Inteligencia Artificial. Revista Iberoamericana de Inteligencia Artificial
ISSN: 1137-3601
revista@aepia.org
Asociación Española para la Inteligencia Artificial
España

Martínez Carreras, Mª Antonia; Gómez Skarmeta, A. F.; Martínez Gracia, Eduardo; Mora González, Miguel
COLAB: a collaborative platform for simulations in virtual laboratories
Inteligencia Artificial. Revista Iberoamericana de Inteligencia Artificial, vol. 8, núm. 24, 2004, pp. 45-53
Asociación Española para la Inteligencia Artificial
Valencia, España

Available in: http://www.redalyc.org/articulo.oa?id=92502406

How to cite
Complete issue
More information about this article
Journal's homepage in redalyc.org

Scientific Information System
Network of Scientific Journals from Latin America, the Caribbean, Spain and Portugal
Non-profit academic project, developed under the open access initiative

COLAB: A Collaborative platform for simulations in virtual laboratories.

Mª Antonia Martínez Carreras, A. F. Gómez Skarmeta, Eduardo Martínez Gracia, Manuel Mora Gónzalez

Facultad de Informática
30071 Campus de Espinardo
Universidad de Murcia
{amart, skarmeta, edumart, mmora}@um.es

Resumen

The use of new technologies has an important influence in educational environments obtaining new ways of producing learning. In order to obtain an educational system which fits with the educational necessities, it is necessary to join pedagogical and technical efforts. More concretely our developments are focused on CSCL (Computer Supported Collaborative Learning) systems, in which the base of the knowledge building is the user collaboration. In this paper we are going to explain how we have designed and implemented an architecture with the main aim of supporting users collaboration using simulation tools. What is more this architecture has been designed in order to support asynchronous communication as well as the synchronous communication. Therefore this system has to incorporate ways to store persistent data and also introduce a synchronous system which enables the online communication between tools and users. In order to allow persistent data in the system, technologies like J2EE have been used in the design and implementation of this architecture. Dealing with the synchronous system it has been designed using notification technologies, especially using Elvin notification server as the core of the communication bus. In this paper it will be also described how the tools can interchange information subscribing and unsubscribing to special events delivered in the communication channel. Finally an example of collaboration inside this system will be showed, and it will be explained how the user can move in this environment as well as how users can coordinate the use of some resources.

Palabras clave: CSCL, J2EE, notification server

1. Introducción.

Los entornos colaborativos educacionales están tomando grandes repercusiones en nuestra sociedad, llevándose a cabo en la actualidad numerosos proyectos de investigación acerca de su construcción y su uso en comunidades educacionales. En esta investigación pedagogos e informáticos aunan sus conocimientos con el objetivo de obtener un entorno educacional que ofrezca las herramientas necesarias para que los alumnos puedan llevar a cabo la tarea colaborativa educativa.

Experiencias anteriores en entornos de educación usando las nuevas tecnologías[Leionen01] han reflejado que la incorporación de estos entornos educacionales puede fomentar el interés del alumno. No se trata de sustituir la enseñanza tradicional, sino de usar estos entornos como un complemento adicional enriqueciendo la enseñanza recibida. En la implementación de un sistema educacional hay que tener en cuenta que la información debe ser mostrada con una interfaz amigable y de fácil uso, porque de lo contrario puede inducir al usuario al total rechazo del entorno.

En la actualidad la Universidad de Murcia está participando como miembro activo en el proyecto europeo COLAB "Collaborative Laboratories for Europe" (IST-2000-25035) el cual tiene como objetivo el desarrollo de un entorno de educacional que permita la colaboración de usuarios a la hora de resolver y simular una serie de problemas en el

ámbito de la física. Este entorno se está desarrollando con el objetivo de ser un complemento a la enseñanza tradicional, fomentando el interés de los alumnos en el mundo de la física a la vez que hacer más sencilla y entretenida el aprendizaje de ésta, a la misma vez que permite reorganizar de una manera más formal las impresiones y trabajos realizados por los alumnos. Con esta plataforma se facilita el uso de experimentos pudiendo planificarlos con diferentes configuraciones para distintos grupos o para establecer diferentes niveles. Como cualquier entorno CSCL[Kligyte01,Stahl02], la base fundamental para la construcción del conocimiento la constituirá la colaboración entre usuarios mediante modelado y simulación de problemas en grupo.

Este entorno debe proporcionar una arquitectura capaz de soportar la colaboración de grupos de trabajo permitiendo que se guarde la información que se ha estado produciendo en dicha colaboración a la vez que soportar la comunicación de estos usuarios habilitando la puesta en común para realizar las tareas necesarias.

Dentro de una sesion colaborativa los usuarios pueden manejar herramientas visuales en las cuales se realiza una simulación y en esa sesión se podrá visualizar el progreso de dicha simulación así como consultar los datos que se generan mediante el uso de herramientas visuales, como pueden ser gráficas o tablas.

2. Arquitectura COLAB.

2.1. Especificación y requerimientos.

En el diseño e implementación de esta arquitectura es fundamental cumplir los siguientes requisitos:

a) La arquitectura debe soportar la comunicación asíncrona, en el sentido de que debe posibilitar el guardado de la información que se maneja en cada sesión a la vez que la recuperación de dicha información en interacciones posteriores.

b) La arquitectura debe ser desarrollada de tal forma que cuente con un bus de eventos a través del cual las herramientas puedan suscribirse a eventos de interés y mostrarlos a los usuarios que estén manejando dichas herramientas.

c) La información manejada debe ser tratada con sesiones independientes, de tal forma que lo que se realice en la sesión de un determinado grupo no sea visible a otros grupos, aún encontrándose en la misma localización de simulación.

d) La arquitectura debe ser modular, tal que permita de una forma sencilla la introducción de nuevas herramientas o la inclusión de herramientas existentes. Además dicha arquitectura deberá ser construida de tal manera que permita separar la parte de interfaz de la parte de programación.

e) Necesidad de una API clara y sencilla que permita a todos los miembros de la comunidad el manejo de esta arquitectura en el diseño e integración de herramientas sin necesidad de tener conocimiento de todas las tecnologías usadas en la elaboración de dicha plataforma.

Además de estos requerimientos se especificó desde la parte pedagógica del proyecto un modelo de gestión de datos que se basa en lo siguiente:

1. La introducción de usuarios en el entorno se hará a través de grupos
2. Los grupos tienen acceso a ciertos edificios (Buildings). La abstracción de edificio viene dada por la necesidad diferenciar los diferentes experimentos (phenomena) realizados por diferentes instituciones.
3. A su vez cada edificio estará formado por plantas, con el objetivo de tener un conjunto de herramientas y simulaciones asociadas, estableciendo en cada planta un nivel de dificultad diferente.
4. A su vez en cada planta existen 4 habitaciones con fines especificados. Dichas habitaciones reciben el nombre de Hall, Meet, Theo y Lab. La habitación Meet y Hall están destinadas a la colaboración de usuarios. Más concretamente Theo incluye las herramientas de modelado de teorías y Lab incluye las herramientas de simulación o laboratorios reales según se configure. Aunque cada habitación tiene un fin especificado, hay herramientas que pueden asignarse a toda la planta de tal forma que sean visibles en todas las habitaciones. Ejemplo de estas herramientas son el chat y el Process Coordinator que sirve para establecer una serie de objetivos en el desarrollo de las tareas.

2.2. Diseño de la arquitectura.

Con el objetivo de obtener una arquitectura que nos permita almacenar la información que se maneje en el entorno, tanto de configuraciones iniciales como de la información a recuperar por los usuarios de acuerdo con la información manejada en una determinada sesión, hemos basado la parte asíncrona de este sistema en tecnologías J2EE (Java 2 Enterprise Edition)[J2EE] usando el servidor JBOSS[JBOSS] en nuestros desarrollos. Las razones para usar dicha tecnologías son las siguientes:

1. El uso de su tecnología, en especial los Entity Beans, facilita el manejo de la base de datos.
2. Su modelo basado en componentes facilita separar la parte de interfaz de la parte de programación a la vez que permite la incorporación de nuevas aplicaciones de una forma sencilla con pocos cambios en el sistema.
3. Facilita la portabilidad de código, de tal forma que puede ser ejecutado en cualquier sistema.
4. Tecnología basada en un conjunto de estándares bien definidos que facilitan la configuración del sistema

Mediante el uso de la tecnología J2EE podemos abordar toda la parte asíncrona, gestión de usuarios, grupos, edificios, herramientas, programas de simulación. Sin embargo esta tecnología por sí sola no es suficiente para abordar temas relacionados con la comunicación síncrona entre herramientas y usuarios. Con lo cual debemos complementar su uso con tecnologías de servicios síncronos. En esta elección nos encontramos con las siguientes posibilidades: JMS (Java Message Service)[JMS], JSDT (Java Shared Data Toolkit)[JSDT] y DSTC Elvin[ELVIN]. Debido a su alto rendimiento, su eficiencia y a los buenos resultados obtenidos en previas investigaciones[Garcia02,Martínez02], hemos elegido el servidor de notificaciones DSTC Elvin como la piedra angular para la comunicación síncrona en el sistema, encargándose de repartir los eventos según las suscripciones de las herramientas, grupos y localización de éstos. De esta forma, especificando edificio, planta y grupo que se maneja podremos establecer diferentes sesiones en las cuales la información que se distribuya no interfiera entre grupos ni usuarios en la misma localización.

El esquema de la arquitectura COLAB es el que se muestra en la Figura 1, en la cual podemos apreciar los diferentes elementos que forman parte de ella, como son el servidor JBOSS (servidor J2EE), el bus de eventos elvin, los clientes y los simuladores de experimentos.

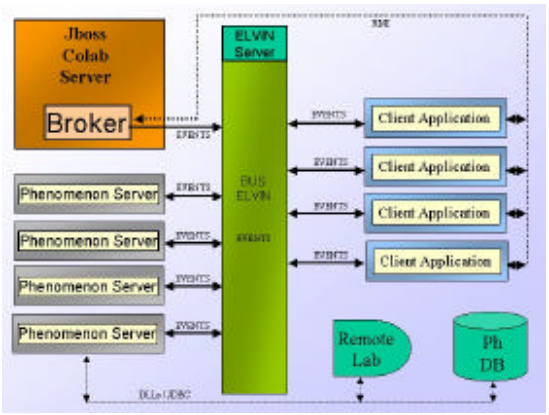

Figura 1. Arquitectura general del sistema.

En el servidor de aplicaciones JBOSS se mantendrá toda la estructura del entorno Colab, incluyendo como aplicaciones todos las herramientas de visualización de resultados (Visual Tools) para seguir una simulación así como todos los simuladores (Phenomenon). Además dentro del servidor son gestionados los datos de usuarios, grupos, edificios, plantas dentro de los edificios y habitaciones. En esta arquitectura los usuarios podrán hacer simulaciones basándose en programas software (PhenomenonServer) y otra se basarán laboratorios remotos (RemoteLab).

En cuanto al diseño de una API clara y sencilla se realizará mediante la inclusión de una serie de clases que ocultarán información tecnológica al desarrollador de aplicaciones, así como una serie de APIs patrón a seguir por las herramientas desarrolladas, las cuales serán expuestas a continuación.

Todas las herramientas visuales así como los servidores para las simulaciones parten de una configuración inicial la cual está especificada en XML. Dicha configuración indica las variables que se pueden manejar en la simulación así como los valores de éstas. Además de una configuración inicial, se puede asignar un fichero XML cuando se asocia cualquier simulador a las plantas de un edificio. De esta forma este fichero descriptor indicará las variables que se pueden manejar a nivel

de habitaciones o plantas. Como cada sesión colaborativa de un grupo producirá una serie de cambios en las herramientas de esa sesión, se guardará la información de cada sesión (ficheros xml) identificada por el edificio, planta y/o habitación y grupo en el que suceden los cambios.

Con el principal fin de facilitar el acceso a cada uno de los EntityBeans en el servidor JBOSS se ha implementado un SessionBean denominado Broker el cual permite, mediante una API bien definida, crear las instancias de cada EntityBeans, modificarlas así como destruirlas encapsulando su uso a los usuarios finales. El esquema de este broker es mostrado en la Figura 2. Como podemos apreciar cada cliente establece una sesión con el Broker y puede manejar los datos del servidor los clientes establecerán comunicación con diferentes instancias del Broker, si bien las operaciones de lectura podrán ser ejecutadas paralelamente, no ocurrirá lo mismo con las de escritura. Además los problemas de concurrencia son solucionados por el servidor J2EE liberando al programador de la implementación de un control de gestión de recursos.

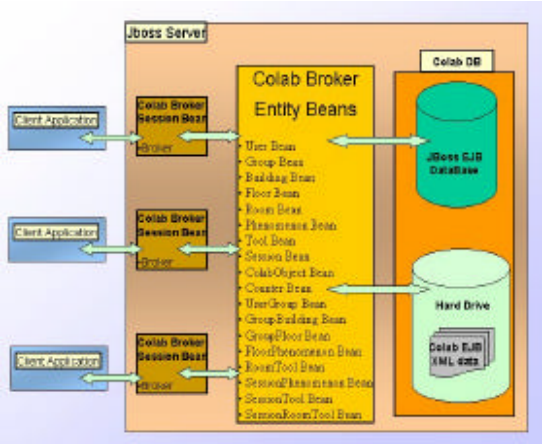

Figura 2. Esquema del Broker, indicando métodos para acceder a los objetos de la B.D.

A su vez el bus Elvin permitirá el intercambio de información entre las herramientas, por ejemplo recogida de datos de los simuladores que se están ejecutando, suscripciones a ciertas variables para mostrarlas en graficas o en tablas, y comunicación entre usuarios usando herramientas como chat o whiteboard. Con el principal objetivo de simplificar el acceso de las herramientas al bus, así como de encapsular la gestión de las conexiones Elvin y al broker se ha diseñado la clase Environment. Además este Environment será el responsable de recibir y comunicar la información deseada a la herramienta en cuestión. En la Figura 3 se muestra el esquema de las funciones del environment el cual será pasado a las herramientas visuales así como a los simuladores en el momento de su ejecución para que así tengan conciencia de ámbito dónde se encuentran.

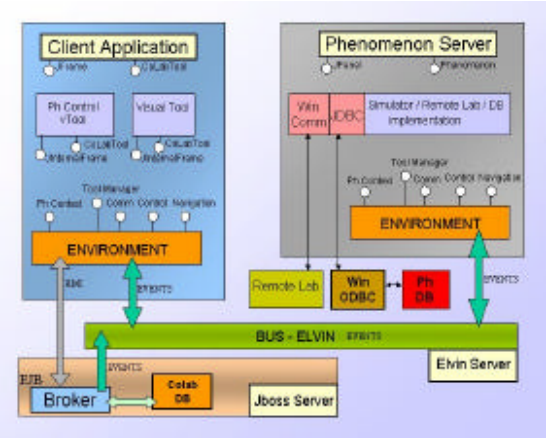

Figura 3. Funciones a realizar en el environment.

Además para conseguir que todas las herramientas visuales del sistema sigan una API bien definida se ha creado la clase ColabTool la cual implementan todas las herramientas visuales. Dicha clase contendrá un método run(Environment e, String config) el cual facilitará la ejecución de las herramientas cada vez que se acceda a la habitación o planta dónde fueron asignados, así como pasarle a cada una de ellas el environment a manejar y la configuración XML.

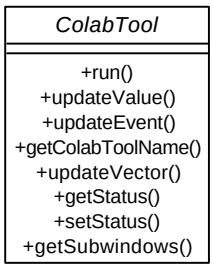

Figura 4. Esquema UML del interfaz ColabTool

De igual forma, se incluyen métodos que permitan a la herramienta recuperar y/o salvar el estado de la sesión en la que se encuentra así como actualizar los valores a mostrar con los métodos update.

De forma similar, cada simulador implementa la clase Phenomenon que contiene una serie de métodos básicos los cuales facilitan el uso de los simuladores de una manera homogénea.

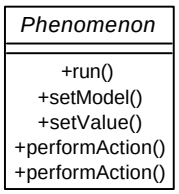

Figura 5. Esquema UML del interfaz Phenomenon.

2.3. Aspectos de la comunicación síncrona.

En todo sistema colaborativo en el que se permiten colaboraciones en el mismo instante de tiempo, es decir colaboración síncrona, es necesario tener en cuenta principalmente los siguientes aspectos[Begole98]:

1. Los usuarios dentro de una misma sesión deben ver la misma información (WorkSpace Awareness)
2. Los recursos en el sistema deben ser accedidos de una manera ordenada lo que implica la necesidad de establecerse políticas de concurrencia.
3. La entrada de usuarios a una misma sesión puede ocurrir en instantes diferentes (Late Comming).
4. Saber que usuarios están conectados en un determinado momento en la misma sesión (Presence Awareness)
5. Control de concurrencia en la utilización y/o creación de algunos recursos.

Para resolver el primer aspecto todas las herramientas están suscritas a una serie de eventos dentro de una sesión y dependiendo del tipo de herramienta del que se trate, de tal forma que todos los usuarios de una misma sesión pueden visualizar la misma información.

Con respecto al segundo punto se ha establecido la figura de leader dentro de cada habitación de tal forma que el primer usuario que entre a una habitación será el encargado de dirigir la forma en que van a ser accedidas y/o manejadas las herramientas. En cualquier punto de la colaboración algún otro usuario puede pedir el liderazgo. Si bien este diseño ha sido requerido por la parte pedagógica del proyecto, se podría integrar en el sistema alguna herramienta más amplia para resolver el cambio de liderazgo.

Para tratar la llegada tarde de usuarios al sistema cuando un usuario se conecte dentro de una sesión activa se le pasará la información actual de dicha sesión (por el leader) o en caso de no haber ningún leader en dicha habitación la recuperará de lo último que se haya guardado en dicha sesión.

Debido a la carencia de Elvin de poder controlar los usuarios que están activos en un cierto momento en el sistema, se ha implementado un mecanismo de sincronización mediante el cual cada cierto tiempo se comprueba si un usuario está activo en el sistema, detectando de esta manera las posibles perdidas de conexión por parte de los clientes. Esta tarea será realizada desde el broker que comprobará mediante un método isAlive, si cada usuario conectado en el sistema está "vivo".

Aunque el uso de EntityBeans facilita el control de concurrencia a la hora de escribir en base de datos, no es suficiente cuando se necesita ejecutar una serie de pasos de manera coordinada. Por lo cual es necesario crear un mecanismo de tal forma que cuando varios usuarios quieran acceder a la vez al código de una sección crítica, sólo se permita un usuario en dicho bloque. Para tratar este aspecto nos hemos basado en la filosofía de token, creando un EntityBean en la base de datos denominado Token (vease Figura 6), que será creado según el edificio, planta y grupo en el que se encuentre el usuario, tal que una sección critica sólo se ejecutará por el primero que recoja dicho token.

Figura 6. Estructura de token para la gestión de bloqueos.

2.4. Gestión dinámica de los simuladores.

Como se ha comentado anteriormente el fin de esta arquitectura dentro del proyecto COLAB es de habilitar las simulaciones dentro del ámbito de la física. Por requerimientos del proyecto las simulaciones que pueden llevarse a cabo pueden implicar la ejecución de un programa, entidades software, o el acceso a laboratorios remotos, entidades hardware.

En el caso de laboratorios remotos (laboratorios físicos) se establecerá que sólo un grupo de usuarios podrá manejar en un momento dado dicha entidad hardware. Para ellos se está desarrollando una gestión de reservas para indicar en que fecha y hora puede ser accedido dicho laboratorio por parte de un grupo.

En cuanto a los simuladores software se ha establecido que diferentes grupos puedan acceder a los simuladores manejando diferente información puesto que cada grupo dentro de una misma planta establecerá una sesión diferente de colaboración. Por lo tanto se ha creado un gestor dinámico de simuladores tal que cuando se detecte que un grupo entra a una planta en el que hay una serie de simuladores asociados dicho gestor se encargue de arrancar los servidores de la simulación para esa sesión. Igualmente cuando se detecta que una sesión no está activa el propio gestor será el encargado de parar los servidores asociados a dicha sesión.

En la Figura 7 se muestra esta herramienta. Dicha herramienta posee opciones para parar cada uno de los servidores, al igual que hacer previos avisos a los usuarios que estén usando estas simulaciones mediante envío de mensajes.

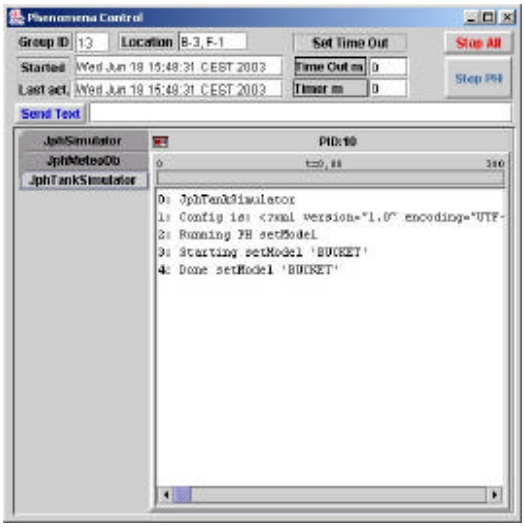

Figura 7. Phenonenon Control para gestión dinámica de servidores de phenomenon.

2.5. Gestión de los Recursos en la Base de Datos.

Para la introducción de datos dentro de la base de datos del entorno se ha creado una aplicación que gestiona dichos datos a través del broker y permite la creación, modificación, asociación y borrado de éstos.

Esta herramienta permite la entrada de datos del sistema y la asignación de ficheros XML a cada una de las acciones que lo requieren. A su vez permite hacer copias y recuperaciones de toda la información almacenada en un determinado servidor.

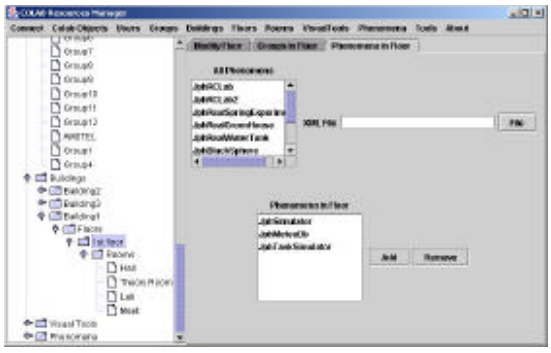

Figura 8. Aplicación Resources Manager para la gestión de datos dentro del entorno COLAB.

3. Cliente Colab.

Con el fin de garantizar que todos los usuarios trabajan con la misma versión de código, y puesto que el diseño en applets restringe muchas de las operaciones a realizar en COLAB, el diseño del cliente ha sido realizado apoyándonos en el uso la tecnología Java Web Start. De esta manera un cliente se conectara a una página y mediante un clic en el enlace dispuesto, Java Web Start será el encargado de analizar si los jars han sido modificados y en caso afirmativo proceder a la descarga. Dicho esquema es presentado en la figura 9.

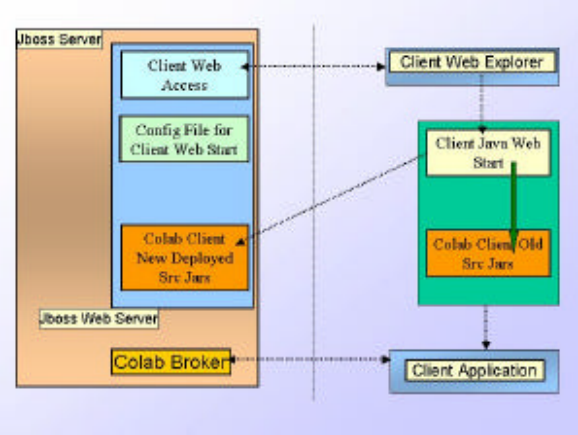

Figura 9. Esquema de carga de la aplicación COLAB usando Java Web Start.

Una vez cargados los jars de la aplicación se procede a la entrada de datos del usuario. En la siguiente sección veremos un ejemplo de uso del entorno COLAB.

4. Ejemplo de colaboración en COLAB.

Previamente a la ejecución de un cliente se debe ejecutar el servidor Elvin, Jboss y el Phenomena Control.

Una vez que el alumno ha realizado los pasos expuestos en la sección 3 se abre la aplicación cliente en la cual, primeramente, se debe introducir login y password del usuario. Estos son validados en el servidor según los datos almacenados en la base de datos de JBoss. En la figura 10 se muestra la primera pantalla de introducción de datos de usuario.

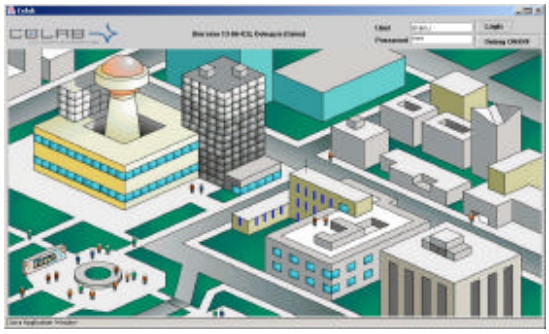

Figura 10. Pantalla de introducción de login y password al entorno COLAB.

Una vez validado usuario, se mostrará una panel con los grupos a los que está asociado dicho usuario. Cuando el usuario haya seleccionado el grupo el usuario seleccionará edificio y planta al que quiera conectarse. Después de estos pasos el usuario entrará automáticamente a la habitación Hall del Floor seleccionando.

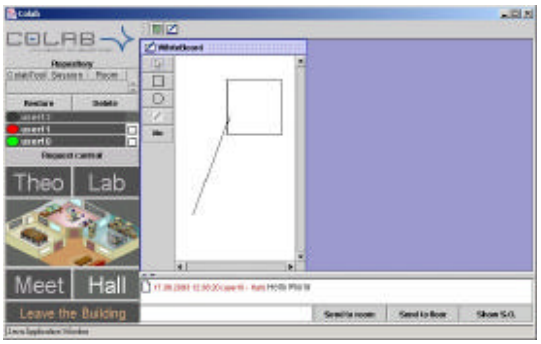

Figura 11. Habitación Hall en la que el usuario puede visualizar el chat y el whiteboard.

En la Figura 11 se muestra el aspecto a la entrada del usuario a la planta. En el margen izquierdo aparecerán las habitaciones por las cuales se puede ir moviendo el usuario: Hall, Meet, Theo y Lab. El usuario sabrá en que habitación se encuentra porque está mantendrá su nombre más iluminado y podrá ir moviéndose haciendo clic en cualquier habitación a la que desee trasladarse.

También en este margen aparecen los usuarios que forman parte del grupo, así como de una forma más iluminada y con un circulo coloreado aquellos que están conectados en ese mismo instante. Con el color verde se indica quien es el leader, con lo cual

será el único con control absoluto sobre algunas herramientas de la habitación.

En la parte superior se encuentran el conjunto de herramientas disponibles en cada habitación, posicionando el ratón por encima se puede ver una breve descripción de éstas.

En la parte inferior aparece el chat el cual puede enviar mensajes a todos los usuarios del grupo en la misma habitación (por defecto) o a nivel de planta.

Como hemos comentado al principio de este artículo cada habitación tiene un fin específico. Más concretamente las habitaciones de Theo y Lab son usadas para la creación de modelos y para la ejecución de una simulación respectivamente.

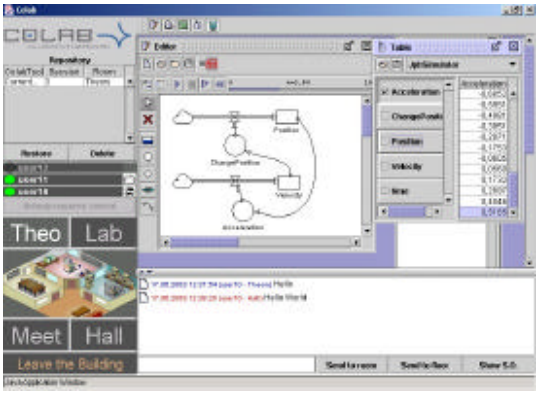

Figura 12. Habitación Theo en la que se muestra la herramienta JEditor de modelos ejecutándose.

En la Figura 12 podemos observar que el usuario user10 se encuentra en la habitación Theo mientras que el usuario11 se encuentra en la habitación Hall. Esto es simbolizado por el icono que se encuentra a la derecha del nombre de usuario. Como ambos no se entorpecen en las tareas que están llevando a cabo puesto que se encuentran en habitaciones diferentes, ambos son leader en sus respectivas habitaciones.

Herramientas como el JEditor permiten al usuario la elaboración de un modelo y simularlo para ver si es correcto o no, pudiendo reflejarse los valores que toman las variables bien a través de las gráficas o las tablas.

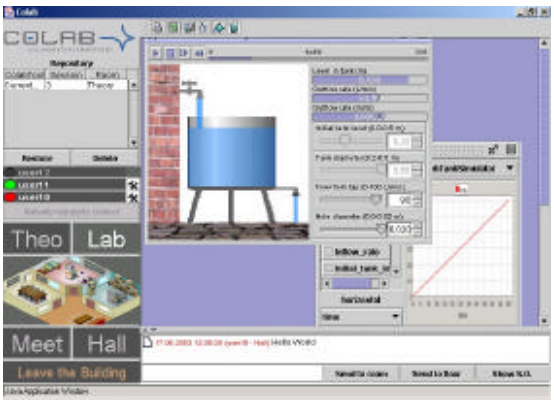

Figura 13. Habitación Lab en la que se muestra la herramienta JvtTankSimulator.

Así pues en la habitación Lab, Figura 13, se puede ver como el alumno ha iniciado una simulación mediante el uso de una herramienta visual la cual va recogiendo los datos de un simulador específico y va mostrando la evolución de está en el tiempo. De igual forma se puede seguir la evolución de esta simulación bien mediante el uso de gráficas bien mediante el uso de tablas haciendo exploraciones más exhaustivas sobre las variables que pueden verse influidas por este proceso.

Una vez que el último alumno abandone la planta se procederá al guardado del estado actual de todas las herramientas, de tal forma que cuando cualquier miembro de ese grupo acceda de nuevo a esa misma planta encontrará la información tal cual se dejó en la ultima sesión, pudiendo continuar de esta forma con el trabajo ya realizado.

5. Conclusiones y vías futuras.

En el desarrollo de este proyecto hemos construido una arquitectura basada en servidores de aplicaciones en la cual se pueden integrar herramientas en el sistema de una forma fácil. Además dichas herramientas se pueden asignar a diferentes plantas con ficheros de configuración diferentes, dependiendo de lo que se quiera mostrar en esa planta.

La configuración del sistema está basada en estándares ampliamente usados como es XML, haciendo fácil la configuración de cada una de las aplicaciones mediante el uso de ficheros siguiendo estructuras XML manejables por las herramientas que van a tratar con ellos.

A su vez toda esta plataforma y conjunto de aplicaciones es mostrado al usuario final como un entorno de fácil manejo, visualmente fácil e intuitivo, no necesitando tener amplios conocimientos de cómo usar cada una de las herramientas de las que dispone este sistema.

Con el desarrollo de esta arquitectura se ha conseguido una plataforma la cual puede ser fácilmente adaptable a otros ámbitos diferentes del abarcado en cuestión en este proyecto. Este características viene dada sobre todo porque el diseño de esta arquitectura se basa en J2EE. Además el esquema síncrono desarrollado es igualmente válido para cualquier tipo de aplicación puesto que las suscripciones se hacen a nivel de herramienta y se maneja de forma general por parte de la arquitectura. Por lo tanto con mínimos cambios dicho diseño podría ser reutilizado para crear un entorno con diferente aplicación.

Agradecimientos.

Los desarrollos e investigaciones realizadas en este entorno han sido financiados dentro del marco del proyecto COLAB *Collaborative Laboratories for Europe* IST-2000-25035.

Referencias.

[Begole98]Begole J. "Flexible Collaboration Transparency: Support Worker Independence in Replicated Application-Sharing Systems" PhD 1998.

[ELVIN]Link Elvin DSTC Elvin http://elvin.dstc.edu.au/

[García01]Pedro García López, Antonio Gómez Skarmeta, Robert Rallo Molla. *"ANTS: a new Collaborative Learning Framework"*. Proc. Of the European CSCL. Maastricht (Holanda) Marzo 2001.

[Garcia02]García López P., Gómez-Skarmeta A. "Plataforma ANTS: Una arquitectura software basada en componentes para el desarrollo de Aplicaciones Distribuidas de Trabajo Colaborativo" Tesis Doctoral (2002).

[J2EE]Link J2EE http://java.sun.com/j2ee/

[JBOSS] Link JBoss http://www.jboss.org/index.html

[JMS]Link JMS Java Message Service http://java.sun.com/products/jms/index.html

[JSDT]Link JSDT Sun Java Shared Data Toolkit ver 2.0-EA1 http://java.sun.com/products/java-media/jsdt

[Kligyte01]Kligyte G, Leinonen T. *"Study of funcionality and and interfaces of existint CSCL/CSCW systems"* ITCOLE project deliverable D3.1 (2001)

[Leinonen01] Leinonen T., Hakkarainen K., Appelt W., Dean P., Gómez-Skarmeta A., Ligorio B., Lipponen L., Merisaari L., Mielonen S., Pontecorvo C., Sligte H., & Vosniadou S. *"ITCOLE Project: Designing innovative technology for collaborative learning and knowledge building"*. Proceedings of the World Conference on Educational Multimedia, Hypermedia & Telecommunications. Tampere. Finland, June 25-30, 2001.

[Martinez02]M. A. Martínez Carreras, A. F. Gómez Skarmeta, J. Tavira Moreno, P. García López "Integración de herramientas síncronas en un sistema BSCW" IE-2002, Vigo 2002

[Stahl02]Stahl G. "Contributions to a Theoretical Framework for CSCL" CSCL 2002