PIAGET - Uma Ferramenta de Suporte à Aprendizagem Colaborativa a Distância

Ismar F. Silveira
Universidade Cruzeiro do Sul, Departamento de Informática
São Paulo, Brasil, 08060-070
ismar.silveira@poli.usp.br

e

Maria Alice G. V. Ferreira
Universidade de São Paulo, Escola Politécnica
São Paulo, Brasil, 05508-900
maria.alice.ferreira@poli.usp.br

Abstract

Most of Computer-Supported Collaborative Distance Learning Systems usually fail on trying to provide interaction among teachers and apprentices, or even among them and the learning environment itself, making these kinds of interaction very limited, if faced to unlimited, wide range of interaction possibilities that exists in a real classroom. Based on this, the present paper introduces PIAGET, a supporting tool for interactive learning processes over a virtual reality environment. PIAGET is a Java-based environment built over an architecture based on distributed objects that uses RMI/IIOP/CORBA, also having a three-dimensional, interactive Java3D/VRML interface, thus allowing educational agents to interact among themselves and with the environment itself, in order to simulate some of the most common interactions that occur in a real learning environment, as well as provides interaction experiences that go beyond simple real-world simulations.

Keywords: Collaborative distance learning, interactive learning tools, immersive virtual worlds, virtual reality, distributed objects.

Resumo

A maioria dos sistemas de apoio à aprendizagem colaborativa a distância auxiliada por computador ainda falham no que diz respeito às formas de interação entre alunos, professores e o próprio ambiente que representa a sala de aula, tornando as simulações de aulas não-presenciais muito aquém do que se obtém numa sala de aula física. O presente artigo apresenta PIAGET, uma ferramenta de suporte à aprendizagem colaborativa via processos interacionistas baseados em um ambiente de realidade virtual. Trata-se de um sistema implementado em Java, usando uma interface em Java3D/VRML e modelado sobre uma arquitetura de objetos distribuídos sobre RMI/IIOP/CORBA, permitindo aos agentes envolvidos em processos educacionais interagir entre si e com o próprio ambiente, simulando as interações comuns em um ambiente real de aprendizagem, além de permitir experiências interativas que extrapolam a simulação do mundo real.

Palavras chaves: Aprendizagem colaborativa a distância, ferramentas interativas de aprendizado, mundos virtuais imersivos, realidade virtual, objetos distribuídos.

1 Introdução

A crescente evolução dos microprocessadores, que promoveu uma maior absorção mercadológica dos computadores pessoais, aliada ao advento das interfaces gráficas, da mesma forma que veio a popularizar o uso de computadores, possibilitou também o surgimento de sistemas educacionais e ferramentas de autoria com maiores recursos de interfaceamento tanto com o professor (elaborador de conteúdo) quanto com o aprendiz. Os anos 1990 foram testemunhas ainda da evolução da WWW (*World Wide Web*), que alavancou o uso comercial da Internet, vindo a acerretar uma demanda, ainda hoje existente, da migração de soluções *stand-alone* para soluções *web-based*. Os sistemas educacionais, como quaisquer outros, foram afetados por isso, resultando em um cenário no qual se tem, como conseqüência natural desse processo evolutivo, uma demanda por sistemas de apoio ao aprendizado que também sigam esse paradigma. Atualmente, encontra-se disponível uma gama variada de *softwares* de apoio ao aprendizado remoto [9], a grande maioria com interfaces desenvolvidas para a *web*, baseadas em padrões amplamente aceitos e difundidos, como HTML, Javascript e afins. Esse fator, apesar de ser determinante para sua boa aceitabilidade social [6], é ao mesmo tempo limitante no que diz respeito às possibilidades de interação entre os agentes do processo educacional –professores e alunos- e destes com o próprio ambiente de aprendizado, uma vez que que a única forma de interação síncrona entre os atores do processo educacional resume-se a salas de bate-papo textual ou com imagens (exceção seja feita aos que se utilizam de videoconferência). Dessa forma, em boa parte desses sistemas, espera-se que o aluno seja capaz de desenvolver, por si só, o currículo disponível a distância, o que contraria o princípio pedagógico defendido por Vygotsky, que ressalta a importância da interação coletiva na construção do conhecimento, por ele denominada construção interpsíquica, que precede a construção individual, intrapsíquica [12][13].

Sabe-se, sem dúvida, que as interações humanas ditas "virtuais" nunca substituirão por completo as interações "reais". Contudo, é possível chegar a uma simulação – e não substituição – da realidade, tão próxima a esta quanto possível, levando-se em conta as limitações às quais estão sujeitos todos os sistemas que envolvem simulação em tempo real. Desta forma, o presente trabalho propõe-se a apresentar PIAGET (*Platform-Independent, Adaptive Generic Environment for Teaching*) [3][5][7], uma ferramenta de suporte à aprendizagem colaborativa via processos interacionistas baseados em um ambiente de Realidade Virtual (RV).

Mais que uma ferramenta de suporte a CSCL (*Computer-Supported Collaborative Learning* - Aprendizagem Colaborativa Auxiliada por Computador) [2][11] , PIAGET apresenta-se como uma proposta genérica e abrangente para problemas comumente vivenciados em sistemas de aprendizado a distância. É proposta uma arquitetura de objetos distribuídos independente de plataforma, o que permite sua aplicação em ambientes heterogêneos que não necessariamente sigam o paradigma "*fat server, thin client*", podendo resultar em diversos benefícios no que se refere à escalabilidade, custo-benefício e performance [10]. Sobre tal arquitetura, é proposta uma interface gráfica tridimensional com recursos de realidade virtual, simulando um ambiente de sala de aula o mais próximo do real quanto possível, de forma a diminuir a "sensação de distância" dos usuários. Partindo do princípio defendido por Moore e Kearsley [4], para quem "a distância é um fenômeno pedagógico", procura-se, com tal interface, obter um ambiente colaborativo que forneça aos agentes do processo educacional meios de interação que de outra forma não poderiam ser obtidos, ou o seriam com maior dificuldade, maiores limitações e menor força metafórica na simulação de ambientes reais. Vale lembrar que, dado o nível de abstração em que a interface é proposta dentro do *framework* do projeto, seus usos podem ser estendidos de forma a obter uma interação multi-modal, incluindo dispositivos (capacetes, luvas e afins) e meios (voz, comunicação gestual, etc.) não-convencionais de I/O, ou ainda utilizando-a em sistemas de realidade mista/aumentada.

O presente trabalho é organizado como se segue: no item 2, é feita uma abordagem geral de PIAGET, melhor detalhado posteriormente nos sub-itens: 2.1, que explana sobre a elaboração da interface com recursos de RV, 2.2, que discute a arquitetura de objetos distribuídos sobre a qual PIAGET foi elaborado, 2.3, que detalha a distribuição de dados e 2.4 que apresenta a integração entre as três camadas citadas. O item 3 apreenta algumas conclusões, além de analisar os impactos pedagógicos do uso de PIAGET.

2 PIAGET

O Projeto PIAGET tem por objetivo estabelecer as bases de arquitetura e interface para um ambiente de RV de interação multi-usuário, de maneira a simular um subconjunto das interações mais comuns em uma sala de aula presencial, seja esta interação entre alunos, entre alunos e professores, ou entre estes e os elementos da sala de aula. Assim, obtém-se um ambiente de incentivo à aprendizagem colaborativa, facilitando o aprendizado através da interação social, através da qual o conhecimento é construído pelos agentes do processo educacional, a partir da própria interação entre agentes e dos agentes com o meio [13]. PIAGET foi elaborado segundo uma arquitetura de objetos distribuídos organizados em três camadas (*3-tier*), na qual procurou-se priorizar o desacoplamento entre camadas e mesmo entre componentes de uma mesma camada, de forma a garantir sua adaptabilidade e extensibilidade. A Figura 1 mostra a organização em camadas de PIAGET.

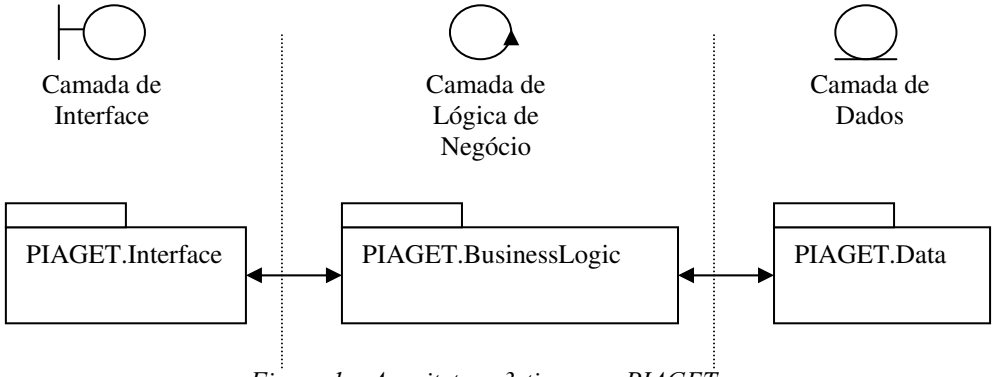

Figura 1 – Arquitetura 3-tier para PIAGET

Como pode ser inferido a partir da Figura 1, o uso de camadas adequadamente desacopladas, além de permitir uma flexibilidade maior no processo de desenvolvimento, também oferece a possibilidade de extensão ou mesmo substituição de qualquer uma das camadas, caso necessário. Por exemplo, a Camada de Interface pode ser modificada, atualizada e reestruturada sem afetar a Camada de Lógica de Negócio. Existe uma delegação de atividades e responsabilidades, pois cada camada exerce uma função independente da outra: caso alguma apresente problemas ou passe por um processo de manutenção ou substituição, a integridade dos demais objetos é mantida. Tem-se também uma maior flexibilidade dinâmica dos componentes, que se adaptam em tempo de execução à mudanças de servidores. Além disso, a Camada de Dados pode apresentar perfis distintos, envolvendo tipos diferentes de servidores e armazenamentos. Os próximos sub-itens irão detalhar cada uma das camadas citadas na Figura 1.

2.1 Camada de Interface

A interface baseia-se inteiramente nas APIs Java *Swing* (para a parte bidimensional) e Java3D (para a parte tridimensional), com suporte a elementos em VRML (*Virtual Reality Modeling Language*) [8]. A Figura 2 exibe o diagrama de classes UML (*Unified Modeling Language*) [1] representando as relações entre as classes da *package PIAGET.Interface* e as classes das APIs acima citadas.

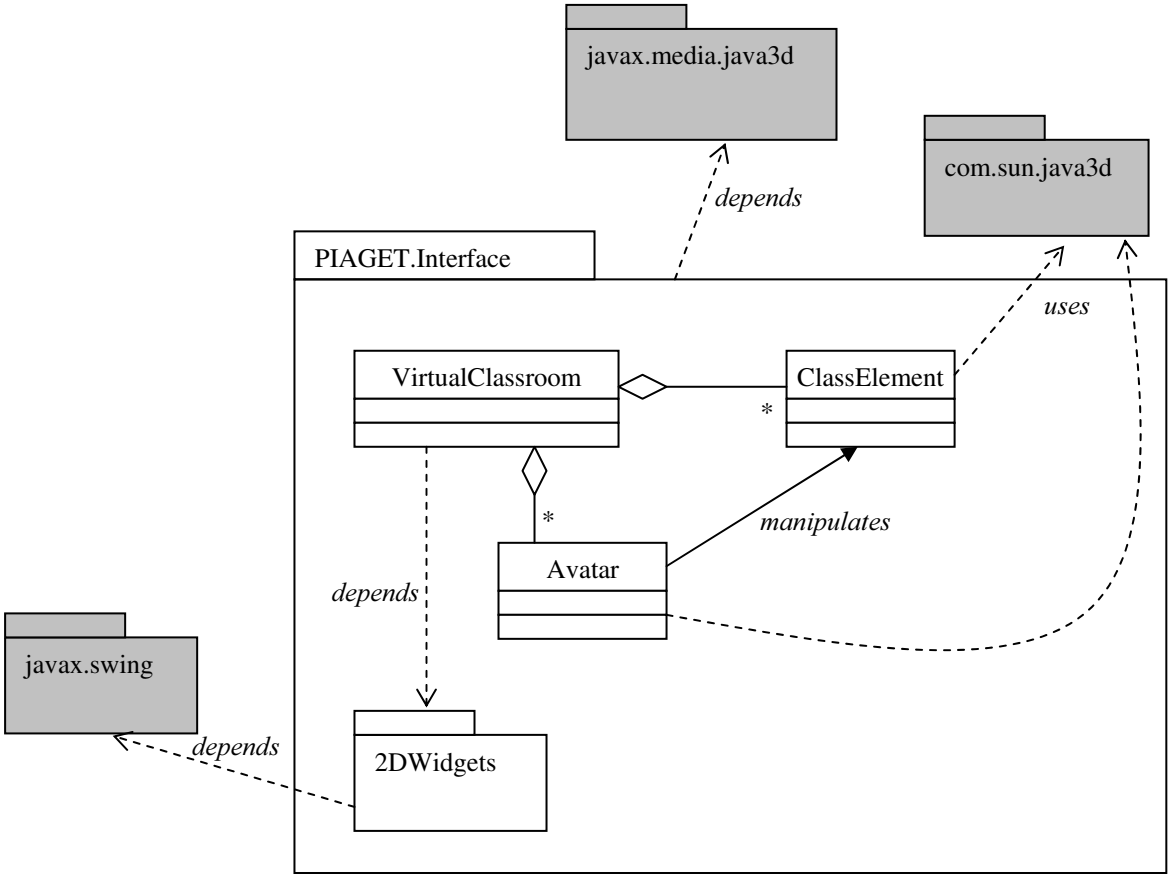

Figura 2 – Diagrama de classes e packages para a Camada de Interface de PIAGET

Como pode ser percebido a partir da Figura 2, a interface de PIAGET consiste em um ambiente de dimensão mista, pois contém um mundo virtual tridimensional representando o ambiente de interação imersivo (representado pela classe *VirtualClassroom*), além de controles bidimensionais, contidos na *package PIAGET.Interface.2DWidgets*, que são alternativas a elementos tridimensionais de controle da interface. Um ponto a se destacar no desenvolvimento da interface é a importância de que as ações possíveis de serem realizadas no mundo tridimensional possuam similares na interface tradicional, mantendo assim um alto nível de consistência conforme citado por Rocha e Baranauskas [6]. Partindo desse princípio, tem-se ressaltada a importância da integração dos componentes *Swing*, que são utilizados na parte bidimensional da interface, com os componentes Java3D, que irão proporcionar uma alternativa mais realística de interação através da tridimensionalidade. Os componentes Java *Swing* são componentes *lightweight*, ou seja, componentes implementados em Java e que não são nativos do sistema operacional. Em contrapartida, o componente *javax.media.java3d.Canvas3d* (componente onde são renderizados os objetos tridimensionais da API Java3D) é *heavyweight*, ou seja, é instanciado utilizando recursos do próprio sistema operacional, uma vez que tem ligações com a API Java AWT (*Abstract Windowing Toolkit*). Isso gera alguns problemas de integração, porém facilmente resolvidos [5].

Desta forma, utilizando-se tais componentes de interface, alunos e professores são metaforizados, no mundo tridimensional, através de seus avatares (instâncias da classe *Avatar*), que são representações gráficas dos agentes do processo interativo, não necessariamente guardando semelhanças físicas com o mundo real. Já no espaço bidimensional, os agentes são organizados em uma árvore (instância de *javax.swing.JTree*).

Além dos agentes, há ainda a representação dos elementos que podem estar presentes durante o processo interativo, presentes como objetos da classe *ClassElement*. Esses elementos podem ser representações tridimensionais de elementos comuns de uma sala de aula real, como cadeiras, cadernos e quadros-negros, ou objetos virtuais sem equivalentes no mundo real, como triângulos retângulos flutuantes sobre os alunos enquanto aprendem sobre o Teorema de Pitágoras, ou um circuito integrado tomando todo o espaço da sala, para permitir aos alunos "navegarem" pelos microcircuitos e aprender sobre Arquitetura de Computadores [5]. De forma a manter o princípio de consistência supracitado, os elementos têm sua representação bidimensional correspondente, conforme visto na Figura 3, a seguir, que mostra um *screenshot* de PIAGET.

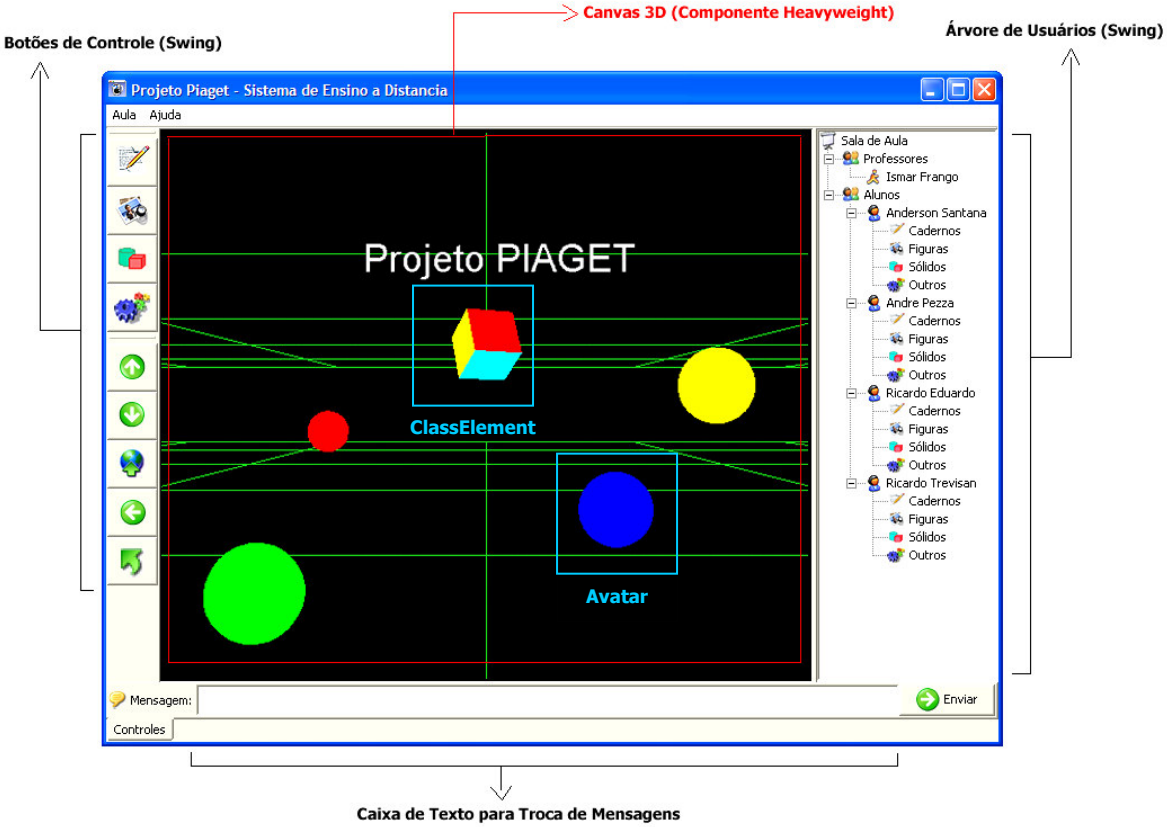

Figura 3 – Screenshot de PIAGET (aumentado a partir de Pezza et al.[5]).

A Figura 3 mostra uma possível interação entre agentes (quatro objetos de *Avatar*) e um elemento (instância de *ClassElement*). Todos os elementos são inseridos na sala de aula virtual pelos agentes, que deles detêm a propriedade, podendo compartilhá-los com todos os demais agentes. Podem haver diversos tipos de elementos presentes em uma sala de aula virtual, os quais serão detalhados na próxima seção.

2.2 Camada de Lógica de Negócio

Indo possivelmente na contramão da grande maioria dos sistemas de CSCL hoje disponíveis, os quais, em quase sua totalidade, são baseados em arquiteturas centralizadas, PIAGET foi elaborado sobre uma arquitetura de objetos distribuídos, utilizando como base a API Java RMI (*Remote Method Invocation*), podendo ser integrado a CORBA (*Common Object Request Broker Architecture)* através de IIOP (*Internet Inter-ORB Protocol*). Por se tratar de uma arquitetura de objetos distribuídos baseada em invocação remota de métodos, cada agente do processo educacional pode representar papel de cliente e servidor dos demais agentes, eliminando assim os altos custos que estariam associados à manutenção de um servidor centralizado com robustez suficiente para suportar toda a demanda dos clientes. Desta forma, o papel do servidor em PIAGET fica restrito a manter um controle dos agentes presentes no sistema, sendo que as ações dos agentes serão processadas localmente, sendo repassadas aos demais agentes via RMI. A Figura 4 exibe um exemplo do emprego de tal tal arquitetura.

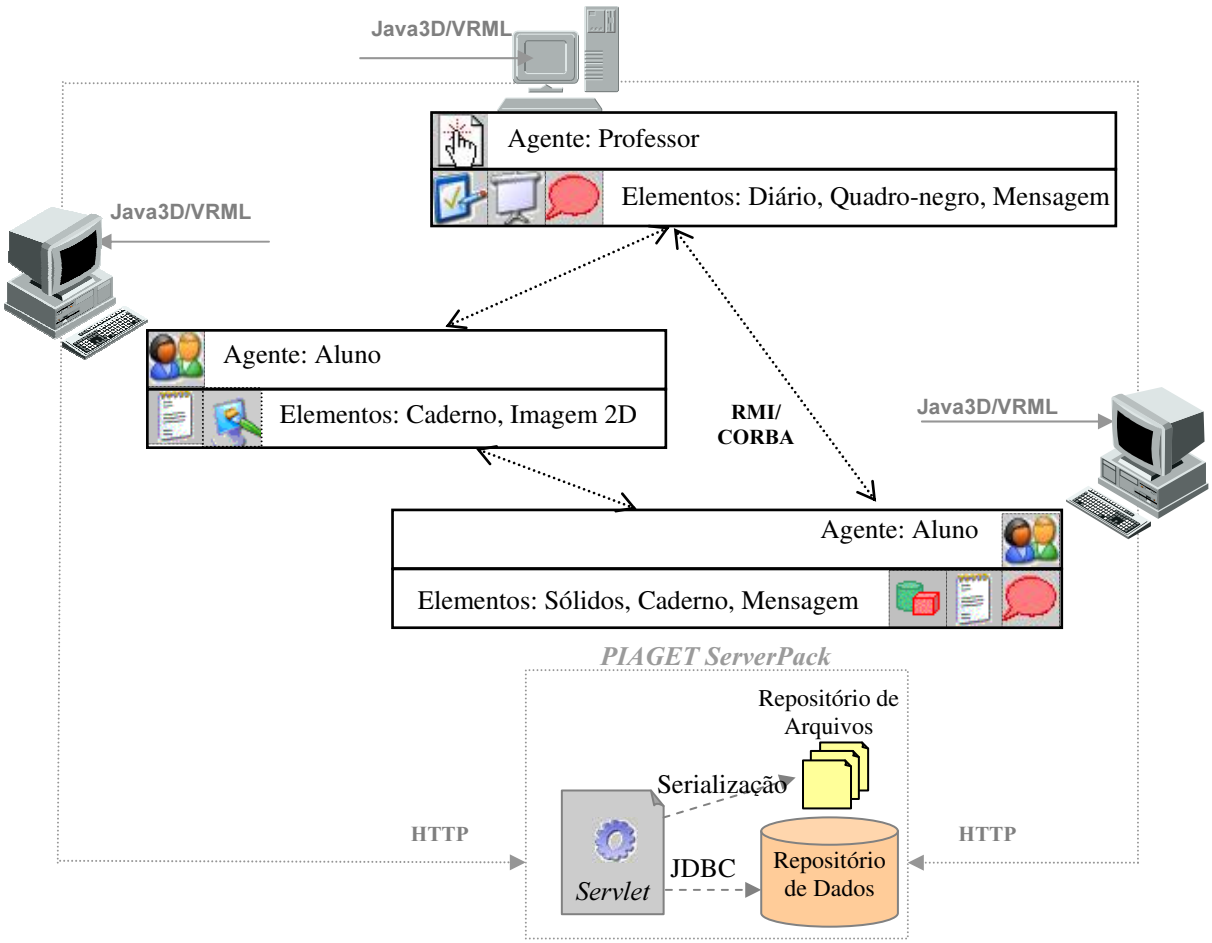

Figura 4 – Exemplo de interação sobre a arquitetura distribuída de PIAGET

A opção pela arquitetura distribuída em detrimento da centralizada deve-se a uma série de fatores, como a redução da necessidade de centralização de processamento e armazenamento, permitindo uma maior escalabilidade do sistema e a independência de um servidor específico para todas as tarefas, o que por sua vez também reduz os custos, já que um elevado número de salas - onde professores e alunos poderiam manipular elementos das mais variadas formas - pode levar à necessidade de servidores mais potentes, e por conseqüência, mais caros [10]. Além disso, há um sem-número de vantagens relativas à adoção do paradigma da orientação em objetos que devem ser levadas em conta, dentre as quais a possibilidade de reuso e extensão da API de PIAGET.

Todas as classes presentes na *package PIAGET.BusinessLogic* encontram-se representadas na Figura 5, que é um diagrama de classes UML [1], exibindo também as dependências em relação à API Java RMI, bem como a relação biunívoca que há com algumas classes da *package PIAGET.Interface*.

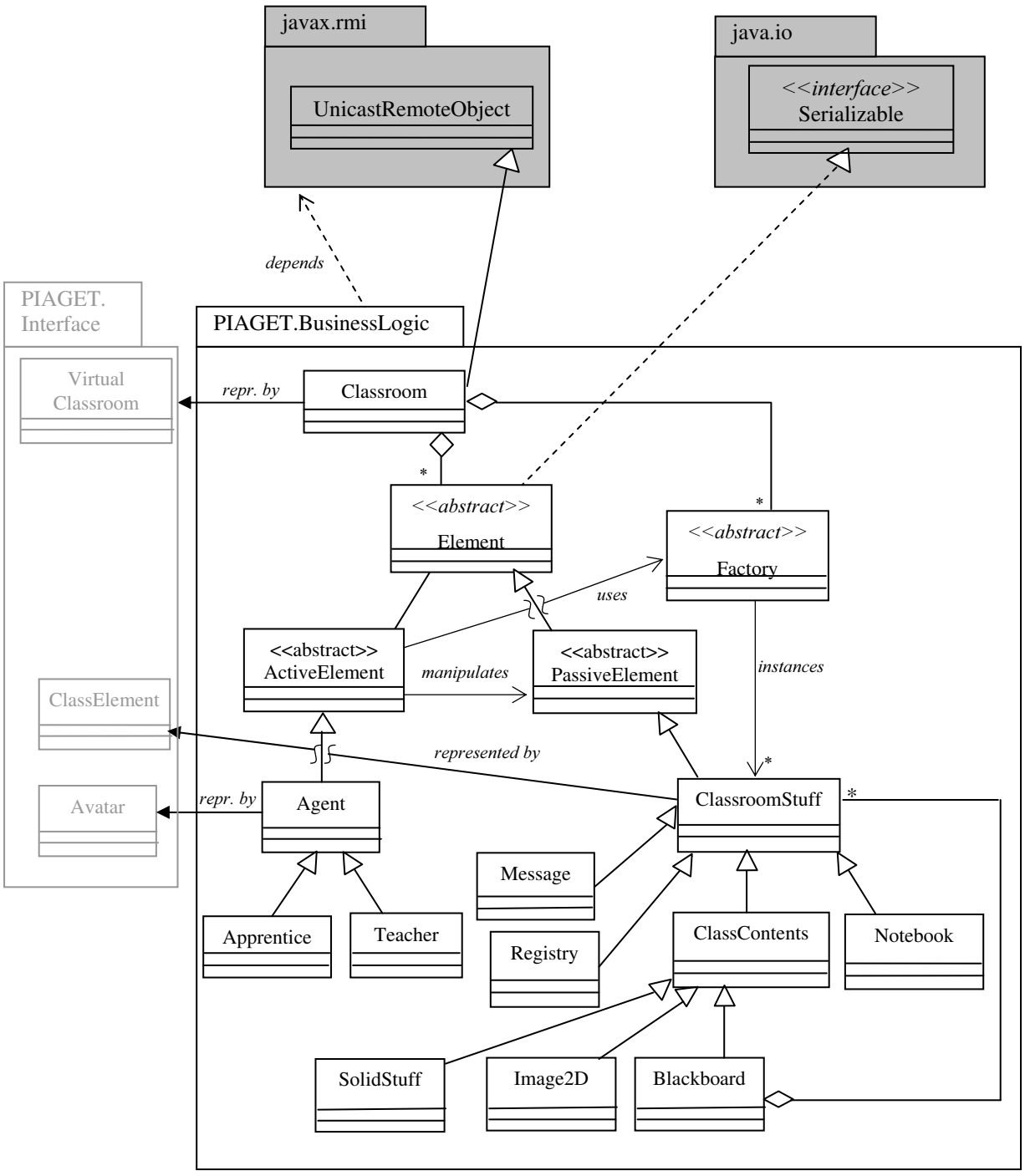

Figura 5 – Diagrama de classes para a Camada de Lógica de Negócio de PIAGET (aumentado a partir de Silveira e Ferreira [7]).

Como pode ser visto na Figura 5, os agentes, representados na Camada de Lógica de Negócio pelas classes derivadas de *Agent* (*Apprentice* e *Teacher*, representando o aluno e o professor, nesta ordem) e na camada de interface pela classe *Avatar*, são os responsáveis pela criação, manipulação e compartilhamento dos elementos da sala de aula virtual. Tais elementos, representados pela classe *ClassElement* na camada de interface, são especificados em *PIAGET.BusinessLogic* pelas classes-filhas de *ClassroomStuff*, descritas a seguir:

- *Message* diz respeito às mensagens trocadas entre os agentes. Na atual versão de PIAGET, tais mensagens são textuais, mas a API pode ser estendida para permitir a interação multi-modal, através de voz, por exemplo.
- *Registry* e *Notebook* são os blocos de anotações particulares do professor e do aluno, respectivamente. Objetos da classe *Notebook* podem ser compartilhados de modo *unicasting*, metaforizando o ato de "emprestar" cadernos, comum em salas de aula reais. O compartilhamento

de objetos *Notebook* dá-se através da serialização de objetos suportada pela *package java.io*.

- *ClassContents* representa os demais elementos presentes na sala de aula virtual, os quais são automaticamente compartilhados em *broadcasting*, também através do processo de chamada remota de métodos cujos parâmetros de entrada são instâncias serializadas desta classe, ou de suas filhas. A Figura 5 apresenta quatro classes derivadas de *ClassContents*, mas este número pode ser aumentado de acordo com a necssidade de extensão da API. As classes são:
    - *SolidStuff*, que diz respeito a qualquer elemento tridimensional que venha a ser inserido no ambiente durante o processo de aprendizagem. Em geral, é utilizada para representar elementos Java3D ou VRML.
    - *Image2D* representa imagens armazenadas em formatos padrões (JPEG, GIF, PNG) e inseridas no ambiente. Por se tratarem de elementos bidimensionais inseridos em um meio tridimensional, as mesmas são visualizadas como texturas para paralelepípedos retângulos "finos", que funcionam como "painéis" de suporte a tais figuras.
    - *Blackboard* é uma classe cujos objetos são instanciados somente pelo professor, servindo de contâiner para objetos de outras classes herdeiras de *ClassroomStuff*.

Como pode ser observado, a Camada de Lógica de Negócio de PIAGET baseia-se inteiramente em RMI, o que garante a distribuição de processamento. O próximo item analisa a distribuição de dados na última camada da arquitetura, a Camada de Dados.

2.3 Camada de Dados

A Camada de Dados é distribuída fisicamente em um Repositório de Arquivos, um Repositório de Dados, além dos armazenamentos locais, como pôde ser observado anteriormente na Figura 4. O Repositório de Dados pode, caso necessário, manter todas as ações significativas ocorridas durante as aulas, de modo que todo o conteúdo apresentado e produzido dentro da sala possa ser organizado para consulta e posterior reutilização. Este conteúdo, representado pelos objetos criados durante o processo de interação, pode ser fisicamente armazenado no Repositório de Arquivos que funciona como um repositório de objetos, serializados em tal servidor durante ou após o processo de interação. Contudo, esse mesmo conteúdo deve, idealmente, ser armazenado local e individualmente para cada agente, obtendo assim a distribuição dos dados. A Figura 6 demonstra as classes componentes da *package PIAGET.Data*, e suas relações com a Camada de Lógica de Negócio de PIAGET e com *packages* da API Java.

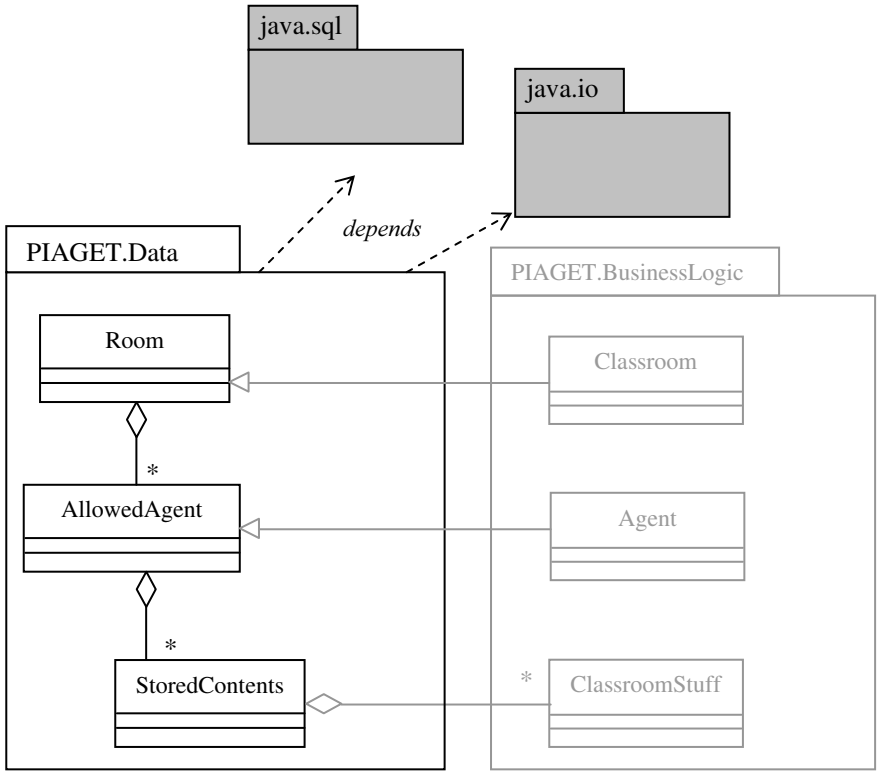

Figura 6 – Diagrama de classes para a Camada de Dados de PIAGET

Na Figura 6, a classe *StoredContents* representa um contâiner da camada de dados para instâncias de *PIAGET.BusinessLogic.ClassroomStuff*. Um objeto da classe *StoredContents* é organizado como uma lista ligada de adjacências, onde cada nó é um objeto de uma das classes derivadas de *ClassroomStuff*, podendo ser armazenado local ou remotamente, de acordo com a escolha do agente. A possibilidade de armazenamento local é de vital importância para a construção intrapsíquica do conhecimento desenvolvido durante o processo de interação social, que é a base da construção interpsíquica deste conhecimento [13], pois permite que cada agente do processo de aprendizagem seja capaz de estabelecer uma ou mais organizações mentais desse conhecimento, revisitando e reordenando o conteúdo discutido, além de proporcionar a distribuição de dados, que age em conjunto com a distribuição de processamento obtida na Camada de Lógica de Negócio. Contudo, caso seja necessário, é possível centralizar o armazenamento, conforme citado anteriormente, bastando manter os objetos da classe *StoredContents* adequadamente armazenados no Repositório de Arquivos. Um outro componente importante da Camada de Dados diz respeito ao gerenciamento das salas de aula virtuais, que consta do registro de alunos e professores que podem ter acesso a determinada sala, além do próprio registro das salas existentes. Esse controle é feito pelas classe *Room*, que representa cada uma das salas existentes, cada uma das quais com uma lista de instâncias de *AllowedAgents*, mantendo controle dos professores e alunos que podem interagir na sala.

2.4 Integração entre as camadas

Após analisadas as camadas da arquitetura, torna-se mister discutir a integração entre as mesmas. A Figura 7 a seguir demonstra a integração que existe entre as três camadas da arquitetura de PIAGET.

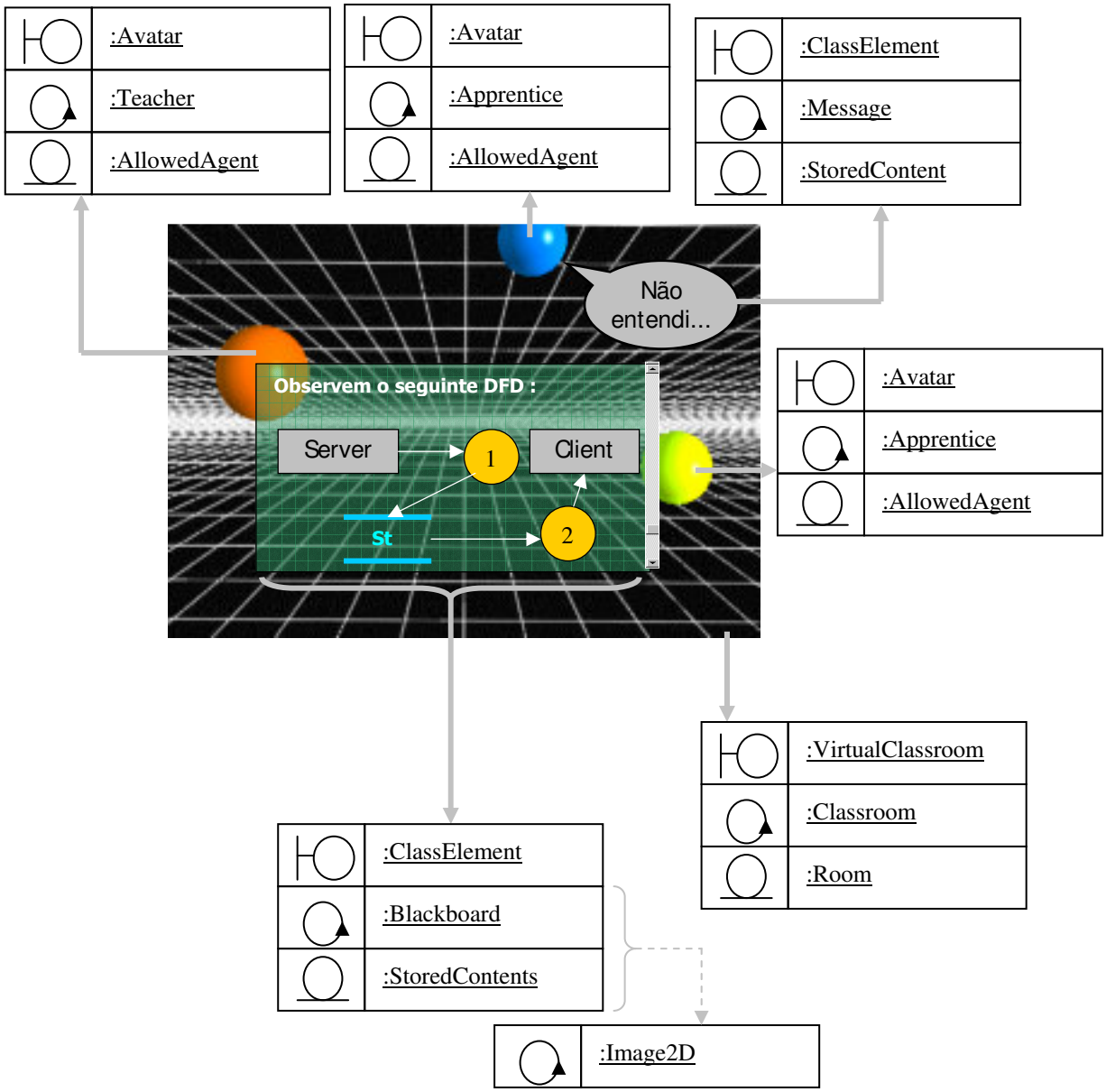

Figura 7 – Integração entre as camadas de PIAGET (aumentado a partir de Silveira e Ferreira [7]).

Pode-se ver, na Figura 7, um exemplo de interação entre três agentes, visualizados na Camada de Interface como objetos da classe *PIAGET.Interface.Avatar*, dois deles representados, na Camada de Lógica de Negócio, por instâncias de *PIAGET.BusinessLogic.Apprentice* e o outro, por um objeto da classe *PIAGET.BusinessLogic.Teacher*. Além disso, a todos eles equivalem instâncias correspondentes de *PIAGET.Data.AllowedAgent*. Cada agente possui, localmente, uma instância de *PIAGET.Interface.VirtualClassroom,* e seus equivalentes nas demais camadas, ou seja, *PIAGET.BusinessLogic.Classroom* e *PIAGET.Data.Room*, representando a sala de aula virtual. Além disso, o agente representado na Camada de Lógica de Negócio por *PIAGET.BusinessLogic.Teacher* compartilha uma instância de *PIAGET.BusinessLogic.Blackboard*, contendo um objeto de *PIAGET.BusinessLogic.Image2D,* ambos representados como instâncias de *PIAGET.Interface.ClassElement*, na Camada de Interface, e na Camada de Dados como objetos da classe *PIAGET.Data.StoredContents*. A Figura 8 exemplifica o que ocorre quando da instanciação de um objeto *PIAGET.BusinessLogic.Blackboard* pelo professor.

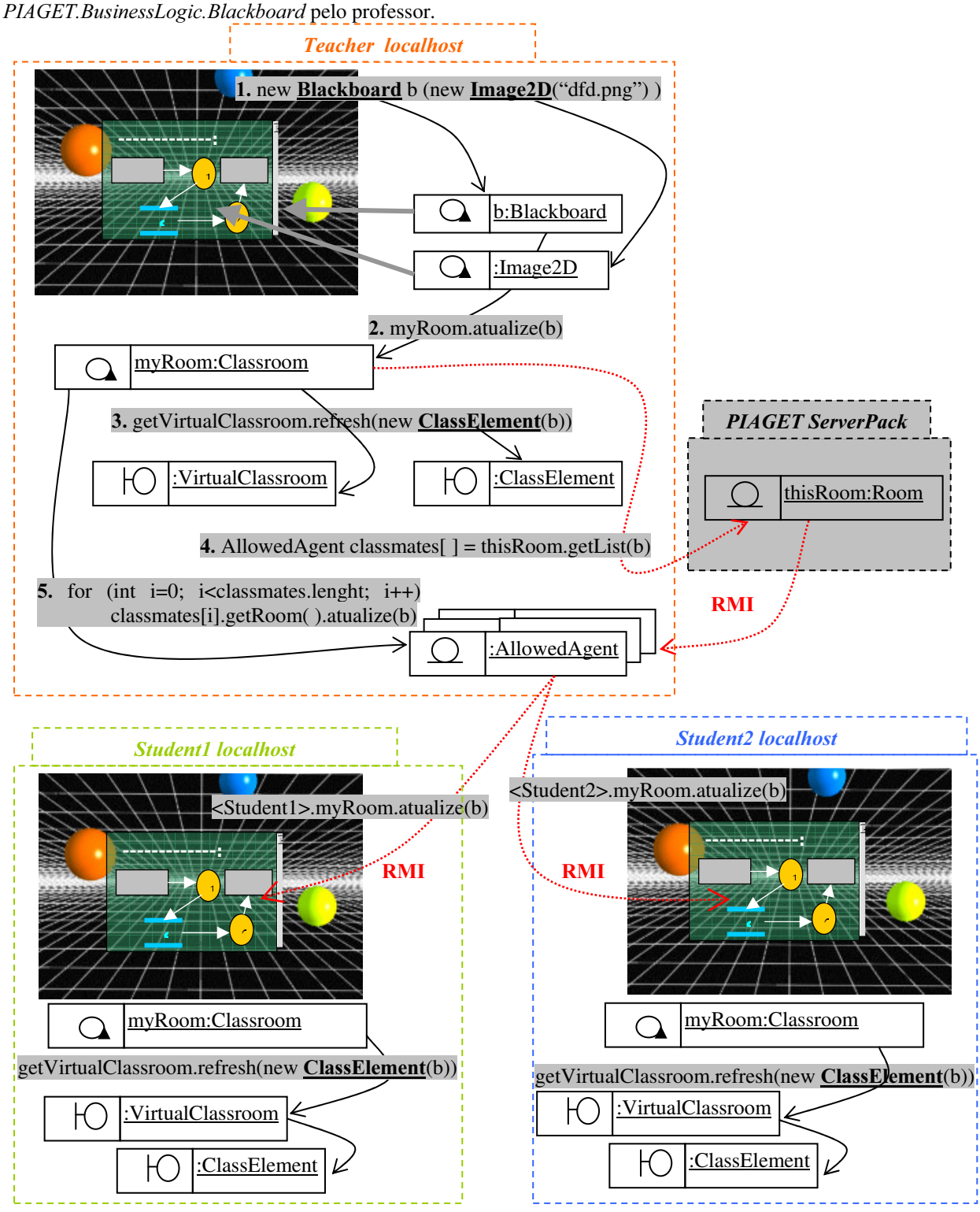

Figura 8: Invocação remota de métodos causada pela criação de objeto Blackboard pelo professor

Através da Figura 8 pode-se esclarecer como é realizado o processo colaborativo de PIAGET baseado na invocação remota de métodos. Sabe-se que toda ação que ocorre na sala de aula virtual é resultado direto da criação ou alteração de objetos da Camada de Interface. Assim, na Figura 8, quando é instanciado um objeto *PIAGET.BusinessLogic.Blackboard* (através de controles bidimensionais da *package PIAGET.Interface.2DWidgets* ou através de solicitação ao objeto *PIAGET.Interface.VirtualClassroom*), além da criação do objeto correspondente na Camada de Interface, faz-se um conjunto de invocações remotas de métodos a cada um dos objetos *PIAGET.BusinessLogic.Classroom* instanciados localmente nas máquinas dos agentes para que atualizem suas interfaces, ou seja, solicitem às instâncias locais de *PIAGET.Interface.VirtualClassroom* que criem um novo objeto *PIAGET.Interface.ClassElement*, inicializado com o objeto *PIAGET.BusinessLogic.Blackboard*. Este, por sua vez, é passado como parâmetro na invocação remota dos métodos via serialização.

Pode-se obter referências aos agentes de forma dinâmica no único ponto de centralização da arquitetura, o *PIAGET ServerPack*, que contém objetos da classe *PIAGET.Data.Room* instanciados em seu Repositório de Dados. Estes objetos, por sua vez, são compostos de instâncias de *PIAGET.Data.AllowedAgent*, contendo informações a respeito dos agentes envolvidos em cada sala de aula virtual, informações estas que possuem, entre outros dados, referências aos endereços dos agentes e formas de *binding* (registro) nos *RMIRegistries* individuais, que funcionam como *binders* no processo de invocação remota [10].

Cada agente, ao entrar numa sala, registra-se com um determinado nome em seu *RMIRegistry* individual, indicando a forma pela qual ele estará disponível para que seus métodos sejam invocados remotamente (através do serviço de *Naming* de Java RMI, também disponível como Serviço CORBA). Ao mesmo tempo, deve ser instanciado, no Repositório de Dados do *PIAGET ServerPack* um objeto de *PIAGET.Data.AllowedAgent* que armazene as informações pertinentes a tal agente.

Assim, quando há qualquer alteração no estado das salas virtuais de qualquer um dos agentes, considerando inclusive a entrada de novos agentes, ocorre, da parte do agente causador da ação, um processo de atualização de referências dos demais agentes e conseqüentes invocações remotas de métodos, implementando dessa forma a arquitetura colaborativa de suporte a processos colaborativos.

3 Conclusões

O presente trabalho apresentou PIAGET, que é um ambiente de suporte a processos de CSCL baseados em processos interacionistas. A contribuição de PIAGET para o desenvolvimento de ambientes de aprendizado encontra-se em diversos fatores, dentre os quais está a mudança de paradigma em relação ao próprio processo de autoria do conteúdo diático. Em PIAGET, temos um cenário no qual tanto o professor quanto o aluno são elementos participantes das fases de criação, organização e apresentação do conteúdo didático.

Um outro ponto a ser destacado em PIAGET é seu impacto pedagógico: por proporcionar um ambiente de suporte à aprendizagem através de processos interacionistas, possibilitando a inserção e manipulação de elementos de diversas mídias, aliado à possibilidade de realizar tal manipulação de maneira colaborativa, PIAGET fornece o ferramental necessário para a experiência sociointerativista defendida por Vygotsky [12][13]. Tanto os professores quanto os alunos realizam a construção do conhecimento através dos dois fundamentos interacionistas de Vygotsky: a interação com o meio e a interação com os demais. Assim, a possibilidade de compartilhamento e armazenamento de toda e qualquer informação presente no ambiente colaborativo permitem, respectivamente, a construção interpsíquica e intrapsíquica do conhecimento.

Há ainda que se citar a independência de plataforma, fator essencial em ambientes inerentemente heterogêneos, como quaisquer sistemas *web-based*. Como a arquitetura de PIAGET é completamente baseada em Java, isso garante um alto nível de independência de plataforma, já que poderá funcionar em qualquer sistema operacional que possua uma JVM (*Java Virtual Machine*). O acesso a dados é feito via JDBC (*Java Database Conectivity*) e também através de serialização. A comunicação entre agentes é realizada através de RMI ou RMI/IIOP/CORBA. Um fator importante a ser citado é que, no caso do uso exclusivo de RMI, uma vez que todos os agentes compartilham o mesmo *framework*, não há a necessidade de uso de interfaces compartilhadas, ou de realizar a serialização de *stubs* dinamicamente, tampouco de realizar o *marshalling* de parâmetros, o que é um ponto favorável em relação ao desempenho [10]. Há que se ressaltar também que as tecnologias empregadas na Camada de Interface, quais sejam, Java *Swing*, Java3D e VRML, são também independentes de plataforma.

A adaptabilidade é outro fator importante a se destacar. PIAGET foi projetado para ser um ambiente adaptativo, no sentido que a interface de usuário é adaptável ao assunto a ser abordado, tipo de sala desejado e perfil do usuário, de acordo com a evolução do curso. Quanto mais professores e alunos usarem PIAGET, mais o sistema irá se adaptar as suas necessidades mais usuais. Por utilizar-se de recursos de RV, o sistema não está restrito às limitações do mundo físico, podendo-se assim imaginar a criação de salas virtuais específicas para cada matéria, construídas de forma a maximizar a assimilação do conteúdo e despertar o interesse dos alunos. Como exemplo, imagine que para uma aula de geometria o professor poderia preparar uma sala de aula virtual composta de figuras geométricas,

enquanto o professor de geografia poderia preparar uma sala com o globo terrestre ao fundo e ainda o de língua portuguesa poderia utilizar uma espécie de biblioteca virtual onde os alunos pudessem se locomover e escolher a obra que fosse de seu interesse para leitura [5].

Por fim, a generalidade de uso de sua interface não exige do usuário nenhum tipo de habilidade além da manipulação básica de periféricos e sistema operacional. A simplicidade da interface é parte vital para PIAGET, já que este poderá ser utilizado por profissionais das mais variadas áreas e estes devem ser capazes de obter o máximo proveito das funcionalidades disponíveis, mesmo tendo somente os conceitos básicos de como utilizar o computador. Uma interface por demasiado complexa e não-intuitiva fatalmente inviabilizaria uma grande aceitação do projeto, daí a importância de utilizar metáforas do mundo real que façam com que os usuários possam se concentrar na criação do conteúdo de ensino e não na operação do sistema em si [6]. O projeto de uma interface de um sistema, mais que um trabalho destinado a *designers* e especialistas em ergonomia, é parte essencial do projeto de um software, visto que a interface com o usuário é quem garantirá o adequado uso do software. No caso específico de software de uso educacional, a interface desempenha um papel ainda maior, uma vez que é fator decisivo na qualidade da aprendizagem. De fato, não raro vêem-se casos onde uma grande porcentagem da carga cognitiva exigida no aprendizado de um certo conteúdo reside justamente na maneira como este é apresentado ao aprendiz, fazendo com que o meio de apresentação às vezes seja de importância tão grande quanto o conteúdo em si.

Referências Bibliográficas

[1] Booch,G., Rumbaugh, J. And Jacobson, I. The Unified Modeling Language User Guide. Reading: Addison-Wesley, 1999.

[2] Dillenburg, P. (Ed.) *Computer supported collaborative learning: cognitive and computational approaches*. Oxford: Pergamon, 1999.

[3] Gastaldo, D. L., Siqueira, S.R.C. e Silveira, I.F (Orientador). Estruturação e Aplicação de Objetos Distribuídos na Educação a Distância. *Trabalho de Graduação Interdisciplinar.* São Paulo: Universidade Presbiteriana Mackenzie, 2001.

[4] Moore, M.G. and Kearsley, G. *Distance Education: A System View*. Belmont: Wadsworth, 1996.

[5] Pezza, A.B., Silva, A.S.S., Trevisan, R.C., Oliveira, R.E.M. e Silveira, I.F. (Orientador). Uso de Java3D para a Criação de Mundos Virtuais Interativos Distribuídos. *Trabalho de Graduação Interdisciplinar*. São Paulo: Universidade Presbiteriana Mackenzie, 2002.

[6] Rocha, H. V. e Baranauskas, M. C. C. *Design e Avaliação de Interfaces humano-computador*. São Paulo: IME-USP, 2000.

[7] Silveira, I. F. and Ferreira, M. A. G. V. Distance Learning – The PIAGET Project. *ICECE 2000 – International Conference on Engineering and Computer Education (Proceedings)*. São Paulo, 2000.

[8] Sun Microsystems. Java 3D API Tutorial. Disponível na Internet. http://java.sun.com/products/java-media/3D/collateral/. Acesso em 15/05/2002.

[9] Tessarolo, M.R.M. Ambiente de Autoria de Cursos a Distância – AutorWeb. 2000. Dissertação (Mestrado). Campinas: Universidade Estadual de Campinas, 2000.

[10] Van Steen, M. And Tanenbaum, A.S. *Distributed Systems: Principles and Paradigms*. New Jersey: Prentice-Hall, 2002

[11] Vargas, I., Quintana, L. y Rudomín, I. Conjunto de Herramientas para la Construción de Mundos Virtuales Colaborativos Sobre Internet. Disponível na Internet. http://veracruz.lania.mx/~smcc/ENC97/ Graficacion/articulo.pdf. Acesso em 28/06/2001.

[12] Vérillon, P. Revisiting Piaget and Vygotsky: In Search of a Learning Model for Technology Education. *Journal of Technology Studies*, v.26, n. 1, (Winter/Spring 2000).

[13] Vygotsky, L.S. A formação social da mente: o desenvolvimento dos processos psicológicos superiores. São Paulo: Editora Martins Fontes, 1998.