Integração de Objetos Instrucionais SCORM com um “Framework” Multiagentes *

Vinicius Heidrich Pinto**

Resumo

Este trabalho se propõe a integração do modelo de ambientes de execução (RTE) para “objetos instrucionais” que utilizam o padrão SCORM, 2004 (Sharable Content Object Reference Model) da ADL (Advanced Distributed Learning) com os chamados ITS (Intelligent Tutoring Systems), através da modelagem de agentes tradutores os quais medirão a comunicação entre os LMS (Learning Management System) que utilize sistema multiagentes, e os SCOs. Neste trabalho serão utilizados agentes no padrão FIPA (Foundation for Intelligent Physical Agents). Adaptando os agentes para a comunicação com a API (Application Program Interface) definida no “release” 1.3 do padrão SCORM.

Palavras-chave: scorm, tutores inteligentes, sistemas multiagentes, fipa.

Abstract

This paper propose the integration of run-time environment (RTE) of “instructional objects” that make uses of SCORM (Sharable Content Object Reference Model) standard created by ADL (Advanced Distributed Learning) with the ITS (Intelligent Tutoring Systems), by modeling translator agents (Learning Management System) who mediate the communication between a LMS, implemented on multiagent systems, and the SCOs. In this work we will adopt agents in the FIPA (Foundation for Intelligent Physical Agents) standard. Adapting the agents to communicate with API (Application Program Interface) defined by the SCORM 1.3 release.

Keywords: scorm, intelligent tutoring, multiagent systems, fipa.

Introdução:

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 para isso que eles tenham um computador com acesso a 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.

Em seu ultimo “release” a ADL, lança uma nova versão do padrão SCORM, a versão 1.3. No livro Overview (ADL, 2004^a) desta versão a ADL faz menção aos ITS.

Este trabalho se propõe a integração de modelos de ambientes de execução baseada em arquiteturas multiagentes, os chamados ITS, com “objetos instrucionais” que utilizam o padrão SCORM. O uso de padrões para o desenvolvimento de um sistema de entrega de conteúdo, utilizando agentes para implementação de um tradutor entre objetos educacionais, neste caso os SCOs, que utilizam o padrão SCORM, o tornaria totalmente reusável e interoperável. Aqui serão modelados agentes seguindo o padrão FIPA para comunicação com a API definida pelo SCORM (ADL, 2004^b).

O Ambiente de Execução SCORM:

O Livro RTE(ADL, 2004^b), descreve um mecanismo comum para o envio de objetos de conteúdo e para a comunicação entre estes objetos e os LMSs (fig 1), além de definir um modelo de dados em comum para armazenar as experiências do aluno em relação aos objetos de conteúdo.

Na fig. 1 o processo “Launch” define uma maneira em comum do LMS inicializar objetos de conteúdo baseados na WEB. O termo objetos de conteúdo é utilizado genericamente aqui para descrever cada pedaço de conteúdo que pode ser inicializado por um aluno. No SCORM, existem dois tipos de objetos de conteúdo: SCOs e Assets. Neste modelo os Assets são a representação de conteúdo mais básica definida pelo modelo que pode ser entregue a um cliente web, enquanto que, os SCOs que são conjuntos de um ou mais assets, representam o mais baixo nível de granularidade de um conteúdo que pode ser rastreado por um Sistema de Gerenciamento de Aprendizagem (LMS) , utilizando o ambiente de execução.

Fig. 1 – SCORM Run-Time Environment.

O processo “Launch” define procedimentos e responsabilidades para o estabelecimento de comunicação entre o conteúdo lançado e o LMS. O mecanismo de comunicação entre eles é padronizado com uma API em comum.

A API é um mecanismo de comunicação para informar o LMS do estado conceitual da comunicação entre o conteúdo e um LMS (e.g., inicializado, terminado ou em condição de erro), e é utilizada para retornar e armazenar informações (e.g., score, limite de tempo, etc.) entre o LMS e o SCO.

O “Data Model” é um padrão de modelo de dados utilizado para armazenar a informação retornada pelo SCO, como por exemplo o score de um questionário ou um

teste. De uma maneira simplificada, o modelo de dados define elementos de modelos de dados que o LMS e o SCO “deveriam conhecer”. O LMS precisa manter o estado do modelo de dados do SCO através das sessões do aluno, e o SCO precisa utilizar estes modelos de dados predefinidos para garantir o reuso através de múltiplas plataformas. A API definida pelo padrão SCORM(ADL, 2004b), será considerada para modelagem dos agentes envolvidos neste trabalho.

Uma proposta multiagente para o LMS:

De acordo com Silveira (Silveira, 2002), o estado da arte em ambientes inteligentes de ensino aponta para o uso de arquiteturas baseadas em sociedade de agentes. A utilização de sistemas multiagentes na criação de tutores inteligentes, pode resultar em um sistema mais versátil, rápido e de baixo custo.

Neste trabalho serão utilizados agentes baseados no padrão FIPA (Foundation for Intelligent Physical Agents), (FIPA) e para implementação será utilizado o “Framework”, FIPA-OS. O objetivo inicial é a modelagem de um agente interpretador que implemente os métodos definidos para comunicação entre os SCOs e o LMS, definidos no padrão SCORM(fig.2), este agente interpretará as mensagens da API

Fig. 2 – API, Instâncias da API e Implementação da

passando essas mensagens, através do protocolo para comunicação de agentes FIPA ACL (Agent Communication Language), para a comunidade de agentes responsável pela implementação do LMS.

A API definida pelo padrão SCORM deixa clara sua sintaxe mas omite detalhes de implementação. Como exemplo veremos a sintaxe para o método Initialize:

return_value = Initialize (Parâmetros).

Descrição: Esta função é usada para inicializar a sessão de comunicação. Ela permite ao LMS especificar parâmetros de inicialização.

Parâmetros : (“”) – string de caracteres vazia. Uma string vazia deve ser passada como parâmetro.

Valor de Retorno: A função pode retornar um ou dois valores. O valor de retorno deve ser representado como uma string de caracteres. Duas aspas (“”) não são parte da string de caracteres retornada, elas são usadas puramente para delinear os valores retornados.

“true” – A string “true” deve ser retornada se a inicialização da sessão de comunicação, como foi determinada pelo LMS, foi bem sucedida.

“false” – A string “false” deve ser retornada se a inicialização da sessão de comunicação, como determinada pelo LMS, foi mal sucedida. A Instacia da API deve setar um código de erro para o valor específico do erro encontrado. O SCO pode chamar a função GerLastError() para determinar o tipo de erro. Uma informação mais detalhada sobre o erro pode ser obtida do LMS através da função GetDiagnostic().

Quando chamado o método específico relativo à informação obtida do SCO o agente mandaria uma mensagem FIPA ao LMS para troca e armazenamento de informações no modelo de dados SCORM sobre os estados do SCO.

Conclusão:

A união de padrões com duas características tão importantes para o desenvolvimento de Ambientes de Ensino à Distância, se torna o ponto chave deste trabalho: a interoperabilidade dos Sistemas Multiagentes, mais particularmente do padrão FIPA, que é um padrão multiplataforma, unido ao reuso de objetos de aprendizagem oferecido pelo padrão SCORM.

Espera-se com este trabalho contribuir para encurtar a distância entre tutores inteligentes que utilizam um “framework” multiagentes e os “objetos instrucionais” que utilizem o padrão SCORM.

Bibliografia:

BRUSILOVSKY, P. [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, 1998. Disponível por: [<http://www2.sis.pitt.edu/~peterb/papers.html>](http://www2.sis.pitt.edu/~peterb/papers.html)
Acesso em 24 de junho de 2003.

Advanced Distributed Learning(ADL)a. **Sharable Content Object Reference Model (SCORM ®) 2004 Overview**. 2004. Disponível por: [<www.adlnet.org>](http://www.adlnet.org)
Acesso em 12 de fev. de 2004.

Advanced Distributed Learning(ADL)b. **Sharable Content Object Reference Model (SCORM ®) 2004 Run-Time Environment Version 1.3**. 2004. Disponível por: [<www.adlnet.org>](http://www.adlnet.org)
Acesso em 12 de fev. de 2004.

SILVEIRA, R. A. VICARI, R. M. **Developing Distributed Intelligent Learning Enviroment with JADE – Java Agents for Distance Education Framework**. Intelligent tutoring systems: 6th Internal conference (ITS 2002), Birratriz, France and San Sebastian, Spain, Junho 2002.

FIPA Specifications Repository. Disponível por:
<<http://www.fipa.org/specs/pesspecs.tar.gz>>
Acesso em 27, junho 2003