Evaluación de Herramientas para la Construcción de Aplicaciones Colaborativas

Javier Díaz, jdiaz@unlp.edu.ar
Claudia A. Queiruga, Claudia M. Banchoff Tzancoff, <claudiaq, cbanchof>@info.unlp.edu.ar
Laboratorio de Investigación en Nuevas Tecnologías Informáticas (LINTI) –UNLP

Palabras Claves: colaboración, Internet, cliente-servidor, NCSA Habanero, Xerox ILU

Resumen

En la actualidad, el uso de las redes de datos y de todas las tecnologías asociadas juegan un rol preponderante en el diseño de las nuevas aplicaciones. La influencia de Internet, se evidencia en las tecnologías en expansión como WWW (en modalidades: pull y push) y la popularidad del lenguaje JAVA (lenguaje de programación de Sun Microsystems) [Gosling, 1996] y sus derivados.

La tendencia actual en el diseño de aplicaciones distribuídas es aprovechar esta infraestructura tecnológica y, potenciar la distribución de recursos tanto humanos como de hardware y software. Sumado a esto, se ha observado el incremento de una nueva modalidad de trabajo en donde ya no es imperante que las personas compartan una misma oficina y/o edificio y donde es posible que un mismo grupo de trabajo resida en distintas ciudades o países. Teniendo en cuenta esta nueva realidad, se requiere de la construcción de herramientas de software que permitan la interacción entre individuos localizados en lugares de trabajo que podrían ser distantes o no, conectados mediante la gran red global.

Existen diferentes proyectos de software que, utilizando como soporte todas las tecnologías asociadas a Internet, permiten a grupos de usuarios compartir información, sin las limitaciones del espacio y del tiempo. En este artículo, se analizan las características de tres proyectos, denominados TANGO [Beca, 1997] [3], NCSA Habanero [2] e ILU (Inter-Languages Unification)[1]. El objetivo principal de este artículo es comparar NCSA Habanero y TANGO como herramientas que brindan una infraestructura para construir aplicaciones colaborativas [Hsu, 1993] [Baecker, 1993] [Ellis, 1993] y, presentar ILU como una “meta herramienta” que provee una infraestructura que facilita la construcción de aplicaciones orientadas a objetos distribuídas sobre plataformas heterogéneas. Estas aplicaciones abarcan a las aplicaciones colaborativas que son la principal línea de trabajo del LINTI. En NCSA Habanero y TANGO el modelo básico de interacción consiste en replicar la información (datos y eventos) en las múltiples estaciones de trabajo aplicando distintas políticas: en NCSA Habanero todas las copias de una aplicación que comparten una sesión tienen los mismos privilegios, en cambio en TANGO cada aplicación define un usuario “master” que coordina esa aplicación dentro de la sesión colaborativa y el resto son usuarios “esclavos”. Por otro lado, las aplicaciones construidas con TANGO son basadas en Web mientras que las construidas con NCSA Habanero se ejecutan sobre un ambiente propietario. Con respecto a ILU, el modelo de interacción es más amplio, ya que es posible construir aplicaciones distribuídas abarcando así a las aplicaciones colaborativas. Por otro lado, tanto NCSA Habanero como TANGO proveen un API para la construcción de las aplicaciones colaborativas, en donde se encuentran todas las clases necesarias para implementar este tipo de interacción. Sin embargo en ILU, a pesar de ser más abarcativo, ya que posibilita la implementación de distintos modelos de interacción, es necesario que estos modelos sean programados según las necesidades.Tanto NCSA Habanero como TANGO proveen un soporte adecuado al lenguaje JAVA, mientras que ILU permite que las aplicaciones sean multilenguaje. Esto es, es posible definir módulos escritos en múltiples lenguajes de programación, que conformen una misma aplicación y vincularlos a través de ILU.

Evaluación de Herramientas para la Construcción de Aplicaciones Colaborativas

Introducción

Actualmente, el uso de Internet está siendo cada vez más popular y, la utilización de todas las tecnologías asociadas juegan un papel muy importante a la hora de diseñar nuevas herramientas de software. La popularidad de las comunicaciones basadas en Web y la evidente expansión de JAVA [Gosling, 1996] es una muestra clara del crecimiento de Internet. La tendencia actual en el diseño de aplicaciones distribuídas es aprovechar esta infraestructura tecnológica y, potenciar la distribución de recursos tanto humanos como de hardware y software.

El diseño de aplicaciones compartidas por múltiples usuarios y de herramientas de software que soporten la colaboración utilizando la infraestructura provista por Internet es un área de sumo interés en el desarrollo de software actual.

Las aplicaciones colaborativas también denominadas de groupware son sistemas basados en computadoras que proveen el soporte necesario para que grupos de personas comprometidas en una tarea común puedan llevarla a cabo de manera conjunta [Hsu, 1993] [Baecker, 1993]. Estas aplicaciones proporcionan a los miembros del grupo un entorno de trabajo común para poder realizar reuniones on-line (chat, pizarras, videoconferencias, etc.), manteniendo un estado de información consistente a todo el grupo.

Los aspectos fundamentales a considerar en las aplicaciones colaborativas son: las tareas comunes, el ambiente compartido y el espacio/tiempo. Las tareas comunes hacen referencia a la actividad común que los miembros del grupo llevan a cabo; el ambiente compartido es el que permite mantener informado a cada usuario sobre el estado del proyecto, lo que cada participante está realizando, etc.; y el espacio/tiempo se centraliza en el tiempo y el lugar en donde se hace la interacción entre los miembros del grupo [Hsu, 1993]. Teniendo en cuenta este último punto la interacción podría ser sincrónica o asincrónica y, dentro de estas dos modalidades, distribuída o centralizada [Ellis, 1993].

Desde el punto de vista de la interacción con el usuario, las aplicaciones colaborativas utilizan la metáfora de interacción WYSIWIS (What You See Is What I See) en donde todos los usuarios que comparten la aplicación pueden visualizar tanto sus propias acciones como las del resto del grupo [Stefik, 1993].

El soporte para las aplicaciones colaborativas está dado por un conjunto de programas que se comunican entre sí basados en el paradigma cliente-servidor [Vaskevitch, 1995].

En este artículo se analizarán, por un lado, dos ambientes que permiten construir aplicaciones colaborativas, con algunas características similares desde el punto de vista del esquema de comunicación utilizado, ellos son NCSA Habanero [Beca, 1997] [3] y TANGO [2] y, por otro lado, un ambiente denominado ILU (Inter-Languages Unification), que provee un infraestructura más rica y compleja para construir aplicaciones distribuídas en general [1].

NCSA Habanero

NCSA Habanero fue desarrollado en el “National Center for Supercomputing Applications” en la Universidad de Illinois en Urbana-Champaign [2]. Es un sistema distribuído, orientado a objetos que provee un soporte de software para construir aplicaciones

colaborativas. Por estar escrito en JAVA™ (lenguaje de programación de Sun Microsystems) [Gosling, 1996], NCSA Habanero soporta múltiples plataformas de hardware. Además de la evidente reducción de costos por tener un único código fuente y, la alta portabilidad del código, JAVA posibilita que los objetos instanciados viajen entre diferentes plataformas de hardware.

Uno de los principales objetivos de diseño de NCSA Habanero fue crear una herramienta simple para desarrollar aplicaciones compartidas por múltiples usuarios. De esta forma es posible tanto transformar aplicaciones monousuario existentes como construir nuevas aplicaciones colaborativas [Chabert, 1998].

NCSA Habanero consiste de un ambiente colaborativo sobre el que se ejecutan las aplicaciones (construídas con NCSA Habanero o las habanerizadas) y, de una infraestructura para construir aplicaciones colaborativas en Internet. La versión de NCSA Habanero disponible gratuitamente a través del web [2] tiene incorporadas un conjunto de aplicaciones que muestran la amplia gama de funcionalidad colaborativa que se puede proveer mediante NCSA Habanero. Además, estas aplicaciones ilustran las diferentes técnicas que utiliza la infraestructura NCSA Habanero para resolver los distintos problemas que se presentan en el desarrollo de software multiusuario. Algunas aplicaciones proveen básicamente comunicación entre los múltiples usuarios, otras coordinación y otras son aplicaciones de un dominio específico. Entre las aplicaciones que proveen comunicación se encuentra el chat basado en texto y, también basado en audio (actualmente esta facilidad solamente esta provista para estaciones de trabajo Sparc), entre las que proveen coordinación se encuentra la pizarra compartida que permite recuperar imágenes tanto de la estación de trabajo local como a través del web. Entre las aplicaciones de dominio específico se encuentran el visualizador de temperaturas atmosféricas, y del cuerpo humano, que originalmente fueron applets y, luego fueron modificados para trabajar con múltiples usuarios.

La arquitectura del ambiente NCSA Habanero consiste de un servidor y, de uno o más clientes en donde están montadas un conjunto de aplicaciones colaborativas. El esquema cliente-servidor utilizado por NCSA Habanero es el de replicación de datos y eventos [Vaskevitch, 1995]. Cada cliente tiene una réplica del estado (datos y eventos) y, toda la interacción entre ellos está coordinada y controlada por un único servidor de aplicaciones.

Las aplicaciones colaborativas se ejecutan en los clientes NCSA Habanero. Los clientes "escuchan" los datos y eventos generados por las aplicaciones y, los envían al servidor NCSA Habanero, quién decide a qué clientes debe replicar el estado recibido. Cada aplicación elige sus reglas de comportamiento (floor control). El comportamiento éstandard provisto por NCSA Habanero es “libre” esto es, cada usuario de la sesión compartida ve las mismas acciones en el mismo orden y, ninguna acción es rechazada.

La arquitectura del servidor consta de tres módulos: a) serializador, que es el encargado de realizar las réplicas de los eventos en el orden en que se generan, b) árbitros, que controlan el orden de ejecución de los eventos en los diferentes clientes, definiendo en algunos situaciones turnos, y, c) un manejador de comunicaciones, quién se encarga de las transmisiones de datos. En el servidor existe un árbitro general que controla a todos los clientes NCSA Habanero y, un árbitro para cada una de las aplicaciones que se ejecutan en los clientes NCSA Habanero. Además, es posible extender los árbitros existentes para que lleven a cabo funciones específicas, por ejemplo en los juegos, para imponer las reglas de los mismos.

Respecto de la infraestructura provista por NCSA Habanero para construir aplicaciones colaborativas, existen dos modalidades de trabajo: a) mediante un asistente (wizard) que permite automáticamente transformar applets Java en hablets e, b) importando la librería de clases definidas en NCSA Habanero y, construir aplicaciones a partir de dicha API [7] [8].

NCSA Habanero además de estar escrito en JAVA, es orientado a aplicación, por lo tanto usando NCSA Habanero es posible acceder directamente a los datos almacenados en la estación de trabajo local así como a los almacenados en el servidor. Sin embargo, bajo este diseño, no es posible actualizar dinámicamente a través del Web el ambiente NCSA Habanero ni cargar las nuevas aplicaciones, sin costo administrativo. En el diseño de NCSA Habanero se consideraron las restricciones de seguridad que tienen las applets y, se priorizaron las aplicaciones stand-alone que permiten leer y escribir sobre los dispositivos locales y que además permiten realizar conexiones arbitrarias mediante sockets.

Teóricamente NCSA Habanero es independiente de la plataforma, sin embargo NCSA Habanero es dependiente de la Máquina Virtual de Java, por lo cual está restringido a sistemas que soporten la Máquina Virtual de Java [Gosling, 1993]. Los testeos realizados usando el Sun JDK sobre Solaris y Windows son satisfactorios; los realizados sobre el MRJ v1.5 sobre Macintosh son relativamente satisfactorios (tiene algunos bugs). Sobre SGI y HP con sus respectivas Máquinas Virtuales funciona bien. Otra restricción que tiene NCSA Habanero respecto de la independencia de la plataforma es el uso de librerías nativas por parte de las aplicaciones que se ejecutan sobre NCSA Habanero. Estas librerías deben compilarse para cada plataforma. Algunas aplicaciones requieren de librerías nativas (Audio Chat y HDF) y actualmente no hay implementaciones de algunas librerías nativas para Macintosh.

TANGO

TANGO fue desarrollado en NPAC (Northeast Parallel Architectures Center, Syracuse University, New York) [Beca, 1997] [3]. Es una plataforma integrada que permite implementar ambientes colaborativos basados en Web. En la actualidad los navegadores de Internet se han convertido en herramientas de software familiares para la mayoría de los usuarios de computadoras. Asímismo, la popularidad alcanzada por Internet sumada a la simplicidad y facilidad de uso de estos navegadores y, fundamentalmente a la aparición del lenguaje Java [Gosling, 1993] y todas las tecnologías asociadas, hacen posible el diseño de una aplicación que tome ventaja de los factores mencionados anteriormente.

TANGO es un sistema abierto y extensible que provee una infraestructura tecnológica para construir sistemas colaborativos. Fundamentalmente está basado en las tecnologías Web y en la la Máquina Virtual de Java [Gosling, 1993].

TANGO permite crear espacios de información compartida, coordinando las aplicaciones que intercambian esta información y además proveyendo visiones independientes de la información relacionada. Es posible presentar la misma información de diferentes modos de acuerdo a la performance y capacidad gráficas de las estaciones de trabajo sobre las que se ejecutan las aplicaciones.

La alta integración con el Web y el uso del lenguaje Java reduce en TANGO el costo administrativo de la distribución de las nuevas versiones de las aplicaciones colaborativas y del núcleo del sistema. Esto hace que el sistema sea muy fácil de instalar, de usar y de extender. Por otro lado, al ser un sistema que se ejecuta sobre el Web, TANGO depende de las limitaciones de performance impuestas por los navegadores.

A pesar de estar construído sobre un ambiente sin estado como es el Web, TANGO es un sistema con estado. Provee todas las funciones de los sistemas colaborativos sincrónicos incluyendo el manejador de sesiones, la distribución de datos y eventos, un "floor control" flexible y múltiple y, niveles de seguridad configurables. No existen restricciones sobre el número de sesiones que se ejecutan concurrentemente ni sobre el número de usuarios o aplicaciones colaborativas.

TANGO también soporta colaboración asincrónica. Todas las interacciones de las sesiones se registran en una base de datos que puede ser consultada permitiendo reveer tanto los datos como los eventos generados en una determinada sesión.

A pesar que TANGO y la mayoría de las aplicaciones están escritas en Java, no existen restricciones en cuanto al lenguaje de programación a utilizar en la implementación de las aplicaciones colaborativas.

TANGO integra aplicaciones "standalone" y basadas en Web, proveyendo una interfaz uniforme para la comunicación con sus instancias en las máquinas remotas. Las aplicaciones pueden estar escritas en cualquier lenguaje de programación siempre asumiendo que la comunicación se realiza a través de sockets.

TANGO permite ejecutar y controlar aplicaciones colaborativas en el ambiente provisto por el navegador de Internet. Provee mecanismos para el control de la sesión (una sesión es un grupo de instancias de aplicaciones que se ejecutan juntas en un modo colaborativo intercambiando información y compartiendo comportamiento) incluyendo autenticación de usuarios, arranque y finalización de una sesión, seguimiento de las actividades de los participantes y cambios de privilegios de los mismos. Dentro de una sesión hay un usuario distinguido considerado el "master" de la sesión que tiene privilegios especiales para controlar el comportamiento de la aplicación y/o controlar el acceso de otros usuarios a esa sesión. Estos privilegios dependen de cada aplicación. Unicamente las aplicaciones que pertenecen a la misma sesión pueden comunicarse. Esta comunicación está basada en el pasaje de mensajes. Existen dos tipos de mensajes: los mensajes de control y los mensajes de las aplicaciones. Los mensajes de control son generados por el sistema TANGO y entre otros permiten realizar el loggin de un usuario al sistema, establecer sesiones, arrancar aplicaciones, etc. Los mensajes de las aplicaciones sirven para realizar la comunicación entre las aplicaciones de usuario. La estructura de estos mensajes depende de cada aplicación y no son interpretados por el sistema de comunicación. El protocolo de comunicación específico de la aplicación está definido por la aplicación misma utilizando la API provista por TANGO.

En TANGO los mensajes de control y de aplicación se almacenan en una base de datos junto con la fecha, hora y la fuente emisora. De esta forma, el acceso a esta información posibilita reveer toda la actividad del sistema en forma asincrónica.

La arquitectura del sistema TANGO está formada por un demonio local, un servidor central, una aplicación de control y aplicaciones de usuario (standalone y appplets). El demonio local mantiene la comunicación entre las aplicaciones de usuario, las applets y el servidor central. Permite arrancar las aplicaciones locales, enviar mensajes entre aplicaciones que se ejecutan en el mismo nodo y provee cierto nivel de funcionalidad que normalmente no está disponible para las applets Java. Está implementado como un plugg-in que debe ser cargado en el navegador.

El servidor central es el elemento principal de comunicación. Todos los demonios locales se comunican con él. Sus principales tareas son: mantener el estado de todo el sistema (participantes, aplicaciones sesiones, etc.), rutear los mensajes entre las aplicaciones que participan en cada sesión y registrar todos los eventos en una base de datos.

Las aplicaciones locales (todas las aplicaciones de usuario que se ejecutan "standalone") pueden estar escritas en cualquier lenguaje de programación y se comunican con el demonio local usando sockets. Las applets Java son también aplicaciones de usuario escritas en Java y cargadas desde la red a partir de un servidor HTTP y, ejecutadas en el ambiente del navegador. La comunicación con el servidor central se realiza a través del demonio local mediante la invocación de funciones.

La aplicación de control es una aplicación especializada que actúa como interfaz de usuario. Se usa para arrancar aplicaciones locales o remotas, crear y unir sesiones, finalizar aplicaciones y conectarse al sistema. Esta aplicación también se comunica con el servidor central a través del demonio local y, puede enviar mensajes del sistema.

TANGO provee una API que permite construir aplicaciones colaborativas que se ejecutan en el sistema TANGO. Mediante esta API todo el pasaje de mensaje y el ruteo de eventos son completamente transparentes para el programador de la aplicación. Existen dos categorías de API: a) TANGO Core API usada por una aplicación para comunicarse con otros participantes de la misma aplicación (es decir, con otras instancias de la misma aplicación) bajo la infraestructura colaborativa brindada por el sistema y, b) TANGO CA API que permite a las aplicaciones acceder a información actual del sistema TANGO tal como sesiones, participantes, modos, identificaciones, listas de aplicaciones, etc. Actualmente se dispone de APIs para diferentes lenguajes como Java, C/C++ y JavaScrip.

ILU (Inter-Languages Unification)

ILU fue desarrollado por la Cía. Xerox Corporation [1]. Como se mencionó anteriormente provee una infraestructura de software que facilita la construcción de aplicaciones orientadas a objetos distribuídos sobre plataformas heterogéneas.

ILU es un sistema que permite construir desde librerías de clases hasta sistemas distribuídos. Las aplicaciones que se construyen con ILU pueden ser multilenguaje. Esto es, es posible definir módulos escritos en múltiples lenguajes de programación, que conformen una misma aplicación y vincularlos a través de ILU.

Responde a los estándares fijados por RPC[5], RMI [6] y CORBA[4]. En particular ILU fue una de las primeras implementaciones de sistemas del tipo CORBA, difiere del estándar en el lenguaje de definición de interfaces (o módulos). El OMG IDL (Interface Description Language, de CORBA) y el ISL (Interface Specification Language, de ILU) tienen una compatibilidad casi completa en el código fuente, pero no en el código binario.

Básicamente, la forma de trabajo en ILU consiste en generar una interfaz escrita en ISL (Interface Specification Language - lenguaje de definición de interfaces orientado a objetos-) en donde se especifican las clases y la funcionalidad exportable por la interfaz. Luego se ejecutan herramientas ILU que generan stubs y skeletons en el lenguaje de programación en el cual se implementará el módulo. Estas herramientas automatizan la traducción al lenguaje de programación específico y todo lo relacionado a la comunicación de datos. El código del stub y del skeleton contiene todo el soporte ILU requerido y, se vincula con el código de la aplicación escrito en el lenguaje de programación específico.

Por ser ILU orientado a objetos, es posible construir jerarquías de interfaces. De esta manera, es posible tener una jerarquía de clases, aún si la aplicación está implementada en un lenguaje de programación no orientado a objetos como por ejemplo C.

La versión ILU 2.0 provee un soporte para los lenguajes Ansi C, C++, Java, Python, Common Lisp, Modula 3, Perl5. ILU fue instalado en distintas plataformas de software: UNIX (SunOS, Solaris, HP-UX, AIX, OSF, IRIX, FreeBSD, Linux, LynxOS, SCO Unix, etc.) y, MS-Windows (3.1, 95, NT). Uno de los principales objetivos de ILU es maximizar la compatibilidad con los sistemas éstandares que existen actualmente. Para ello, ILU provee el soporte necesario para el uso del lenguaje de descripción de interfaces de CORBA (OMG CORBA IDL), con lo cual ILU se transforma en un sistema CORBA. Asimismo, ILU provee una implementación propia del ONC RPC, con lo cual es posible describir y usar servicios RPC

como objetos ILU y, además incluye una implementación propia de HTTP que permite implementar browsers y servidores web.

TANGO y NCSA Habanero - Comparación

A partir de la descripción de los sistemas anteriores es posible notar una gran similitud entre dos de ellos: TANGO y NCSA Habanero. ILU es más abarcativo dado que permite no solo construir aplicaciones colaborativas sino distribuídas (algo más general).

En la siguiente tabla se compara TANGO y NCSA Habanero, citando las principales características de cada uno de ellos en la siguiente tabla:

NCSA Habanero	TANGO
Basado en aplicación	Basado en Web
Escrito en Java	Escrito en Java
Se ejecuta sobre cualquier plataforma de hardware y software que soporte la Máquina Virtual de Java	Se ejecuta sobre cualquier navegador "Java-enabled"
Actualmente, las aplicaciones colaborativas son aplicaciones Java	Actualmente, las aplicaciones colaborativas son applets Java
Basado en el modelo cliente-servidor	Basado en el modelo cliente-servidor
Modelo de interacción es broadcast universal	Modelo de interacción es broadcast universal
Se provee colaboración sincrónica	Se provee colaboración sincrónica y asincrónica
Todas las instancias de una misma aplicación colaborativa tienen los mismos privilegios	Para una misma aplicación se define una instancia "master" y las restantes son "slaves"
Las nuevas versiones y aplicaciones, deben instalarse y configurarse manualmente	Distribución dinámica de aplicaciones y versiones
Alto costo de administración del sistema	Bajo costo de administración del sistema
Provee un asistente para transformar aplicaciones monousuario en colaborativas	-----
Provee una API para la construcción de las aplicaciones colaborativas	Provee una API para la construcción de las aplicaciones colaborativas
------	Limitado por la performance impuesta por el navegador sobre el que se ejecuta
Se provee mayor nivel de seguridad	Nivel de seguridad impuesto por el navegador
Acceso a los recursos de cómputo locales de la estación de trabajo -no vía red-	-----

Como se mencionó antes, ILU es un sistema que abarca no sólo las aplicaciones colaborativas sino las distribuidas en general. Esto permite construir sistemas que pueden tener características más convenientes y adecuadas al tipo de comunicación requerida por la aplicación. Por otro lado, cabe señalar que en ILU debe ser programado tanto el modelo de comunicación como la aplicación en si misma, etc. Las únicas herramientas provistas son compiladores y debbuggers.

Conclusión

Con este trabajo se analizaron y evaluaron diversos sistemas existentes para la construcción de aplicaciones colaborativas. Se estudiaron y analizaron los sistemas NCSA Habanero versión 2.0 beta 1, TANGO e ILU versión 1.8. De este análisis se concluye que para

construir aplicaciones donde el modelo de interacción necesario se corresponde con un modelo de broadcast universal, la utilización de TANGO o de NCSA Habanero es más adecuada. Ambos proveen herramientas que facilitan la construcción de las aplicaciones colaborativas e inclusive Habanero provee un asistente para este trabajo. En ILU siempre es necesario programar el modelo de interacción y no cuenta con herramientas que automaticen el modelo de comunicación requerido. Como contrapartida en ILU una vez programado el modelo de interacción, éste puede ser reusado en otras situaciones.

NCSA Habanero y TANGO son más convenientes que ILU debido a la rápida implementación de aplicaciones colaborativas y el uso fácil, sin embargo las aplicaciones deben ejecutarse dentro de un ambiente propietario (en el caso de NCSA Habanero) o de un navegador de Internet (en el caso de TANGO) y requerir de un esquema de comunicación específico (broadcast universal). Por otro lado, ILU es extensible ya que no está limitado a una única modalidad de interacción, es más abarcativo, permite integrar módulos implementados en diferentes lenguajes, se abstrae de la plataforma de software, etc., pero para construir una aplicación donde se requiere de un modelo de interacción de broadcast universal, es necesario realizar muchos pasos de programación. Además, como ya se ha mencionado, ILU no cuenta con herramientas para automatizar estos procesos de generación de aplicaciones colaborativas (debe programarse el modelo de comunicación, la aplicación, etc.). Las herramientas provistas consisten en compiladores y debbuggers.

La experiencia descripta en este artículo se basó en el actual estado de desarrollo de las versiones analizadas. Actualmente, se está profundizando en modelos de interacción que no coincidan exactamente con el modelo de comunicación correspondiente al broadcast universal, como también evaluando otras infraestructuras similares a las mencionadas aquí.

Referencias Bibliográficas

[Gosling, 1996]: Gosling, B. Joy, G. Steele. *The JAVA Language Specification.* Addison Wesley, Sunsoft JAVA Series, 1996.

[Beca, 1997]: Beca L., Cheng G., Fox G., Jurga T. *TANGO – a Collaborative Envoronment for the world Wide Web.* White Paper, NPAC (Northeast Parrallel Architectures Center), Syracuse University, New York, USA, 1997.

[Hsu, 1993]: Hsu, Lockwood. *Collaborative Computing.* Publicado en la Revista Byte. Edición de Marzo de 1993.

[Ellis, 1993]: Ellis, Gibbs, Rein. *Groupware. Some Issues and Experiences.* Publicado en [Baecker, 1993].

[Baecker, 1993]: Baecker, Ronald M. *Readings in Groupware and Computer-Supported Cooperative Work: Assisting Human-Human Collaboration.* Morgan Kaufmann Publishers, 1993.

[Stefik, 1993]: Stefik, Bobrow, Foster, Lanning, Tatar. *WYSIWIS Revised: Early Experiences with Multiuser Interfaces.* Publicado en [Baecker, 1993].

[Vaskevitch, 1995]: Vaskevitch, David. *Client/Server Strategies, Second Edition : A Survival Guide for Corporate Reengineers*. Publicado por IDG Books Worldwide, 1995

[Chabert, 1998]: Annie Chabert, Ed Grossman, Larry Jackson, Stephen Pietrowicz, Chris Seguin. *Java Object-Sharing in Habanero*. Publicado en Communication of the ACM, Junio 1998, Vol. 41, N^{o}. 6

URL's

[1] Inter-Language Unification, ftp://beta.xerox.com/pub/ilu/ilu.html

[2] NCSA Habanero, http://www.ncsa.uiuc.edu/SDG/Software/Habanero

[3] TANGO, http://trurl.npac.syr.edu/tango

[4] Object Management Group, http://www.omg.org

[5] Remote Procedure Call (RPC) Reference Manual SR-2089 9.0, http://hpccsrv.gsfc.nasa.gov:8080/library/all/SR-2089

[6] Java™ Remote Method Invocation Specification, http://java.sun.com/products/jdk/1.2/docs/guide/rmi/spec/rmiTOC.doc.html

[7] NCSA Habanero, Class Hierarchy, http://www.ncsa.uiuc.edu/SDG/Software/Habanero/API/packages/tree.html

[8] Habanero 2.0 API Overview, http://www.ncsa.uiuc.edu/SDG/Software/Habanero/API/Habanero_overview.html