aprendizagem surgiram no inicio da década de 90, onde vários grupos isolados iniciaram trabalhos e estudos com o conceito de normalizar a apresentação de conteúdos na *Web*. O Grupo de Metadados de Objetos de Aprendizagem (*Learning Object Metadata Group*) do Instituto Nacional de Ciência e Tecnologia (*National Institute of Science and Technology*) e a CEdMA (*Computer Education Management Association*) investiram seus esforços no estudo de objetos de aprendizagem (LO), incluindo modularidade, orientação para armazenamento em banco de dados e o uso de elementos de marcação, que evoluíram para os metadados.

Para a reutilização, um objeto de aprendizagem precisa ser modular, interoperável e ter a capacidade de ser descoberto. Muitas organizações e grupos de pesquisas vêm trabalhando no sentido de alcançar essas características e também no sentido de aprimorar a eficiência e eficácia desses objetos. A maioria dos esforços concentra-se na definição de padrões. Iniciativas como o *Learning Tecchnology Standard Comitee* (LTSC) do *Institute of Electrical and Electronics Engineers* (IEEE, 2005), a *Alliance of Remote Institute of Electrical and Distribution Networks for Europe* (ARIADNE) (ARIADNE, 2005), o IMS *Global Learning Consortium* (IMS, 2005), a Canadiam Core (CanCore, 2005) e a *Advanced Distributed Learning initiative* (ADL, 2005) têm contribuído significativamente na definição de padrões de indexação (metadados). Estruturas de metadados contêm informações que descrevem o conteúdo educacional carregado pelo objeto de aprendizagem, o que facilita a tarefa de encontrar o objeto que melhor se adapta a uma demanda específica.

Este Capítulo, trata do relacionamento dos conceitos e características dos Objetos de Aprendizagem Reutilizáveis (*Reusable Learning Objects* - RLOs) que vêm sendo apontados como uma solução eficiente para os problemas de padronização com baixo custo de desenvolvimento de conteúdo

educacional (Gomes et al., 2004) (Pereira et al., 2003) (Barbeira and Santos, 2002). Serão mostrados os conceitos mais comuns dados aos Objetos de Aprendizagem Reutilizáveis, bem como os padrões de metadados educacionais *Instructional Management Systems* (IMS), *Learning Object Metadata* (LOM) e *Sharable Content Object Reference Model* (SCORM).

## 5.2 O que são Objetos de Aprendizagem?

Um Objeto de aprendizagem é qualquer entidade, digital ou não, que possa ser usada, reusada, ou referenciada durante a aprendizagem suportada por tecnologia (IEEE, 2005).

Segundo Jacobsen (Jacobsen, 2004), trata-se de uma coleção de material reutilizável utilizado para apresentar e dar apoio a um único objetivo de aprendizagem ou (EDtech, 2005) de um pequeno "componente instrucional" que pode ser utilizado para suportar a aprendizagem em ambientes diferentes.

Pode-se concluir que um Objeto de Aprendizagem é um módulo, unidade, capítulo ou lição que se pretende ensinar um conceito, uma idéia, um fato, um procedimento, um processo ou um princípio.

Tendo este princípio de componentes, vários ambientes de ensino diferentes podem utilizar o mesmo Objeto de Aprendizagem, por meio de sistemas de gerência de aprendizagem (*Learning Management System* - LMS) ou sistemas de gerência de conteúdos de aprendizagem (*Learning Content Management Systems* - LCMS).

Os elementos componentes dos Objetos de Aprendizagem são identificados por meio de *tags* XML, seguindo rígidas diretrizes de especificações com padronização internacional. Desta maneira, os RLOs podem ser armazenados em repositórios para futuras recuperações. Outra vantagem do uso de *tags* de identificação de elementos nos RLOs é que facilitam os mecanismos de busca na localização do objeto quando se faz uma pesquisa em um repositório de objetos educacionais.

## 5.3 Metadados Educacionais

Mesmo que o conceito de metadado anteceda a Internet e a *Web*, mundialmente, o interesse por padrões de metadados e suas aplicações tiveram seu maior crescimento com o aumento de publicações eletrônicas e "sobrecarga de informações", resultando em uma vasta quantidade de dados digitais não diferenciados (DCMI, 2001).

Metadados são definidos como informação acerca de informação. Segundo a W3C (W3C, 2005) são dados que descrevem dados mais complexos.

Metadados de Objetos de Aprendizagem são informações sobre os dados que compõem LOs, sejam eles físicos ou digitais. Atualmente, existe um crescimento exponencial de objetos de aprendizagem disponíveis na *Web*. Paralelamente, cresce na mesma proporção a necessidade por esse aprendizado, porém, a falta de informações sobre os conteúdos dos LOs já desenvolvidos impõe limitações e críticas fundamentais à capacidade de se pesquisar, gerenciar e utilizar os LOs (IEEE, 2002). O uso de padrões para definir e catalogar, disciplinar e fazer uma descrição do conteúdo dos LOs permite a uniformização e ampliação da qualidade da base de LOs e também a reutilização ampla dos mesmos. Várias organizações procuram criar padrões para metadados de LOs, sendo vários os padrões existentes, dentre eles destacam-se o *Learning Object Metadata* (LOM) do

IEEE *Learning Technology Standards Committee* (LTSC) [Sessão 5.3.1], o do IMS *Global Learning Consortium* [Sessão 5.3.2] e o *Sharable Content Object Reference Model* (SCORM) [Sessão 5.3.3].

### 5.3.1 *Learning Object Metadata* (LOM)

O *Learning Object Metadada* (LOM) surgiu dos projetos ARIADNE[1] e IMS[2] e também de estudos sobre metadados desenvolvidos pelo grupo do Dublin Core[3] (LOM, 2005).

Padrão desenvolvido pela IEEE é uma organização credenciada para desenvolvimento de normas. O *Learning Technology Standards Committee* (LTSC) foi instituído pelo IEEE, pela *Computer Society* e pelo *Standards Activity Board* com o objetivo de desenvolver normas, orientações e práticas recomendadas na área de aprendizado suportado por computador.

Cerca de 20 grupos de estudos constituídos no LTSC classificam-se nas categorias de atividades gerais, atividades relacionadas ao aprendiz, atividades relacionadas ao conteúdo, dados e metadados, sistemas de gerência e aplicações.

#### 5.3.1.1 Objetivos

Um dos principais objetivos da norma IEEE-LOM é o de facilitar a busca, avaliação, aquisição e uso de objetos de aprendizagem por parte de aprendizes (alunos), instrutores (professores) ou processos automatizados de *software*. Como objetivo secundário, o LOM busca facilitar o compartilhamento e intercâmbio de LOs independentemente da plataforma ou sistema que o utilizará (LMS), permitindo a geração de catálogos e inventários ao mesmo tempo em que se leva em consideração a diversidade de contextos culturais e de linguagem onde os LOs e seus metadados possam ser empregados.

Com o uso de especificação de um esquema conceitual padrão, a norma garante que *bindings* (representações desses metadados, XML, por exemplo) entre metadados de LOs terão melhor grau de interoperabilidade semântica.

O padrão ou norma pretende especificar um esquema base que pode ser usado para facilitar, por exemplo, a colocação automática e adaptável de LOs em seqüência por parte de agentes de *software*.

#### 5.3.1.2 Estrutura Básica dos Metadados

O LOM (LOM, 2005) possui um esquema conceitual de dados que define a estrutura da instância de metadados de um objeto de aprendizagem. Uma instância de metadados de um LO descreve as características relevantes do LO ao qual se aplica e é composta de elementos de dados. Esses elementos de dados compõem uma hierarquia, que são: nós intermediários[4] e folhas, que se enquadram em categorias:

- características gerais: possuem informações gerais que descrevem o Objeto de aprendizagem (LO) como um todo;
- ciclo de vida: são as características relacionadas à história e ao estado corrente desse LO;
- meta-metadados: agregam as informações sobre a instância de metadados propriamente dita;

[1] http://www.ariadne-eu.org
[2] http://www.imsproject.org
[3] http://dublincore.org
[4] No LOM V1.0, nós intermediários não possuem campos de valores ou tipos de dados

- técnicas: agrupam as características e requisitos técnicos do LO;
- educacionais: têm as características pedagógicas e educacionais do LO;
- direitos: responsáveis pelos direitos de propriedade intelectual e condições de uso do LO;
- relações: são as características das relações entre LO e outros LOs (quando existem essas relações);
- anotação: agrupam os elementos que contêm comentários sobre o uso educacional do LO e sobre por quem e quando os comentários foram feitos;
- classificação: agrupam os elementos que descrevem o LO com relação a um sistema de classificação específico.

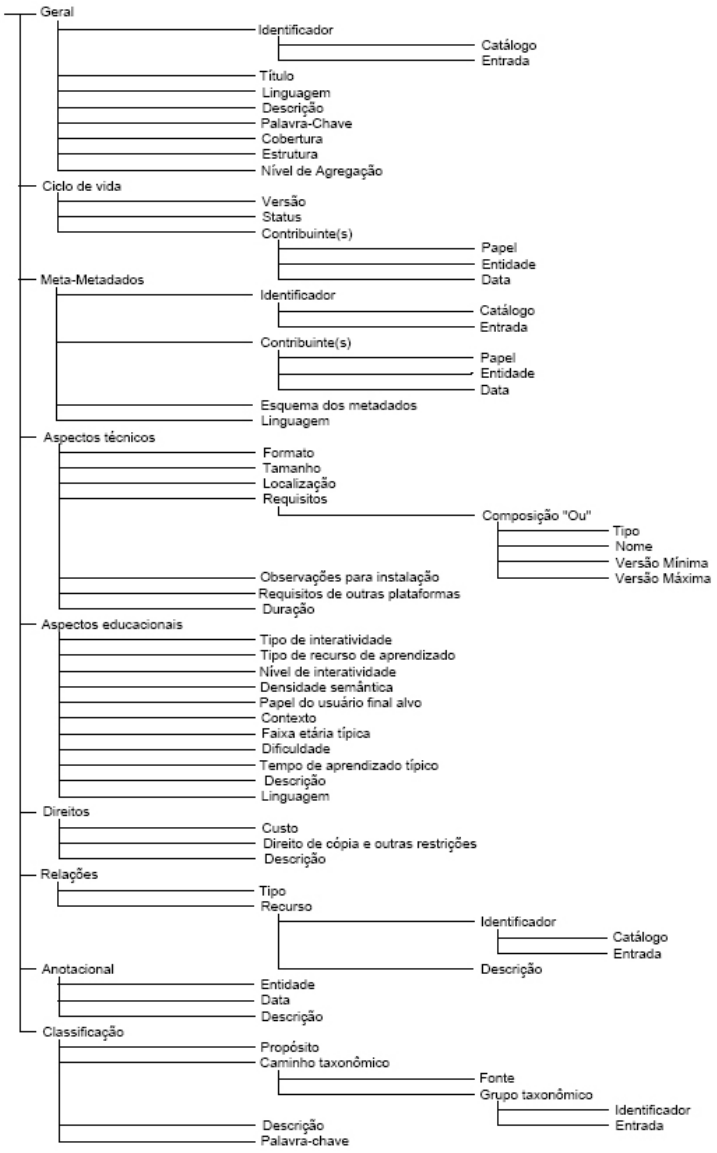


**Figura 5.1:** Árvore completa do LOM V1.0 (LOM, 2005).

A Figura 5.1 mostra a árvore completa do LOM V1.0, que mostra os diversos níveis de hierarquia e onde as colunas são, para cada elemento de dado, como abaixo:

1. o número (1, 1.1, 1.1.2 ...);
2. o nome;
3. a explanação (descrição em prosa do elemento);
4. o número de valores permitidos;
5. a ordem dos valores permitidos (não especificada, ordenado e não ordenado);
6. o espaço de valores (domínio e/ou norma aplicável);
7. tipo de dado;
8. exemplo(s).

### 5.3.2 IMS *Global Learning Consortium, Inc*

O *Instructional Management Systems* (IMS) *Global Learning Consortium, Inc.* surgiu em 1997 como um projeto dentro da US *National Learning Infrastructure Initiative* (NLII) e originalmente focada em educação superior. O projeto IMS tem o intuito de promover especificações não proprietárias, como definição de metadado e especificação de questionários/avaliações, para prover o estudo *on-line* distribuído (*e-learning*) (IMS, 2005) (Muzio et al., 2001). O IMS conta com a participação de diversas organizações, tais como: *Advanced Distributed Learning* (ADL), Apple *Computers*, Artesia *Technologies*, Califórnia *State University*, Oracle,*University of* Cambrige, *University of* Michigan, PUC-Rio, dentre outras.

Os objetivos do projeto IMS são: (*i*) definir técnicas padrões para a interoperabilidade de aplicações e serviços em aprendizagem distribuídos; (*ii*) suportar a incorporação de especificações IMS em produtos e serviços; e (*iii*) a adoção das especificações que possibilitem que ambientes de aprendizagem/ensino distribuído e conteúdos de vários autores operem (trabalhem) em conjunto.

Algumas das atividades desenvolvidas pelo IMS para auxiliar atividade de ensino/aprendizagem *on-line* são (IMS, 2005):

- localização e uso de conteúdos educacionais;
- controle do progresso do aprendizado/estudo;
- relatório do desempenho do aprendizado;
- troca de registros de estudantes entre sistemas administrativos (LMS).

Como exemplo, a Figura 5.2 mostra como podemos usar uma idéia simples de "como preparar uma xícara de café perfeito" é utilizada para demonstrar como construir um Objeto de Aprendizagem seguindo o padrão IMS (Bettio, 2003).

Segundo o Padrão IMS, Figura 5.3, um Objeto de Aprendizagem deve ser bem estruturado e dividido em três partes: Objetivos, Conteúdo instrucional e Prática e *feedback*.

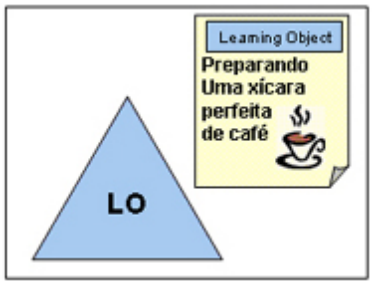


**Figura 5.2:** Objeto de Aprendizagem (Bettio, 2003).


**Figura 5.3:** Divisão de um Objeto de Aprendizagem (Bettio, 2003).

Objetivos: essa parte do objeto tem como intenção demonstrar ao aprendiz (aluno) o que ele poderá aprender a partir do estudo desse objeto. Também poderá conter uma lista dos conhecimentos prévios (pré-requisitos) necessários para um bom aproveitamento de todo o conteúdo disponível.

Conteúdo instrucional: é nessa parte que realmente é apresentado todo o material didático e o conteúdo, podendo ser essa apresentação em várias mídias como: Flash, Texto, Imagens, Animações, http e outras mídias *Web*. No final desse conteúdo e com os objetivos alcançados, é que o aprendiz (aluno) pode finalizar o conteúdo.

Prática e *feedback*: uma das características importantes do paradigma Objetos de Aprendizagem é que, a cada final de aprendizado, julga-se necessário que o aprendiz (aluno) verifique se o seu desempenho atingiu as expectativas. Caso contrário, o aprendiz deve ter a liberdade para voltar a se utilizar do objeto quantas vezes julgar necessário.

A abordagem do IMS à problemática da normalização da criação de conteúdos de aprendizagem é modular, sendo criados vários grupos de trabalhos para definir as especificações de normalização (IMS, 2005):

- IMS *Content Packaging Specification*: o objetivo desse grupo é especificar uma norma para o empacotamento e estruturação dos conteúdos educacionais, definindo um curso como um conjunto de pequenos objetos, ou unidades educacionais agregados (unidos) em uma estrutura em forma de árvore (onde se pode acessar até os níveis desses mesmos objetos). A estrutura é definida em um arquivo XML, também chamado manifesto de um curso, onde está definida a estrutura do curso, as relações com os componentes físicos (arquivos) somado com a definição de metadado sobre todos esses aspectos;

- IMS Metadata *Specification*: torna o processo de encontrar e utilizar recurso de aprendizagem mais eficiente, por fornecer uma estrutura de elementos que descrevem um conteúdo

pedagógico/aprendizagem. Pode ser visto como uma etiqueta que se agrega aos diversos conteúdos com as suas principais características e descrições;

- IMS *Enterprise Specification*: o objetivo desse grupo é definir um modelo de dados a utilizar pelos diversos LMSs com base para a interoperabilidade entre os sistemas. É responsável por descrever como sistemas de gerenciamento institucionais (LMS) são interoperáveis com outros sistemas, usando para suporte operações de uma organização, bem como administração de treinamento, de estudantes e sistemas de recursos humanos. Dessa maneira, interações com qualquer sistema compatível com a norma possuem uma estrutura padrão (*standard*) de dados, os quais podem ser utilizados (independentemente do LMS) em questões e testes;

- IMS *Question & Test Interoperability* (QTI): responsável por descrever uma estrutura básica para a representação de dados de questões e testes/avaliações, usando a adaptação QTI para a troca de dados para testes/avaliações entre LMS. Essa proposta está baseada em XML, buscando que avaliações e testes obedeçam esta normalização. Resulta na criação de questionários em uma estrutura XML que podem ser reutilizados pelos LMSs que suportam a norma;

- IMS *Learner Information*: o objetivo desse grupo de trabalho é definir um modelo de dados normalizados para possibilitar o intercâmbio entre sistemas de informação relativa aos seus utilizadores (alunos) e disponibilizar um protocolo para a troca dos dados e informações entre sistemas. Esse modelo utiliza XML para evitar incompatibilidade das bases de dados envolvidas;

- IMS *Competency Definitions*: grupo dedicado ao estabelecimento de um modelo descritivo das diferentes competências que podem ser abordadas no âmbito de uma plataforma de *e-learning*. Essa norma busca criar uma referência nas competências que permita serem referidas por significados comuns.

Com o padrão IEEE-LOM, o projeto IMS desenvolveu uma representação de metadado em XML, utilizando esse padrão somado a algumas extensões para elementos e estruturas de metadados proprietários. Para evitar a perda da interoperabilidade, o IMS definiu duas maneiras de tratamento para todas as extensões proprietárias (IMS, 2005) (Muzio et al., 2001):

- *Document Type Definition* (DTD): são extensões que utilizam um elemento que deve ser utilizado quando se quer construir extensões usando um arquivo de controle (DTD). Trata-se de elemento opcional para cada parte da estrutura de metadados. O IMS utiliza a linguagem XML *Schema Definition* quando utiliza extensões;

- XML *Schema Definition* (XSD): as extensões de metadados que utilizam XML *Schema Definition* são possíveis através da utilização de outros elementos LOM com novos elementos definidos em *namespace* específico.

#### 5.3.2.1 Armazenamento

O IMS define uma forma padrão de armazenagem de informações necessárias para uma indexação dos Objetos de Aprendizagem, de modo que a característica Indexação e Procura possa ser cumprida.

A maneira encontrada pelos pesquisadores envolvidos no consórcio IMS foi a criação de metadados (Muzio et al., 2001)(IMS, 2005).

A Figura 5.4 mostra como os metadados possuem informações referentes a um LO, ou seja, como ele descreve os Objetos de Aprendizagem utilizando uma linguagem de marcação (XML), de maneira que qualquer plataforma de *e-learning* que siga o padrão IMS poderá procurar por objetos em qualquer *Learning Content Repository* (Bettio, 2003).

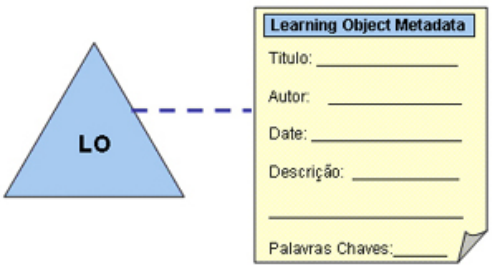


**Figura 5.4:** Exemplo de Metadado de um Objeto de Aprendizagem (Bettio, 2003).

#### 5.3.2.2 IMS *Content Packing*

O *Content Packing* é o padrão de especificação da maneira como os Objetos de Aprendizagem são organizados dentro das plataformas de EAD. O objetivo dessa organização é permitir e facilitar a distribuição de objetos entre plataformas diferentes, assim, facilitando a criação e divulgação dos mesmos.

Outra função do *Content Packing* é como será feita a visualização do LO pelo estudante. Ele pode ser utilizado para criar árvores de conhecimento, dividindo assim os Objetos de Aprendizagem em grupos que ficam mais bem organizados (IMS, 2005).

O padrão IMS que define um Modelo de Agregação do Conteúdo tem por objetivo empacotar os Objetos de Aprendizagem de uma maneira que estes possam ser compartilhados, permitindo sua reusabilidade. A Figura 2.14 mostra como esse modelo pode ser definido como um pacote, que é referenciado no padrão IMS como *Package Interchange File* (PIF), que é dividido em duas áreas, o "Manifesto" e os Arquivos Físicos.

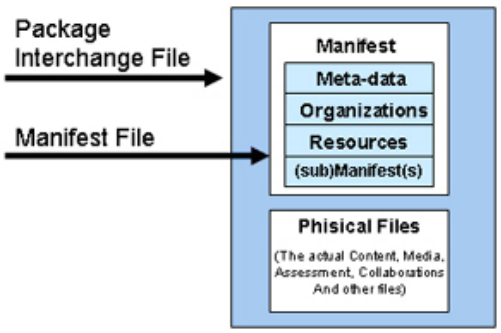


**Figura 5.5:** *Package Interchange File* (PIF) (IMS, 2005).

*Manifest*: o manifesto é estruturado utilizando *tags* da linguagem XML, que foi subdividido para uma melhor organização em quatro subitens:

- *Meta-data*: são os dados que descrevem o pacote, explicitam que tipo de informação o pacote contém;
- *Organizations*: responsável pela organização do pacote, este campo contém a taxonomia organizacional dos dados referenciados no pacote;
- *Resources*: contém as informações sobre os arquivos físicos que estão presentes no *Physical Files* (Arquivos Físicos);
- (sub)Manifestos: faz a ligação com outros manifestos, possibilitando assim a criação da árvore de conhecimento.

Arquivos Físicos (*Physical Files*): contém os arquivos Físicos referenciados no *Resources* do Manifesto. Esses arquivos podem ser de diversos tipos de mídia como: documento, vídeo, áudio ou qualquer outro tipo de mídia que possa ser usada como material de ensino.

### 5.3.3 *Sharable Content Object Reference Model* (SCORM)

A norma Sharable Content Object Reference Mode (SCORM) (Looms and Christensen, 2002) (SCORM, 2005) surgiu de uma iniciativa do governo dos Estados Unidos, através do seu Departamento de Defesa e da união da indústria de tecnologia, em meados de 1997, para iniciar o movimento pela adoção de um padrão único para os sistemas de educação a distância. Dessa união, surgiu o consórcio ADL (*Advanced Distribuited Learning Co-Labs*), que no início de 1999 apresentou o SCORM como o padrão ideal, pois reuniu todos os padrões disponíveis no mercado.

Desta maneira, SCORM é um modelo que descreve um conjunto de especificações técnicas e de referência para a apresentação de conteúdo de ensino e aprendizagem via *Web* (*e-learning*). A norma SCORM divide-se em duas partes:

- *Content Aggregation Model*: responsável por definir a forma como os conteúdos educacionais devem ser criados e agrupados;
- *Run-Time Environment*: responsável pela definição da forma como o sistema de gerenciamento (*Learning Management System* - LMS) do ambiente disponibiliza os conteúdos educacionais e como estes comunicam com o sistema.

Assim, utilizando a normalização do SCORM, é possível dissociar o armazenamento do conteúdo educacional de sua exibição.

#### 5.3.3.1 *Content Aggregation Model*

O Modelo de Agregação de Conteúdo (Looms and Christensen, 2002) (SCORM, 2005) define a forma como os conteúdos de ensino devem ser criados e agrupados. Assim, o modelo de agregação de conteúdo é constituído pelos seguintes componentes: modelo de conteúdo, metadado e empacotamento de conteúdo.

O modelo de conteúdo (Content Model) é onde se localizam as informações educacionais e didáticas do ambiente. Os componentes do modelo são os *Assets*, *Sharable Content Objects* (SCO) e *Content Aggregation Model*. Os *Assets* representam a forma mais básica de um conteúdo educacional, podendo fazer parte de um *Asset*: texto, imagens, som, páginas HTML e qualquer formato *Web*. Porém, para que o sistema de gerência possa apresentar um *Asset*, este deve ser sempre inserido em SCO.

O *Sharable Content Objects* (SCO) representa um conjunto de um ou mais *Assets* e, obrigatoriamente, inclui um mecanismo de comunicação com o sistema de gerência do ambiente (LMS), como se pode ver na Figura 5.6. Por padrão, o mecanismo de comunicação é codificado em JavaScript.

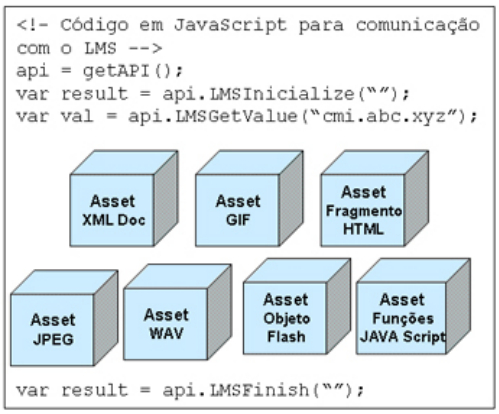


**Figura 5.6:** Exemplo de SCO, adaptado de (SCORM, 2005).

O *Content Aggregation* trata-se de um mapa usado para reunir os recursos educacionais de forma ordenada e coesa (como por exemplo, um curso ou um capítulo), estabelecendo uma estrutura seqüencial pelas quais os conteúdos educacionais serão exibidos aos usuários do ambiente.

Metadado é freqüentemente identificado como sendo dado acerca de dado, ou informação sobre informação que está no espaço digital e virtual. Pode ser visto como um sumário de informações sobre a forma e o conteúdo de um recurso eletrônico, ou não (Hansen, 2003). No modelo SCORM, Metadado consiste na disponibilidade de prover um meio coerente de descrever o conteúdo de cada componente (*Assets*, SCO e *Content Aggregation*), de modo que esses componentes possam ser arquivados e pesquisados de uma forma rápida e eficiente.

O SCORM respeita o *standard* IEEE LTSC *Learning Object Meta-Data* (LOM)[5] e utiliza o IMS *Learning Resource Meta-Data* XML *Binding Specification*[6] para guardar a informação em formato *eXtensible Markup Language* (XML).

O empacotamento de conteúdo é um conjunto de regras e normas para agregar conteúdos educacionais em blocos (*packeges*), com o objetivo de facilitar sua transferência. Os *packeges* dividem-se em duas partes:

- um documento XML que descreve os conteúdos e a organização do bloco;
- arquivos dos recursos educacionais (SCOs).

[5]IEEE *Information Technology - Learning Object Meta-Data* (LOM) http://ltsc.ieee.org/
[6]IMS *Learning Resource Meta-Data Specification* http://www.imsglobal.org

São armazenados em arquivos do tipo *Package Interchange File* (PIF) para facilitar sua distribuição pela *Web*. O PIF pode ter diversos formatos, sendo os mais utilizados: zip, jar, rar, arj, tar e cab (Shih et al., 2003).

O modelo de *Content Packaging* é utilizado de duas formas ou perfis (Barbeira and Santos, 2002):

- *Resource Package Application Profile*: é quando o manifesto que acompanha os conteúdos apenas é constituído pela seção <*resources*>, são apenas um conjunto de recursos (SCOs), sem nenhuma estrutura, apenas aglomerados em um pacote para intercâmbio;
- *Content Aggregation Package Application Profile*: acontece nos casos em que, além da definição dos recursos, é acompanhado de uma descrição da sua organização e estrutura, tendo o manifesto que acompanha o pacote, além da seção <*resources*>, a utilização seção de <*organizations*>.

Um modelo conceitual desse tipo de perfil pode ser visto na Figura 5.7.

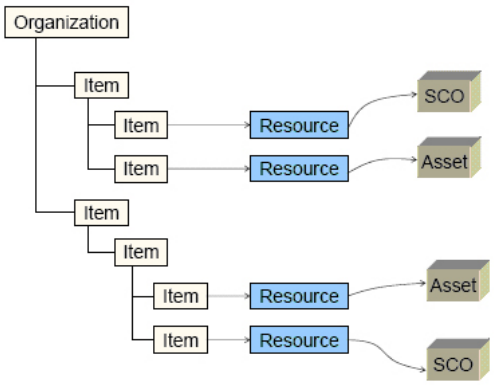


**Figura 5.7:** Diagrama da seção *Organizations* (SCORM, 2005).

#### 5.3.3.2 *Run-Time Environment*

O *Run-Time Environment* (Looms and Christensen, 2002) (SCORM, 2005) trata-se de um ambiente de execução que tem como objetivo permitir que os conteúdos de ensino possam ser visualizados em diferentes sistemas de gerenciamento (LMS) e administrar o comportamento dos conteúdos educacionais.

Para que isso seja possível, é necessário determinar a forma como os SCOs são enviados para o *browser* e definir o protocolo que os SCOs e o LMS irão usar para se comunicar. A Figura 5.8 mostra a interação entre o sistema cliente e servidor e pode ser observado que o ambiente de execução é constituído pos três elementos (Shih et al., 2003)(ADL, 2005):

- o mecanismo de *Launch*: que é responsável por enviar o SCO para o navegador (*browser*) do cliente e efetuar todas as configurações para que o SCO possa comunicar com o LMS;
- a *Application Program Interface* (API) *Adapter*: que trata-se de um dispositivo que irá criar um canal de comunicação entre os SCOs e o LMS;

- o *Data Model*: que é responsável por definir a "linguagem" usada pelos SCOs e pelo LMS para que ambos entendam ao se comunicarem. O Modelo de Dados utilizados pelo SCORM é o AICC CMI *Data Model*[7], em que CMI é o acrônimo para *Computer Managed Instructions*.

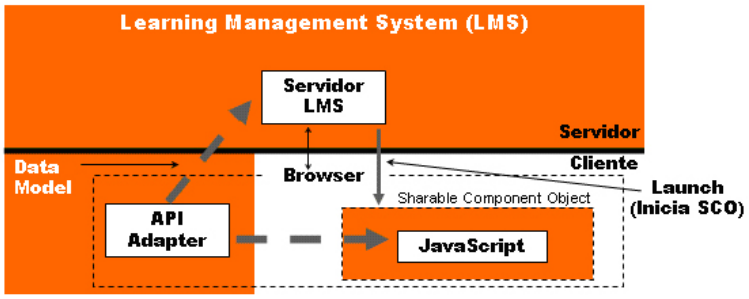


**Figura 5.8:** *Run-Time Environment* de uma plataforma de *e-learning*, adaptado de (SCORM, 2005).

O *Launch* é um mecanismo presente em todos os LMSs, que possui a responsabilidade de enviar os recursos de ensino ao navegador do aluno (*browser*). Os componentes do *Content Model* que podem ser enviados pelo LMS são os *Assets* e os SCOs. (SCORM, 2005).

No caso de um SCO, o LMS tem que disponibilizar uma estrutura de comunicação entre o SCO e o LMS. Esse mecanismo é o API *Adapter*. Os *Assets* não comunicam com o LMS, portanto, a responsabilidade do *Launch* é enviar o recurso via HTML.

A *Application Program Interface* (API) é o mecanismo ou o meio pelo qual os SCOs e o LMS se comunicam. A finalidade da API *Adaper* é facilitar a interoperabilidade e a reutilização dos conteúdos de ensino, criando uma forma padrão (*standard)* de comunicação, encapsulando os detalhes da comunicação entre o aluno e o ambiente de EAD (SCORM, 2005).

A API *Adapter* é um *software* que implementa as funções definidas pela norma SCORM. Todas as comunicações entre o SCO e o LMS são iniciadas pelos SCOs. Desta maneira, após o LMS enviar o SCO ao navegador, através do *Launch*, cabe ao SCO localizar a API *Adapter* e iniciar a comunicação com o LMS para pedir e armazenar dados no modelo de dados. As funções disponíveis pela API são:

- Estado de Execução
  - LMSInicialize(""): essa função indica ao LMS que o SCO pretende comunicar. O SCO deve sempre executar o LMSInicialize antes de invocar qualquer outra função da API, podendo obter como resposta, "*true*" ou "*false*", indicando se foi bem ou mal sucedida;
  - LMSFinish(""): essa função deve ser invocada quando o SCO já não precisa mais se comunicar com o LMS, liberando o API, podendo obter como resposta, "*true*" ou "*false*", indicando se foi bem ou mal sucedida;
- Controle de Erros
  - LMSGetLastErro(""): essa função permite que o SCO possa saber se a chamada a alguma das funções da API foi bem sucedida ou não, e no caso de não ter sido bem sucedida,

[7] *Aviation Industry* CBT *Committee* / CMI CMI001 *Guidelines for Interoperability* - http://www.aicc.og/

retornar o código do erro gerado na última chamada à API. A Tabela 5.1 mostra os tipos de erros que a norma estabelece, podendo obter de resposta "0" se não tiver erro ou o número correspondente ao erro;

- LMSGetErroString(*erronumber*): essa função permite ao SCO obter uma descrição textual do erro ocorrido, quando o SCO chama a função passando como argumento o código do erro e recebendo de resposta uma *string* com a descrição do erro;
- LMSGetDiagnostic(*parameter*): essa função possibilita obter a descrição específica de cada LMS para os erros ocorridos, fornecendo uma descrição mais detalhada.

- Transferência de Dados
  - LMSGetValue(*element*): essa função permite ao SCO solicitar dados ao LMS. O elemento é solicitado através do seu parâmetro, respeitando o modelo de dados da norma. As respostas são devolvidas pela função no formato "*string*";
  - LMSSetValue(*element*, *value*): essa função possibilita ao SCO enviar informações para o LMS. A função possui dois parâmetros: o primeiro indica o local no modelo de dados que deve ser armazenado e o segundo o valor. Pode-se obter de resposta "*true*" ou "*false*" indicando se foi bem ou mal sucedida;
  - LMSCommit(*parameter*): essa função garante ao SCO que a informação enviada ao LMS, pelo LMSSetValue() será persistida. Nos casos em que a API *Adaper* foi implementada utilizando memória *cache*, informa que o dado deve ser persistido em uma memória física (banco de dados ou arquivo). Pode-se obter de resposta "*true*" ou "*false*" indicando se foi bem ou mal sucedida;

**Tabela 5.1:** Código dos erros mais freqüentes (SCORM, 2005)

| Tipo de erro | Código | Descrição |
|---|---|---|
| | 0 | No erro |
| 100's General errors | 101 | General exception |
| 200's Syntax errors | 201 | Invalid argument |
| | 202 | Element cannot have children |
| | 203 | Element no an array - cannot have count |
| 300's LMS errors | 301 | Not initialized |
| 400's Data model errors | 401 | Not implemented error |
| | 402 | Invalid set value, element is a keyword |
| | 403 | Element is read only |
| | 404 | Element is write only |
| | 405 | Incorrect Data Type |

#### 5.3.3.3 *Data Model*

O Modelo de Dado foi criado para permitir que diferentes LMSs possam processar e manipular um conjunto definido de informações provenientes de qualquer SCO. Desta maneira, é capaz de proporcionar aos SCOs uma forma padrão de acessar os dados do LMS. O Data Model é um

protocolo para possibilitar a comunicação entre o SCO e o LMS em uma "linguagem" que ambos entendam (SCORM, 2005).

O SCORM optou pela utilização do modelo de dados criados pela *Aviation Industry* CBT *Committee* (AICC) o AICC CMI *Data Model* (CMI, 2005). CMI é o acrônimo de *Computer Managed Instructions*. O CMI possui algumas regras gerais para a utilização de dados:

- a identificação do modelo de dados é feita pelo primeiro símbolo do elemento, sendo que o SCORM utiliza o padrão CMI e por isso todos os elementos começam por "cmi". Esta regra é importante para que uma mesma API trabalhe com modelos de dados diferentes;
- o CMI estabelece algumas palavras reservadas para obter informações sobre o modelo de dados. Elas são: "*_version*"(para determinar a versão do modelo de dados suportado pelo LMS),"*_children*" (para determinar os elementos contidos em um nó de dados) e "*_count*" (que determina o número de elementos contidos em uma lista);
- o Modelo de Dados estabelece que todas as listas comecem com "0" e os elementos devem ser inseridos nessas listas de maneira seqüencial;
- cada SCO possui sua implementação de modelo de dados. Desta maneira, cada SCO dispõe de campos específicos para guardar os seus dados e não pode acessar a dados de outros SCOs.

A Tabela 5.2 mostra os elementos mais importantes (cmi.core, cmi.objectives e cmi.interactions) do modelo de dados com sua descrição (CMI, 2005).

**Tabela 5.2:** CMI Data Model (SCORM, 2005) (CMI, 2005)

| cmi.interactions<br>contém a informação sobre as interações dos alunos com o SCO,<br>exemplo: LMSSetValue ("cmi.interactions.1.type", "*true-false*"); | |
|---|---|
| cmi.interactions.n.id | Campo com a identificação da interação. |
| cmi.interactions.n.time | Armazena o tempo utilizado pelo aluno para completar a interação. |
| cmi.interactions.n.type | Campo para identificar o tipo de interação que é armazenada neste registro. (aceitando apenas 7 valores). Vocabulário CMI: *true-false* (questões falso ou verdadeiro); *choice* (questões de múltipla escolha); *fill-in* (questões com resposta de completar "uma palavra"); *matching* (questões associativas); *performace* (semelhante às questões de múltipla escolha, mas a resposta deve ser dada pela realização de uma tarefa ou ação); *likert* (grupo de alternativas dentro de uma série contínua); e *numeric* (quando a resposta é um valor numérico). |
| cmi.interactions.n.correct _responses.n.pattern | Contém as possíveis respostas de uma interação "n". Permite mais de uma resposta correta. |
| cmi.interactions.n.weighting | Campo numérico que atribui pesos diferentes a cada interação. |

| | |
|---|---|
| cmi.core<br>contém a informação mais relevante para os SCO,<br>exemplo: var coreChildren = LMSGetValue("cmi.core._children"); | |
| cmi.core.student_id | Armazena um código (ID) alfanumérico que identifica um determinado aluno. (pode ser número de matrícula). |
| cmi.core.student_name | Campo que armazena o nome completo do aluno. |
| cmi.core.lesson_location | Armazena o ponto do SCO que aluno estava visualizando quando abandonou a sessão. O valor é passado para o LMS quando o aluno sai do SCO e pode ser utilizado quando ele retornar. |
| cmi.core.lesson_status | Possui o estado de cada SCO (aceitando apenas 6 valores): *passed* (passou); *completed* (completo); *failed* (falhado); *incomplete* (incompleto); *browsed* (no navegador, sendo acessado neste momento); e *not attempted* (não acessado). |
| cmi.core.entry | Informa se o aluno já visitou o SCO (aceitando apenas 3 valores): *ab-initio* (SCO nunca foi visitado); *resume* (SCO já foi visitado); e " " (*String* vazia, para outros casos). |
| cmi.core.score | Armazena o desempenho do aluno, possuindo três filhos: "*raw*", "*min*" e "*max*". São utilizados nos SCOs que implementam a avaliação do aluno (como teste, exercícios e exames) |
| cmi.core.score.raw | Armazena o resultado da avaliação do aluno no SCO. |
| cmi.core.socre.min | Contém o valor mínimo que um aluno pode obter em uma avaliação no SCO (aceita inteiro de 0 a 100). |
| cmi.core.score.max | Contém o valor máximo que um aluno pode obter em uma avaliação no SCO (aceita inteiro de 0 a 100). |
| cmi.core.total_time | Campo que contém o tempo acumulado de todas as sessões do aluno no SCO. (formato: HHHH:MM:SS.SS). |
| cmi.core.exit | Campo que possui o motivo ou forma que o aluno abandonou (saiu) a sessão (aceitando apenas 3 valores). Vocabulário CMI: *time-out*; *suspend*; e "" (*String* nula, quando a saída é feita ao completar o SCO). |
| cmi.core.session_time | Campo que contém o tempo gasto pelo aluno numa sessão. |
| cmi.objectives<br>contém a informação sobre o desempenho dos alunos quanto aos objetivos propostos do SCO, exemplo: LMSSetValue ("cmi.objectives.1.status", "*failed*"); | |
| cmi.objectives.n.id | Campo com a identificação definida pelo criador do SCO. (podendo ser qualquer texto: P1, 1, objetivo_1 ou etc.). |
| cmi.objectives.n.score | Semelhante ao campo cmi.core.score (Possui três filhos: "*raw*", "*min*" e "*max*"). |
| cmi.objectives.n.status | Campo que armazena o desempenho do aluno para um determinado objetivo "n". (aceitando apenas 6 valores): *passed* (passou); *completed* (completo); *failed* (falhado) - *incomplete* (incompleto) - *browsed* (no navegador, sendo acessado neste momento) - *not attempted* (não acessado). |

## 5.4 Conclusão

Neste Capítulo, dicutimos os conceitos e características dos Objetos de Aprendizagem Reutilizáveis (*Reusable Learning Objects*), que vêm sendo apontados pela literatura como uma solução eficiente para a diminuição do custo de desenvolvimento de cursos ministrados a distância.

A Figura 5.9 ilustra o grafo de referência entre as especificações do IEEE, IMS, AICC, ADL SCORM, ARIADNE e Dublin Core, entendidas como sendo os principais padrões usados (Pereira et al., 2003). Na Figura 5.9, a seta tracejada deve ser interpretada como "referencia" ou "usa". Pode-se observar que os padrões IEEE-LOM (*Learning Object Metadata*), IMS e SCORM (nessa ordem) definem uma estrutura base, onde o padrão representado mais acima "contém" o mais abaixo.

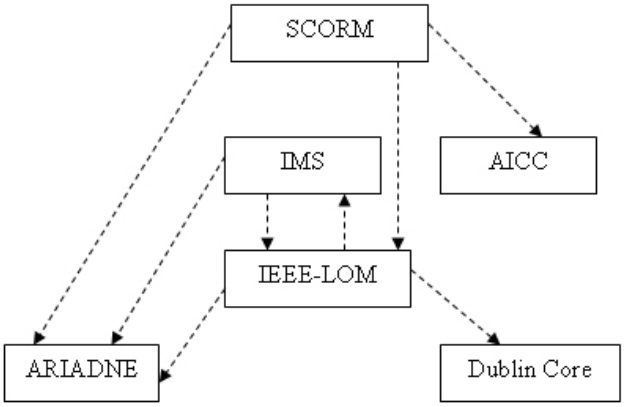


**Figura 5.9:** Grafo de referência entre as especificações do IEEE, IMS, AICC, SCORM, ARIADNE e Dublin Core, adaptado de (Pereira et al., 2003).

Foram discutidos, também, tipos de metadados educacionais, com destaque especial à proposta fornecida pela ADL (*Sharable Content Object Reference Model*).

# CAPÍTULO 6

# Trabalhos Relacionados

Neste Capítulo serão apresentados alguns trabalhos que possuem algum relacionamento ao que foi desenvolvido nesta pesquisa. Começaremos por analisar a estratégia de *e-learning* utilizada pela Cisco, que é um dos pioneiros do uso de objetos de aprendizagem. Logo após, é apresentada a adequação para a norma SCORM da plataforma de *e-learning* do Grupo Portugal Telecom. Em seguida, será apresentada uma ferramenta (*Toolkit*) para a construção de objetos de aprendizagem (padrão SCORM). Por fim, será mostrada uma iniciativa brasileira (latino-americana) de repositório de objetos de aprendizagem.

## *6.1* Introdução

Diversas iniciativas estão sendo desenvolvidas para difundir os objetos de aprendizagem, inclusive através do desenvolvimento de *software*, como forma de apoiar a sua criação e sua distribuição. Porém, uma das dificuldades encontradas para o cumprimento deste objetivo ocorre no momento de se reaproveitar recursos educacionais já existentes, por isso, existem poucos ambientes e soluções que promovem o reaproveitamento de recursos educacionais.

Os trabalhos relacionados têm como principais objetivos: mostrar inciativas e soluções que auxiliem o entendimento das tecnologias e conceitos envolvidos nesta pesquisa; detectar o que a comunidade científica está produzindo na área da Informática da Educação e no uso de objetos de aprendizagem; e, por fim, investigar e identificar lacunas e eventuais necessidades que esses trabalhos possam ter, possibilitando, assim, o desenvolvimento desta pesquisa.

O Capítulo inicia mostrando uma das primeiras soluções de *e-learning* a utilizar o conceito de objetos de aprendizagem, que apesar de se tratar de uma proposta proprietária da Cisco, essa solução traz grandes contribuições sobre o entendimento deste conceito. O segundo trabalho relacionado mostra a adequação para a norma SCORM de um ambiente de EAD Português. Em seguida, são mostrados alguns serviços da ferramenta *Toolkit* que trabalham na construção de objetos de aprendizagem, permitindo visualizar suas funcionalidades e limitações. No último trabalho é possível entender o funcionamento de um repositório de objetos de aprendizagem.

## 6.2 A Proposta da Cisco Systems para Objetos de Aprendizagem Reutilizáveis (RLOs)

Esta Seção faz um estudo da pesquisa da Cisco "*Reusable Learning Object* (RLO) *Strategy*" (Kelly and Barritt, 1998)(Barrit, 2001). Na proposta da Cisco, é reconhecida a necessidade de abandonar-se a prática da criação e apresentação de cursos a distância compostos por blocos de treinamento grandes e inflexíveis, passando a utilizar objetos de aprendizagem armazenáveis em banco de dados que podem ser reutilizados, pesquisados e modificados por diversos ambientes de EAD, tornando-se, assim, a solução da Cisco para *e-learning*. Possibilita também que a Empresa desenvolva, gerencie e entregue todo o tipo de informação e treinamento via Internet (WWW). Outra característica interessante dessa solução é o fato dessa arquitetura ser independente de plataforma e de ferramenta, suportando uma variedade de aplicações de autoria tradicional e eletrônica, de entrega e de gerência de aprendizado.

A Figura 6.1 ilustra a arquitetura de solução para *e-learning* da Cisco, a qual contém as seguintes camadas: acesso, plataforma de aplicação e infra-estrutura de rede. Cada camada será explicada com mais detalhes a seguir:

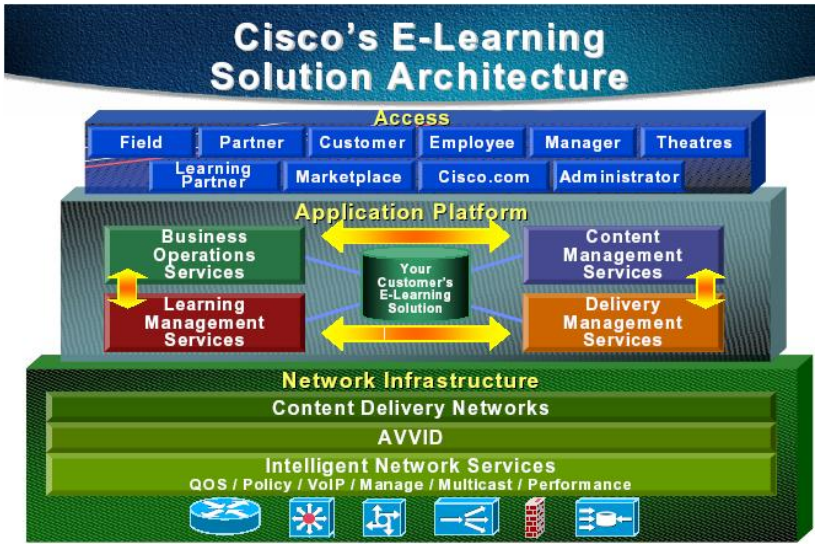


**Figura 6.1:** Arquitetura de solução para *e-learning* da Cisco (Kelly and Barritt, 1998).

- camada de acesso - é responsável por atender a múltiplas audiências. Cada uma dessas audiências pode ter seus próprios esquemas ou projetos de portais de acesso adaptados às necessidades específicas. Outra função dessa camada é fornecer um canal de comunicação com os usuários que acessam a solução de *e-learning*;
- camada de plataforma de aplicação - trata-se de uma camada intermediária que é responsável pela criação, armazenamento, gerência e entrega de um objeto de aprendizagem. Essa camada está dividida em quatro sub-partes: processos de concepção, desenvolvimento, entrega e gerência em áreas de responsabilidade distintas. As sub-partes são: serviços de operações de negócio, gerência de conteúdo, gerência de entrega e gerência de aprendizado;
- camada de infra-estrutura de rede - é responsável pela distribuição dos serviços providos pela camada de aplicação. Além disso, essa camada deve garantir disponibilidade, escalabilidade

e desempenho. A camada possui muitas funcionalidades relativas a rede, como possibilitar o tráfego de dados, voz e vídeo, assim como gerenciar a segurança do ambiente, Qualidade de Serviço (QoS) e a distribuição de conteúdo de aprendizagem.

### 6.2.1 Criação de Conteúdo de Aprendizagem

Na arquitetura da Cisco, um importante ponto são os Serviços de Gerência de Conteúdo (Kelly and Barritt, 1998)(Barrit, 2001), os quais permitem que os "provedores de conteúdo" criem, registrem, gerenciem e publiquem seus conteúdos de aprendizagem. Na arquitetura, o registro de conteúdo deve ser, sem exceção, executado antes da publicação. Funciona inicialmente pelo envio ao gerenciador e sua categorização e rotulação (descrição), através do uso de metadados.

A arquitetura ressalta a importância da integração da ferramenta de construção de conteúdos tradicionais, como Word, PowerPoint, Flash e DreamWeaver, entre outros, com os serviços de gerência. Além disso, o gerenciador de armazenamento deve suportar dados sem estrutura como: imagens, programas executáveis e arquivos binários quaisquer.

Deve-se dispor de mecanismos de busca dos conteúdos de aprendizagem armazenados para que se possa facilmente localizá-los e utilizá-los na montagem de cursos, capítulos, seções, entre outros, sempre priorizando a reusabilidade desses conteúdos.

O *Internet Learning Solutions Group* (ILSG) da Cisco combina Objetos de Informação Reutilizáveis (RIOs), no intuito de formar estruturas de aprendizagem maiores, denominadas pela Cisco de RLOs. A Figura 6.2 mostra um RLO que encapsula um módulo, o qual possui uma visão geral do RLO, um módulo que o sumariza, um módulo que permite a avaliação do aproveitamento por parte do aluno de 5 a 9 ($7 \pm 2$) RIOs (Barritt, 2001).

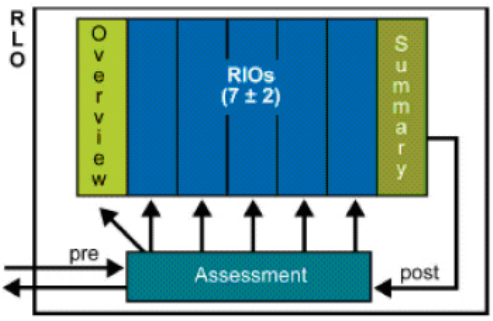


**Figura 6.2:** Estrutura Interna do RLO (Barritt, 2001).

A Figura 6.3 mostra, da proposta da Cisco, que um RLO tem um único objetivo pré-definido dentro de uma seqüência definida de RLOs que formam uma "lição" (Barritt, 2001). Na ilustração, se observa que um RIO é composto por itens: **conteúdo, prática e avaliação**. Além disso, cada RIO é desenvolvido de forma a cumprir uma seqüência que atende ao objetivo geral do RLO.

A Cisco utiliza, como é ilustrado na Figura 6.4, uma hierarquia de níveis em agregação de conteúdo de aprendizagem, que vai de currículo, unidade, módulo, lição (RLO) e seção (RIO), possibilitando combinar RIOs para formar estruturas maiores, tais como módulos, unidades e cursos. Assim, a estrutura RLO/RIO define uma hierarquia em dois níveis. A ILSG adotou a nomenclatura "lição" para denominar um RLO e o de seção para cada RIO dentro de uma lição. Os RLOs

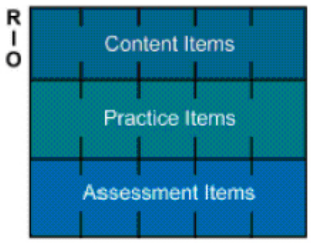


**Figura 6.3:** Estrutura interna de um RIO (Barritt, 2001).

se agrupam em estruturas maiores, dependendo da necessidade, dos requisitos de negócio e do empacotamento da oferta no sistema de gerência de aprendizado.

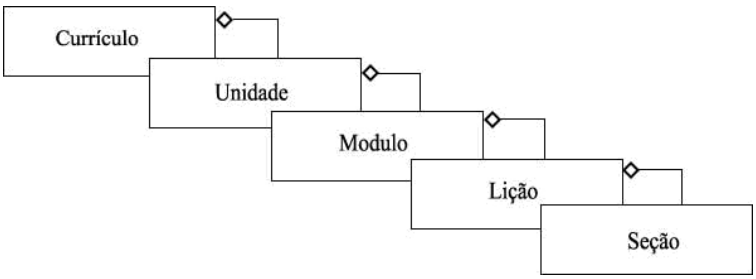


**Figura 6.4:** Modelo da Cisco para níveis de agregação em conteúdos de aprendizado, adaptado de (Kelly and Barritt, 1998).

## *6.3* Portugal Telecom Inovação e o FORMARE

A Portugal Telecom Inovação (PT Inovação) é responsável, desde 1993, pela Formação Tecnológica e de Serviços (FTS) do Grupo Portugal Telecom[1], sendo considerada uma das pioneiras no uso e desenvolvimento de *e-learning* em Portugal (Barbeira and Santos, 2002).

A PT Inovação tem seguido, desde 1996, uma metodologia pedagógica que está em constante evolução, seguindo as teorias de EAD de Moore e Keegan (Barbeira and Santos, 2002). Essa metodologia foi estabelecida pelo Ministério da Educação de Portugal, sendo implementada no projeto Formare[2] e programa Prof2000[3].

A concepção de cursos para auto-aprendizagem e *e-learning* depende, normalmente, de uma equipe de trabalho pluridisciplinar e especialistas em tecnologia de formação para a criação de material educacional. Desta maneira, adequar um curso tradicional (presencial) para o formato de ensino a distância não é uma tarefa fácil, sendo necessárias preocupações como: cativar, motivar e estimular remotamente o aluno.

Assim, a PT Inovação desenvolve seus próprios conteúdos multimídia, desde 1996, com um variado conjunto de conteúdo educacional:

[1]Portugal Telecom - http://www.telecom.pt/
[2]Formare - http://www.formare.pt/
[3]Prof2000 - http://www.prof2000.pt/

- "*Multimedia System and Services*; LAN e WAN; *Data Communication*; SDH; ATM; IP; *Telecommunication*, ISDN; SS7; *Digital Switching*; *Corporate Service and Networks*; *Telecommunication Management Network*; PBX; *High Bandwidth Networks*; Internet; *e-Learning* and Formare"(AICC, 2005).

Com o objetivo de redução dos custos de criação de conteúdo e possibilitar a incapacidade de reaproveitar os conteúdos já desenvolvidos, ou parte dele, a PT Inovação passou a estudar a normalização da criação de conteúdo, para possibilitar a incapacidade de reutilizar conteúdos de aprendizagem, uma vez que suas implementações (Formare) de *e-learning* não seguiam nenhuma norma.

A partir do ano de 2001, a PT Inovação adotou a norma SCORM para a normalização da criação de conteúdos de *e-learning* (Barbeira and Santos, 2002) (Formare, 2005). Desta maneira, foi possível o desenvolvimento de um ambiente de navegação uniforme para diferentes cursos desenvolvidos de acordo com a norma, separando o que é conteúdo de aprendizagem do que é estrutura.

### 6.3.1 Adequação da Plataforma Formare

A PT Inovação adota o Formare para criação de solução de *e-learning*, porém, o Formare não foi projetado para atender as especificações do SCORM, resultando na necessidade de adequar essa ferramenta para a norma. A adaptação da plataforma FORMARE de modo a responder aos requisitos técnicos especificados pelo SCORM foi planejada em 6 fases (Barbeira and Santos, 2002):

- funcionalidade para disponibilizar conteúdos de acordo com a estrutura definida no manifesto desses mesmos conteúdos;
- implementação de modelo de dados definido pela norma SCORM, no lado do servidor e lado cliente;
- desenvolver as funcionalidades de comunicação de dados entre conteúdos e a plataforma (API *Adapter*);
- criação de mecanismo de prover (gerir) conteúdos dentro da plataforma FORMARE:
  - inserção de conteúdos no sistema;
  - associação de conteúdos a ações de ensino;
  - validação dos conteúdos inseridos com a norma SCORM (trabalha a desenvolver);
  - criação e pesquisa num repositório de conteúdos de aprendizagem normalizados (trabalha a desenvolver);
- implementação de ferramenta de relatório (*report*) para alunos, tutores, coordenadores e administradores;
- indexação das ações de ensino aos conteúdos (trabalha a desenvolver).

#### 6.3.1.1 Disponibilizar conteúdos

A primeira fase de adequação da plataforma Formare foi possibilitar que o ambiente conseguisse disponibilizar conteúdos de aprendizagem normalizados (SCOs), seguindo a estrutura definida pelo manifesto do SCO.

Esse desenvolvimento teve a implementação de um mecanismo de criação dinâmica da estrutura de navegação, a partir da leitura do manifesto de qualquer curso desenvolvido de acordo com a norma SCORM 1.2 (Barbeira and Santos, 2002).

O mecanismo implementado realiza uma análise XML do manifesto do curso, mapeando a estrutura e respectiva ligação aos componentes físicos (arquivos) que constam no manifesto analisado, como é mostrado na Figura 6.5.

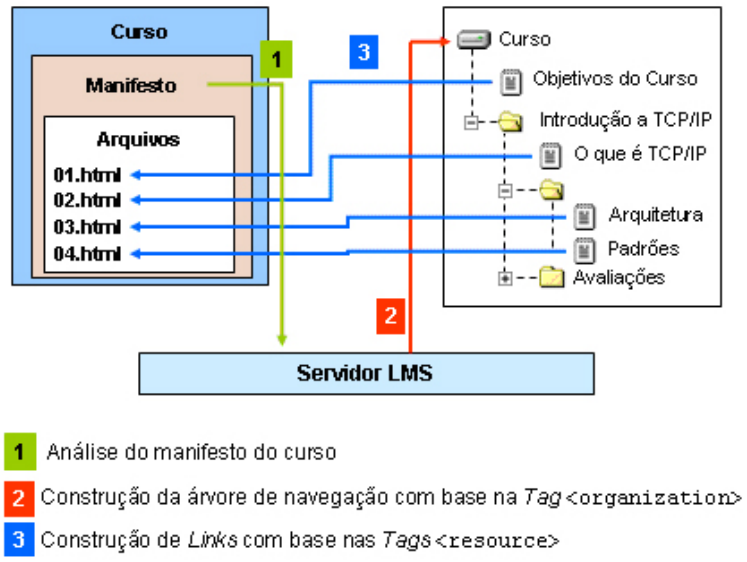


**Figura 6.5:** Estrutura dinâmica de navegação, adaptado de (Barbeira and Santos, 2002).

Assim, é possível criar um ambiente de navegação uniforme para diferentes cursos desenvolvidos de acordo com a norma, sempre respeitando as limitações da linguagem HTML que a norma SCORM estabelece.

#### 6.3.1.2 Modelo de Dados

A segunda etapa da adequação foi no desenvolvimento do modelo de dados de acordo com a norma SCORM [Barbeira, 2002]. Foi optado por retirar a carga do servidor, implementando o modelo de dados com um conjunto de objetos definidos em JavaScript e disponíveis através do *Document Object Model* (DOM[4]) do navegador do cliente.

No servidor, foi montada uma estrutura de base de dados com a capacidade de guardar os dados armazenados no DOM sempre que o cliente (conteúdo) ordene a persistência dos mesmos. Foram implementados dois métodos para tratar a manipulação dos dados, um de "*set*" e outro de "*get*" dos valores, tendo esses métodos embutido a lógica de permissão. A Figura 6.6 mostra como esses métodos funcionam na plataforma.

[4]W3C *Document Object Model* - http://www.w3.org/DOM/

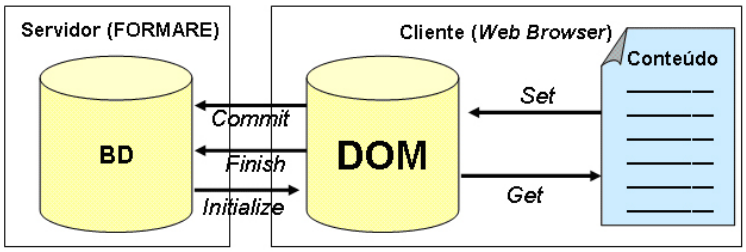


**Figura 6.6:** Arquitetura e Modelo de Dados (Barbeira and Santos, 2002).

Essa solução permite que os dados sejam sempre manipulados do lado do cliente, apenas efetuando comunicação com o servidor na iniciação e finalização das sessões e nas chamadas LMSCommit(“”). Mesmo que essa solução prejudique a portabilidade, trata-se de uma solução válida para a norma.

#### 6.3.1.3 *Application Program Interface* (API)

A API realiza as funcionalidades para a gestão do modelo de dados, respeitando a norma SCORM. Como o modelo implementado utiliza o DOM no navegador, deve ser acessado via JavaScript.

Como já foi mencionado, foram implementados os comandos LMSGetValue e LMSSetValue do lado cliente, onde se passam as funções “*get*” e “*set*” dos parâmetros passados, identificando se a chamada provém do sistema ou do SCO.

Os comandos LMSInitialize, LMSFinish e LMSCommit foram implementados no servidor. Essas funções realizam a comunicação com o servidor de base de dados. Foram utilizados componentes ADOdb[5] via ASP para desenvolver a API e também se utilizou o *Remote Scripting* (ActiveX) para permitir as funcionalidades do lado servidor (ASP) através do cliente (JavaScript), para evitar a necessidade de atualização da página para realizar as comunicações com o servidor.

#### 6.3.1.4 Gestão de Conteúdo

Após a implementação do API, foram desenvolvidas as funcionalidades para a gestão de conteúdo dentro da plataforma Formare, sendo que esse trabalho não possui uma versão final (Barbeira and Santos, 2002).

A plataforma suporta conteúdos de aprendizagem normalizados ou não, onde ao inserir um novo conteúdo, deve ser informado se é normalizado ou não. No caso de conteúdo normalizado, é necessária a criação de arquivo de manifesto (imsmanifest.xml), que indica a estrutura de navegação a ser construída.

Após essa tarefa, é necessário associar os diversos conteúdos às ações de ensino da área correspondente, para disponibilizar aos alunos. Na inclusão de um conteúdo normalizado, é necessário realizar essa gestão, definindo uma série de parâmetros para os vários SCOs que constituem o conteúdo:

- se o SCO possuir avaliações, este deve ser armazenado no Bando de Dados;

[5] ADOdb é uma abstração de base de dados

- quantas oportunidades o aluno terá para realizar o SCOs;
- qual é o peso de cada SCO na nota final.

#### 6.3.1.5 Relatórios

Este módulo tem como objetivo disponibilizar na plataforma Formare uma série de ferramentas que possibilitam o rastreamento da atividade dos alunos ao longo dos conteúdos. Esse acompanhamento possibilita um modo de gerir as informações relevantes (estabelecidas pela norma) sobre a atividade dos alunos nos conteúdos.

## *6.4* Click2learn ToolBook

A Click2learn foi fundada, em 1984, por Paul Allen (co-fundador da Microsoft) da *Asymetrix Learning Systems Inc.*. A Click2learn desenvolveu os sistemas *Aspen Enterpriser Learning Platform* e ToolBook com a finalidade de possibilitar às Organizações e Empresas a criação, disseminação e administração de suas atividades de aprendizagem e gestão do conhecimento através de uma arquitetura única (E-learning, 2005).

Atualmente, a Click2learn se associou a Docent, resultando na SumTotal *System*[6], empresa líder em soluções de produtividade voltada para as 2.000 principais organizações Globais e agências governamentais que buscam a melhoria no desempenho dos negócios e a produtividade da força de trabalho. Ela provê soluções para *Accenture, American Airlines, Anheuser Busch, AT&T, Boeing, Century 21, Fujitsu, Microsoft, Pfizer, the National Guard (Guarda Nacional), The Thomson Corporation, Symantec, Verizon , Vodafone* e etc..

O ToolBook é um popular programa para o desenvolvimento de sistemas educacionais (*courseware*), sendo definido por Crawford (Crawford, 2001) como o "principal provedor de serviços para soluções de *e-learning* destinados a empresas, agências governamentais e instituições educacionais ao longo do mundo". O ToolBook vem sendo desenvolvido e melhorado desde 1984 e, como conseqüência, é um produto extensamente usado, sendo responsável pela produção de uma grande quantidade de material educacional.

### 6.4.1 Neuron da plataforma ToolBook

A SunTotal identificou na Internet um grande potencial para sistema educacional, pois dispensa a necessidade de ter instalado localmente sistema de *e-learning* (*software*). Assim, para que a Empresa disponibilize suas soluções neste meio (Internet), foi criado o Neuron, que possibilita que os arquivos do ToolBook trabalhem sobre a Internet (Ostyn, 2003).

O Neuron é um "*plug-in*" ou ActiveX *control* para navegadores *web* (*web browsers*). Ele permite executar o ToolBook e suas ferramentas na Internet ou qualquer rede TCP/IP sem a necessidade de baixar (*download*) e instalar os arquivos do sistema (ToolBook). O Neuron é compatível com os navegadores *web* (*web browsers*): Microsoft Internet Explorer 4.0 (ou superior) e Netscape Navigator 4.0 (ou superior) (Neuron, 2005) (Crawford, 2001).

[6]SumTotal *System* - http://www.sumtotalsystems.com/

### 6.4.2 Ferramenta do ToolBook

A SunTotal possui diversas soluções e implementações voltadas a *e-learning*, porém duas dessas soluções se destacam por fornecer soluções que se assemelham a proposta deste trabalho, que são as ferramentas para recurso SCORM (SCORM *Resource Kit*) do ToolBook: o SCORMisizer e o SCORMAggregator.

#### 6.4.2.1 SCORMisizer

O SCORMisizer é um "*wizard*" (assistente) para transformar um documento ou um *asset* em um pacote SCORM de maneira simples e fácil. O pacote criado por esse programa é um pacote SCORM 1.2 que inclui um manifesto e metadado (Academia_E-learning, 2005).

Essa ferramenta pode transformar arquivos texto (.txt), arquivos gráficos (.jpg, .gif, etc.), páginas *web* (.htm, .html), etc. em pacotes SCO. Dependendo das configurações do servidor e dos *plug-ins* disponíveis no navegador do cliente, outros formatos de arquivos podem ser entregues, como Adobe Acrobat (.pdf), arquivos de mídia, documentos do Microsoft Word ou Excel, etc..

Apresentações em Microsoft PowerPoint 2000 ou PowerPoint XP salvos como HTML podem ser utilizados pelo SCORMisizer para se transformarem em pacotes de SCORM.

**Requisitos do sistema:** o SCORMisizer é uma aplicação do ToolBook e necessita de alguns requisitos para ser executado. Necessita do Microsoft Windows e requer ou o ambiente de desenvolvimento ToolBook 8.6 (ou superior) ou o programa Neuron 8.5 (ou superior). O Microsoft MSXML 3.0 ou MSXML 4.0 também é requerido. No caso de utilização do Microsoft Internet Explorer 5.5 (ou superior), MSXML já é instalado no sistema.

A Figura 6.7 mostra o funcionamento da ferramenta: em (a), inicialmente, é necessário selecionar, localmente no computador, um *Asset* compatível (mídia que possa ser visualizada em um navegador, como um arquivo: html, flash, pdf, txt, gif, jpg, etc.). Em seguida é necessário em (b) criar o metadado associado a esse recurso. Abrindo o editor de metadado, observe que em (c) pode se acessar todas as categorias estabelecidas pela norma e em (d) são inseridas as informações. Ao finalizar essa operação é gerado o metadado XML do SCO (arquivo: "metadado.xml") e, por fim, em (e) é necessário criar o pacote SCORM onde o *Asset* é inserido em um arquivo html junto com o código JavaScript necessário para a comunicação do SCO com o LMS e ainda são gerados os arquivos XML esquemas ("adlcp_rootv1p2.xsd", "ims_xml.xsd", "imscp_rootv1p1p2.xsd" e "imsmd_rootv1p2p1.xsd").

O *Asset* e os arquivos gerados são armazenados em uma pasta no disco local. Para se enviar o SCO gerado a um repositório, é necessário compactar manualmente todos os arquivos em um arquivo PIF antes de enviá-lo.

#### 6.4.2.2 SCORMAggregator

O SCORMAggregator também é um "*wizard*" (assistente) para gerar *Content Aggregation*, que consiste em um conjunto de regras e normas para agregar conteúdo de ensino (SCO) em um bloco (por exemplo, um curso, um capítulo, um módulo, etc.), com o objetivo de permitir a transferência desses mesmos blocos entre sistemas diferentes (Academia_E-learning, 2005).

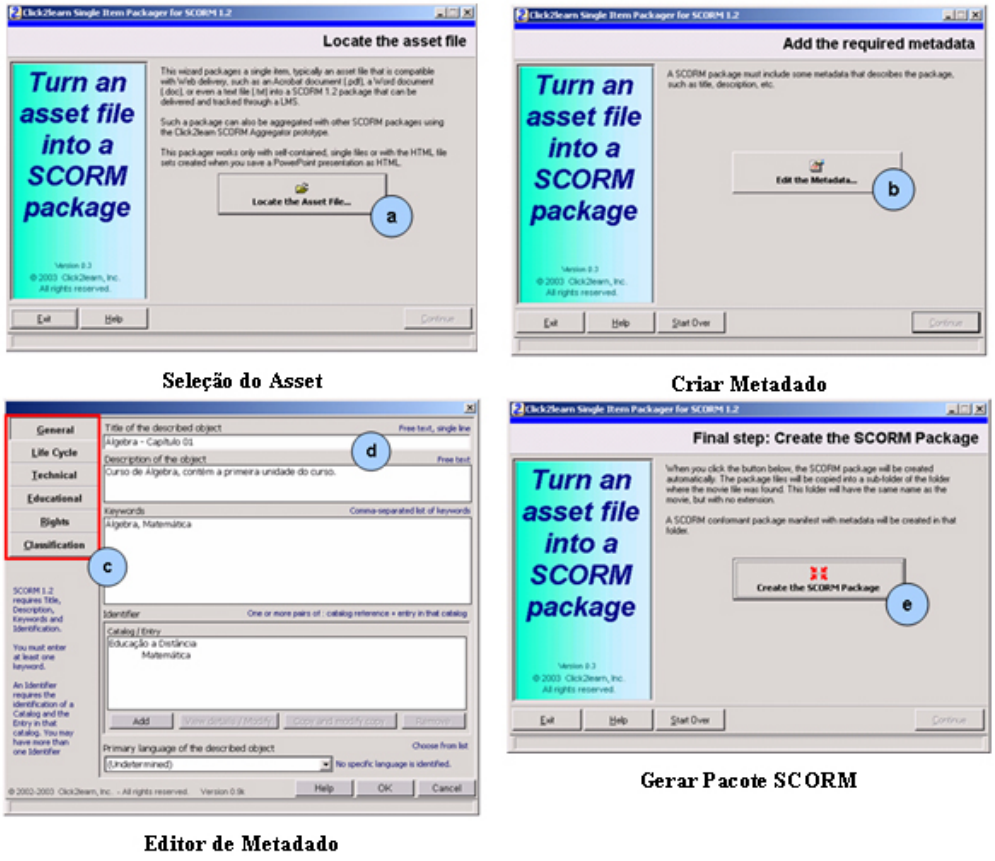


**Figura 6.7:** Telas e funcionamento do SCORMisizer.

A ferramenta permite trabalhar com SCOs produzidos pelo SCORMisizer ou criados por outras ferramentas, inclusive pacotes complexos. Permite definir metadado para o pacote agregado e organizar a estrutura do pacote de um modo seqüencial ou hierárquico.

**Requisitos do sistema:** o SCORMAggregator também é uma aplicação do ToolBook e necessita dos mesmos requisitos do SCORMisizer para ser executado. Necessita do Microsoft Windows e requer ou o ambiente de desenvolvimento ToolBook 8.6 (ou superior) ou o programa Neuron 8.5 (ou superior). O Microsoft MSXML 3.0 ou MSXML 4.0 também é requerido. No caso de utilização do Microsoft Internet Explorer 5.5 (ou superior), MSXML já é instalado no sistema.

A Figura 6.8 mostra o funcionamento da ferramenta: em (a) inicialmente é necessário selecionar a pasta que contenha os SCOs, sendo que todos os recursos necessários para gerar o bloco devem, obrigatoriamente, estar na mesma pasta do disco local; em (b) é mostrada a estrutura do bloco, podendo ser seqüencial ou hierárquico. É necessário criar o metadado associado ao bloco (curso ou capítulo), sendo que esse metadado vai conter o objetivo e a descrição do pacote agregado, utilizando o editor de metadado (o mesmo utilizado da ferramenta SCORMisizer); em (d) pode se acessar todas as categorias estabelecidas pela norma e em (e) são inseridas as informações. Ao finalizar essa operação, é gerado o metadado XML do SCO (arquivo: "metadado.xml") e, por fim, em (f) cria o pacote agregado SCORM (*Content Aggregation*), onde todos os SCOs são colocados em sub-pastas e são gerados os arquivos XML esquemas ("adlcp_rootv1p2.xsd",

"ims_xml.xsd", "imscp_rootv1p1p2.xsd" e "imsmd_rootv1p2p1.xsd") juntamente com o manifesto ("imsmanifest.xml") que possui a estrutura do pacote agregado.

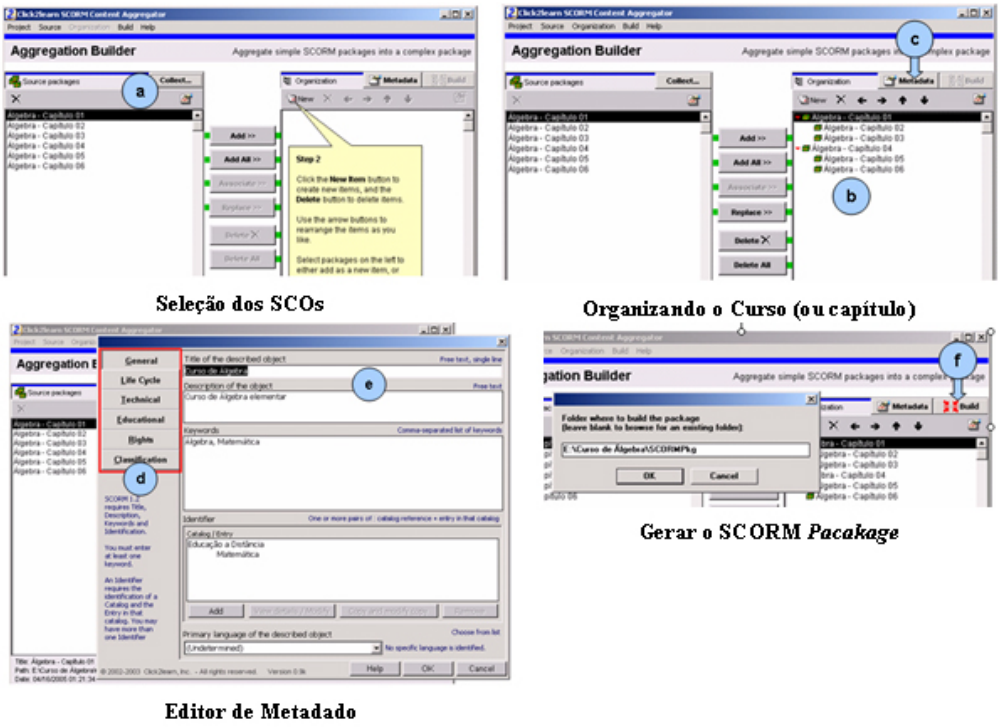


**Figura 6.8:** Telas e funcionamento do SCORAggregator.

## 6.5 Rede Internacional Virtual de Educação (RIVED)

A Rede Internacional Virtual de Educação para o Melhoramento da Aprendizagem de Ciência e Matemática na América Latina (Rived), inicialmente também denominada *International Virtual Education Network* (IVEN), foi criada para se constituir em programa de aperfeiçoamento e de apoio à distância ao trabalho de professores que atuem no ensino de Matemática, de Biologia, de Física e de Química (Menezes, 2004)(Nascimento, 2004).

O projeto nasceu de uma iniciativa proposta dentro do acordo Brasil-Estados Unidos sobre o desenvolvimento de tecnologia para uso pedagógico, assinado em 1997. Seu planejamento iniciou em 1999 e foi lançado em 2000. Os países participantes são Brasil, Venezuela, e Peru (este último substituindo a Colômbia). O RIVED é patrocinado por recursos dos países participantes, mas no seu inicio, foi assistido com recursos do Banco de Desenvolvimento Americano e da UNESCO.

No Brasil, o projeto é desenvolvido pelo Ministério da Educação do Brasil por meio da Secretaria de Educação a Distância (SEED) e da Secretaria de Educação Básica (SEB). O material pedagógico é elaborado por uma equipe multidisciplinar, contando com especialista em conteúdo, *designer* instrucional, pedagogo, programadores, ilustrador e *designers* de multimídia. É responsabilidade desse grupo o desenvolvimento e seqüência de atividades de ensino que contemple os objetivos de aprendizagem do currículo do ensino médio brasileiro (RIVED, 2005).

### 6.5.1 Especificação

O objetivo mais importante do projeto RIVED é permitir o compartilhamento e a reutilização de suas produções tanto no aspecto pedagógico como no aspecto técnico. Buscando esse objetivo, o projeto optou por adotar especificações de padrões educacionais, que documentam um objeto de aprendizagem. Esses padrões configuram uma estrutura conceitual de dados e aspectos técnicos.

As especificações selecionadas são:

- *Instructional Management System* (IMS):
  - a especificação IMS *Learning Resource Meta-Data Information Model* preocupa-se em documentar um objeto de aprendizagem fisicamente;
- *Educational Modelling Language* (EML):
  - descreve um objeto educacional em seus termos mais pedagógicos, onde preocupa-se em documentar o planejamento pedagógico de um Módulo Educacional, as atividades que o compõe e os objetivos educacionais a serem alcançados com a sua utilização.

A partir de estudos e análise sobres as especificações IMS e EML, originou-se uma base de dados que utiliza informações extraídas das duas especificações, permitindo o armazenamento de informações para pesquisa por um objeto de aprendizagem dentro dos dois aspectos de compartilhamento: pedagógico e o técnico. Essa base de dados constitui um repositório, onde os objetos de aprendizagem podem ser acumulados e catalogados para uma distribuição ampla. O uso dessas especificações auxilia a busca, avaliação e utilização dos objetos pelos alunos e professores, além de possibilitar o gerenciamento de cada material criado, como versões, autorias, permissões de uso, entre outros campos.

### 6.5.2 Fábrica Virtual

O projeto RIVED envolve desde o *design* instrucional de ensino/aprendizagem, produção de material pedagógico multimídia, capacitação de pessoal, rede de distribuição de informação até o desenvolvimento de estratégias de avaliação de aprendizagem e do programa. São desenvolvidos módulos (objetos de aprendizagem) que abordam unidades curriculares das áreas de conhecimento específico, que serão disponibilizados aos professores das escolas públicas por meio da Internet (Nascimento, 2004).

Além disso, a expansão desse projeto através da Fábrica Virtual pode facilitar a inclusão digital e social tanto dos alunos quanto dos professores, pois aproxima ambos do computador e fornece um alicerce para conhecimentos futuros.

A Fábrica Virtual é um projeto para produção de objetos de aprendizagem multimídia em institutos de ensino superior. Trata-se de uma expansão do projeto RIVED que facilita o processo de ensino/aprendizagem ao incentivar a produção de módulos pedagógicos virtuais, *softwares* que auxiliem na aprendizagem do conteúdo escolar para as séries do ensino médio (RIVED, 2005).

Por meio de um edital disponibilizado na Internet, equipes multidisciplinares de 33 universidades públicas inscreveram propostas para criação desses módulos. Dessas, 16 foram selecionadas pela Rived para serem habilitadas em um curso de capacitação. A partir da análise dos objetos entregues,

haverá mais uma seleção, e das 16 instituições inscritas ficarão apenas 12. Essas 12 é que vão, efetivamente, compor a Fábrica Virtual da Rived. Elas assinarão contrato de um ano com o MEC, periodicamente renovável, para uma produção média de 48 módulos anuais.

Os módulos produzidos pelas universidades se juntarão aos já concluídos pela própria Rived e estão disponíveis em um repositório virtual. Por meio desse repositório, hospedado na Internet (http://rived.eproinfo.mec.gov.br), as escolas poderão ter acesso aos objetos de aprendizagem produzidos pela Fábrica Virtual e usá-los como uma ferramenta adicional de ensino (RIVED, 2005).

#### 6.5.2.1 Desenvolvimento dos Módulos de Aprendizagem

Inicialmente, o RIVED buscava para os seus módulos construir uma coleção de material didático, cobrindo todas as unidades do currículo em todo o tipo de mídia. Posteriormente, o foco do projeto mudou e passou a enfatizar mais o desenvolvimento das atividades para uso do computador. Assim, para o melhor uso dos objetos de aprendizagem, o guia do professor que acompanha os módulos sugere várias atividades complementares, práticas de avaliação e material de referência (Figura 6.9).

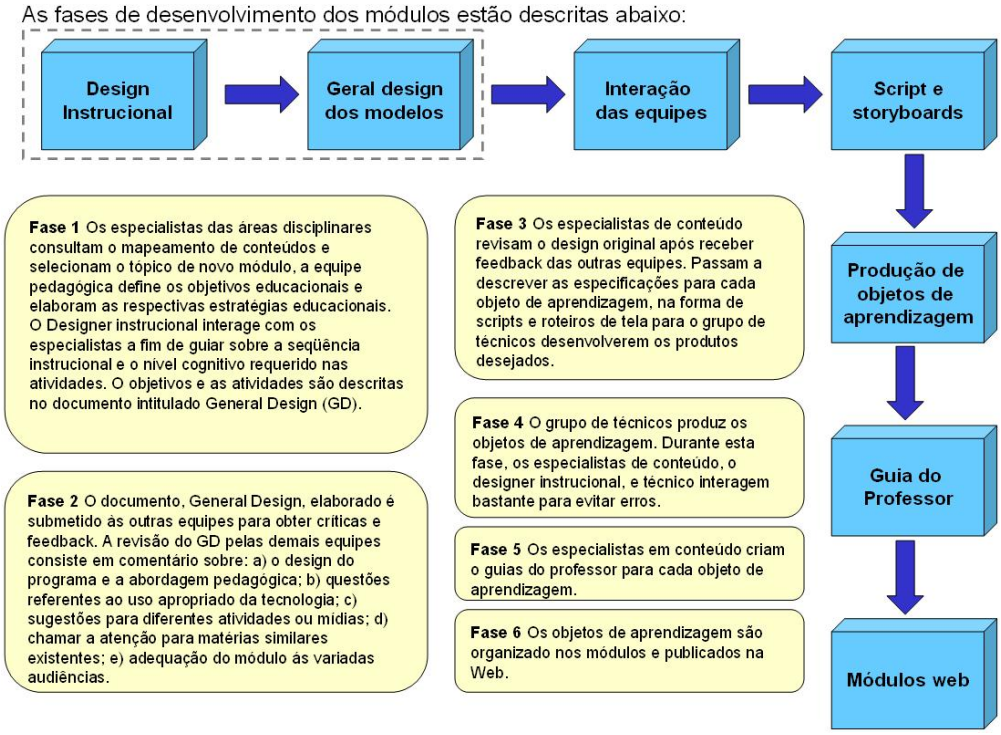


**Figura 6.9:** Processo de produção dos módulos e objetos de aprendizagem (Nascimento and Morgado, 2004).

Fase 1: os especialistas das áreas disciplinares consultam o mapeamento de conteúdos e selecionam o tópico do novo módulo. A equipe pedagógica define os objetivos educacionais e elaboram as respectivas estratégias educacionais. O *designer* instrucional interage com os especialistas a fim de guiar sobre a seqüência instrucional e o nível cognitivo requerido nas atividades. Os objetivos e as atividades são descritos no documento intitulado General *Design* (GD).

Fase 2: o documento General *design* elaborado é submetido às outras equipes para obter críticas e *feedback*. A revisão do GD pelas demais equipes consiste em comentários sobre: a) o *design*

do programa e a abordagem pedagógica; b) questões referentes ao uso apropriado da tecnologia; c) sugestões para diferentes atividades ou mídia; d) chamar a atenção para materiais similares existentes; e) adequação do modulo às variadas audiências.

Fase 3: os especialistas de conteúdo revisam o *design* original após receber *feedback* das outras equipes. Passam a descrever as especificações para cada objeto de aprendizagem, na forma de *scripts* e roteiros de tela, para que o grupo de técnicos desenvolva os produtos desejados.

Fase 4: o grupo de técnicos produz os objetos de aprendizagem. Durante essa fase, os especialistas de conteúdo, o *designer* instrucional e os técnicos interagem bastante para evitar erros.

Fase 5: os especialistas em conteúdo criam o guia do professor para cada objeto de aprendizagem.

Fase 6: os objetos de aprendizagem são organizados nos módulos e publicados na *Web*.

## 6.6 Conclusão e considerações sobre os trabalhos relacionados

Os estudos da Cisco mostram a importância do uso de objetos de aprendizagem reutilizáveis em um ambiente de EAD. O trabalho realizado na Portugal Telecom Inovação mostra as etapas e adequações necessárias para adequar um ambiente de *e-learning* para a norma SCORM, possibilitando a obtenção de subsídios necessários para a elaboração da proposta do sistema. Além disso, mostra como há uma falta de mecanismos que auxiliem essa adequação e a complexidade inerente nessa atividade. O projeto RIVED mostra a importância e complexidade dos repositórios de objetos de aprendizagem na atividade de educação a distância, principalmente em países do dito terceiro mundo.

Já as ferramentas SCORMisizer e SCORMAggregator permitem visualizar claramente como devem ser implementados e quais os serviços são necessários para a adequação de repositórios tradicionais, no intuito de disponibilizar seus recursos educacionais, reaproveitando-os em outros cursos e sistemas (LMS). Isso possibilita definir os serviços que devem ser criados no protótipo, utilizando as idéias e soluções computacionais aplicadas para tal (criação de conteúdo para norma SCORM). Um comparativo entre as ferramentas pode ser conferido na Tabela 6.1 e posteriormente discutido a fim de justificar o desenvolvimento deste trabalho.

**Tabela 6.1:** Tabela comparativa.

| | **SCORMisizer** | **SCORMAggregator** | **Fábrica ACEAS** |
|---|---|---|---|
| Trabalha com objetos locais (no disco rígido) | X | X | |
| Trabalha com objetos remotos (na rede) | | | X |
| Editor de Metadado | X | X | X |
| Gera SCO | X | | X |
| Gera Pacote Agregado | | X | X |
| Gera Pacote SCORM | X | X | X |
| Gera arquivo PIF (compactado) | | | X |
| Gera Repositório SCORM | | | X |

Assim como as ferramentas SCORMisizer e SCORMAggregator, a Fábrica de Adequação trabalha na criação de Objetos de Aprendizagem (LO) para a norma SCORM. A diferença é que a Fábrica é capaz de tratar tanto a criação de um único objeto (SCO) quanto de pacotes agregados (que contêm uma estrutura de SCOs). Além disso, a Fábrica trabalha com objetos de aprendizagem remotos, localizados em repositórios tradicionais (que não respeitam a norma SCORM), adequando esses objetos para a norma e gerando um repositório normalizado (SCORM) para outros cursos e ambientes de *e-learning* (LMS).

A partir dos estudos realizados, elaborou-se a proposta da arquitetura, as funcionalidades e a implementação da Fábrica de Adequação que permitem a interoperabilidade e normalização de objetos de aprendizagem. Essa solução faz uso da tecnologia de *Web Services* e Agentes de *Software* apresentados nos Capítulos anteriores.

CAPÍTULO 7

# Fábrica de Adequação: ambiente de adequação de recurso educacional para Objetos de Aprendizagem SCORM

Nos Capítulos anteriores foram apresentados aspectos relacionados à tecnologia de Agente de *Software*, Meta-dado (SCORM) e *Web Services*, os quais dispõem de mecanismo de autonomia, interação, invocação, acesso e reuso de serviços. Desta maneira, essas tecnologias definem uma estrutura básica para promover a integração entre sistemas.

Este Capítulo apresenta a proposta para promover a interoperabilidade de conteúdos educacionais promovendo a sua normalização para a norma SCORM. A solução utiliza Agentes de *Software* baseados em *Web Services*, os quais permitem uma eficiente descrição de serviços através de suas tecnologias padrões como a WSDL, SOAP e UDDI (todas baseadas em XML).

Trata-se do desenvolvimento de um conjunto de serviços para a manipulação e adequação de recursos educacionais. Este Capítulo também propõe o paradigma de múltiplos contextos para os objetos de aprendizagem, onde se possibilita uma melhor descrição, aproveitamento e evolução dos recursos educacionais.

Assim, integrando essas tecnologias aos conceitos de objetos de aprendizagem, é possível descrever de maneira bastante detalhada qualquer tipo de recurso educacional, facilitando sua interoperabilidade e reaproveitamento.

## *7.1* Introdução

O conteúdo de ensino, dentro de um contexto de EAD, revela-se um dos elementos mais importantes de todo o processo educacional e, por conseqüência, exige cuidados no momento da sua concepção, produção, desenvolvimento e criação, tornando o desenvolvimento desse conteúdo uma atividade relativamente cara ou onerosa (Rosember, 2002).

Promover e estimular o reuso de objetos educacionais tanto diminui os custos quanto melhora a qualidade dos sistemas gerados, pois, com o aproveitamento de conteúdos já existentes e testados

anteriormente, é possível a criação de ambientes de EAD com mais qualidade, rapidez e com baixos custos (Barbeira and Santos, 2002)(Kratz et al., 2005).

Como se pode observar no decorrer deste trabalho, a padronização da apresentação de conteúdos educacionais via *Web* é uma eficiente solução para otimizar e desonerar o processo de aprendizagem a distância. Dentre os diversos esforços, estudos e padrões sobre objetos de aprendizagem e metadados educacionais (Barbeira and Santos, 2002)(Pereira et al., 2003)(Gomes et al., 2004)(Kratz et al., 2005), o SCORM é considerado a especificação mais completa e utilizada no mercado, definindo assim a sua escolha para esta pesquisa.

Para promover a reutilização de conteúdos de aprendizagem (LOs) é necessário promover a normalização desses conteúdos para que esses LOs possam funcionar corretamente em qualquer sistema de *e-learning* (LMS). A normalização permite: uma fácil reutilização; a portabilidade dos conteúdos criados; a padronização dos processos de criação; e a gestão dos conteúdos de aprendizagem. A normalização propicia automatizar o acompanhamento do aluno ao longo do seu curso, tornando possível, em tempo real, gerar toda a informação necessária para que o aluno possa controlar e acompanhar o seu progresso no ambiente de EAD.

Porém, normalizar um ambiente de EAD já existente não é uma tarefa trivial e necessita de uma adequação tanto dos conteúdos de aprendizagem quanto do sistema de *e-learning*, pois toda a arquitetura é alterada, sendo necessário separar o conteúdo da estrutura (Barbeira and Santos, 2002).

Este trabalho busca permitir a adequação de recursos educacionais para Objetos de Aprendizagem Reutilizáveis e possibilitar que um objeto de aprendizagem tenha múltiplos contextos.

## 7.2 Arquitetura de um Repositório de Conteúdo de Aprendizagem Tradicional

Diversos ambientes de EAD surgiram e foram desenvolvidos ao longo dos tempos. Em sua maioria, não foram projetados e nem se preocuparam em criar formas de reaproveitamento dos seus conteúdos educacionais, mas foram desenvolvidos para atender as necessidades e objetivos de sua ementa. A Figura 7.1 mostra que apesar dos diferentes ambientes de EAD disponibilizarem o mesmo curso de "Álgebra", cada solução desenvolveu seu próprio sistema de gerência de aprendizagem (LMS) e seu próprio conteúdo de aprendizagem (LO). A este cenário, dentro desta pesquisa, foi dado o nome de "repositório tradicional".

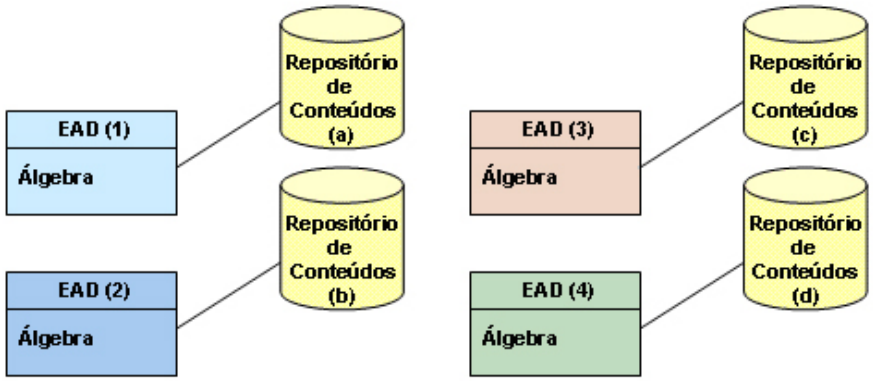


**Figura 7.1:** Ambientes de EAD tradicionais.

Buscando otimizar este cenário, a Figura 7.2 apresenta outro cenário onde, ao invés de cada ambiente de EAD desenvolver seu próprio conteúdo de aprendizagem, exista um repositório de conteúdos compartilhado entre os diversos ambientes. Desta forma, a solução diminui os custos, tempo de desenvolvimento e a quantidade dos objetos educacionais gastos no desenvolvimento de curso a distância e permite a reutilização dos conteúdos de forma racional.

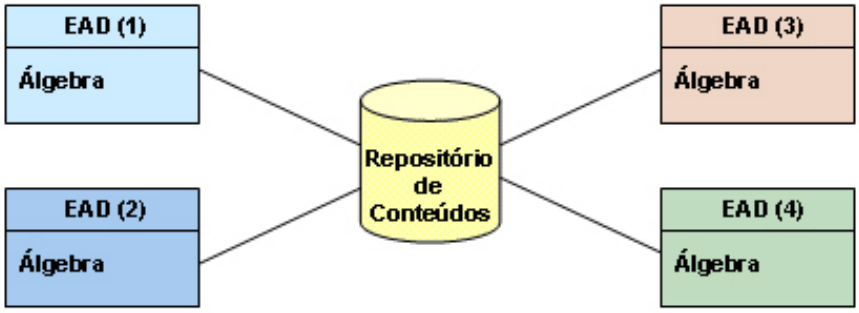


**Figura 7.2:** Ambientes de EAD com repositório compartilhado.

Porém, para que esse novo cenário seja possível, são necessários vários cuidados para que os conteúdos possam ser exibidos em diferentes ambientes de EAD da mesma forma (Albuquerque, 2005)(RIVED, 2005)(IMS, 2005)(LOM, 2005)(ADL, 2005).

## 7.3 Visão geral da arquitetura proposta ACEAS (Ambiente de Construção de Etiquetas de Adaptação SCORM)

O Ambiente de Construção de Etiquetas de Adaptação SCORM (ACEAS) foi criado para possibilitar ao usuário (repositório educacional tradicional) um conjunto de serviços de normalização de recursos educacionais em um ambiente de execução distribuído.

A Figura 7.3 ilustra a arquitetura, mostrando como um único recurso educacional pode ser associado a uma ou mais etiquetas (*labels*) ("Etiqueta 1", "Etiqueta 2"e "Etiqueta 3"). Essas etiquetas funcionam como Agentes *Web Services* que fazem a adequação do recurso para a norma SCORM. As diferenças de adequações podem estar na descrição (metadado) do objeto de aprendizagem e/ou na estrutura de apresentação (manifesto). Assim, os ambientes de *e-learning* (a) e (b) usam o mesmo recurso educacional, mas de maneiras diferentes conforme o agente associado ao recurso, permitindo uma interoperabilidade de ambientes de EAD que seguem o padrão SCORM (Kratz et al., 2005).

A Estrutura do Agente possui uma Camada *Web Service* responsável por fornecer uma interface de acesso aos ambientes de *e-learning* (LMS) e uma Camada de Etiqueta. A Camada *Web Service* (*Web Service Layer*) usa um ObserverAgent que permite que uma única etiqueta (*label*) possa ser acessada por diferentes clientes (LMSs) e trabalha com um *proxy* para o acesso. A Camada Etiqueta (*Label Layer*) possui as adequações SCORM do objeto de aprendizagem.

O ACEAS permite uma melhor utilização de um recurso educacional, possibilitando o desenvolvimento de diversas descrições deste recurso e, assim, sua melhor reutilização em novos contextos. O Ambiente é formado por três camadas: Camada de Interface, responsável por interagir com o usuário; Camada de Serviço, responsável por fornecer as funcionalidades do sistema de adequação; e a Camada de Persistência, responsável pelos agentes de adequação.

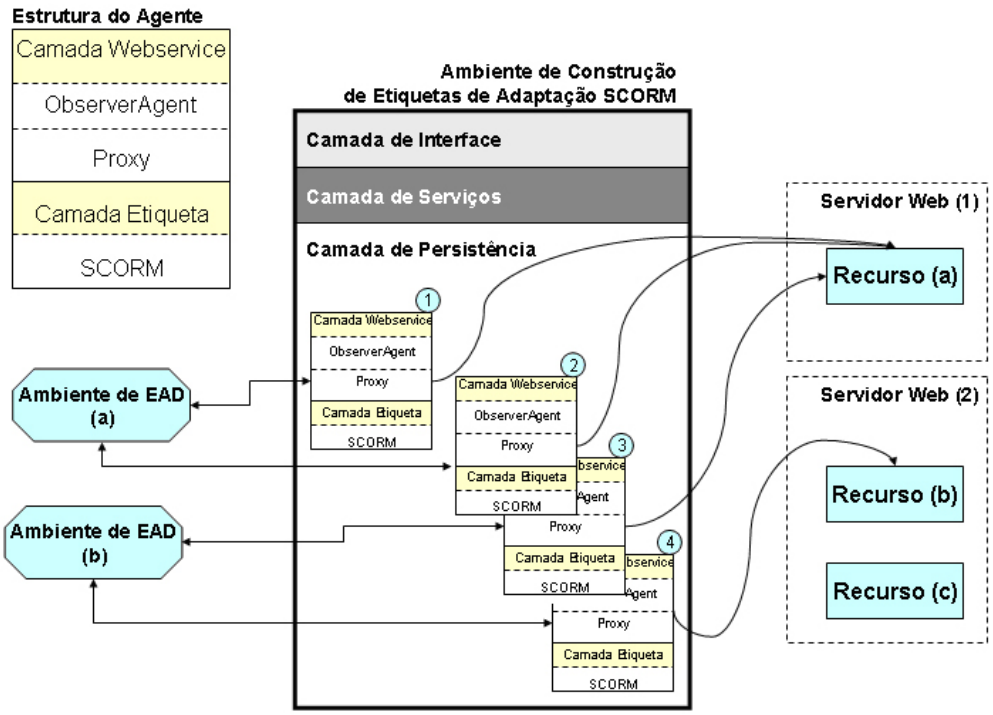


**Figura 7.3:** Arquitetura geral da Fábrica de Adequação (ACEAS) (Kratz et al., 2005).

O sistema tem que oferecer uma série de serviços e uma interface para que os responsáveis pelos repositórios tradicionais possam registrar os recursos a serem adequados à norma SCORM. Como pode ser observado na Figura 7.4, ao se cadastrar um SCO na Fábrica de Adequação, é necessário oferecer uma interface ao usuário para a criação de um pacote com apenas um recurso (SCO) ("SCORM *Single Item Packager*") e de um pacote com vários SCOs agregados ("SCORM *Package Aggregator*"). Internamente, o sistema cria a etiqueta associada ao recurso, inclui os códigos de comunicação com LMS e empacota (compacta) todos os arquivos em um único arquivo, gerando um pacote SCORM. Por fim, o sistema armazena o pacote SCORM em um Repositório Local da Fábrica de Adequação.

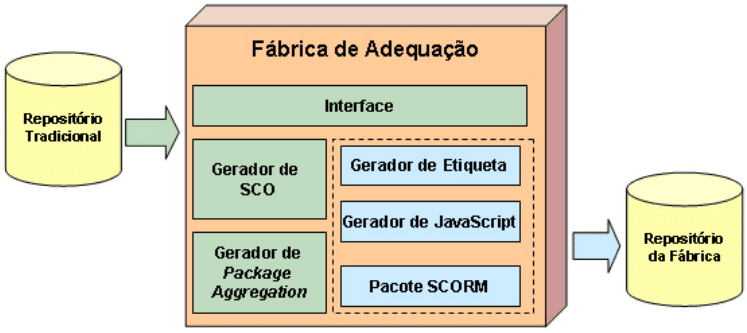


**Figura 7.4:** Serviços da Fábrica de Adequação.

Realizada a adequação dos repositórios tradicionais, a Fábrica passa a funcionar como um único repositório de recursos educacionais, respeitado a norma SCORM. A Figura 7.5 mostra que sistemas

de *e-learning* (LMS) que respeitam as especificações do SCORM podem acessar o "Repositório da Fábrica" como se fosse um repositório SCORM comum, sendo que toda a adequação foi abstraída e embutida no Sistema de Adequação (ACEAS). Desta maneira, os LMSs podem enviar SCO para o navegador do cliente (aluno), seguindo as especificações técnicas do SCORM. A ilustração ainda mostra que o *frameset* "sco.htm" possui um *link* associado com o recurso educacional (arquivo físico, como: documento pdf, imagem jpeg, etc), que, por sua vez, é visualizado no navegador via HTML, como qualquer *home page*.

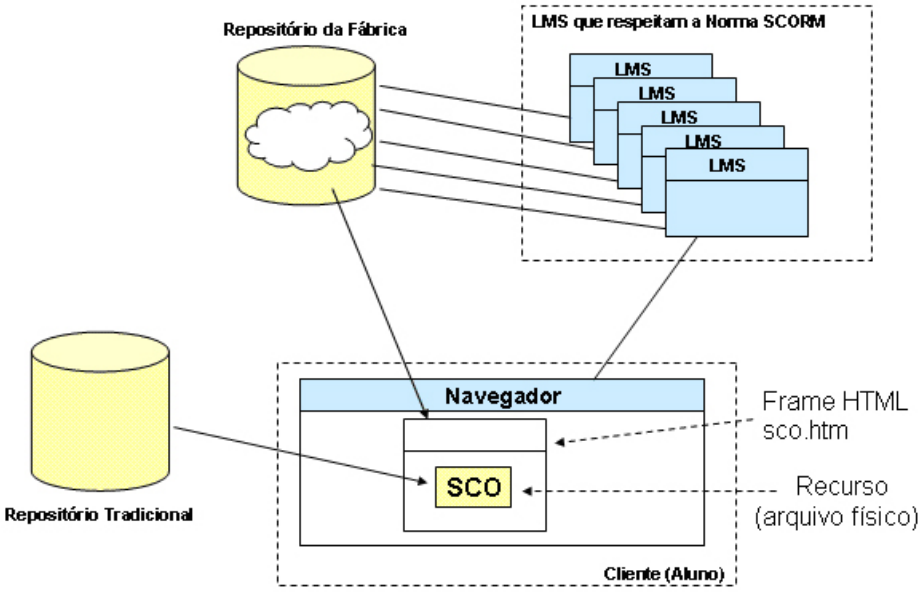


**Figura 7.5:** Visão geral da Arquitetura (vista pelo LMS).

### 7.3.1 Diagramas da Fábrica de Adequação

A Figura 7.6 apresenta os principais atores que manipulam a Fábrica de adequação (OMG, 2003).

A Figura 7.7 mostra o cenário geral da Fábrica, onde apresenta o digrama de seqüência com as interações dos atores com o sistema. O Configurador de Sistema permite que os usuários possam usar o sistema montando um conjunto de serviços oferecidos pela Fábrica. Assim, um Repositório Adaptador pode começar a adequar seus recursos educacionais à norma SCORM, gerando, desta maneira, objetos de aprendizagem normalizados para o repositório da Fábrica. O Pesquisador de LO (LMS, professor, orientador, estudante, responsável, gerente, etc.) procura no repositório de objetos (recursos) e os seleciona para seus próprios propósitos e intenções pedagógicas.

Dentro do UML, o diagrama de classe é utilizado para descrever os tipos de objetos e seus relacionamentos em um sistema, sendo capaz de modelar a estrutura das classes e seu conteúdo, utilizando modelos de elementos como: classe, objetos, *packages*, etc. (Sommerville, 2003) (OMG, 2003).

O modelo de classes da Fábrica de Adequação é apresentado na Figura 7.8, onde se tem uma visão da arquitetura com o objetivo de ser o ponto de partida para a definição de uma "arquitetura padronizada" de sistemas dentro de um determinado escopo. É desenvolvido como um precursor para qualquer atividade de padronização. Pode ser entendido como uma estrutura conceitual cujo

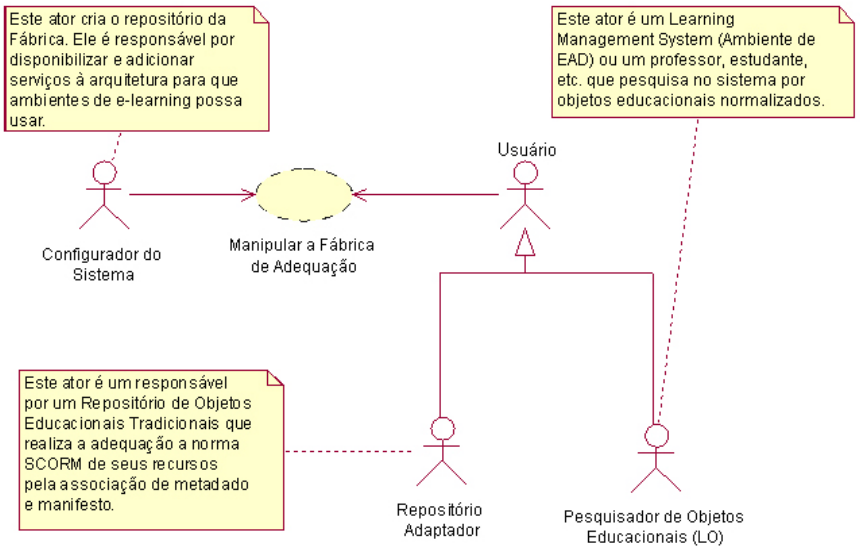


**Figura 7.6:** Diagrama de Caso de Uso.

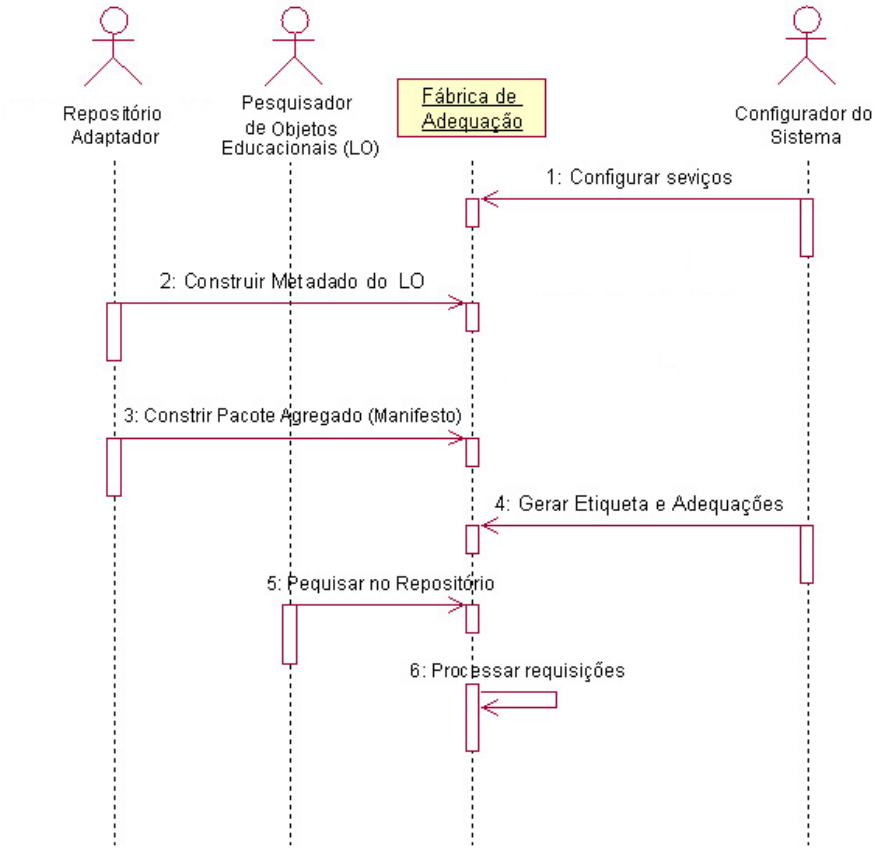


**Figura 7.7:** Diagrama de Seqüência Geral da Fábrica.

propósito é dividir o trabalho de padronização em fragmentos gerenciáveis, bem como mostrar, em um nível geral, como esses fragmentos estão relacionados uns com os outros. É composto da classe

cliente (Pesquisado de LO) e da classe etiqueta que possui adequações (Comunicação em JavaScript, Metadado e Manifesto).

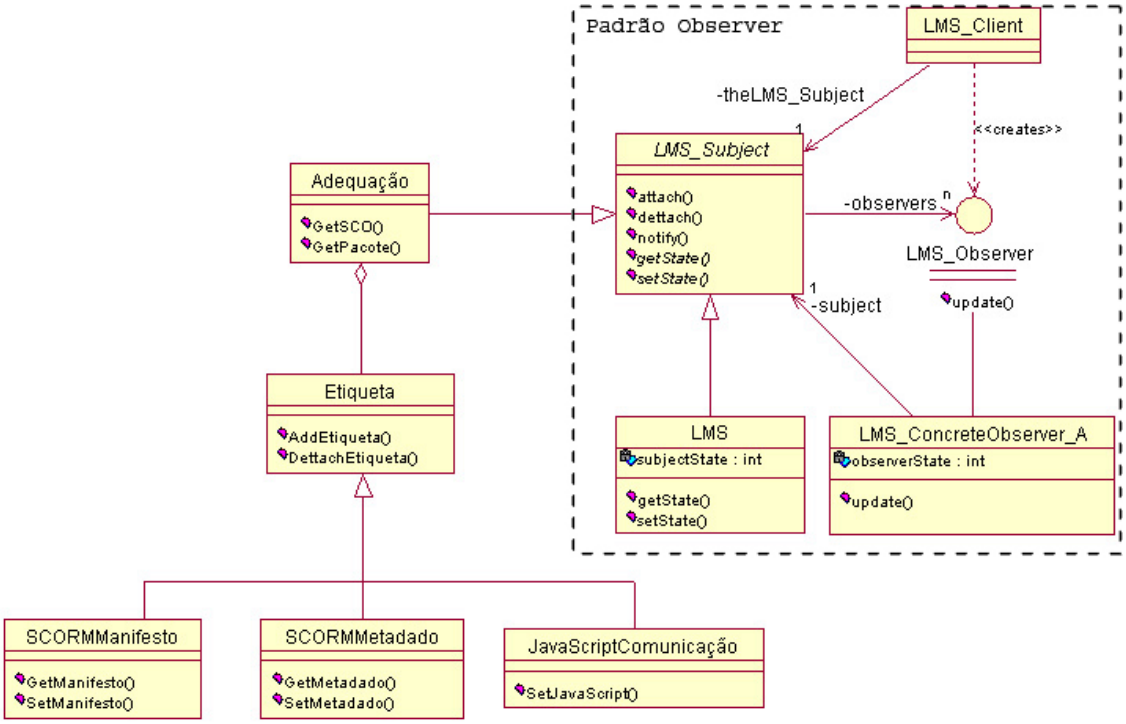


**Figura 7.8:** Diagrama de Classes.

O padrão de projeto *Observer* define uma dependência um-para-muitos entre objetos para que quando um objeto mudar de estado, todos os seus dependentes sejam notificados e atualizados automaticamente (Gamma et al., 2000). Nesta pesquisa, o objeto observado é uma etiqueta associada a um recurso educacional, que quando muda de estado tem que notificar todos os LMS que puder (os "*Observers*"), da forma mais transparente possível. O Observer permite que tanto observadores quanto sujeitos observados podem ser reutilizados e ter sua interface e implementação alteradas sem afetar o sistema. Além disso, o acoplamento forte implicado pelo relacionamento bidirecional é reduzido com o uso de interfaces e classes abstratas.

O uso do padrão *Observer* resolve o problema de escalabilidade, já que é possível ocorrer um excesso de etiqueta (explosão de etiqueta) para o mesmo recurso educacional, pois um recurso é associado com um contexto e este pode ser acessado, ao mesmo tempo, por diversos ambientes de EAD. O *Observer* atua como uma central, permitindo que vários observados busquem informações sobre o objeto a ser utilizado. Por esta razão, é possível obter uma economia quanto ao número de etiquetas.

O *Web Service* WsObserverAgent contém os reais métodos que são invocados pelos sistemas de *e-learning*. A classe WsProxy contém os ponteiros para invocação do *Web service*, como pode ser constatado na Figura 7.9. Nesta classe são indicados o endereço onde o agente *web service* está hospedado e os métodos que devem ser invocados pelos ambiente (LMS) que instanciam o objeto de aprendizagem.

```
namespace proxy.WsObserverAgent
{
      using System.Diagnostics;
      using System.Xml.Serialization;
      using System;
      using System.Web.Services.Protocols;
      using System.ComponentModel;
      using System.Web.Services;


      /// <remarks/>
      [System.Diagnostics.DebuggerStepThroughAttribute()]
      [System.ComponentModel.DesignerCategoryAttribute("code")]
      [System.Web.Services.WebServiceBindingAttribute(Name="ObserverSoap",
Namespace="http://tempuri.org/")]
      public class ObserverService : System.Web.Services.Protocols.SoapHttpClientProtocol
      {

            /// <remarks/>
            public Observer()
      {
            this.Url = "http://10.16.165.208/scorm/wsObserverAgent.asmx";
      }

            /// <remarks/>

      [System.Web.Services.Protocols.SoapDocumentMethodAttribute("http://tempuri.org/exAcessoAoBanco",
                  RequestNamespace="http://tempuri.org/", ResponseNamespace="http://tempuri.org/",
                  Use=System.Web.Services.Description.SoapBindingUse.Literal,
                  ParameterStyle=System.Web.Services.Protocols.SoapParameterStyle.Wrapped)]
```

**Figura 7.9:** Classe Proxy de acesso ao *Web service*.

## *7.4* Camada Interface

A Camada de Interface de Adequação é responsável por estabelecer a comunicação dos repositórios tradicionais com a Fábrica de Adequação (Ambiente de Construção de Etiquetas de Adaptação SCORM - ACEAS). Essa interface deve oferecer aos usuários duas formas de adequações (Camada de Serviços): SCORM *Single Item Package*, para a criação de um SCO, e o SCORM *Package Aggregator*, para criação de um bloco de SCO (como um capítulo ou curso).

O protótipo foi implementado utilizando a linguagem de programação Microsoft.NET ASP VBScript e C# e banco de dados SQL Server. O sistema opera com Servidor *Web* Microsoft *Internet Information Services* (IIS) Versão 5.1. A Figura 7.10 mostra a interface principal do protótipo implementado, onde é possível realizar o cadastro e a autenticação dos usuários.

Foi escolhida a plataforma de desenvolvimento Microsoft .NET, principalmente pela sua facilidade em lidar com *Web Services*, pela ampla utilização para o desenvolvimento *Web* e pelo meu prévio conhecimento desta linguagem.

A interface deve permitir o cadastro de novos repositórios tradicionais e realizar o *login* dos usuários cadastrados para, assim, realizarem as adequações de conteúdos. Desta forma, é necessário o uso de uma base de dados. Outra funcionalidade disponibilizada por essa interface é a pesquisa de objetos de aprendizagem adequados, permitindo o acesso aos contextos armazenados no Repositório local.

Os usuários, para acessarem o sistema (Fábrica), deverão primeiro efetuar o registro do repositório origem, fornecendo os seguintes dados (Figura 7.11): identificador do repositório, nome do repositório, descrição do repositório, endereço eletrônico (URL), responsável e *e-mail*. Esses dados são obri-


**Figura 7.10:** Tela principal da Fábrica de Adequação.

gatórios e permitem a autenticação dos repositórios. Além disso, algumas informações são utilizadas na criação da etiqueta, associando o recurso educacional com a adequação gerada pelo sistema.

## 7.5 Camada de Serviços

A Camada de Serviços contém as funcionalidades implementadas que realizam a adequação dos conteúdos de aprendizagem para a norma SCORM. De acordo com a norma SCORM são necessários oferecer os seguintes serviços:

- Interface de adequação de objeto de aprendizagem (SCO);
  - Sistema de Geração de um SCO (SCORM *Single Item Package*);
  - Sistema de Geração de Pacote Agregado (SCORM *Package Aggregator*);
- Gerador de Etiqueta de adequação;
- Inclusão de mecanismo de comunicação (JavaScript) entre o SCO e o LMS;
- Gerador de Pacote SCORM.

### 7.5.1 SCORM *Single Item Package*

O SCORM *Single Item Package* é uma funcionalidade da camada de serviço responsável pela normalização (adequação) de recursos educacionais para a norma SCORM. Ele é responsável por


**Figura 7.11:** Tela cadastro de novos repositórios.

gerar todas as adequações necessárias para que esses recursos sejam pacotes SCOs do SCORM, gerando o metadado, o manifesto e a etiqueta de adequação associada a esse recurso. Além disso, é responsável por incluir o mecanismo de comunicação do SCO com o LMS e, assim, normalizar esse recurso para um objeto de aprendizagem reutilizável, podendo tais recursos serem utilizados em novos cursos ou por outros LMSs.

Essa funcionalidade também é capaz de criar múltiplos contextos de objetos de aprendizagem, capacidade que é detalhada na Seção 7.8.

Para adequar um recurso educacional, primeiro é necessário o cadastramento de um repositório tradicional a partir da camada de interface. Assim, pode-se passar para a etapa de adequação, onde é necessário informar ao sistema (veja Figura 7.12) o endereço eletrônico (URL) do recurso a ser adequado à norma SCORM.

#### 7.5.1.1 Metadado

O próximo passo é criar o arquivo de Metadado que descreve o objeto de aprendizagem. Metadado é definido como informação acerca de informação, assim, na norma SCORM, metadado consiste em informações que descrevem o conteúdo do SCO (ADL, 2005). A aplicação de metadado respeita o padrão IEEE LSTC *Learning Object Meta-Data* (LOM) e utiliza o IMS *Learning Resource Meta-data* XML *Binding Specification* para armazenar as informações em formato XML, ambos descritos anteriormente.

O sistema oferece duas maneiras de se criar o metadado: importar um metadado já existente, podendo editá-lo, ou criar um novo arquivo. Para isso foi desenvolvido um editor de metadado (XML parse) que facilita a criação e edição de metadados SCORM.

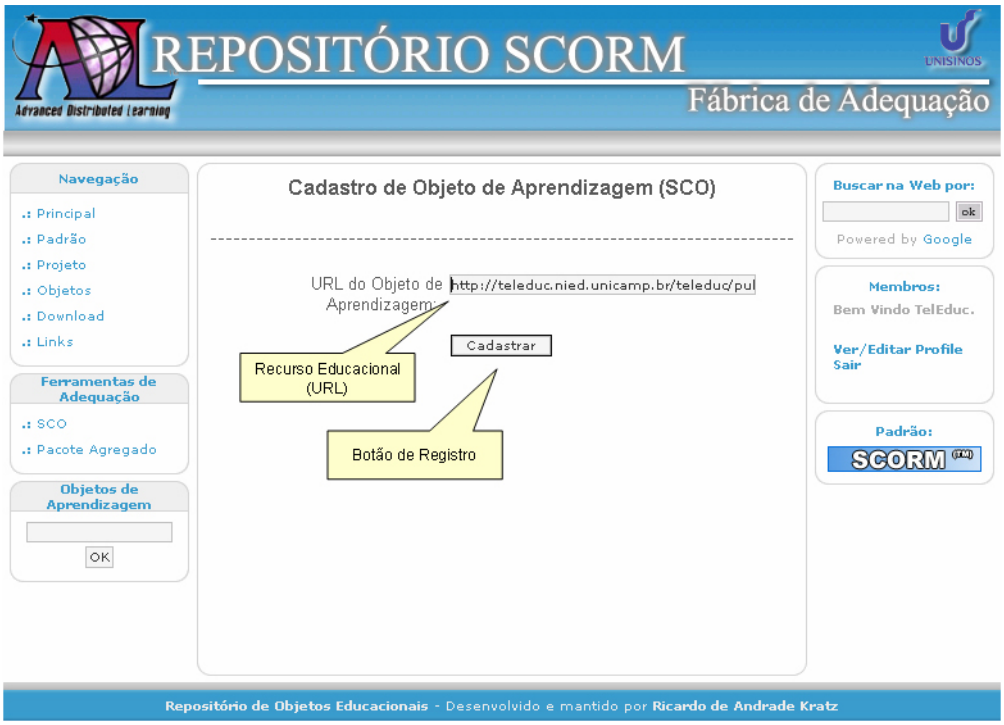


**Figura 7.12:** Adequação de Recuso Educacional (URL).

A Figura 7.13 mostra como se pode importar um arquivo de metadado. Esse arquivo pode ser mantido em sua forma original ou editado.


**Figura 7.13:** Importação de Arquivo de Metadado.

O Editor de Metadado pode ser visto na Figura 7.14, o qual facilita a criação do metadado, fornecendo ao usuário uma maneira de gerar o metadado do recurso educacional sem que tenha que conhecer XML. No "Apêndice A" é possível ver um exemplo de metadado gerado pelo sistema (Editor de Metadado).


**Figura 7.14:** Editor de Metadado (XML *Parse*).

#### 7.5.1.2 Manifesto

Apesar de se tratar de um único recurso educacional que está sendo adequado para se tornar um SCO, a norma define que seja necessária a presença de um manifesto no pacote SCORM (ADL, 2005). O manifesto é gerado automaticamente pelo sistema, sem a necessidade de um editor, como no caso do metadado, ou pode ser importado. Ele contém as informações sobre o pacote, como: os recursos, a organização, as versões do SCORM e o ADL Esquema utilizado.

Sua principal função é informar a estrutura de apresentação dos recursos. Como neste caso existe apenas um recurso, a organização vai possuir apenas um item. O "Apêndice B" traz um exemplo de um arquivo de manifesto gerado.

### 7.5.2 SCORM *Package Aggregator*

O SCORM *Package Aggregator* é o segundo serviço (funcionalidade) implementado na Fábrica de Adequação, o qual permite adequar uma série de recursos educacionais de um repositório tradicional

para um Objeto (SCO) da norma SCORM. Consiste em um conjunto de regras e normas para agregar conteúdo de ensino (SCO) em um bloco (por exemplo, um curso, um capítulo, um módulo, etc.), com o objetivo de permitir a transferência desses mesmos blocos entre sistemas LMSs diferentes (ADL, 2005).

Desta maneira, são geradas todas as adequações necessárias para que esse recurso seja um pacote agregado do SCORM, gerando o metadado, o manifesto e uma etiqueta de adequação associada a esse recurso.

A Figura 7.15 mostra a interface para a criação de um pacote agregado SCORM. É necessário utilizar o SCORM *Single Item Package* (Seção 7.5.1) para adequar os objetos de Aprendizagem (SCOs) que serão usados na estrutura agregada. Realizadas essas adequações, para se gerar um pacote agregado foi desenvolvido um Editor de Estruturação de Agregação capaz de selecionar os objetos desejados (que já foram previamente adequados) e, assim, adicioná-los a um pacote agregado. É possível organizar os objetos de forma seqüencial ou hierárquica com os comandos de manipulação. Por fim, o botão "Manifesto" finaliza o pacote.

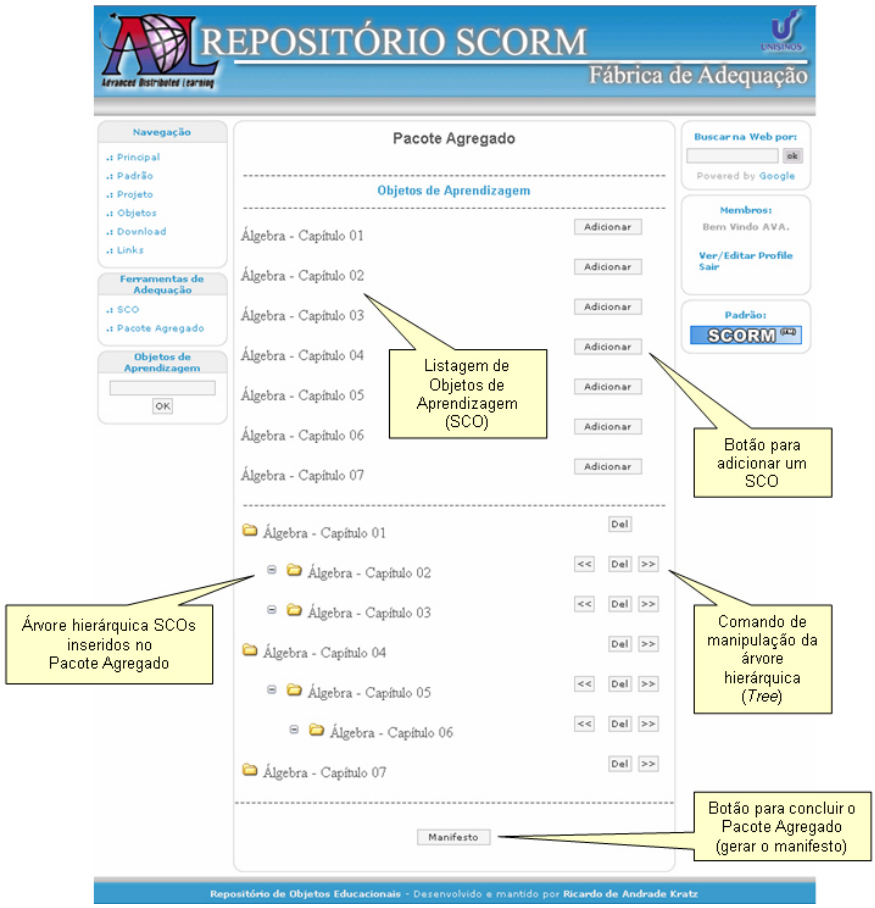


**Figura 7.15:** Editor de Estruturação de Agregação.

### 7.5.2.1 Metadado

Criado o pacote agregado, é necessário criar o metadado associado ao Pacote responsável por descrevê-lo. O metadado é criado como no caso do SCORM *Single Item Package*: é utilizado o

Editor de Metadado (Seção 7.8) para criar o metadado XML. A diferença é que para o pacote agregado, o metadado descreve o bloco, como por exemplo: em um curso de Álgebra o metadado descreve os objetivos do curso, seus pré-requisitos, entre outras coisas. São informações gerais do pacote agregado que permitem ao usuário (repositório tradicional) definir um recurso educacional mais complexo e coeso.

#### 7.5.2.2 Manifesto

O manifesto no *Content Aggregation* permite um mapeamento dos SCOs que agregam os conteúdos educativos em uma estrutura funcional coerente (por exemplo, um curso, módulo, lição, etc.). Essa agregação e estruturação são feitas utilizando o modelo de *Content Packaging* do IMS (IMS, 2005). Os vários níveis de agregação podem ser acompanhados de metadados específicos para *Content Aggregation*, tal como a norma SCORM (ADL, 2005) define.

A Figura 7.16 mostra como o manifesto na *tag* <*organization*> estrutura a organização do pacote agregado que está sendo adequado. As informações geradas no Editor de Estruturação permitiram registrar a organização do Pacote Agregado SCORM. A *tag* <*resources*> recebe os SCOs do pacote.

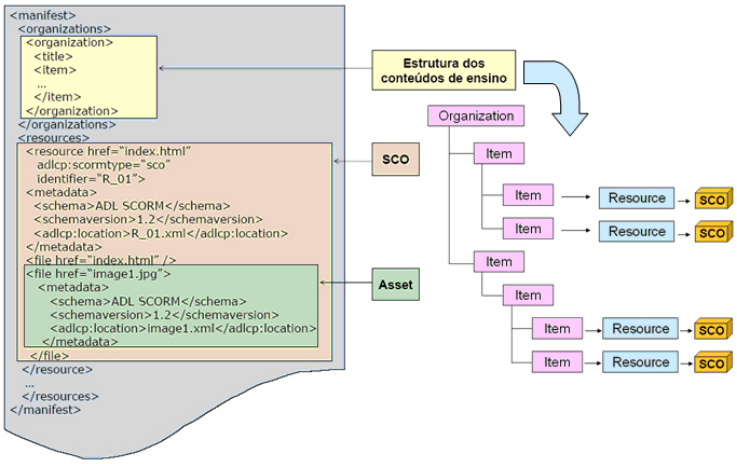


**Figura 7.16:** Manifesto de um *Content Aggregation*.

### 7.5.3 Mecanismo de Comunicação

A norma SCORM estabelece que o *Sharable Content Objects* (SCO) deve, obrigatoriamente, incluir funcionalidade de comunicação (código JavaScript) com o sistema de *e-learning* (LMS). Como o trabalho opera na normalização de repositórios educacionais tradicionais, esse mecanismo não está presente nos recursos educacionais.

O Mecanismo de Comunicação é uma funcionalidade responsável por adicionar esse código, sendo que a solução adotada é a introdução do recurso educacional em uma página html juntamente com o código em JavaScript.

#### 7.5.3.1 Funcionamento

Inicialmente é criada uma página HTML *frame* (moldura) com dois arquivos: "sco.htm", que recebe o código de comunicação e a associação do recurso do repositório tradicional; e uma página em branco invisível "dummy.htm" (Figura 7.17).

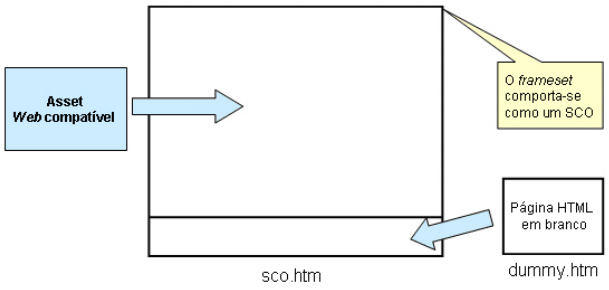


**Figura 7.17:** Objeto de Aprendizagem com mecanismo de comunicação.

O código JavaScript que possui as funções de comunicação do SCO com a API do LMS foi colocado em um arquivo JavaScript "SCORMGenericLogic.js" que é invocado dentro da página "sco.html". O código genérico JavaScrit de comunicação pode ser visto no "Apêndice C".

A Figura 7.18 mostra um exemplo de um código de um SCO em uma página HTML (sco.htm). A função "function SCOInitData()" é responsável pela associação (via URL) de recurso físico do repositório tradicional. A *tag* "<script src="SCORMGenericLogic.js"type="text/javascript" language="JavaScript">" adiciona o código de comunicação (JavaScript) na página HTML.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>---</title>
 <script src="SCORMGenericLogic.js" type="text/javascript"
 language="JavaScript">
 </script>
<script type="text/javascript" language="JavaScript">
 function SCOInitData() {
       content.location.href="http://URL do repositório
 tradicional/<recurso>" //(ex.: documento.pdf)"
 }
function SCOSaveData() {
      SCOSetValue("cmi.core.lesson_status","completed")
  SCOSetValue("cmi.core.exit","") }
</script>
</head>
<frameset rows="100%,*"
  onload="SCOInitialize()" // Inicia a comunicação com API do LMS
  onunload="SCOFinish()"   // Término a comunicação
  onbeforeunload="SCOFinish()">
<frame name="content" src="dummy.htm" scrolling="auto" title="content frame"/>
<frame name="empty" src="dummy.htm" scrolling="no" title="empty frame"/>
</frameset>
<noframes>
  <body>
  <p>Esta página usa frames, mas seu navegados no é capaz de exibir.</p>
  </body>
</noframes>
</html>
```

**Figura 7.18:** Código do "sco.htm".

A página em branco (invisível) "dummy.htm" está presente porque a função function SCOInitData() envia para o *frame* ativo "sco.htm" o recurso educacional.

### 7.5.4 Pacote SCORM

O pacote SCORM (*Packages*) é o local onde se armazenam os arquivos, no formato PIF (*Package Interchange File*), para facilitar a sua distribuição pela *Web*. Os formatos mais comuns de arquivos PIF são: zip, jar, rar, tar e cab. Este serviço deve armazenar em um arquivo PIF os arquivos gerados em uma adequação, para poder ser armazenado no repositório da Fábrica.

A opção implementada foi o formato ZIP que é mundialmente utilizado e trabalha em vários sistemas operacionais. A Figura 7.19 ilustra um exemplo de um pacote SCORM gerado pela Fábrica de Adequação e de como recuperar esse pacote.

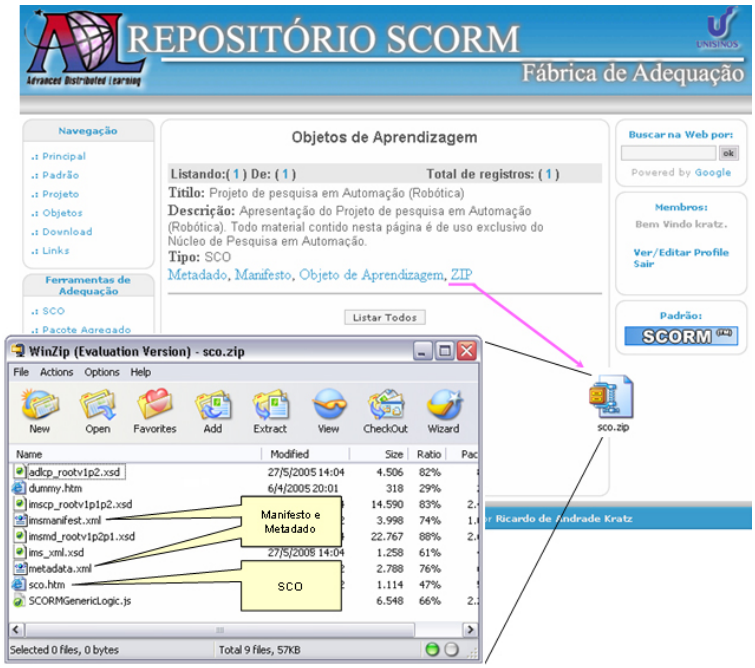


**Figura 7.19:** Pacote Scorm (sco.zip).

## 7.6 Camada de Persistência

Na Fábrica de Adequação, a camada de persistência detém os dados, neste caso as etiquetas, sendo que cada etiqueta está associada a um recurso educacional de um repositório tradicional. A Figura 7.20 mostra que a etiqueta é responsável por armazenar os dados do recurso educacional (a) com suas adequações, sendo composta por (*i*) uma página html no formato de *frame*, sendo que a essa página foi dado o nome "sco.html". Ela possui um *link* apontando para o recurso físico do repositório tradicional e um mecanismo de comunicação em JavaScript para permitir a interação do SCO com o LMS. Possui ainda (*ii*) arquivo de metadado "metadata.xml"; (*iii*) um arquivo manifesto "imsmanifest.xml" e (*iv*) os XML esquemas padrões da norma SCORM (ADL, 2005): "adlcp_rootv1p2.xsd", "ims_xml.xsd", "imscp_rootv1p1p2.xsd" e "imsmd_rootv1p2p1.xsd".

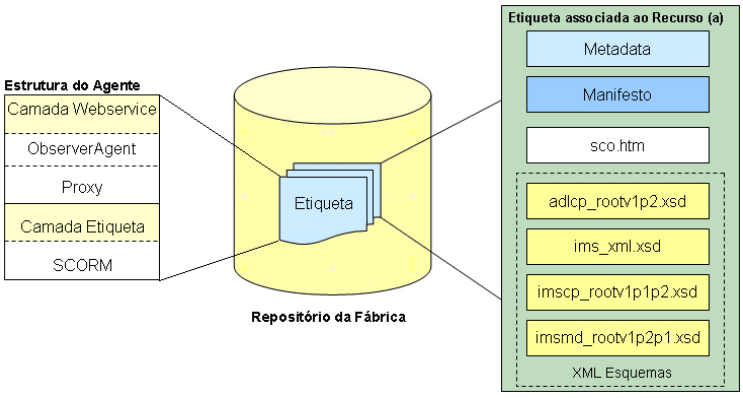


**Figura 7.20:** Estrutura da etiqueta da Fábrica de Adequação.

Foi desenvolvido o *web method* "criaEtiqueta" para o *Web Service* WsEtiqueta, responsável por promover a associação dos recursos educacionais com as adequações da norma SCORM armazenadas na Fábrica de Adequação. Os parâmetros de entrada e saída do mesmo, entre outras informações, são apresentados na Figura 7.21.

**criaEtiqueta**

Cria a Etiqueta associando um recurso educacional às adequações SCORM. Retorna o código da etiqueta

**Test**

To test the operation using HTTP POST protocol, click the 'Invoke' button.

| Parameter | Value |
|---|---|
| url: | |

Invoke

**SOAP**

The following is a sample SOAP request and response. The **placeholders** shown need to be replaced with actual values.

```
POST /scorm/WsEtiqueta.asmx HTTP/1.1
Host: localhost
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://tempuri.org/criaEtiqueta"

<?xml version="1.0" encoding="utf-0"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLShema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schema.xmls
  <soap:Body>
    <criaEtiqueta xmlns="http://tempuri.org/">
      <url>string</url>
    </criaEtiqueta>
  </soap:Body>
</soap:Envelope>
```

```
HTTP/1.1 200 OK
Content-Type: text/xml; charser=uft=8
Content-Length: length
<?xml version="1.0" encoding="utf-0"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLShema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soap="http://schema.xmls
  <soap:Body>
    <criaEtiquetaResponse xmlns="http://tempuri.org/">
      <criaEtiquetaResult>etiquetaID</userListResult>
    </criaEtiquetaResponse>
  </soap:Body>
</soap:Envelope>
```

**Figura 7.21:** Descrição do *Web method* criaEtiqueta.

Na Figura 7.22 é apresentado o código fonte, em .NET C#, com a definição do serviço "criaEtiqueta". A implementação do serviço realiza, basicamente, uma inclusão do URL e do ID do

usuário na base de dados presente na camada de persistência com a finalidade de associar o recurso educacional com as adequações SCORM geradas pelo sistema.

```
[WebMethod (Description="Insere Etiqueta no Repositório/Banco")]
            public bool url(string[] aData, string conect)
            {
                  SqlConnection conn = new SqlConnection(conect);
                  SqlCommand cmd = new SqlCommand();
                  conn.Open();
                  try
                  {
                        cmd = new SqlCommand("sp_insert_etiqueta",conn);//"insert_Etiqueta", conn);
                        cmd.CommandType = CommandType.StoredProcedure;
                        // idUsuario
                        SqlParameter pId = new SqlParameter("@id_etiqueta",SqlDbType.Char);
                        pId.Value = aData[0];
                        cmd.Parameters.Add(pId);
                        // type
                        SqlParameter pSubject = new SqlParameter("@tipo",SqlDbType.Char);
                        pSubject.Value = aData[1];
                        cmd.Parameters.Add(pSubject);
                        // location
                        SqlParameter pLocation = new SqlParameter("@url",SqlDbType.Char);
                        pLocation.Value = aData[2];
                        cmd.Parameters.Add(pLocation);
                        // date
                        SqlParameter pDate = new SqlParameter("@data",SqlDbType.DateTime);
                        pDate.Value = aData[3];
                        cmd.Parameters.Add(pDate);
                        SqlParameter pTimeEnds = new SqlParameter("@timeEnds",SqlDbType.DateTime);
                        pTimeEnds.Value = aData[5];
                        cmd.Parameters.Add(pTimeEnds);
                        // folder
                        SqlParameter pBody = new SqlParameter("@pasta",SqlDbType.Text);
                        pBody.Value = aData[6];
                        cmd.Parameters.Add(pBody);
                        cmd.ExecuteNonQuery();
                        // status
                        SqlParameter pBody = new SqlParameter("@status",SqlDbType.Text);
                        pBody.Value = aData[6];
                        cmd.Parameters.Add(pBody);
                        cmd.ExecuteNonQuery();

                        return true;
                  }
                  catch(Exception e)
                  {
                        string t = e.Message;
                        return false; }
                  finally
                  {
                        conn.Close(); }
            }
```

**Figura 7.22:** Fonte do *Web method* criaEtiqueta.

## 7.7 Mecanismo de Pesquisa de Objetos de Aprendizagem

Os materiais educacionais adequados para objetos de aprendizagem estão disponíveis aos educadores e sistemas de *e-learning* através do repositório local da Fábrica de Adequação, que permite a recuperação dos recursos digitais de aprendizagem. Trata-se de uma biblioteca de conhecimento, permitindo que diferentes cursos utilizem um mesmo objeto.

Foi desenvolvido um agente de pesquisa seguindo o modelo definido por Carlos Lucena (Lucena et al., 2001), que é capaz de localizar os dados e obter acesso. Esse agente fornece acesso aos objetos de aprendizagem através de mecanismos de comunicação, que também podem ser acessados por outros agentes.

O mecanismo de busca do repositório consiste de pesquisas realizadas sobre os campos que compõem os medadados dos objetos de aprendizagem. O formalismo para a entrada da pesquisa é semelhante a maioria dos sistemas de busca da *web* conhecidos como *search engines* (Sullivan, 2005). Operadores lógicos "e" e "ou" são utilizados, bem como operadores de adição e exclusão (respectivamente "+" e "-"). O texto de consulta passa então por um processamento para que sejam definidos todos os parâmetros da consulta. Através de um agente são geradas uma pilha de operadores e uma pilha de frases (palavras ou frases delimitadas por aspas). O agente é responsável então pela decomposição do texto da pesquisa em elementos que irão formar a consulta sobre a fonte de dados.

A Figura 7.23 mostra como se pode encontrar objetos de aprendizagem. Para fazer a pesquisa é necessário digitar uma palavra ou área de conhecimento. Exemplo: Para recuperar uma atividade de matemática relacionada a "Álgebra" digite: Álgebra.


**Figura 7.23:** Mecanismo de Pesquisa de Objeto de Aprendizagem.

O agente trabalha com busca simples, que consiste de uma consulta realizada nos campos: nome, descrição e palavras chaves dos objetos de aprendizagem. A única responsabilidade do usuário é entrar com o texto para a pesquisa e acionar a busca. O agente faz diferenciação entre caracteres acentuados, por exemplo, "computação" e "matemática". Já a Figura 7.24 mostra uma resposta exemplo de uma pesquisa no repositório da Fábrica de Adequação.

## *7.8* Múltiplos Contextos

A melhor maneira do desenvolvimento de curso através da *Web* é um ponto que tem sido amplamente discutido. A maneira de disponibilizar conteúdo deve ser adaptada aos objetivos principais do curso em si, adequando aos recursos tecnologias disponíveis para atender às necessidades do seu criador.

Apesar de todo o esforço que vem sendo empreendido na área, há muito trabalho para o uso eficiente dos objetos de aprendizagem (Downes, 2002). É necessária a construção de um ambiente educacional no qual este objeto funcione. Os possíveis usuários precisam localizar os objetos de


**Figura 7.24:** Resultado da pesquisa por objeto de Aprendizagem.

aprendizagem e então arranjá-los de acordo com algum fim pedagógico. Segundo Stephen Downes, o tanto de trabalho necessário para o uso de um objeto de aprendizagem em diferente contexto leva-nos a acreditar que necessitamos, na verdade, é de objetos de aprendizagem mais flexíveis, que sejam mais facilmente descobertos e que possam ser melhor descritos com suas experiências.

A usabilidade[1] é a definição do perfil dos usuários que utilizarão um ambiente de EAD, neste caso, o sistema educacional. Desta maneira, é possível o desenvolvimento da interface e da lógica de navegação e interação do sistema, com o objetivo de facilitar a aprendizagem (Nielsen, 1999). Mas, o uso de objetos de aprendizagem apresenta mais desafios do que apenas definir os conteúdos para um determinado curso de EAD.

Torna-se necessário aos pedagogos e desenvolvedores pensar na reusabilidade dos objetos de aprendizagem em novos contextos, não apenas para suas próprias soluções. Trata-se de uma nova ótica sobre os recursos educacionais, que passam a ser unidades separadas que devem ser modulares, interoperáveis e tenham a capacidade de serem descobertas.

## 7.9 Conclusão

Através de uma arquitetura baseada em agente é possível realizar a adequação de recursos educacionais em objetos de aprendizagem SCORM, com o objetivo de reaproveitar esses recursos, ou parte deles, adequando-os em um repositório SCORM que, por sua vez, podem ser reaproveitados em novos ambientes de *e-learning* (padrão SCORM).

[1] Área do conhecimento que estuda o desenvolvimento de interface de sistema como o objetivo de minimizar o tempo necessário de aprendizagem para uso do sistema.

Como foi visto na introdução, a produção de conteúdos educacionais multimídia é uma atividade relativamente onerosa e que envolve a necessidade de uma equipe especializada para o seu desenvolvimento. A normalização de recurso já existente permite a redução dos custos e a possibilidade de gerar novas soluções com menor tempo operacional.

O diferencial desta solução é a capacidade de simplificar a adequação de repositórios tradicionais (que não respeitam a norma SCORM) para o padrão SCORM, sem a necessidade de adequar o LMS (ambiente de EAD) para o padrão, possibilitado, desta maneira, a geração de objetos de aprendizagem reutilizáveis.

Assim, como visto neste Capítulo, é possível promover a interoperabilidade de objetos de aprendizagem utilizando tecnologia de agentes de *software* e de *Web Service*. Com o objetivo de testar e validar a Fábrica de Adequação, o próximo Capítulo apresenta estudos de casos, o qual utiliza a Fábrica para adequar recursos educacionais e verificar esses recursos em um ambiente de EAD. Por fim, os resultados são validados em uma ferramenta que é capaz de verificar a compatibilidade de objetos de aprendizagem com a norma SCORM.

# CAPÍTULO 8

# Estudo de Caso

Este Capítulo apresenta os estudos de casos da Fábrica de Adequação com a finalidade de verificar a sua usabilidade, além de avaliar e verificar eventuais restrições da arquitetura. São realizados três estudos buscando avaliar a compatibilidade e consistência do sistema. Por fim, os resultados são validados em um ambiente de teste fornecido pela ADL.

## *8.1* Introdução

No sentido de avaliar e testar as funcionalidades da ferramenta desenvolvida, foram desenvolvidos três estudos de casos usando a "Fábrica de Adequação". O primeiro e segundo estudos mostram a adequação para a norma SCORM de dois repositórios educacionais (repositórios tradicionais) e o terceiro estudo mostra o desenvolvimento de um ambiente de *e-learning* (LMS), o qual acessa os recursos normalizados nos dois estudos iniciais.

Para finalizar os estudos de casos, os resultados obtidos foram validados com a ferramenta "ADL SCORM *Conformance Test Suite*" que é fornecida pelo consórcio responsável pelo SCORM capaz de validar componentes da norma (ADL, 2005).

Para realizar esses estudos, foi utilizado um ambiente de teste controlado dentro do Laboratório de Engenharia de *Software* do Programa Interdisciplinar de Pós-Graduação em Computação Aplicada da Unisinos. O ambiente controlado foi montado com a seguinte topologia:

- Rede Local *Fast Ethernet* de 100Mb;
- Internet com Largura de Banda 8Mbps;
- Servidor *Web*
  - Banco de Dados SQL Server 2000;
  - Servidor Microsoft *Internet Information Services* (IIS) Versão: 5.1;
  - Sistema Operacional Windows XP Professional com *Service Pack* 2;
  - Configuração: Intel Pentium IV CPU 2.26GHz, 1GB de RAM, 80GB de Memória Física, Placa de Video S3 *Graphics* ProSavageDDR 16Mb e Placa de Rede VIA Technologies, Inc. *Fast Ethernet Adapter* 10/100Mb;

- Estação de Trabalho A
  - Sistema Operacional Windows XP Professional com *Service Pack* 2;
  - Navegador *Web* Internet Explore 6.0.2900.2180 SP2;
  - Configuração: Intel Pentium IV CPU 2.26GHz, 1GB de RAM, 80GB de Memória Física, Placa de Video S3 *Graphics* ProSavageDDR 16Mb e Placa de Rede VIA Technologies, Inc. *Fast Ethernet Adapter* 10/100Mb;
- Estação de Trabalho B
  - Sistema Operacional Windows XP Professional com *Service Pack* 2;
  - Navegador *Web* Opera Versão: 8.5;
  - Configuração: Intel Pentium IV CPU 2.26GHz, 1GB de RAM, 80GB de Memória Física, Placa de Video S3 *Graphics* ProSavageDDR 16Mb e Placa de Rede VIA Technologies, Inc. *Fast Ethernet Adapter* 10/100Mb;
- Estação de Trabalho C
  - Sistema Operacional Windows XP Professional com *Service Pack* 2;
  - Navegador *Web* Netscape *Browser* Versão 8.0.4;
  - Configuração: Intel Pentium IV CPU 2.26GHz, 1GB de RAM, 80GB de Memória Física, Placa de Video S3 *Graphics* ProSavageDDR 16Mb e Placa de Rede VIA Technologies, Inc. *Fast Ethernet Adapter* 10/100Mb;;
- Estação de Trabalho D
  - Sistema Operacional Linux Umbutu versão 5.04;
  - Navegador *Web* Mozilla Firefox Versão 1.0.2;
  - Configuração: Intel Pentium III CPU 800Hz, 256MB de RAM, 80GB de Memória Física, Placa de Video Intel810(TM) *Graphics* Ac 2Mb e Placa de Rede VIA Technologies, Inc. *Fast Ethernet Adapter* 10/100Mb;

## 8.2 Primeiro Estudo de Caso

Este estudo de caso baseia-se na adequação do repositório de *e-learning* TelEduc da Universidade Estadual de Campinas (UNICAMP). O TelEduc é um ambiente para a criação, participação e administração de cursos na *Web*. Ele foi concebido tendo como alvo o processo de formação de professores para Informática Educativa, baseada na metodologia de formação contextualizada desenvolvida por pesquisadores do Núcleo de Informática Aplicada a Educação (Nied) (TelEduc, 2005).

Neste estudo de caso, foi adequada uma amostragem composta por 65 recursos educacionais do repositório do TelEduc. A Figura 8.1 mostra quantos e quais foram os tipos de recursos adequados. Todos esses recursos obtiveram um novo contexto, desta maneira, os recursos passaram a funcionar como um objeto de aprendizagem SCORM (SCO) e podem ser pesquisados e utilizados.

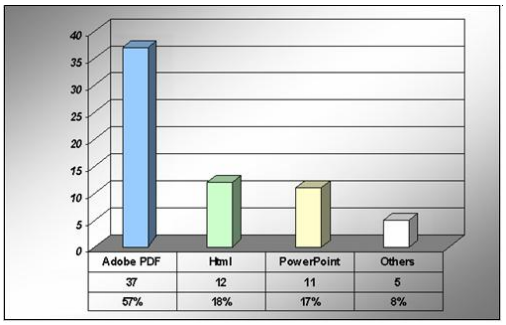


**Figura 8.1:** Tipos de Mídias do TelEduc.

A amostragem foi dividida em blocos para avaliar a consistência e compatibilidade da Fábrica de Adequação em diferentes Sistemas Operacionais e navegadores *web*. A divisão dos recursos educacionais do TelEduc adequados foram: 17 recursos na estação A, 16 na estação B, 16 na estação C e 16 na estação D.

A adequação desses recursos não apresentou incompatibilidade do sistema (Fábrica de Adequação), seja com o Sistema Operacional ou com o tipo de navegador das estações de trabalhos utilizadas. Além disso, a arquitetura da ferramenta também permite a criação de múltiplos contextos desses recursos, conforme sua evolução e interação com os alunos e professores.

## *8.3* Segundo Estudo de Caso

No segundo estudo de caso foi utilizado o repositório tradicional de *e-learning* AVA (Ambiente Virtual de Aprendizagem) da Universidade do Vale do Rio dos Sinos (Unisinos). O Ambiente Virtual de Aprendizagem (AVA) é a solução de *e-learning* da Unisinos. Ele decorre de uma concepção interacionista de construção do conhecimento, na qual o aluno é o centro do processo de aprendizagem e de construção do próprio ambiente. (Pinto et al., 2002)(AVA, 2005).

No AVA foi realizado um estudo dentro do curso da disciplina de Análise de Algorítmos do Mestrado em Computação Aplicada da Unisinos, onde obtivemos a autorização de seu responsável para utilizar os recursos existentes neste estudo de caso. Foram selecionados 92 recursos. Na Figura 8.2, pode-se ver quantos e quais foram os tipos de recursos adequados do repositório do AVA.

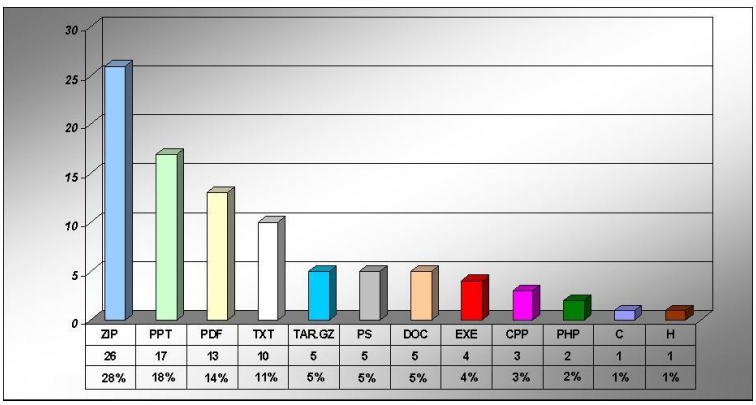


**Figura 8.2:** Tipos de Mídias do AVA.

Seguindo a metodologia adotada no primeiro estudo de caso, a amostragem deste repositório foi dividida em blocos para avaliar a consistência e compatibilidade da Fábrica de Adequação em diferentes Sistemas Operacionais e navegadores *web*. A divisão dos recursos educacionais do AVA adequados foram: 23 recursos na estação A, 23 na estação B, 23 na estação C e 23 na estação D. Também neste estudo de caso, as adequações não apresentaram incompatibilidade ou inconsistência com o Sistema Operacional ou com o tipo de navegador das estações de trabalhos.

## *8.4* Terceiro Estudo de Caso

Este estudo de caso que pretende desenvolver um protótipo de um servidor de *e-learning* que respeite a norma SCORM, construído sobre a tecnologia Microsoft .NET. Esse ambiente busca acessar o Repositório da Fábrica de Adequação e, assim, avaliar a interoperabildade dos recursos educacionais em um ambiente de EAD.

### 8.4.1 Protótipo de LMS

O protótipo desenvolvido é uma verão simplificada de um ambiente de *e-learning*, uma vez que o objetivo deste estudo de caso é avaliar a arquitetura da Fábrica. O protótipo recebeu o nome de "Sistema de *E-learning*" e, em linhas gerais, ficam então definidos os seguintes componentes deste protótipo (LMS): Interface do ambiente *web* e API *Adapter*.

A primeira etapa do Sistema de *E-learning* consiste no desenvolvimento da Interface do ambiente (LMS), o qual fornece os cursos (pré-estabelecidos) e os recursos educacionais do repositório de Fábrica aos usuários (alunos). A Figura 8.3 mostra a interface principal do LMS.

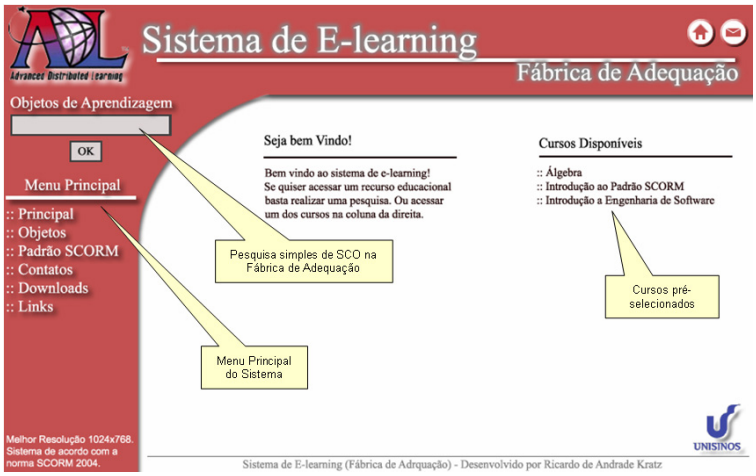


**Figura 8.3:** Interface do LMS.

Na segunda etapa, foram implementadas as funcionalidades do API *Adapter*, as quais são responsáveis por disponibilizar aos SCOs as funções previstas na norma SCORM:

- LMSInitialize("");
- LMSFinish("")

As demais funções: LMSGetLastError(), LMSGetErrorString(errornumber), LMSGetDiagnostic(parameter), LMSGetValue(element), LMSSetValue(element, value) e LMSCommit("") não foram implementadas porque não são essenciais e nem comprometem o objetivo deste estudo de caso.

As funções implementados estão disponíveis em um objeto com o nome API criado em ECMAScript (JavaScript). Assim, quando o SCO quiser, por exemplo, invocar o LMSInitialize, só tem que localizar o objeto API (que deverá estar numa janela-pai daquela em que o SCO se encontre) e utilizar o método LMSInitialize desse objeto. O trecho de código em ASP (Figura 8.4) mostra as funções do API desenvolvidas.

```
var contador, sco;
function resultado() {
this.res;
this.ID_SCO; } //guarda o código do SCO que está a chamar a API
//Funções da API Adapder
function LMSInitialize(str) {
resultado.res = null; //reinicializa a variável res
//Este contador serve para limitar o nr de tentativas feitas para
//tentar contactar. Assim, se o serviço não estiver disponível por
//algum motivo, o browser não irá ficar bloquado à espera da resposta
contador = 0;
//coloca no atributo 's' de service, na frame auxiliar
//a indicação do serviço que está a ser utilizado.
//Esta informação é necessária apenas na segunda parte do código.
service.s = "LMSInitialize";
//Invoca o serviço e passa os parâmetros necessários, neste caso "SCO"
svcElm.Api.callService("LMSInitialize", "",
resultado.ID_SCO);
//Executa um ciclo de espera até que o LMSService responda
//isto é, coloque um resultado em 'resultado.res'
//Esta verificação é feita em intervalos de 500 milisegundos,
//até a um número máximo de 10 vezes, ou seja, irá esperar no
//máximo 5 segundos pela resposta.
while (resultado.res == null && contador < 10) {
pause(500);
contador++; }
return resultado.res;
}
function LMSFinish(str) {
resultado.res = null;
Toolbar.service.s = "LMSFinish";
Toolbar.svcElm.Api.callService("LMSFinish", "",
resultado.ID_SCO);
contador = 0;
while (resultado.res == null && contador < 5) {
pause(500);
contador++; }
return resultado.res;
}
//Cria uma 'classe' com os métodos da API que pode invocar qualquer um
//dos métodos implementados pelas funções acima.
function APIfunc() {
this.LMSInitialize = LMSInitialize;
this.LMSFinish = LMSFinish;}
var API = new APIfunc();//Cria o objeto API
```

**Figura 8.4:** API *Adapter*.

### 8.4.2 Acessando um Objeto de Aprendizagem da Fábrica de Adequação

Existem duas maneiras de acessar os recursos adequados pela Fábrica de Adequação: a primeira é realizando uma pesquisa simples no repositório e selecionando o objeto de aprendizagem desejado; e a segunda forma é acessar os cursos pré-estabelecidos.

As Figuras 8.5 e 8.6 mostram o sistema acessando um curso pré-estabelecido de "Álgebra", podendo ser observado que a árvore de conhecimento é formada a partir das informações contidas no arquivo manifesto do curso alvo. A árvore de conhecimento permite acessar aos conteúdos de aprendizagem normalizados (SCOs), seguindo a estrutura definida pelo manifesto do SCO.

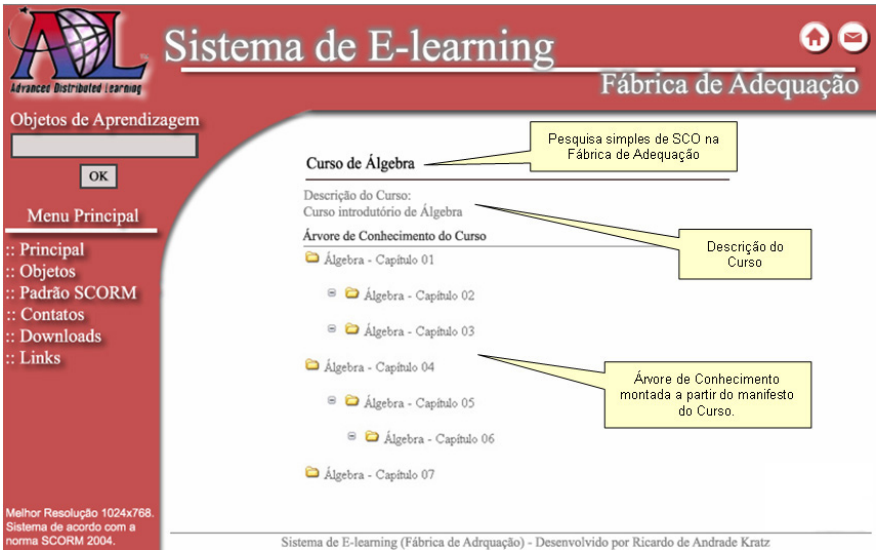


**Figura 8.5:** Curso de Álgebra do Sistema de *E-learning*.

Este desenvolvimento teve a implementação de um mecanismo de criação dinâmica da estrutura de navegação, a partir da leitura do manifesto de qualquer curso desenvolvido de acordo com a norma SCORM, onde se realiza uma análise XML do manifesto do curso, mapeando a estrutura e respectiva ligação aos componentes físicos (arquivos) que constam no manifesto analisado.

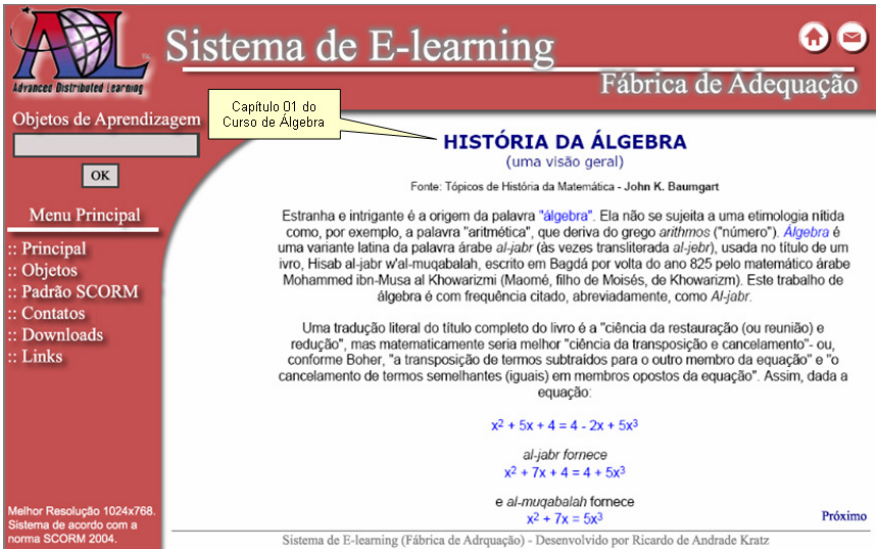


**Figura 8.6:** Capítulo 01 do Curso de Álgebra.

## 8.5 ADL SCORM *Conformance Test Suite*

O SCORM *Conformance Test Suite* é uma ferramenta fornecida pela *Advanced Distributed Learning* (ADL) para avaliar (*self-test*) a conformance ou compatibilidade de sistemas de gerência da aprendizagem (LMS), de objetos de aprendizagem compartilháveis (SCOs), dos metadados XML e de pacotes agregados com a norma SCORM (ADL, 2005).

A atual versão da ferramenta é o SCORM 2004 *Conformance Test Suite* 1.3.3 que trabalha com a atualização do SCORM 2004. O *Test Suite* pode ser usado para avaliar a compatibilidade das quatro áreas principais do SCORM, fornecendo cinco componentes de teste como segue:

- Teste de compatibilidade do LMS (*LMS Conformance Test;*);
- Teste de compatibilidade do Pacote de conteúdo (*Content Package Conformance Test*);
- Teste de compatibilidade do SCO (*Sharable Content Object Run-Time Environment Conformance Utility Test*);
- Teste de compatibilidade do Metadado (*Metadata Conformance Utility Test*);
- Teste de compatibilidade do Manifesto (*Manifest Utility Test*).

Neste trabalho são feitos os testes de compatibilidade dos objetos de aprendizagem (SCO), dos metadados e dos manifestos.

### 8.5.1 *Sharable Content Object (SCO) Run-Time Environment (RTE) Conformance Utility Test*

O *SCO RTE Conformance Utility Test* verifica se o objeto (SCO) testado está de acordo com o ambiente de execução (*Run-Time Environment*) da norma SCORM. Esse teste verifica se o objeto pode ser inicializado (*launched*) por um LMS compatível com a norma.

### 8.5.2 *Metadata Conformance Utility Test*

O *Metadata Conformance Utility Test* verifica se o *Asset*, o SCO ou o *Content Aggregation* Metadado estão de acordo com o padrão. O teste verifica se o arquivo XML está formatado de acordo com o SCORM e compara o metadado com a definição IEEE LOM Version 1.0 *Schema* (XSD).

### 8.5.3 *Manifest Utility Test*

O *Manifest Utility Test* verifica se o arquivo de manifesto é compatível com as regras definidas pelo SCORM. Esse teste pode receber arquivos compactados (PIF) ou não. O teste verifica se o arquivo XML está formatado de acordo com o SCORM e compara o manifesto com a definição IEEE LOM Version 1.0 *Schema* (XSD).

### 8.5.4 Resultados

Foram verificados com a *Conformance Test Suite* todos os recursos educacionais adequados dos estudos de casos anteriores. A Tabela 8.1 mostra os resultados obtidos. Como se pode observar, todos os 157 recursos educacionais adequados estão compatíveis com a norma SCORM 2004.

**Tabela 8.1:** Resultados da validação dos Objetos de Aprendizagem.

| Tipos de Testes | TelEduc | AVA | Sucesso | Fracasso |
|---|---|---|---|---|
| Capacidade de Localizar o API *Adapter* | 65 | 92 | 157 | 0 |
| Capacidade de Invocar LMSInitialize() | 65 | 92 | 157 | 0 |
| Capacidade de Invocar LMSGetValue(cmi.core.lesson_mode) | 65 | 92 | 157 | 0 |
| Capacidade de Invocar LMSGetValue(cmi.core.lesson_status) | 65 | 92 | 157 | 0 |
| Capacidade de Invocar LMSSetValue(cmi.core.lesson_status, incomplete) | 65 | 92 | 157 | 0 |
| Capacidade de Invocar LMSSetValue(cmi.core.session_time, 0000:13:14.64) | 65 | 92 | 157 | 0 |
| Capacidade de Invocar LMSSetValue(cmi.core.lesson_status, completed) | 65 | 92 | 157 | 0 |
| Capacidade de Invocar LMSSetValue(cmi.core.exit,) | 65 | 92 | 157 | 0 |
| Capacidade de Invocar LMSFinish() | 65 | 92 | 157 | 0 |
| **Total de Objetos de Aprendizagem validados** | **157** | | - | |

A Figura 8.7 mostra um exemplo de registro de *log* gerado pela *Conformance Test Suite*, onde se pode observar quais ítens passaram ou não no teste, onde se verifica que os objetos adequados estão compatíveis com a norma SCORM.

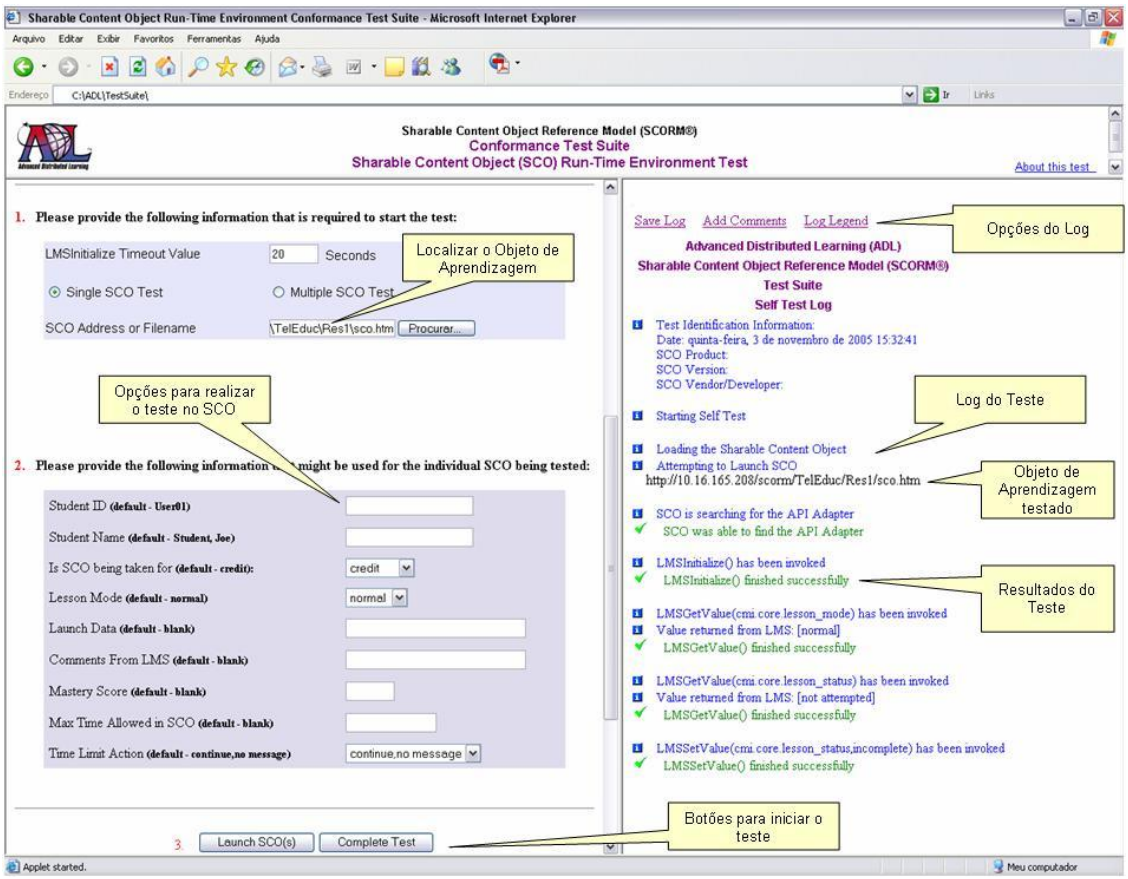


**Figura 8.7:** Exemplo de teste do *Conformance Test Suite* em um Objeto de Aprendizagem.

## 8.6 Conclusão

Os dois primeiros estudos de casos mostraram a adequação de dois repositórios educacionais (TelEduc e AVA), onde se pode constatar a consistência e compatibilidade da Fábrica de Adequação operando em diferentes Sistemas Operacionais e navegadores *web*.

O terceiro estudo traz uma implementação simplificada de um ambiente de *e-learning*, onde se verifica o acesso aos objetos armazenados no repositório da Fábrica.

Já no intuito de validar e testar a interoperabilidade dos objetos de aprendizagem, foi verificada a compatibilidade ou conformidade dos objetos gerados a partir da Fábrica com a norma SCORM. Para isso foi utilizada a ferramenta de certificação da ADL, onde, com os resultados obtidos (sub-Seção 8.5.4), é possível comprovar a validade dos objetos normalizados.

# CAPÍTULO 9

# Conclusões e Trabalhos Futuros

Considerando a grande preocupação com os custos e qualidade dos cursos oferecidos a distância (*e-learning*), principalmente em países em desenvolvimento como o Brasil, esta proposta permite promover a interoperabilidade dos conteúdos educacionais e a adequação de recursos educacionais para Objetos de Aprendizagem Reutilizáveis respeitando a norma SCORM.

Para seu desenvolvimento, foi realizado um estudo sobre a Educação a Distância, destacando a educação via *Web* e o surgimento de especificações no aprendizado eletrônico (*e-learning*). Juntamente com essa investigação, os trabalhos relacionados foram fundamentais para entender o funcionamento de ambientes e repositórios que utilizam o paradigma de objetos de aprendizagem reutilizáveis, além de permitir a identificação da funcionalidade e serviços necessários para a adequação de recursos educacionais à norma SCORM.

Posteriormente, foram estudadas técnicas computacionais necessárias para o desenvolvimento desta pesquisa. Assim, foram realizados estudos sobre Agentes de *Software*, *Web Services* e Objetos de Aprendizagem Reutilizáveis. Essas técnicas e conceitos permitiram desenvolver a implementação de uma solução capaz de atender os objetivos propostos.

Para a implementação deste trabalho foi escolhida a plataforma de desenvolvimento Microsoft .NET, principalmente pela sua facilidade em lidar com *Web Services* e pela ampla utilização para o desenvolvimento *web*.

Assim, tendo como base os estudos e experimentos realizados, foi possível modelar a sua arquitetura base e criar os diagramas de caso de uso, classes e seqüência. Finalizada a etapa de modelagem, foram implementados os serviços a serem disponibilizados: Sistema de Geração de um SCO (SCORM *Single Item Package*) e Sistema de Geração de Pacote Agregado (SCORM *Package Aggregator*). Esses serviços, juntamente com os agentes da camada de persistência, promovem a adequação de recursos educacionais para o padrão SCORM, gerando objetos de aprendizagem que são armazenados no repositório local, que podem ser recuperados através do Mecanismo de Comunicação.

Como forma de testar e validar os serviços desenvolvidos, foram realizados estudos de casos, onde foi possível avaliar a Fábrica de Adequação. Dentro desses estudos, inicialmente foram realizados testes para verificar a compatibilidade e consistências da Fábrica, onde se observa que o sistema desenvolvido não apresentou problemas com o ambiente de execução, seja com o tipo de mídia dos recursos, os tipos de navegadores *web* e o tipo do Sistema Operacional.

O terceiro estudo de caso buscou avaliar o funcionamento do repositório da Fábrica de Adequação através da implementação de um ambiente de *e-learning*, o qual foi capaz de acessar os objetos adequados nos dois primeiros estudos de casos, levando à conclusão que a arquitetura é capaz de realizar o objetivo proposto.

Outra forma de validação foi aplicada nesta pesquisa, através da ferramenta *Conformance Test Suite* fornecida pelo Consórcio ADL responsável pela norma SCORM. Assim, todos os objetos de aprendizagem adequados nos estudos de casos foram considerados compatíveis (válidos) com a norma SCORM, levando à conclusão que a Fábrica de Adequação é capaz de adequar (normalizar) recursos educacionais para o padrão SCORM.

A pesquisa ainda obteve as seguintes contribuições:

- Um editor de Metadados, capaz de criar e editar metadados padrão SCORM para objetos de aprendizagem;
- Um editor de Estruturação de Agregação, capaz de gerar objetos de aprendizagem mais complexos através da criação de uma árvore de conhecimento (hierárquica ou seqüencial) representada no arquivo manifesto do objeto;
- Um repositório SCORM, capaz de promover a interoperabilidade de conteúdos educacionais e, ainda, criar múltiplos contextos para um mesmo objeto de aprendizagem.

Ainda, tivemos a oportunidade de publicar o artigo intitulado "*An architecture for courseware adequacy to re-useable learning objects* (SCORM)" (Kratz et al., 2005) no IEEE *International Conference on Next Generation Web Services Practices* (NWeSP'05), onde foi possível mostrar a arquitetura da Fábrica de Adequação para a comunidade científica e ainda discutir a importância da interoperabilidade e do aproveitamento dos recursos educacionais.

Desta forma, os objetivos deste trabalho foram plenamente alcançados, na medida que, dada sua conclusão, a comunidade de desenvolvedores tem à disposição as funcionalidades necessárias para promover a interoperabilidade de recursos educacionais e a adequação destes para o padrão SCORM.

## *9.1* Trabalhos Futuros

As possíveis linhas de estudo para a continuação ou aprimoramento deste trabalho são apresentadas a seguir:

- Manter e aprimorar os serviços desenvolvidos com o objetivo de atender o maior número de demandas possíveis com garantia de flexibilidade, interoperabilidade e qualidade;
- Inserir outros Agentes de Pesquisa (Mecanismo de Pesquisa de Objetos de Aprendizagem):
  - Desenvolver um sistema de "Busca Avançada" que permita que sejam selecionados múltiplos campos dos metadados dos objetos de aprendizagem, desta maneira, refinado a pesquisa de uma ou mais categorias do metadado para se fornecer uma consulta ainda mais detalhada e precisa;

- Estudar mecanismos para a visualização dos objetos e seus relacionamentos em forma de grafos:
  - O usuário poderia visualizar mapas de objetos de aprendizagem ou quaisquer grupos de recursos cadastrados no repositório da Fábrica. A visualização poderia ser feita através da estrutura de um grafo onde os vértices seriam os SCOs e as arestas seriam os relacionamentos entre eles. Os níveis de profundidade do grafo, bem como os tipos de relacionamentos de interesse que seriam definidos pelo usuário;
  - A visualização e navegação sobre um mapa de objetos de aprendizagem poderiam agilizar o processo de busca por recursos educacionais ou complementares dentro do repositório da Fábrica de Adequação;
- Aumentar ou extender a oferta de Padrões de forma a possibilitar adequação de objetos de aprendizagem para mais ambientes de EAD:
  - Permitir que a Fábrica trabalhe na interoperabilidade de conteúdos seguindo outras especificações e normas como o IMS (IMS, 2005) e o LOM (LOM, 2005);
  - Adicionar outras especificações, como por exemplo a especificação *Educational Modelling Language* (EML) que permite adicionar mais informações pedagógicas aos recursos educacionais, assim como foi implementado no projeto RIVED (Rived, 2005);
- Investigar a possibilidade da introdução de Agentes de *Software* Inteligentes na criação de contextos (metadados) automáticos dos objetos de aprendizagem.

# Referências


(Albuquerque, 2005) Albuquerque, T. (2005). Rived Ativa: Padrões de OA Rived. Rede Internacional Virtual de Educação (RIVED).

(Aroyo and Kommers, 1999) Aroyo, L. and Kommers, P. (1999). Preface - Intelligent agents for educational computer-aided systems. *Journal of Interactive Learning Research, v. 10, n. 3/4, p. 235-242.*

(Barbeira and Santos, 2002) Barbeira, B. and Santos, A. (2002). Desenvolvimeno de conteúdo normalizado para ambiente de e-learning: um estudo de caso na PT inovação. *6º Congresso Iberoamericano de Informática Educativa p. 1-12.*

(Barrit, 2001) Barrit, C. (2001). *Reusable Learning Object Strategy, version 4.0. p. 1-48.* Cisco Systems.

(Bettio, 2003) Bettio, R. (2003). Avaliações gráficas e dinâmicas aplicadas a ambientes virtuais de aprendizagem. Dissertação de Mestrado, Universidade Federal de Santa Catarina. Florianópolis. Dissertação de Mestrado.

(Bradshaw, 1997) Bradshaw, J. (1997). *Software Agents.* Cambridge: MIT p. 490.

(Bray et al., 2000) Bray, T., Paoli, J., McQueen, C., and Maller, E. (2000). Extensible Markup Languagem. 6 de outubro de 2000.

(Brusilovsky, 1998) Brusilovsky, P. (1998). Adaptive educational systems on the world-wide-web: A review of available technologies. *In: Proceedings of Workshop "WWW-Based Tutoring" at 4th International Conference on Intelligent Tutoring Systems (ITS'98), San Antonio, TX, August 16-19,.*

(Cerami, 2002) Cerami, E. (2002). *Web Services Essentials - Distributed Applications with XML-RPC, SOAP, UDDI and WSDL.* O'Reilly, Primeira Edição p. 304.

(Crawford, 2001) Crawford, E. (2001). Toolbook on the web with neuron: a preliminary case study with steps. *Maths, Stats & OR (MSOR Connections) ISSN 1473-4869 p. 20-22.*

(Downes, 2002) Downes, S. (2002). Smart learning objects. *maio.*

(Fallon and Brown, 2002) Fallon, C. and Brown, S. (2002). *E-learning Standards: A Guide to Purchasinig, Developing, and Deploying Standards-Cpnfprmant e-Learning.* St. Lucie Press, ISBN: 1574443453 272p.

(Ferris and Farrell, 2003) Ferris, C. and Farrell, J. (2003). What are web services? *Communications of the ACM ISSN:0001-0782 v. 46 31p.*.

(Franklin and Graesser, 1996) Franklin, S. and Graesser, A. (1996). Is it an agent or just a program? a taxonomy for autonomous agents. *3ž International Workshop on Agent Theories, Architectures, and Languages. Springer-Verlag p. 21-35*, v.46.

(Gamma et al., 2000) Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (2000). *Padrões de Projeto: Soluções Reutilizáveis de Software Orientado a Objetos.* Tradução de Luiz A. Meirelles Salgado. Porto Alegre: Bookman.

(Goñi and Fuks, 2002) Goñi, J. and Fuks, H. (2002). Comparação de ambientes de ensino na web baseados em plataforma ims a partir dos papéis dos atores envolvidos. *PUC-RioInf.MCC14/02.*

(Gomes, 2005) Gomes, E. (2005). Objetos Inteligentes de Aprendizagem: uma abordagem em agentes para objetos de arendizagem. Dissertação de Mestrado, Universidade Federal do Rio Grande do Sul, UFRGS. Dissertação de Mestrado.

(Gomes et al., 2004) Gomes, E., Silveira, R., and Viccari, R. (2004). Objetos de aprendizagem: uma abordagem baseada em agentes para objetos de aprendizagem. *XV Simpósio Brasileiro de Informática na Educação - SBIE2004. Manaus.*

(González, 2000) González, L. (2000). Metodologia e ferramenta de elaboração de curso com navegação dinâmica. Dissertação de Mestrado, Escola Politécnica da Universidade de São Paulo, São Paulo. Dissertação de Mestrado.

(Graham et al., 2002) Graham, S., Simeonov, S., Boubez, T., Davis, D., Daniels, G., Yuichi, N., and Neyama, R. (2002). *Building Web Services With Java.* SAMS.

(Hansen, 2003) Hansen, R. (2003). Gluescript: Uma linguagem específica de domínio para composição de web services. Dissertação de Mestrado, Universidade do Vale do Rio dos Sinos - Unisinos. Dissertação de Mestrado.

(Holmberg, 1985) Holmberg, B. (1985). Educación a distancia: situación y perspectivas. *Buenos Aires: Editorial Kapeluz. p. 1-25.*

(IEEE, 2002) IEEE (2002). IEEE P1484.12.1/D6.4 draft standard for learning object metadata. Janeiro 2002.

(Jagiello and Júnior, 2003) Jagiello, I. and Júnior, E. (2003). Web services uma solução para aplicações distribuídas na internet. In *Pontifícia Universidade Católica do Paraná.*

(Keegan, 1991) Keegan, D. (1991). Foundations of distance education. *2a.ed. Londres: Routledge. p. 224.*

(Kelly and Barritt, 1998) Kelly, T. and Barritt, C. (1998). *Cisco's white paper on Reusable Learning Objects.* Cisco Systems Inc.

(Klusch, 1999) Klusch, M. (1999). *Intelligent information agents: agent based discovery and management on the Internet.* Springer-Verlag. ISBN:3540651128 p. 521.

(Kratz et al., 2005) Kratz, R., Pinto, S., and Scopel, M. (2005). An architecture for courseware adequacy to re-useable learning objects (scorm). *IEEE International Conference on Next Generation Web Services Practices (NWeSP'05).*

(Looms and Christensen, 2002) Looms, T. and Christensen, C. (2002). *Advanced Distributed Learning Emerging and Enabling Technologies for the Design of Learning Object Repositories Report.* Version 1 Advanced Distributed Learnin. Alexandria, VA. 66p.

(Loyolla and Prates, 1999) Loyolla, W. and Prates, M. (1999). Educação à Distância mediada por computador (EAD): Uma proposta pedagógica. *PUCCAMP.*

(Lucena and Fuks, 2000) Lucena, C. and Fuks, H. (2000). Professores e aprendizes na Web: A educação na era da Internet. *Rio de Janeiro: Clube do Futuro. ISBN: 85-88011-01-8.*

(Lucena et al., 2001) Lucena, C., Silva, V., and Fuks, H. (2001). Contentnet: A framework for the interoperability of educational content using standard IMS. *Computers & Education Journal, Elsevier Science Press, v. 37 n. 3/4 p. 273-295.*

(McCormick, 2004) McCormick, R. (2004). Issues of leraning and knowledge in technology education. *International Journal of Technology and Design Education, n. 1, v. 1 p. 21-44.*

(Menezes, 2004) Menezes, L. (2004). Uma avaliação em percurso de rede internacional virtual de educação para o melhoramento da aprendizagem de ciência e matemática na américa latina (rived). *In: Guiomar Namo de Mello. (Org.) Ofício de Professor na América Latina e Carobe. SãoPaulo, v., p.157-162.*

(Moore and Kearsley, 1996) Moore, M. and Kearsley, G. (1996). *Distance education: a systems view.* Belmont, California: Wadsworth. Segunda Edição 336p.

(Muzio et al., 2001) Muzio, J., Heins, T., and Mundell, R. (2001). *Experiences with Reusable eLearning Objects: Theory to Practice.* Victoria, Canadá.

(Nascimento, 2004) Nascimento, A. (2004). Taking the next step with the project. *Proceedings of ED-MEDIA 2004 World Conference on Educational Multimedia, Hypermedia and Telecommunications. 6p.*

(Nascimento and Morgado, 2004) Nascimento, A. and Morgado, E. (2004). Um projeto de colaboração internacional na américa latina. *Artigo e Textos RIVED.*

(Neuron, 2005) Neuron (2005). Ajuda (help) do software neuron 8.5.

(Newcomer, 2002) Newcomer, E. (2002). *Understanding Web Services. Independent Technology Guide.* Dadid Chappell, Series Editor.

(Nielsen, 1999) Nielsen, J. (1999). *Designing Web Usability: The Practice of Simplicity.* USA: New Riders. ISBN 1-56205-810-X. 432p.

(Nunes, 1994) Nunes, I. (1994). Noções de educação a distância. *Revista Educação a Distância n. 4/5, p. 7-25.*

(Nwana, 1996) Nwana, H. (1996). *Software agents: an overview.* Knowledge Engineering Review, p. 205-244.

(OMG, 2000) OMG (2000). Agent technology - green paper. Object Management Group. Versão 1.0, 2000.

(Ostyn, 2003) Ostyn, C. (2003). Cooking up a scorm: A scorm 1.2 content cookbook for developers. *Learning Standards Strategist Click2learn, Inc. White Paper , Version 1.2 - Draft 0.8, 2003.*

(Pereira et al., 2003) Pereira, L., Porto, F., Machado, M., and Nascimento, R. (2003). Objetos de Aprendizado Reutilizáveis (RLOs): Conceitos, padronização, uso e armazenamento. *PUC-RioInf.MCC10/03.*

(Pinto et al., 2002) Pinto, S., Schlemmer, E., Santos, C., Pérez, C., and Rheinheimer, L. (2002). Ava: Um ambiente virtual baseado em comunidades. *In: XIII Simpósio Brasileiro de Informática na Educação (SBIE), 2002, São Leopoldo. XIII Simpósio Brasileiro de Informática na Educação, 2002. v. 1. p. 31-38.*

(Rheinheimer, 2002) Rheinheimer, L. (2002). Jlearningservices: Um framework para serviços síncronos em ambientes para ead. Monografia em informática. São Leopoldo: Curso de Informática da UNISINOS.

(Rheinheimer, 2004) Rheinheimer, L. (2004). Wsagent: Um agente baseado em web services para promover a interoperabilidade entre sistemas heterogêneos no domínio da saúde. Dissertação de Mestrado, Universidade do Vale do Rio dos Sinos - Unisinos. Dissertação de Mestrado.

(Rosemberg, 2002) Rosemberg, M. (2002). *e-Learning.* São Paulo: Editora Makron. p. 112.

(Sá, 2000) Sá, P. (2000). Gerador automático de arquivo html de ajuda para aplicação em educação (Gaaha). Dissertação de Mestrado, São Carlos: Instituto de Ciência Matemáticas e de Computação da USP. Dissertação de Mestrado.

(Salvador, 1995) Salvador, V. (1995). Hipermídia interativa - a educação do futuro, no presente. *Tecnologia Educaional. v. 22, n. 123/124, p. 22-23.*

(Santos, 2002) Santos, D. (2002). RDF na interoperabilidade entre domínios na Web. Dissertação de Mestrado, Universidade Federal de Campina Grande. Dissertação de Mestrado.

(Simonson et al., 2002) Simonson, M., Smaldino, S., Michael, A., and Zvacek, S. (2002). *Teaching And Learning At Distance: Foundations Of Distance Education.* New Jersey: Prentice Hall. Segunda Edicção. ISBN: 013094629X. p. 320.

(Sommerville, 2003) Sommerville, I. (2003). *Engenharia de Software.* Tradução André Masurício de andrade Ribeiro - Addison Wesley, São Paulo.

(Souza, 2004) Souza, V. (2004). Swservice: uma biblioteca para a escrita da língua brasileira de sinais baseada em Web Services. Dissertação de Mestrado, Universidade do Vale do Rio dos Sinos - Unisinos. Dissertação de Mestrado.

(Walsh, 2002) Walsh, A. (2002). *UDDI, SOAP and WSDL The Web Services Specification Reference Book.* Prentice Hall Books. ISBN: 0596000952. 260p.

(Zaina, 2002) Zaina, L. (2002). Acompanhamento do aprendizado do aluno em cursos a distância através da web: metodologias e ferramenta. Dissertação de Mestrado, Escola Politécnica da Universidade de São Paulo, São Paulo. Dissertação de Mestrado.

(Zaina et al., 2004) Zaina, L., Ruggiero, W., and Bressan, G. (2004). Metodologia para acompanhamento da aprendizagem através da web. *SBC: Revista Brasileira de Informática na Educação. ISSN: 1414-5685, v. 12, n. 1, p. 20-27.*

# *Web*-Referências


(Academia_E-learning, 2005) Academia_E-learning (2005). Click2learn, inc. - Scorm 1.2 resource kit. [http://academiaelearning.com/contenido/scorm/] acessado em (02/04/2005).

(ADL, 2005) ADL (2005). Advanced Disributed Learning. The Scorm section of the resource center contains available versions of the sharable content object reference mode. [http://www.adlnet.org/] acessado em (10/01/2005).

(AICC, 2005) AICC (2005). AICC Guidelines and Recommendations - Aviation Industry CBT Committee. [http://www.aicc.org/] acessado em (13/04/2005).

(ARIADNE, 2005) ARIADNE (2005). Alliance of Remote Institute of Electrical & Distribution Networks for Europe. [http://ariadne.unil.ch/] acessado em (20/01/2005).

(AVA, 2005) AVA (2005). Ambiente Virtual de Aprendizagem da Universidade do Vale do rio dos Sinos. [http://www.ava.unisinos.br/] acessado em (05/09/2005).

(Blackboard, 2005) Blackboard (2005). Blackboard Commerce Suite. [http://www.blackboard.com/] acessado em (01/03/2005).

(CanCore, 2005) CanCore (2005). Canadian Core about. [http://www.cancore.ca/about.html] acessado em (22/01/2005).

(CEdMA, 2005) CEdMA (2005). Computer Education Management Association (CEdMA). [http://www.cedma.net/] acessado em (05/04/2005).

(DCMI, 2005) DCMI (2005). DCMI Education Working Group. [http://www.dublincore.org/groups/education/] acessado em (24/02/2005).

(E-Learning, 2005) E-Learning (2005). E-learning Brasil. Curriculum Kevin Oakes. [http://www.elearningbrasil.com.br/congresso/2005/palestrantes/kevin.asp] acessado em (02/04/2005).

(EDtech, 2005) EDtech (2005). National University of Singapore, centre for instructional technology, courseware development. [http://courseware.nus.edu.sg/Standards/rlo.asp] acessado em (12/01/2005).

(eProInfo, 2005) eProInfo (2005). Ambiente colaborativo de aprendizagem e-proinfo. [http://www.eproinfo.mec.gov.br/] acessado em (27/02/2005).

(Filho, 2005) Filho, A. (2005). Entendento o XML. [http://www.guj.com.br/java.artigo.19.1.guj] acessado em (25/03/2005).

(FIPA, 2005) FIPA (2005). Foundation for intelligent physical agents. specifications repository. [http://www.fipa.org/specs/pesspecs.tar.gz/] acessado em (11/08/2005).

(Formare, 2005) Formare (2005). Formare - soluções globais de elearning. [http://www.formare.pt/] acessado em (17/05/2005).

(IEEE, 2005) IEEE (2005). Learning technology standards committee. "specifications". [http://ltsc.ieee.org/] acessado em (15/01/2005).

(IMS, 2005) IMS (2005). Instructional management systems. IMS Specifications. [http://www.imsproject.org/specifications.html] acessado em (17/01/2005).

(Jacobsen, 2004) Jacobsen, P. (2004). E-learning magazine. [http://www.ltimagazine.com/ltimagazine/article/articleDetail.jsp?id=5043] acessado em (11/12/2004).

(LOM, 2005) LOM (2005). IEEE information technology - Learning Technology - Learning Objects Meta-data LOM: Working drafty. resource meta-data specification. [http://ltsc.ieee.org/] acessado em (15/02/2005).

(OMG, 2003) OMG (2003). OMG Unified Modeling Language Specification, version 1.5. [http://www.omg.org/technology/documents/formal/uml.htm] acessado em (12/04/2005).

(RIVED, 2005) RIVED (2005). Portal da Rede Internacional Virtual de Educação (RIVED) no Brasil. [http://rived.eproinfo.mec.gov.br] acessado em (02/06/2005).

(SCORM, 2005) SCORM (2005). The scorm section of the resource center contains available versions of the sharable content object reference model (Scorm). [http://www.adlnet.org/index.cfm?fuseaction=srchrslt&filterid=35] acessado em (10/01/2005).

(Sullivan, 2005) Sullivan, D. (2005). Webmaster's guide to search engines. [http://searchenginewatch.internet.com] acessado em (17/08/2005).

(TelEduc, 2005) TelEduc (2005). Teledu repositório e ambiente de EAD. Universidade Estadual de Campinas (Unicamp). [http://teleduc.nied.unicamp.br/teleduc/] acessado em (10/08/2005).

(W3C, 2005) W3C (2005). World Wide Web Consortium. [http://www.w3.org/] acessado em (11/03/2005).

(WG12, 2005) WG12 (2005). Learning Object Metadata group (LOM). [http://ltsc.ieee.org/wg12/] acessado em (10/02/2005).

# APÊNDICE A

# Arquivo exemplo de Metadado "metadata.xml"

Este Apêndice traz um arquivo de metadado exemplo gerado pelo Editor de Metadado.

```
<?xml version="1.0"?>
<lom xmlns="http://www.imsglobal.org/xsd/imsmd_rootv1p2p1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemalocation="http://www.imsglobal.org/xsd/imsmd_rootv1p2p1
imsmd_rootv1p2p1.xsd"
xsi:schemaLocation="http://www.imsglobal.org/xsd/imsmd_rootv1p2p1
imsmd_rootv1p2p1.xsd">
   <general>
      <title>
         <langstring xml:lang="x-none">Exemplo de SCORM Single Item
Package</langstring>
      </title>

      <catalogentry>
         <catalog>EAD</catalog>

         <entry>
            <langstring xml:lang="x-none">Exemplo</langstring>
         </entry>
      </catalogentry>

      <description>
         <langstring xml:lang="x-none">Este documento é um exemplo de metadado
de uma SCORM Single Item Package</langstring>
      </description>

      <keyword>
         <langstring xml:lang="x-none">Metadado</langstring>
      </keyword>

      <keyword>
         <langstring xml:lang="x-none">XML</langstring>
      </keyword>

      <keyword>
         <langstring xml:lang="x-none">Exemplo</langstring>
      </keyword>

      <keyword>
         <langstring xml:lang="x-none">SCORM</langstring>
      </keyword>
   </general>

   <lifecycle>
      <version>
         <langstring xml:lang="x-none">1.0</langstring>
      </version>

      <status>
         <source>
            <langstring xml:lang="x-none">LOMv1.0</langstring>
         </source>
```

```
         <value>
            <langstring xml:lang="x-none">Final</langstring>
         </value>
      </status>

      <contribute>
         <role>
            <source>
               <langstring xml:lang="x-none">LOMv1.0</langstring>
            </source>

            <value>
               <langstring xml:lang="x-none">Author</langstring>
            </value>
         </role>

         <centity>
            <vcard>BEGIN:vCard VERSION:3.0 FN:Ricardo Kratz ORG:Unisinos
LABEL:Av Unisinos, São Leo EMAIL;TYPE=INTERNET:rkratz@turing.unisinos.br
END:vCard</vcard>
         </centity>

         <date>
            <datetime>2005-04-18</datetime>
         </date>
      </contribute>
   </lifecycle>

   <metametadata>
      <metadatascheme>ADL SCORM 1.2</metadatascheme>
   </metametadata>

   <technical>
      <format>application/pdf</format>
      <format>text/html</format>
      <size>88035</size>
      <location>.</location>
   </technical>

   <educational />

   <rights>
      <cost>
         <source>
            <langstring xml:lang="x-none">LOMv1.0</langstring>
         </source>

         <value>
            <langstring xml:lang="x-none">no</langstring>
         </value>
      </cost>

      <copyrightandotherrestrictions>
         <source>
            <langstring xml:lang="x-none">LOMv1.0</langstring>
         </source>

         <value>
            <langstring xml:lang="x-none">no</langstring>
         </value>
      </copyrightandotherrestrictions>

      <description>
         <langstring xml:lang="x-none">No Copyright</langstring>
      </description>
   </rights>

   <classification>
```

```
      <purpose>
         <source>
            <langstring xml:lang="x-none">LOMv1.0</langstring>
         </source>

         <value>
            <langstring xml:lang="x-none">Idea</langstring>
         </value>
      </purpose>

      <description>
         <langstring xml:lang="x-none">Este documento é um exemplo de metadado
de uma SCORM Single Item Package</langstring>
      </description>

      <keyword>
         <langstring xml:lang="x-none">Metadado</langstring>
      </keyword>

      <keyword>
         <langstring xml:lang="x-none">XML</langstring>
      </keyword>

      <keyword>
         <langstring xml:lang="x-none">Exemplo</langstring>
      </keyword>

      <keyword>
         <langstring xml:lang="x-none">SCORM</langstring>
      </keyword>
   </classification>
</lom>
```

**Figura A.1:** Exemplo de metadado de um SCO "metadado.xml".

# APÊNDICE B

# Arquivo exemplo de Manifesto "imsmanifest.xml"

Este Apêndice traz um arquivo de manifesto exemplo gerado pelo Editor de Estruturação de Agregação.

```
<?xml version="1.0"?>
<manifest identifier="exemplo.PDF.200503111527-B263E582-435A-4433-38B8-
EC889D6KDE2E" version="20050418165114"
xmlns="http://www.imsproject.org/xsd/imscp_rootvlplp2"
xmlns:adlcp="http://www.adlnet.org/xsd/adlcp_rootvlp2"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.imsproject.org/xsd/imscp_rootvlplp2
imscp_rootvlplp2.xsd http://www.imsglobal.org/xsd/imsmd_rootvlp2pl
imsmd_rootvlp2pl.xsd http://www.adlnet.org/xsd/adlcp_rootvlp2
adlcp_rootvlp2.xsd">

   <metadata>
      <schema>ADL SCORM</schema>

      <schemaversion>1.2</schemaversion>

      <lom xmlns="http://www.imsglobal.org/xsd/imsmd_rootvlp2pl"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemalocation="http://www.imsglobal.org/xsd/imsmd_rootvlp2pl
imsmd_rootvlp2pl.xsd">

         <general>
            <title>
               <langstring xml:lang="x-none">Exemplo de SCORM Single Item
Package</langstring>
            </title>

            <catalogentry>
               <catalog>EAD</catalog>

               <entry>
                  <langstring xml:lang="x-none">Exemplo</langstring>
               </entry>
            </catalogentry>

            <description>
               <langstring xml:lang="x-none">Este documento é um exemplo de
metadado de uma SCORM Single Item Package</langstring>
            </description>

            <keyword>
               <langstring xml:lang="x-none">Metadado</langstring>
            </keyword>

            <keyword>
               <langstring xml:lang="x-none">XML</langstring>
            </keyword>
```

```
            <keyword>
               <langstring xml:lang="x-none">Exemplo</langstring>
            </keyword>

            <keyword>
               <langstring xml:lang="x-none">SCORM</langstring>
            </keyword>
         </general>

         <lifecycle>
            <version>
               <langstring xml:lang="x-none">1.0</langstring>
            </version>

            <status>
               <source>
                  <langstring xml:lang="x-none">LOMv1.0</langstring>
               </source>

               <value>
                  <langstring xml:lang="x-none">Final</langstring>
               </value>
            </status>

            <contribute>
               <role>
                  <source>
                     <langstring xml:lang="x-none">LOMv1.0</langstring>
                  </source>

                  <value>
                     <langstring xml:lang="x-none">Author</langstring>
                  </value>
               </role>

               <centity>
                  <vcard>BEGIN:vCard VERSION:3.0 FN:Ricardo Kratz ORG:Unisinos
LABEL:Av Unisinos, São Leo EMAIL;TYPE=INTERNET:rkratz@turing.unisinos.br
END:vCard</vcard>
               </centity>

               <date>
                  <datetime>2005-04-18</datetime>
               </date>
            </contribute>
         </lifecycle>

         <metametadata>
            <metadatascheme>ADL SCORM 1.2</metadatascheme>
         </metametadata>

         <technical>
            <format>application/pdf</format>

            <format>text/html</format>

            <size>88035</size>

            <location>.</location>
         </technical>

         <educational />

         <rights>
            <cost>
               <source>
                  <langstring xml:lang="x-none">LOMv1.0</langstring>
```

```
               </source>

               <value>
                  <langstring xml:lang="x-none">no</langstring>
               </value>
            </cost>

            <copyrightandotherrestrictions>
               <source>
                  <langstring xml:lang="x-none">LOMv1.0</langstring>
               </source>

               <value>
                  <langstring xml:lang="x-none">no</langstring>
               </value>
            </copyrightandotherrestrictions>

            <description>
               <langstring xml:lang="x-none">No Copyright</langstring>
            </description>
         </rights>

         <classification>
            <purpose>
               <source>
                  <langstring xml:lang="x-none">LOMv1.0</langstring>
               </source>

               <value>
                  <langstring xml:lang="x-none">Idea</langstring>
               </value>
            </purpose>

            <description>
               <langstring xml:lang="x-none">Este documento é um exemplo de
metadado de uma SCORM Single Item Package</langstring>
            </description>

            <keyword>
               <langstring xml:lang="x-none">Metadado</langstring>
            </keyword>

            <keyword>
               <langstring xml:lang="x-none">XML</langstring>
            </keyword>

            <keyword>
               <langstring xml:lang="x-none">Exemplo</langstring>
            </keyword>

            <keyword>
               <langstring xml:lang="x-none">SCORM</langstring>
            </keyword>
         </classification>
      </lom>
   </metadata>

   <organizations default="ORG_1">
      <organization identifier="ORG_1" structure="hierarchical">
         <title>Content</title>

         <item identifier="ITEM1" identifierref="RES1"
isvisible="true">
            <title>Content</title>
         </item>
      </organization>
   </organizations>
```

```
    <resources>
      <resource identifier="RES1" type="webcontent" adlcp:scormtype="sco"
href="sco.htm">
        <file href="sco.htm" />

        <file href="dummy.htm" />

        <file href="SCORMGenericLogic.js" />

        <file href="exemplo.PDF" />
      </resource>
    </resources>

</manifest>
```

**Figura B.1:** Exemplo de manifesto de um SCO "imsmanifest.xml".

# APÊNDICE C

# Código Genérico JavaScrit de Comunicação "SCORMGenericLogic.js"

Este Apêndice traz o código genérico utilizado na Fábrica de Adequação respeitando as especificações da norma SCORM.

```
// Iniciação da API INTERFACE e demais Funções da norma SCORM //
var g_nFindAPITries = 0;
var g_objAPI = null;
var g_bInitDone = false;
var g_bFinishDone = false;
var    g_bSCOBrowse = false;
var g_dtmInitialized = new Date(); // sera ajustado após ser iniciado

function AlertUserOfAPIError(strText) {
      if (g_bShowApiErrors) alert(strText)
}
// Procura a API do LMS
function FindAPI(win) {
      while ((win.API == null) && (win.parent != null) && (win.parent !=
win)) {
            g_nFindAPITries ++;
            if (g_nFindAPITries > 500) {
                  AlertUserOfAPIError(g_strAPITooDeep);
                  return null;
            }
            win = win.parent;
      }
      return win.API;
}
// API INTERFACE Encontrada
function APIOK() {
      return ((typeof(g_objAPI)!= "undefined") && (g_objAPI != null))
}
// Inicia o SCO
function SCOInitialize() {
      var err = true;
      if (!g_bInitDone) {
            g_bInitDone = true;
            if ((window.parent) && (window.parent != window)){
                  g_objAPI = FindAPI(window.parent)
            }
            if ((g_objAPI == null) && (window.opener != null))  {
                  g_objAPI = FindAPI(window.opener)
            }
            if (!APIOK()) {
                  AlertUserOfAPIError(g_strAPINotFound);
                  err = false
            } else {
                  err =  g_objAPI.LMSInitialize("");
```

```
                  if (err == "true") {
                        g_bSCOBrowse =
(g_objAPI.LMSGetValue("cmi.core.lesson_mode") == "browse");
                        if (!g_bSCOBrowse) {
                              if
(g_objAPI.LMSGetValue("cmi.core.lesson_status") == "not attempted") {
                                    err=
g_objAPI.LMSSetValue("cmi.core.lesson_status","incomplete")
                              }
                        }
                  } else {
                        AlertUserOfAPIError(g_strAPIInitFailed)
                  }
            }
            if (typeof(SCOInitData) != "undefined") {
                  // A função SCOInitData pode ser definida em outro script
do SCO
                  SCOInitData()
            }
      }
      g_dtmInitialized = new Date();
      return (err + "") // Força o  tipo para string
}
// Finaliza o SCO
function SCOFinish() {
      if ((APIOK()) && (g_bFinishDone == false)) {
            if (typeof(SCOSaveData) != "undefined"){
                  SCOReportSessionTime()
                  // A função SCOSaveData pode ser definida em outro script
do SCO
                  SCOSaveData();
            }
            g_bFinishDone = (g_objAPI.LMSFinish("") == "true");
      }
      return (g_bFinishDone + "" ) // Force type to string
}

// Chamada as outras funções estabelecida pelo norma SCORM
// Função para solicitar dados ao LMS
function SCOGetValue(nam)              {return
((APIOK())?g_objAPI.LMSGetValue(nam.toString()):"")}
// Função Commit, garante que o dados envidado seja persistida
function SCOCommit(parm)               {return
((APIOK())?g_objAPI.LMSCommit(""):"false")}
// Busca o número de erro
function SCOGetLastError(parm){return
((APIOK())?g_objAPI.LMSGetLastError(""):"-1")}
// Busca a descricão do erro
function SCOGetLastError(parm){return
((APIOK())?g_objAPI.LMSGetLastError(""):"-1")}
// Busca a descrição do erro
function SCOGetErrorString(n)    {return
((APIOK())?g_objAPI.LMSGetErrorString(n):"No API")}
// Busca o diagnostico do erro
function SCOGetDiagnostic(p)     {return
((APIOK())?g_objAPI.LMSGetDiagnostic(p):"No API")}

// Função de envio de valor ao LMS
var g_bMinScoreAcquired = false;
var g_bMaxScoreAcquired = false;

function SCOSetValue(nam,val){
      var err = "";
      if (!APIOK()){
            AlertUserOfAPIError(g_strAPISetError + "\n" + nam + "\n" +
val);
            err = "false"
      } else if (nam == "cmi.core.score.raw") err = privReportRawScore(val)
      if (err == ""){
            err =  g_objAPI.LMSSetValue(nam,val.toString() + "");
            if (err != "true") return err
```

```
        }
        if (nam == "cmi.core.score.min"){
                g_bMinScoreAcquired = true;
                g_nSCO_ScoreMin = val
        }
        else if (nam == "cmi.core.score.max"){
                g_bMaxScoreAcquired = true;
                g_nSCO_ScoreMax = val
        }
        return err
}
function privReportRawScore(nRaw) { // chamado apenas pelo SCOSetValue
        if (isNaN(nRaw)) return "false";
        if (!g_bMinScoreAcquired){
                if
(g_objAPI.LMSSetValue("cmi.core.score.min",g_nSCO_ScoreMin+"")!= "true")
return "false"
        }
        if (!g_bMaxScoreAcquired){
                if
(g_objAPI.LMSSetValue("cmi.core.score.max",g_nSCO_ScoreMax+"")!= "true")
return "false"
        }
        if (g_objAPI.LMSSetValue("cmi.core.score.raw", nRaw)!= "true")
        return "false";
        g_bMinScoreAcquired = false;
        g_bMaxScoreAcquired = false;
        if (!g_bMasteryScoreInitialized){
                var nMasteryScore =
parseInt(SCOGetValue("cmi.student_data.mastery_score"),10);
                if (!isNaN(nMasteryScore)) g_SCO_MasteryScore = nMasteryScore
   }
        if (isNaN(g_SCO_MasteryScore)) return "false";
        var stat = (nRaw >= g_SCO_MasteryScore? "passed" : "failed");
        if (SCOSetValue("cmi.core.lesson_status",stat) != "true") return
"false";
        return "true"
}

//Converte uma duração de milhe segundos para o formato 0000:00:00.00
function MillisecondsToCMIDuration(n) {
        var hms = "";
        var dtm = new Date();     dtm.setTime(n);
        var h = "000" + Math.floor(n / 3600000);
        var m = "0" + dtm.getMinutes();
        var s = "0" + dtm.getSeconds();
        var cs = "0" + Math.round(dtm.getMilliseconds() / 10);
        hms = h.substr(h.length-4)+":"+m.substr(m.length-2)+":";
        hms += s.substr(s.length-2)+"."+cs.substr(cs.length-2);
        return hms
}

// SCOReportSessionTime é chamdado automaticamente por este script,
// mas pode ser chamda a qualquer momento pelo SCO
function SCOReportSessionTime() {
        var dtm = new Date();
        var n = dtm.getTime() - g_dtmInitialized.getTime();
        return
SCOSetValue("cmi.core.session_time",MillisecondsToCMIDuration(n))
}

// Função que permite informar o status do SCO, seguindo a norma
function SCOSetStatusCompleted(){
        var stat = SCOGetValue("cmi.core.lesson_status");
        if (SCOGetValue("cmi.core.lesson_mode") != "browse"){
                if ((stat!="completed") && (stat != "passed") && (stat !=
"failed")){
                        return SCOSetValue("cmi.core.lesson status","completed")
                }
        } else return "false"
}
```

**Figura C.1:** Código de comunicação do SCO com o LMS "SCORMGenericLogic.js".