LeanSim: Un sistema de simulación para el entrenamiento de personal especializado dentro de sistemas complejos.

Pau Fonseca i Casas
Departamento de Estadística e Investigación Operativa. Universidad Politécnica de Cataluña.
Barcelona, Catalunya, 08034, SPAIN

y

Josep Casanovas i García
Departamento de Estadística e Investigación Operativa. Universidad Politécnica de Cataluña.
Barcelona, Catalunya, 08034, SPAIN
y

Jordi Montero i García
Departamento de Estadística e Investigación Operativa. Universidad Politécnica de Cataluña.
Barcelona, Catalunya, 08034, SPAIN

RESUMEN

La simulación se usa no solo para determinar posibilidades en sistemas reales y evaluar las diferentes alternativas que se presentan en cada modelo, sino también, y cada vez más, para poder entrenar a personal especializado en la ejecución de tareas que pueden ser mas o menos críticas. Es en este punto, donde el simulador se trasforma para pasar a ser una herramienta donde lo importante es la capacidad para emular el sistema real a partir de, no únicamente los comportamientos asociados a los elementos que participan del modelo, sino también de la representación que estos adoptan en el ordenador. Es por ello que las técnicas de representación virtual se convierten en claves obligadas para poder desarrollar sistemas de entrenamiento que cumplan con las expectativas que se depositan en ellos.
En el presente artículo se muestra un sistema denominado LeanSim, desarrollado en su totalidad en el LCFIB (Laboratorio de Cálculo de la Facultad de Informática de Barcelona), que permite el entrenamiento de personal especializado en el manejo de herramientas definidas dentro del modelo.

Palabras clave. Simulación, Realidad virtual, VRML, Sistema distribuido, Internet, Sistema de entrenamiento, Agente.

1. INTRODUCCIÓN: LA ESTRECHA RELACIÓN ENTRE REALIDAD VIRTUAL Y SIMULACIÓN.

La realidad virtual ha pasado de ser una forma más para poder hacer la representación de los modelos de simulación a ser casi una necesidad, que permite al simulador mostrar de una forma realista los diferentes procesos que tienen lugar dentro del modelo.
Lejos quedan ya aquellos entornos de simulación, como GPSS/H o SLAMM, [9] que únicamente, a partir de una introducción previa de datos (los datos con los que se alimentaba el simulador, y la parametrización de los diferentes elementos que participaban en el modelo), permitían obtener unos resultados estadísticos que daban respuesta al comportamiento de las diferentes variables que previamente habíamos preseleccionado. Eran simulaciones que no tenían representación visual, y por lo tanto ofrecían pocas facilidades para realizar interacciones con el usuario, (aunque evidentemente si que era posible realizar estas interacciones, a partir de la entrada de datos por parte del usuario con *ampervariables*, para mas información ver [9], en el caso de GPSS/H, por ejemplo, es evidente no obstante, que sin una representación visual para un usuario poco acostumbrado a trabajar con datos estadísticos podría resultar una interacción difícil).
Actualmente, y cada vez mas los simuladores se usan en el campo del entrenamiento, y es por ello que es necesario la incorporación de técnicas de representación lo mas detalladas y realistas posibles. Evidentemente esta necesidad de detalle implica el desarrollo de técnicas de representación de realidad virtual.
El problema de combinar una representación virtual rica y detallada con un modelo de simulación, posiblemente complejo es evidente, y normalmente pasa por una construcción a dos fases, que en el mejor de los casos se podrían superponer en el tiempo.
Una primera fase consistiría en la construcción del modelo lógico, la segunda seria la representación de los elementos y la conexión de esta representación con el modelo de simulación.
LeanSim permite una integración total del modelo de simulación con las diferentes representaciones virtuales de los elementos del sistema.
Uno de los criterios de diseño más importantes que se tuvieron en cuenta para la construcción de la arquitectura del sistema, era que se deseaba que en la definición de todos los objetos del modelo, se pudiera definir una representación virtual asociada, así como comportamientos intrínsecos de los objetos desde el punto de vista de esa representación virtual, como por ejemplo cambios en la representación de los objetos a partir del estado en el que se encuentran, o movimientos específicos de los mismos, independientemente de su estado.
A partir de esta facilidad, el sistema permite definir un complejo y detallado modelo de simulación con técnicas de representación basadas en realidad virtual, sin que ello conlleve un coste de tiempo adicional, como acostumbra a pasar con cualquier otro software de simulación estándar.

Esta estructuración se basa en una clara separación entre los tres elementos fundamentales que se pueden encontrar en todo modelo de simulación, es decir:

- Entidades.
- Operaciones o actividades
- Procesos.

Estos elementos se corresponden con la mayoría de los elementos que podemos encontrar descritos en los libros de texto como elementos fundamentales de simulación [1] o [3].
En LeanSim no obstante, aparece un cuarto elemento que hay que tener en cuenta en los modelos, y que facilita enormemente la vinculación de la representación virtual con el modelo, denominado máquina.

Funcionalidad	Familia	Elementos
Creación de entidades que entran en el sistema.	Generador	Generador Estocástico
		Generador Bajo Demanda
		Generador Frecuencia
		Generador Script
Destrucción de entidades que abandonan el sistema	Destructores	Destructor
Creación de entidades a partir de otras entidades	Empaquetadores	Empaquetador
		Desempaquetador
		Transformadores
Elementos de control de operaciones	Recursos	Recursos Estáticos
		Recursos Dinámicos
		Recursos Humanos
Almacenamiento de entidades	Almacenes	Silo
	Estantes	Estantes
	Almacén de baja estacionalidad	Colas con Time Out
		Colas Estándares
Procesadores de tiempo	Servidores	Servidor Estático
		Servidor Dinámico
Control de procesos	Controladores	Bifurcadotes
		Selectores
		Enrutadores
		Señalizadotes
Tranposrtes de entidades	Transportes	Vehículos
	Cintas	Tapices rodantes Cintas de rodillos

Tabla 1: Elementos de LeanSim.

Estas máquinas (Tabla 1), son en primera instancia, una especie de contenedores de operaciones que indican las posiciones físicas dentro del modelo en las que las operaciones tienen lugar. Una segunda función de las máquinas es la de representar a los recursos, (elemento escaso necesario para efectuar alguna operación) definidos durante el ciclo de vida de un proceso.
De esta forma en cada operación se indica, no simplemente la máquina en la que se tiene que efectuar, sino también las máquinas que necesita para que esta operación se pueda efectuar.
Hay que tener presente que las operaciones pueden ser extremadamente amplias, como las mas sencillas, que simplemente representarían una demora en un proceso, hasta las que permitirían mover las entidades de un punto a otro, que estarían representadas en máquinas cintas o máquinas de transporte.

A continuación en la siguiente tabla se resumen las principales operaciones que pueden usarse dentro del marco de un modelo de entrenamiento basado en LeanSim.

Nombre	Efecto
Abandonar	La entidad abandona el sistema.
Almacenar	Almacena la entidad en el almacén donde se realiza la acción.
Bifurcar	La entidad puede cambiar de proceso en función de una condición.
Desempaquetar	Realiza el desempaquetado del contenido de una entidad.
Destruir	Destruye la entidad y recoge los estadísticos.
Empaquetar	Realiza el empaquetado de una o más entidades en un paquete.
Enviar señal	Envía una señal de sincronismo y/o comunicación.
Enrutamiento	Cambios de procesos en función de una distribución acumulativa.
Entrar	Una entidad entra en el sistema.
Generar	Genera una entidad o conjunto de entidades y las introduce en el modelo.
Liberar recurso	El elemento libera el recurso especificado en la acción.
Modificar	Modifica un atributo de una entidad o de un elemento.
Mover	Mueve un elemento de simulación a una posición especifica,
Selección elemento	Solicita una instancia de un conjunto de elementos.
Solicitar recurso	Petición de uno o más recursos.
Transformar entidad	Transforma la entidad en un conjunto de nuevas entidades.
Ejecutar Script	Ejecución de un script de programación
Bifurcar Script	La entidad cambia de proceso según un script de programación.
Route Script	Combinación de las dos operaciones anteriores (Ejecutar y Bifurcar) .
Usuario	Acciones definidas por el usuario la entidad se demora un tiempo.

Tabla 2: Operaciones

La secuencia de operaciones es lo que define un proceso y es a partir de esta definición de procesos que se puede especificar el comportamiento del modelo.

2. ARQUITECTURA DE LEANSIM

No es el objetivo del presente articulo detallar la arquitectura del sistema, no obstante, si que se mostrará un esbozo de la estructura externa que presenta, así como el sistema de comunicación existente entre cada uno de los diferentes elementos que lo engranan.
Todos los elementos han sido desarrollados fundamentalmente en C++ [5] [6] y Java [4], mientras que para la representación virtual se ha usado VRML [8]. La conexión con elementos programados con otros lenguajes, como por ejemplo Delphi ® o VisualBasic, se ha realizado a partir de controles ActiveX [7].

En el siguiente diagrama se representa a *grosso modo* los elementos más importantes que constituyen LeanSim.

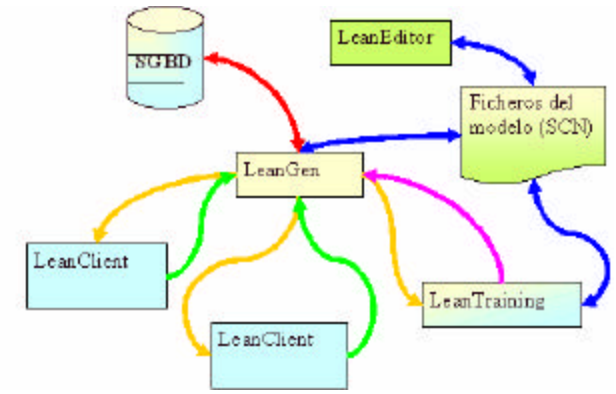

Figura 1: Arquitectura de LeanSim

En este esquema simplificado de la arquitectura de LeanSim, se pueden observar dos instancias de un módulo denominado LeanClient, y una de otro denominado LeanTraining. Realmente no importa el número de instancias que existan de cada uno de estos dos elementos, dado que, como se explicará mas adelante, no es en ellos donde reposa el núcleo central de la simulación, sino que básicamente sirven para que diferentes usuarios interactúen independientemente y diferentemente según su perfil o rol con el sistema.
Los ficheros que contienen la descripción de las plantillas a usar en los modelos que se simularan, así como la descripción de los modelos, son ficheros de texto con extensión SCN. Estos ficheros son escritos por el editor de plantillas de simulación (LeanEditor), para su posterior personalización en el motor de simulación (LeanGen).

Brevemente el funcionamiento seria el siguiente:

1. LeanEditor genera los ficheros SCN (Scenery files) necesarios para definir una plantilla (Template) del proyecto con el que se trabaja.
2. LeanGen a partir de las plantillas, personaliza un proyecto concreto, y ejecuta una simulación.
3. Los sistemas de representación distribuida muestran la simulación en diferentes dispositivos visibles desde la red asignada al motor de simulación..
4. Los sistemas de entrenamiento LeanTrainig, se conectan al motor de simulación LeanGen, e interactúan con el.
5. Finalmente se recogen los datos.

A continuación se enuncían las principales funcionalidades de cada componente.

LeanEditor
Es el editor de modelos de Leansim. Con el se pueden definir las entidades, las máquinas, los recursos y los procesos que fluirán por el modelo.
Es importante remarcar que el modelo de simulación construido a partir del editor se basa en el paradigma de interacción de procesos, donde a partir de la definición de los mismos queda completamente definido el modelo de simulación.

LeanEditor cuenta con tres zonas importantes en su interfaz:

Estructura arbórea: Permite la visualización estructurada y jerárquica de los diversos elementos, entidades, procesos y objetos que se hayan añadido al modelo

Visión 2D: donde se puede contemplar una representación plana del modelo.

Visión virtual: donde se puede observar el modelo en formato de realidad virtual, y el movimiento que las entidades seguirán a través de las rutas.

En la figura 2 se muestra el aspecto de LeanEditor con las tres zonas mencionadas.

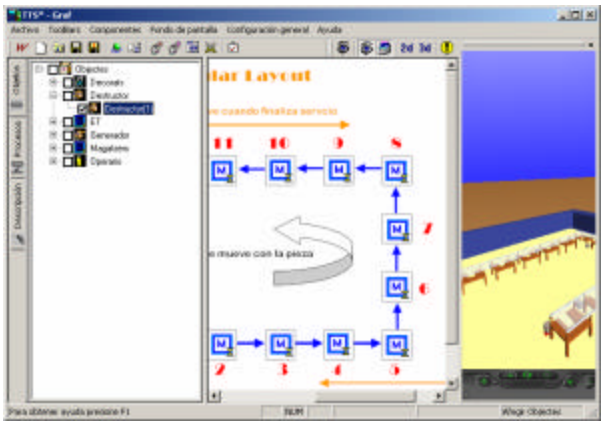

Figura 2: Aspecto de LeanEditor

LeanGen
Contiene el motor de simulación propiamente dicho, así como dos motores de representación (uno para la representación 2d y otro para la representación en formato de realidad virtual) y el subsistema de adquisición de datos (recolector de estadísticos).
El proceso de construcción de un modelo de simulación con el sistema LeanSim presenta una serie de fases que se describen a continuación.

A partir de un modelo construido con LeanEditor, denominado *Template* (plantilla), se permite personalizar cada uno de los objetos de simulación a partir de modificar los tiempos de servicio de las diferentes operaciones que constituyen los procesos. Esta personalización da lugar a la construcción de un proyecto de simulación, que ya contiene los datos adecuados para realizar el primer conjunto de experimentos.

En la figura 3 se puede observar el aspecto que presenta LeanGen durante la ejecución de una simulación.

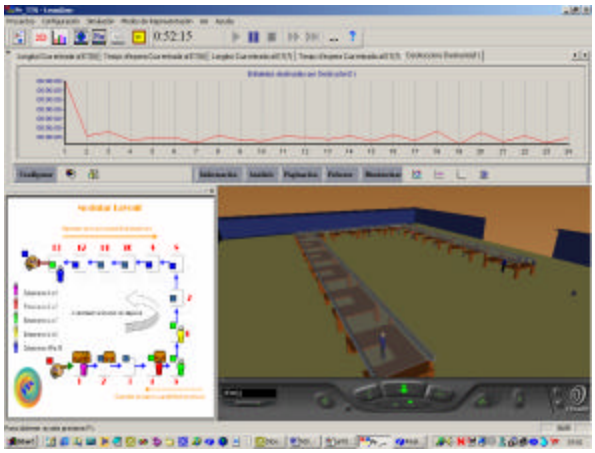

Figura 3: Aspecto de LeanGen

Fundamentalmente consta de tres áreas claramente diferenciadas, que permiten mostrar la simulación de tres formas diferentes.

Área de trabajo 2D: La primera área muestra una representación clásica en 2D, a partir de los iconos que representan cada una de las máquinas del modelo. Un problema que suele aparecer en este tipo de representación, es como representar un modelo que tiene diferentes niveles (altura). La solución a veces es representar en diferentes posiciones del mismo plano los diferentes niveles del modelo, como por ejemplo se puede hacer en Witness®. En LeanGen, solo se muestra en la ventada de 2D un nivel cada vez, y es el usuario el que puede seleccionar que nivel desea ver en cada momento.

Área de realidad virtual: La segunda área muestra una representación en realidad virtual de las mismas máquinas mostradas en 2D. Puede representarse también de forma distribuida a través del LeanClient.

Área de estadísticos: La tercera forma de representar la simulación es a partir de la visión de los valores que van tomando los diferentes estadísticos que se recogen.

Actualmente la representación de los estadísticos puede desvincularse del motor de simulación y representarse de forma distribuida a partir de un control de representación de estadísticos denominado LeanStatistics.

LeanClient

Este modulo de Leansim permite representar en una máquina cliente una simulación virtual, descargando considerablemente el cómputo en la máquina en la que se esta ejecutando la simulación.

Integra la posibilidad de abrir una sesión de FTP, para poder descargar el proyecto que se esta ejecutando en la máquina remota y el avatar asociado a los diferentes usuarios de los diferentes módulos LeanClient que estén conectados al motor de simulación.

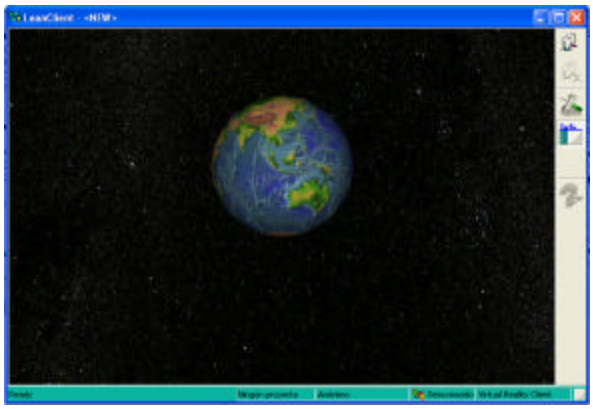

Figura 4: Aspecto de LeanClient

En la figura 4 se puede observar el aspecto de LeanClient antes de establecer una comunicación con el motor de simulación LeanGen.

En las siguientes figuras se puede observar el proceso de simulación de un modelo MM3 (Figura 5), y su representación en una máquina remota a través del cliente (Figura 6).
En este ejemplo se definen 6 máquinas:

- Un generador de entidades.
- 4 servidores estáticos.
- Un destructor.

El proceso que rige el comportamiento del sistema está constituido por las siguientes operaciones (entre paréntesis está indicada la máquina donde tiene lugar la operación):

1. Generar (generador de entidades)
2. Seleccionar máquina (Servidor estático)
3. Operación de usuario (Servidor estático)
4. Destruir (destructor).

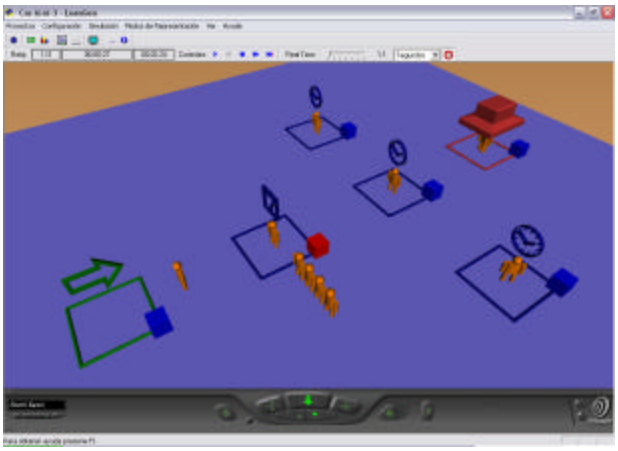

Figura 5: Simulación de un modelo MM3

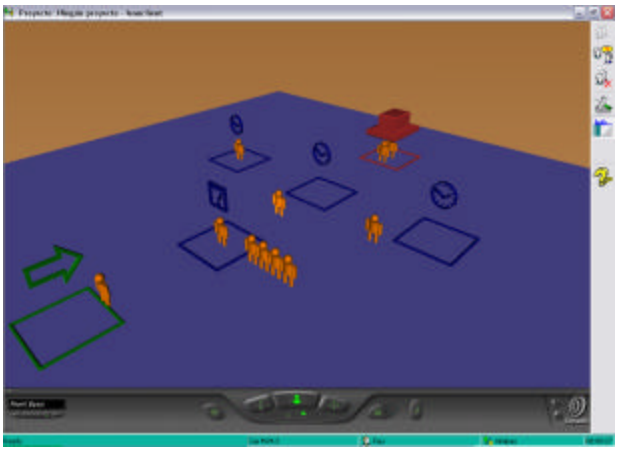

Figura 6: Representación del modelo MM3 en un cliente remoto

LeanTraining

En un modelo de simulación puede haber diversos *puntos de interacción*, es decir, puntos a partir de los que un usuario puede modificar el comportamiento del sistema.

Estos puntos de interacción, se comunicarán con el usuario a partir de los módulos denominados LeanTraining. De esta forma, estos módulos pueden permitir el entrenamiento de personal especializado. La arquitectura externa de estos módulos reposa sobre una especialización del módulo de representación virtual distribuido LeanClient.

Es en este punto donde se hace mas patente la necesidad de poder distribuir la simulación entre diferentes ordenadores, puesto que cada usuario de LeanTraining necesita, evidentemente disponer de su propio dispositivo (PC, Tablet PC, Hand Held) para interacturar con la máquina asignada en el proceso de entrenamiento, pero no obstante tendrá que trabajar conjuntamente con otros operarios para evaluar los efectos de todas sus acciones sobre el mismo sistema.

La construcción de un modulo de entrenamiento se basa en VRML [8], conectado a partir de un sistema denominado EAI (External Authoring Interface) con las clases C++ del cliente, eventualmente se usa también Java [4] para la personalización del comportamiento de algún elemento del sistema de entrenamiento. Esta estructura permite integrar el sistema de entrenamiento fácilmente dentro del cliente de representación. Así mismo gracias a un sistema de mensajes, las acciones que el usuario realiza sobre el módulo de entrenamiento tienen una respuesta dentro del motor de simulación (LeanGen)

La comunicación con LeanGen se traduce en cadenas de texto enviadas a la máquina servidor a través de TCP/IP.

Este mecanismo permite desvincular la ejecución de los diferentes sistemas de entrenamiento del motor central de simulación, facilitando su uso y la velocidad de cálculo del modelo.

Cada módulo de entrenamiento se tiene que definir, fundamentalmente, a partir de querer representar un *punto de interacción* de un usuario especifico, definido normalmente a partir de un perfil (el rol que desarrolla dentro del sistema), con el sistema general.

La interacción con el modelo se realiza a través de la modificación de la simulación que se ejecuta en el motor de simulación (LeanGen), en la máquina servidor.

De esta forma, en un caso sencillo, el punto de interacción representaría normalmente la máquina con la que el operario trabaja habitualmente. Por este motivo, debido a que interesa que sea lo más parecida a la máquina real, que el uso de la realidad virtual es fundamental cuando hablamos de LeanTraining.

La definición de un módulo de LeanTraining, sigue los siguientes pasos:

1. Creación de la **estructura virtual** que representa la máquina a través de la que el usuario interactúa con el sistema.
2. Definición de la **mensajeria** (señales), que puede enviar y recibir el punto de interacción (máquina). Esta definición hace únicamente referencia a la estructura conceptual de la información necesaria para modificar el curso normal de la simulación.
3. Definición de los **actuadores de salida** que permiten **enviar** mensajes al motor de simulación LeanGen. Son activados por el usuario, por ejemplo a partir de activar la representación de un palanca (que representaría una palanca de la máquina real con la que el operario trabajaría).
4. Definición de los **actuadores de entrada** que permiten **recibir** mensajes a partir del servidor de simulación. Estos actuadores normalmente pueden simplemente ser pantallas sobre las que se representan las consecuencias de las acciones que el usuario provoca en el modelo, por ejemplo, bajo la forma de paneles informativos.
5. Definición de las **lógicas de comportamiento** internas específicas del punto de interacción. Esto permite dotar a los puntos de interacción de una lógica adicional, que permite por ejemplo filtrar la información que se muestra en los actuadores de entrada.

La arquitectura interna de un sistema LeanTraining es la mostrada en la figura 5.

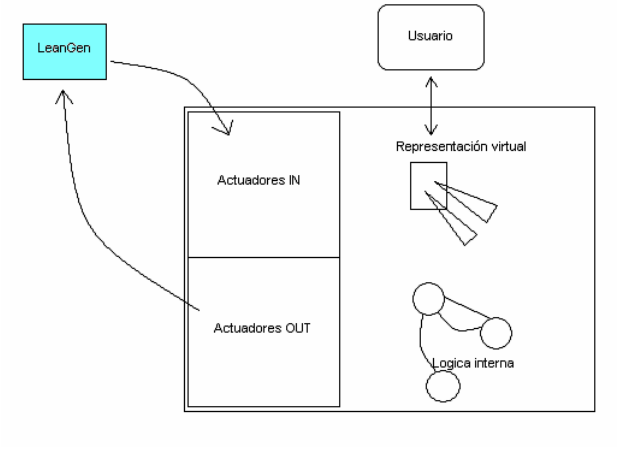

Figura 7: Arquitectura de LeanTraining.

Como se puede observar, esta arquitectura es muy similar a la usada para la implementación de alguna tipología de agentes inteligentes [2].

También en este diagrama se puede observar como la interacción del usuario con el sistema, definido en el motor de simulación, únicamente tiene lugar a través de la

representación virtual del punto de interacción con el que esta familiarizado el operario.
Esto facilita enormemente el uso del modelo de entrenamiento por parte del usuario, y no requiere de otros elementos, a parte de los necesarios para representar correctamente la realidad virtual, para poder realizar el entrenamiento.

3. MENSAJES DE LEANTRAINING

Las señales permiten establecer la comunicación entre el usuario y el modelo de simulación a través del punto de interacción.
Estas señales se envían a determinados objetos del modelo de simulación, lo que permite modificar el proceso que rige la lógica del modelo
.
Las señales tienen su origen en un elemento de simulación, denominado emisor, y tienen como destino uno o más elementos, receptor o receptores, que reciben la señal.

Los destinos típicos de las señales pueden ser:

- CA: Cualquier elemento de simulación.
- S: Silos.
- G: Generadores de entidades.

El mecanismo de activación de las señales en Leansim es consecuencia directa de los cambios que tienen lugar en el elemento emisor, tal y como se expresa en los siguientes puntos:

- Un cambio de estado en el elemento emisor, como por ejemplo el bloqueo en la entrada de productos del sistema cuando se registra una anomalía en el sistema de recepción de una factoría.
- Una condición de llenado de una cola o buffer dedicado del emisor, como por ejemplo, forzar la avería de uno o más receptores al detectar que la cola de entrada del emisor es igual a tres. Esta cola podría estar asociada a una entidad de tipo "pieza defectuosa", y la maquina con la señal avisaría de la existencia de un determinado número de piezas defectuosas en su cola.
- Una condición de vaciado de una cola o buffer dedicado del emisor, como por ejemplo cada vez que la cola del emisor es inferior a un valor prefijado por el usuario.
- El proceso de encolar o desencolar una entidad de una cola o buffer dedicado, es decir, cada vez que se detecta la entrada o salida de una entidad del buffer dedicado del elemento emisor se enviaría la señal.
- Una condición asociada a un atributo del elemento emisor, es decir un atributo de la máquina, al llegar a cierto valor activaría el envío de la señal.
- Sensores definidos a partir del punto de interacción. A partir de los cuales se podría enviar a uno o más emisores una señal concreta. Los sensores responderían a la posición que ocupa el usuario que se esta entrenando en el mundo virtual, o al tiempo.
- Actuadores definidos en el punto de interacción, a partir de los que el usuario podría interactuar de forma activa con el modelo.

Los mensajes definidos en el sistema son los siguientes:

Identificador	Efecto en el destino	Destino
StopSimulation	Finaliza el estudio de simulación	CA
StopProcess	Finaliza el proceso actual1	CA
PauseProcess	Detiene temporalmente el proceso actual	CA
RestartProcess	Arranca el proceso temporalmente detenido	CA
GenerateByDemand	Petición de generación de una entidad bajo demanda	S, G
CloseOutDoor	Cierra la salida del elemento, no puede transmitir o propagar las entidades a otros elementos	CA
OpenOutDoor	Abre la salida del elemento, puede transmitir o propagar las entidades a otros elemento	CA
CloseInDoor	Cierra la entrada del elemento, no admite entidades para ser procesadas	CA
OpenInDoor	Abre la entrada del elemento, no admite entidades para ser procesadas	CA
ForceLostEntity	Fuerza la perdida de la entidad que se esta procesando	CA
ForceSetup	Recuerda al elemento que deberá realizar una operación de mantenimiento cuando inicie una nueva actividad con una nueva entidad.	CA
ForceFail	Provoca una avería en el elemento	CA

Tabla 3: Mensajes definidos.

4. INTERFÍCIES VIRTUALES VS INTERFÍCIES FÍSICAS

Multitud de sistemas de entrenamiento basan su interfície en un sistema físico, más o menos complejo sobre el que reposa el modelo. Esta estructura física se construye con la finalidad de emular las herramientas como las que el operario, al que se desea entrenar, trabaja habitualmente.

Las ventajas de construir un sistema con estas características son claras:

1. Permiten que el usuario se involucre completamente con el modelo, reduciendo drásticamente la curva de aprendizaje, puesto que desde un primer momento puede usar ya los diferentes elementos que en el sistema real usará.
2. Permiten evaluar al individuo, al poder monitorizar el resultado de todas sus acciones. Esta monitorización puede tener diferentes utilidades, desde corregir vicios del operario, hasta detectar problemas de operativa, etc.

No obstante como principales contrapartidas tenemos fundamentalmente dos:

1. El alto coste de cada interfaz. Este coste no se tiene que entender simplemente desde el punto de vista económico, sino también temporal.
2. Debido a la especialización de las interfaz estas tienen una clara dificultad para poder ser reaprovechadas en otros modelos.

Para solucionar estos dos problemas, sin perder las claras ventajas que proporcionan, es necesario usar interfaces de realidad virtual completamente inmersivas pero de uso general, es decir que no estén basadas en estructuras físicas concretas, –guantes, gafas o incluso trajes de realidad virtual-, permitan reproducir cualquier máquina con la que un operario pudiera trabajar.

La ventaja de un sistema así es que ahora ya no tendremos una interfaz especializada en un modelo concreto, entendiendo por interfaz, los elementos físicos necesarios para permitir la comunicación entre el usuario y el modelo, sino que se podrá reutilizar en multitud de modelos, con el consecuente ahorro de dinero, simplemente se tiene que reconstruir el mundo virtual que simula el punto de interacción con el que trabaja el usuario.

5. LA NECESIDAD DE LA SIMULACIÓN DISTRIBUIDA

La simulación con representación distribuida permite la construcción de modelos virtuales mucho más complejos, puesto que gracias a ello se pueden separar completamente los procesos lógicos del modelo respecto los que controlan la representación virtual.
En sistemas de entrenamiento es no obstante fundamental poder tener esta separación, puesto que cada usuario que interactúa con el modelo ha de poder hacerlo desde diferentes lugares, evidentemente para no interferir unos con otros.

En el siguiente diagrama, figura 6, se muestra un hipotético esquema del sistema LeanSim usado para entrenar a dos operarios en un determinado modelo. En este esquema se han incluido también agentes inteligentes, para mostrar un esquema mas completo de las posibilidades del sistema.
Los agentes inteligentes se podrían incluir también en un sistema de entrenamiento basado en LeanSim a partir de usar el mismo sistema de mensajes definido para los sistemas LeanTraining.

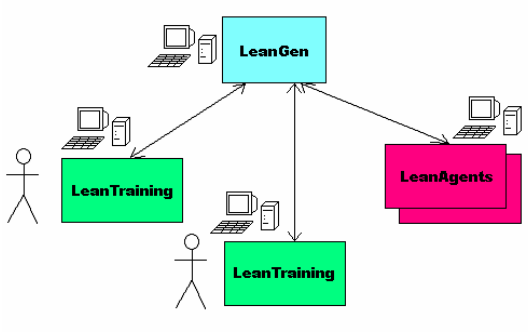

Figura 8: Esquema de sistema de entrenamiento basado en LeanSim.

En la figura se puede observar que cada usuario se comunica con el modelo a través de una instancia de LeanTraining, que se ejecuta en una máquina diferente, no obstante todos los agentes pueden estar ejecutándose simultáneamente en la misma máquina (aunque no es necesario).
Comparando la arquitectura básica de un sistema de entrenamiento y la de los agentes inteligentes, dentro del marco del sistema de entrenamiento que se esta describiendo, podríamos observar como principal diferencia, que los agentes no requieren una interacción con el usuario para poder interactuar con el modelo, es decir, la lógica interna suple la necesidad de un usuario que guíe sus operaciones.
En un sistema de entrenamiento el uso de agentes inteligentes puede ser extremadamente interesante, para poder dotar al sistema de un mayor grado de realismo, simulando el comportamiento de estructuras, entidades, o máquinas que poseen un tipo de comportamiento relativamente complejo. Mas aún, como se indica en [10], los agentes también, podrían aprender estrategias y competir en cierto modo con los operarios reales, que manipulan los LeanTrainig, y de esta forma mejorar su entrenamiento.
Es decir, si seguimos la clasificación de las diferentes tipologías de agentes con las que nos podríamos encontrar a partir de [11] tenemos:

1. Agentes inteligentes: fundamentalmente agentes dotados de algún mecanismo que les permita tener el calificativo de inteligente.
2. Agentes que aprenden.
3. Agentes móviles, es decir que se puedan mover de una máquina a otra.
4. Agentes creíbles, es decir que tengan algún tipo de representación que permita establecer una comunicación mas “natural”.

Los agentes a los que se hace referencia serian fundamentalmente elementos que aprenderían del usuario que se está entrenando para poder ponerlo a prueba a partir de complicar las situaciones, del entorno, en el que se encuentra.

6. CONCLUSIONES

Los sistemas de entrenamiento para personal especializado, cada vez son más importantes para reducir su curva de entrenamiento, además que permiten disponer de un personal más versátil, debido a la capacidad de adaptarse a nuevas máquinas a partir del entrenamiento que puntualmente puedan recibir.
Como principales ventajas se pueden citar las siguientes:

- El coste de la realización de una sesión de entrenamiento utilizando un simulador es muy inferior al coste que tendría realizar la sesión en un entorno real. La maquinaria vería reducido su ciclo de vida al manejarla personal inexperto, el coste de los materiales en algunos casos es prohibitivo (como pueden ser casos de sistemas de entrenamiento médico con cadáveres o animales vivos), etc.
- Permite dotar al usuario de una mayor experiencia al poder recrear en el modelo de simulación cualquier situación sin tener que esperar a que estas se den en una sesión real.
- Cualquier proceso puede ser repetido tantas veces como sea necesario y en las mismas condiciones. Este hecho, en un sistema real sería prácticamente imposible.
- Disponibilidad del simulador mucho más elevada que en un sistema real.
- Posibilidad de análisis de las sesiones realizadas y probar diferentes alternativas a un mismo problema.

No obstante los sistemas de entrenamiento presentan, como principal desventaja, su elevado coste, y su vinculación, casi completa, con el sistema real al que están destinados a emular, no permitiendo, prácticamente nunca su adaptación a un sistema diferente.
Es evidente que la realidad virtual es un elemento fundamental para la construcción de sistemas de entrenamiento, ya que permite una interacción del usuario con el modelo de una forma más natural.
En última instancia interesaría que el operario interactuara con el modelo como si del sistema real se tratase. No obstante la construcción de sistemas que incorporen realidad virtual es costosa, no solamente por la construcción de los diferentes elementos físicos, es decir los actuadores sobre los que interactuará el usuario, sino también porque no existen herramientas de simulación genéricas que permitan la descripción de sistemas, y la interacción con los mismos mediante mecanismos de realidad virtual.
LeanSim permite definir estos modelos, y a través de LeanTraining definir interfases de acceso, puntos de interacción para los usuarios que se desea entrenar, fundamentalmente gracias al mecanismo de señales descrito anteriormente, y a la posibilidad de distribuir la representación.
La posibilidad de representar estos puntos de interacción en formato de realidad virtual no obliga a la construcción de sofisticados aparatos para representar los diferentes actuadores que el operario usará para comunicarse con el modelo.
Además la posibilidad de describir el modelo de forma directa en formato de realidad virtual permite al usuario reducir sensiblemente los costes derivados de la construcción del modelo.
En definitiva, esto permitiría que empresas con recursos más modestos pudieran acceder a sistemas de entrenamiento, lo que permitiría reducir el coste destinado a la preparación de personal especializado, y de esta forma aumentar su competitividad y seguramente reducir sensiblemente, por ejemplo, los riesgos laborales asociados.

7. REFERENCIAS

[1] Antoni Guasch, Miquel Àngel Piera, Josep Casanovas, Jaime Figueres, **Modelado y simulación. Aplicación a procesos logísticos de fabricación y servicios**. Edicions UPC, 2002

[2] Ulises Cortés García, Javier Béjar Alonso, Antonio Moreno Ribas **Inteligencia Artificial** , Edicions UPC,1994

[3]. Law, A. M., Kelton, W. D. **Simulation modeling and analysis**. McGraw-Hill, 1991.

[4] Jaime Jaworski, **Java 1.2 al descubierto**. Editorial Prentice Hall, 1999

[5]. **Microsoft Foundation Classes version 6.0 for Visual C++**. Microsoft Development Network. Visual Studio 6.0.

[6]. **Microsoft Development Network Library. Visual C++ Section**. MSDN.

[7]. **The Direct X Programmer Reference**. Graphics and Multimedia Services. Microsoft Developer Network. 1998-1999.

[8] Andrea L. Ames ; David R. Nadeau ; John L. Moreland **VRML 2.0 Sourcebook**, 2E. 688.

[9] Geoffrey Gordon. **System simulation**. Editorial Prentice Hall 1978.

[10] Michael Luck, Peter McBurney, Chris Preist; **Agent Technology: Enabling Next Generation Computing**; AgentLink 2003

[11] Richard Murch, Tony Johnson; **Intelligent Software Agents**; Prentice Hall 1999.