Java y Software Libre

Álvaro Sánchez-Mariscal (`mariscal@javahispano.org`)

basado en los trabajos de

Alberto Molpeceres (`al@javahispano.org`)
Martín Pérez (`martin@javahispano.org`)

javaHispano
`http://www.javahispano.org`

4 de abril de 2003

Contenido

- Evolución de Java: desde 1995 a la actualidad.
- El JCP y las especificaciones.
- Algunos datos estadísticos sobre el JCP.
- Los peligros de una posible liberación.
- Ejemplo: un entorno totalmente libre.
- Dudas y preguntas.

Java, un nuevo lenguaje

- Nace en 1995.
- Lenguaje de programación propietario de SUN.
  - Sin embargo, SUN era consciente de que era necesario implicar a cuantas más empresas y desarrolladores fuese posible.
- Problemas:
  - La propiedad del software será siempre de SUN y/o sus licenciatarios.
  - No se permitían implementaciones libres de las librerías (`java.*`, `javax.*` y `sun.*`).
  - Se prohibía distribuir el JDK o el JRE de SUN desde sitios externos a la propia SUN.

- Aparecen ya grupos de software libre, como *Kaffe* y *Japhar*.
- Esta situación era muy negativa para el mundo del software libre.
  - Sin embargo, mucha gente estaba de acuerdo con esta situación.
    - Así se favorece un desarrollo uniforme y se protege la plataforma de versiones devaluadas (kit de desarrollo de Microsoft).

La creación del JCP

- Poco a poco Java se iba extendiendo cada vez más.
- Comenzaban a surgir numerosas tecnologías: EJB, Servlets/JSP, etc.
- Surgió la necesidad de estandarizar las especificaciones.
  - Hacía falta que un organismo imparcial controlara los distintos desarrollos.
- Primera opción: la ISO.
  - Descartada. SUN decía que la excesiva burocracia de estos organismos acabaría con el dinamismo de Java.
  - Ejemplo: CORBA.
  - Esta decisión provocó muchas críticas.
- Por ello, SUN decide crear su propio organismo: el JCP.

El JCP: Java Community Process

- Creado el 8 de diciembre de 1998.
- Organismo encargado de asegurar la evolución de Java.
- Controla la creación, evolución y aprobación de las especificaciones.
- Formado por más de 500 empresas y particulares del mundo de la informática y las telecomunicaciones: SUN, IBM, Oracle, HP, Motorola, Nokia, BEA, Siemens, Fujitsu, Borland, SAP, Ericcson, etc...
  - Como era de esperar, Microsoft no ha querido formar parte.

Estructura del JCP

- Dos comités ejecutivos.
- Uno encargado de J2ME, y otro encargado de J2SE/J2EE.
- Formado cada uno por 16 miembros.
- Una parte de ellos son elegidos mediante votación pública por todos los miembros del JCP. El resto son elegidos por los licenciatarios más importantes de SUN.
- Éste último tipo debe ser ratificado en votación pública por todos los miembros del JCP.

¿Quién puede participar?

- Cualquier persona, organización (con o sin ánimo de lucro) o empresa.
- Precio:
  - Entidades comerciales: 5000\$.
  - Organizaciones académicas y sin ánimo de lucro: 2000\$.
  - Personas individuales: 0\$.
  - Licenciatarios de SUN: 0\$.

Java Specification Participation Agreement (JSPA)

- Es el contrato que hay que firmar para ser miembro del JCP.
- Dura un año y es renovable (pagando las cuotas).

Individual Expert Participation Agreement (IEPA)

- De esta forma no es necesario convertirse en miembro del JCP, sino que se puede participar como experto individual en alguna tecnología.
- Estos expertos deben firmar el *Java Specification Participation Agreement for an Individual Expert Group Participant*.
- El experto se compromete a no revelar las interioridades del trabajo realizado por el grupo de expertos en que participe.

Java Specification Request (JSR)

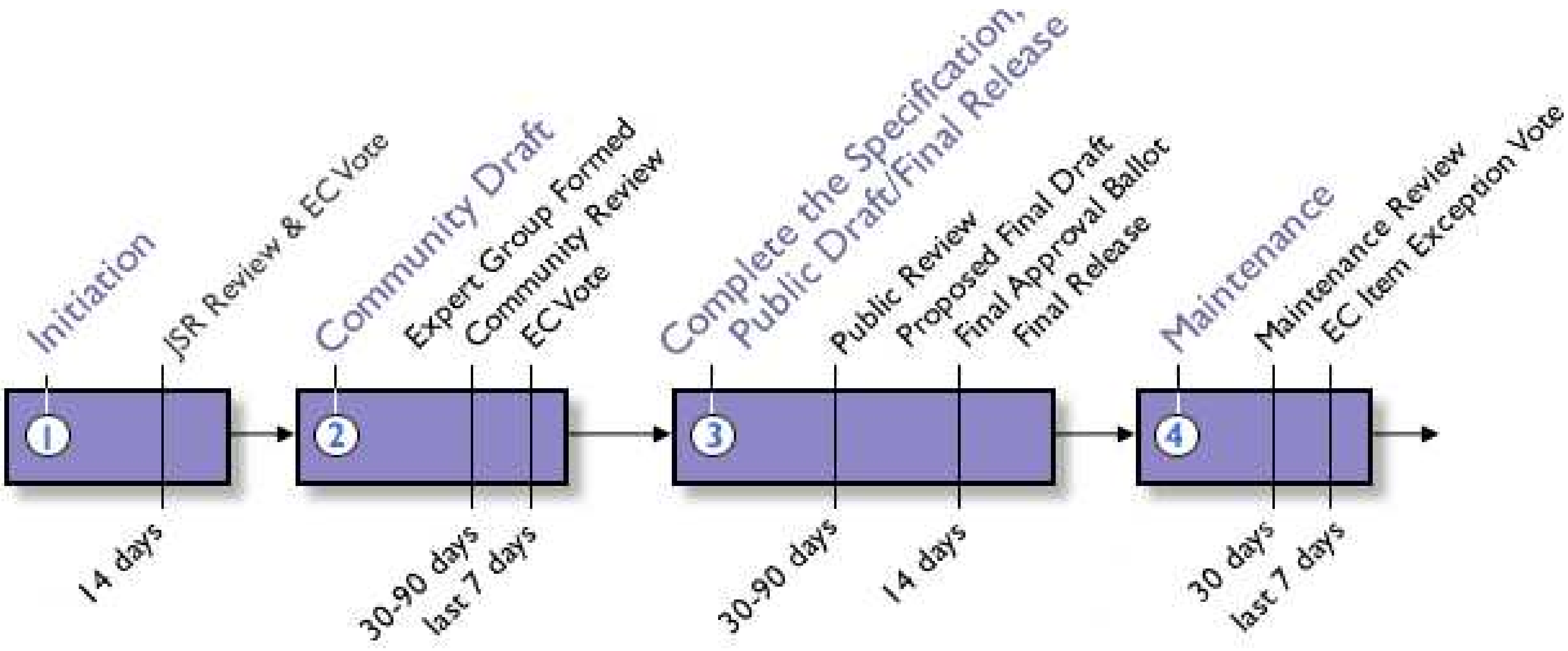
- Es un documento que una persona/empresa crea cuando ven necesaria una determinada tecnología dentro de alguna de las plataformas de Java.
- Dentro de este documento se especifica por qué es necesaria dicha tecnología, por qué no se pueden abordar los problemas que soluciona con las tecnologías existentes, cómo afectará esta tecnología a otras tecnologías existentes.

Umbrella JSR (UJSR)

- Son JSR's en los que SUN tiene privilegios especiales.
- Hay 3 UJSR's, y son los JSR's que definen las plataformas de Java: J2ME, J2SE y J2EE.
- Además de estos privilegios, SUN cuenta con otros mecanismos de protección hacia el exterior:
  - Tiene propiedad de los dominios `java.*` y `javax.*`. Cualquier cambio en esas jerarquías debe ser gestionada a través del JCP.
  - Estos *paraguas* evitan cualquier cambio en el lenguaje o en la máquina virtual.
  - Además, no se puede realizar ningún cambio sobre *Java Native Interface, JNI* ni modificar ninguno de los paquetes distribuídos con la plataforma estándar, J2SE.

- Estas medidas no son muy bien vistas desde el exterior.
- SUN argumenta que son necesarias para proteger las plataformas de alteraciones indebidas, puesto que son las especificaciones más importantes.
- Permitir cambios en estas especificaciones pondría en peligro aspectos tan importantes de Java como la compatibilidad hacia atrás o la portabilidad.
- Asimismo, permitir desarrollos bajo `java.*` y `javax.*` podría dar lugar a diferentes versiones del lenguaje que no harían más que confundir al usuario o incluso poner en duda una vez más la multiplataforma.
  - Ejemplo: Microsoft Visual J++.

El proceso de desarrollo en el JCP



- **Envío de la especificación:**
  - Lo primero que debemos hacer es proponer la especificación al comité que corresponda del JCP.
  - Tras un par de semanas, el comité aprobará o rechazará nuestra propuesta.
- **Revisión de la comunidad:**
  - En esta fase, los miembros del JCP (y sólo ellos) pueden revisar la especificación y aportar sus comentarios.
  - Esta fase dura de 1 a 3 meses. La última semana después de cerrarse la revisión pública, se procede a una votación en la que se decide si la especificación puede pasar a su fase final.

- **Revisión pública:**
  - Llegados a este punto, deberemos preparar y retocar nuestra especificación para su revisión pública.
  - En esta fase, cualquiera se la puede descargar, revisarla, y opinar para tratar de mejorar la especificación, **incluso sin ser miembro del JCP**.
  - Durante esta fase (que dura también de 1 a 3 meses) es cuando tenemos que terminar de desarrollar la implementación de referencia (RI) y el conjunto de pruebas (TCK) que se pondrán a disposición del público.
  - Si todo va bien, habrá una nueva votación por parte del comité correspondiente.
  - Si finalmente es aprobada, nuestra especificación es ya un estándar más dentro del mundo Java.

- **Fase de mantenimiento:**
  - Aquí se produce el mantenimiento de nuestra especificación. Podemos añadir y corregir características.
  - Cualquier modificación require pasar por las fases de revisión pública y por las votaciones respectivas.

Las partes de un JSR

La especificación

- Es simplemente un PDF que describe la tecnología resultado del esfuerzo realizado por el grupo de trabajo del JSR en cuestión.

El test de compatibilidad (TCK)

- Conjunto de pruebas que garantizan que una determinada implementación es compatible con la especificación a la que corresponden los tests.
- El JCP pone a disposición de los creadores de cada especificación una serie de herramientas (*Java Compatibility Test Tools*) para crear estos tests.

La implementación de referencia

- Toda especificación debe proporcionar también una implementación de referencia. Esta implementación debe haber pasado además los tests de compatibilidad.
- Las implementaciones de referencia son importantes porque...
  1. Aparecen mucho antes que las versiones comerciales/libres, por lo que permiten familiarizarse rápidamente con la tecnología.
  2. Al haber pasado los tests de compatibilidad, nos aseguramos que los desarrollos que hagamos sobre ella se puedan portar a otras implementaciones que también sean compatibles.
  3. Como suelen ser más sencillas que las comerciales, permiten una adapatación mucho más simple a las diferentes tecnologías.
- Antes de usar una implementación de referencia, debemos asegurarnos que sea apta para producción.

Problemas del antiguo JCP

- Aunque la labor del JCP parezca adecuada, analizada desde el punto de vista del software libre vemos que tiene muchas limitaciones:

  TCK's Era necesario pagar unas cuotas de varios miles de dólares para poder acceder a las *Java Compatibility Test Tools*.
  Sin embargo, el pasar no pasar estos test no impide crear implementaciones libres: es simplemente una cuestión de marketing (una *medalla*).
  Ejemplo: estándares ISO o certificados *Common Criteria* de la seguridad de los sistemas operativos.

  Derecho de veto SUN tenía el privilegio de poder anular cualquier especificación.

Cuotas Para pertenecer al JCP era necesario pagar unas cuotas, destinadas a paliar los gastos administrativos que se generan. Esto perjudica gravemente a los grupos de software libre.

Patentes Si alguna especificación tiene patentes dentro de ella, las implementaciones deberán respetarlas.

Licencias No se permite modificar la licencia de ninguna especificación, test de compatibilidad o implementación de referencia. Normalmente esta licencia estaba basada en la SCSL. Esto supone grandes inconvenientes para el desarrollo de proyectos libres.

El presente: el JCP 2.5

- Era obvio que la versión 2.1 del JCP ponía muchas trabas al desarrollo del software libre.
- Una de las organizaciones afectadas era la *Apache Software Foundation*.
- Por un lado, estaba desarrollando implementaciones de referencia de algunas especificaciones, como la de Servlets/JSP, JDOM o Axis.
- Por el otro, la fundación Apache ofrecía gratuitamente el código de estas implementaciones, con una licencia libre (la *Apache Software License*, y su proceso de desarrollo no se basaba en el JCP sino que era totalmente público.
- Todo esto iba absolutamente contra todas las licencias del JSPA, poniendo a la fundación Apache en una situación de ilegalidad.

- No obstante, las relaciones entre SUN y la fundación Apache eran magníficas y habían estado trabajando conjuntamente durante mucho tiempo.
- El apoyo de SUN a Apache era de sobra conocido, sin embargo la propia ética de la fundación obligó a ésta a exigir un remodelamiento del JCP.
- Una muestra del descontento de la fundación Apache es que durante durante los últimos 2 años emitió un NO en la fase de aprobación de **14** JSR's diferentes, casi siempre por problemas en las licencias, como por ejemplo el API de *logging* del JDK 1.4, que prácticamente ilegalizaba a *log4j* (de la fundación Apache), un proyecto estable y de gran éxito.

- Concretamente, la fundación Apache exigió que el nuevo JSPA cumpliera al menos las siguientes condiciones:
  1. Una licencia de una especificación no debería prohibir la creación de una implementación compatible. Se deberían permitir implementaciones liberadas bajo la licencia Apache u otras más libres.
     Si una especificación tiene alguna parte de código sujeta a alguna patente que no permita su uso libre, dicho código ha de estar disponible de un modo no restringido y libre de todo coste.
  2. Se debe permitir la creación de implementaciones de referencia y/o tests de compatibilidad liberados bajo una licencia libre (una vez más, como mínimo, la de Apache). Hasta la fecha eso ya se había conseguido con las especificaciones de Servlets, JSP y Taglibs.

3. Se ha de permitir que un grupo de expertos utilice foros de discusión públicos así como publicar *drafts* públicos cuando lo deseen, incluso antes de la fase de revisión pública. Sólo se protegerían las discusiones explícitamente confidenciales.
4. Se debe permitir que grupos de software libre tengan acceso a los TCK's sin tener que pagar las cuotas necesarias hasta el momento.

- SUN, consciente de la situación y consciente de la importancia de tener la colaboración de una fundación como Apache, decidió remodelar el JCP.
- Don Deutsch, vicepresidente del departamento de estándares, estrategia y arquitectura de Oracle, presidió grupo formado formado por ejecutivos del JCP, cuya misión era estudiar y revisar los derechos y deberes de todos los miembros y licenciatarios del JCP.
- El objetivo era ayudar a pequeños desarrolladores, organizaciones educativas, entidades sin ánimo de lucro, etc. a competir de forma más efectiva con las grandes empresas.
- El proceso de liberacio duró 25 meses, y se hizo público en marzo de 2002. Tres meses después, la fundación Apache y SUN llegaron a un acuerdo definitivo: el JCP 2.5 entraría en vigor en noviembre de 2002.

Ventajas del JCP 2.5

- **Cuotas**: un comité se encargará de decidir si un grupo de software libre, entidad sin ánimo de lucro, universidad, etc, es apto para entrar en el JCP o pasar los tests de compatibilidad. La idea es intentar evitar la diversificación de las implementaciones.
  Este comité está formado por 3 personas: un representante del mundo del software libre o del mundo académico, un representante de SUN y otro miembro escogido democráticamente por el JCP.
- **Derecho de veto**: SUN ha renunciado al derecho de veto.
- **Patentes**: si una especificación tiene patentes, las implementaciones no tienen por qué respetarlas.

- **Licencias**: sin duda la novedad más importante es que ahora es posible implementar cualquier especificación, test de compatibilidad o implementación de referencia **bajo cualquier licencia, por libre que sea**.

  Esto es un hito importantísimo dentro de la historia de Java, puesto que ahora se legitiman implementaciones libres de multitud de especificaciones.

El futuro

- El proceso de liberación no es inmediato. Inicialmente, todas las especificaciones iniciadas a partir del 29 de octubre de 2002 operan bajo la versión 2.5 del JCP.
- Respecto a las especificaciones ya existentes pueden darse 2 casos diferentes. Por un lado las nuevas versiones pasan a regirse bajo el nuevo JCP. Para el resto de versiones, se deja la decisión de modificar el JCP por el que se rigen al líder de la especificación.

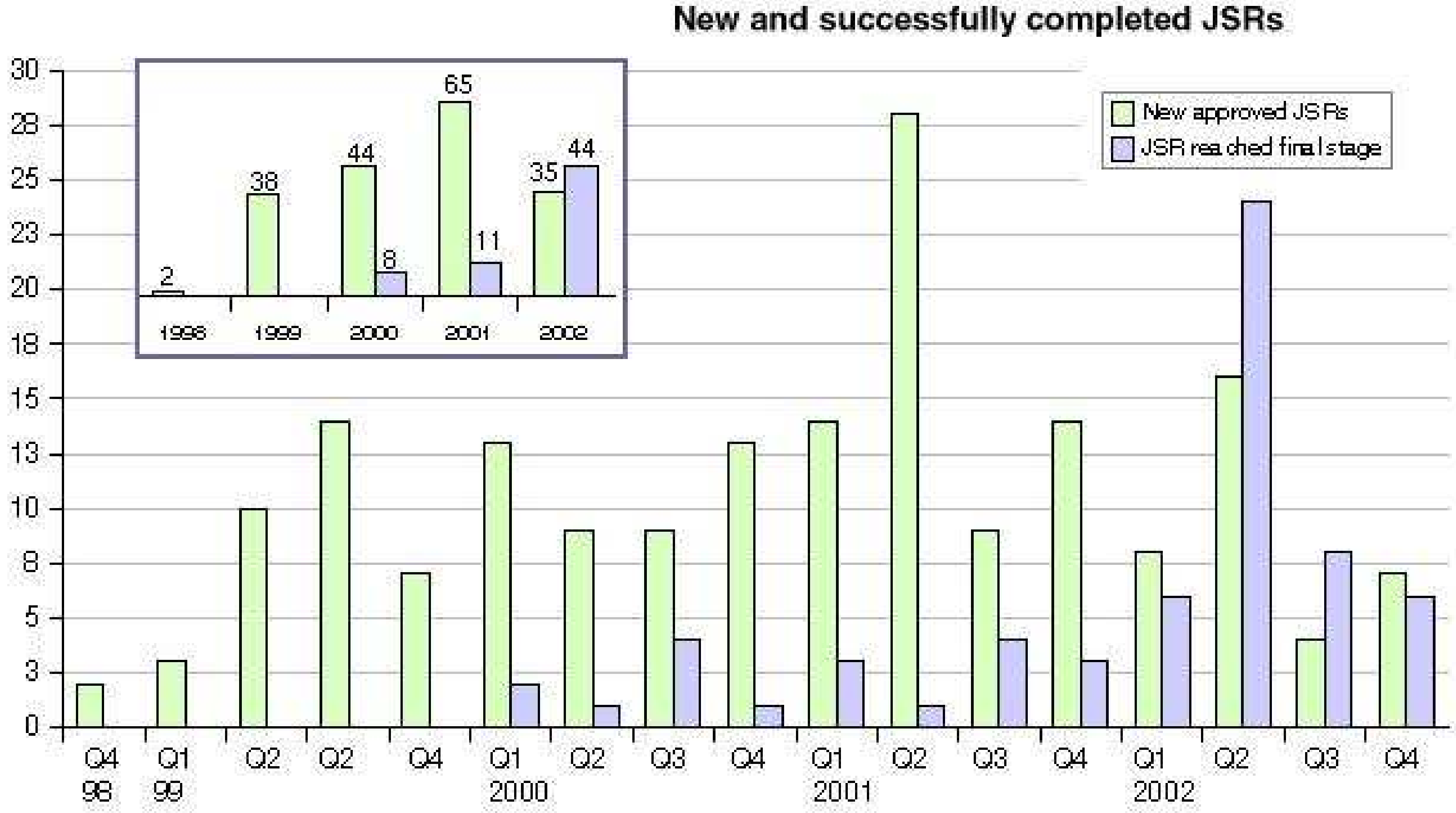
¿Qué falta por hacer?

- Aunque los cambios en el JCP han sido importantísimos, SUN afirma que aún no es suficiente.
- Esta opinión coincide con la de la mayoría de las organizaciones y empresas vinculadas al mundo Java.
- La mayor parte de la comunidad coincide que los siguientes pasos deben encaminarse a:

1. Liberar las especificaciones afectadas por los viejos JSPA's.
2. Liberar los UJSR's. La especificación del lenguaje, así como algunos UJSR's donde SUN tiene privilegios especiales, no se encuentra bajo el JCP, por lo que no se les puede aplicar estas nuevas políticas.
   La comunidad anhela la liberación de estos JSR's, aunque esta liberación no está exenta de peligros.

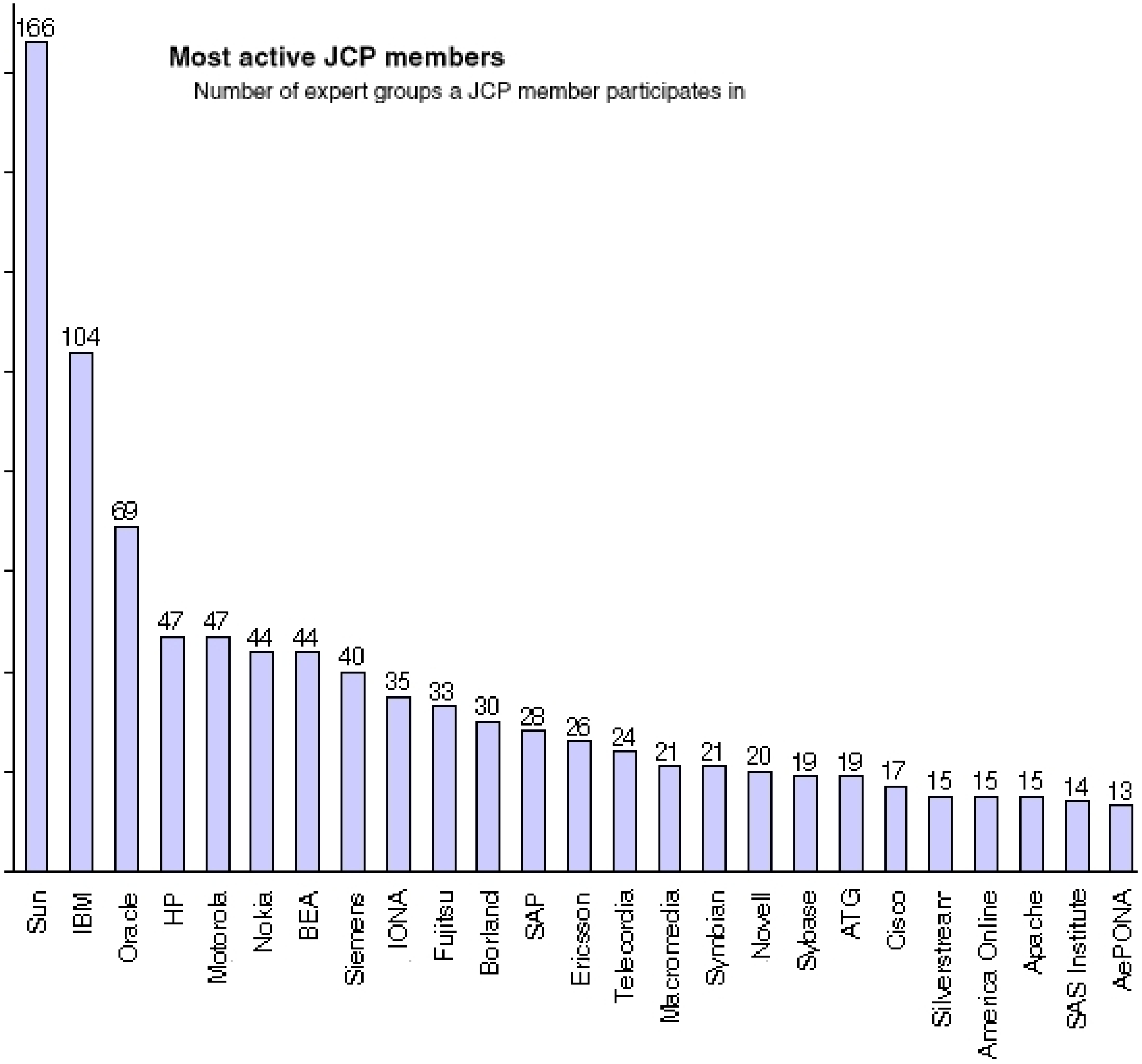
Datos estadísticos

JSR's creados y terminados

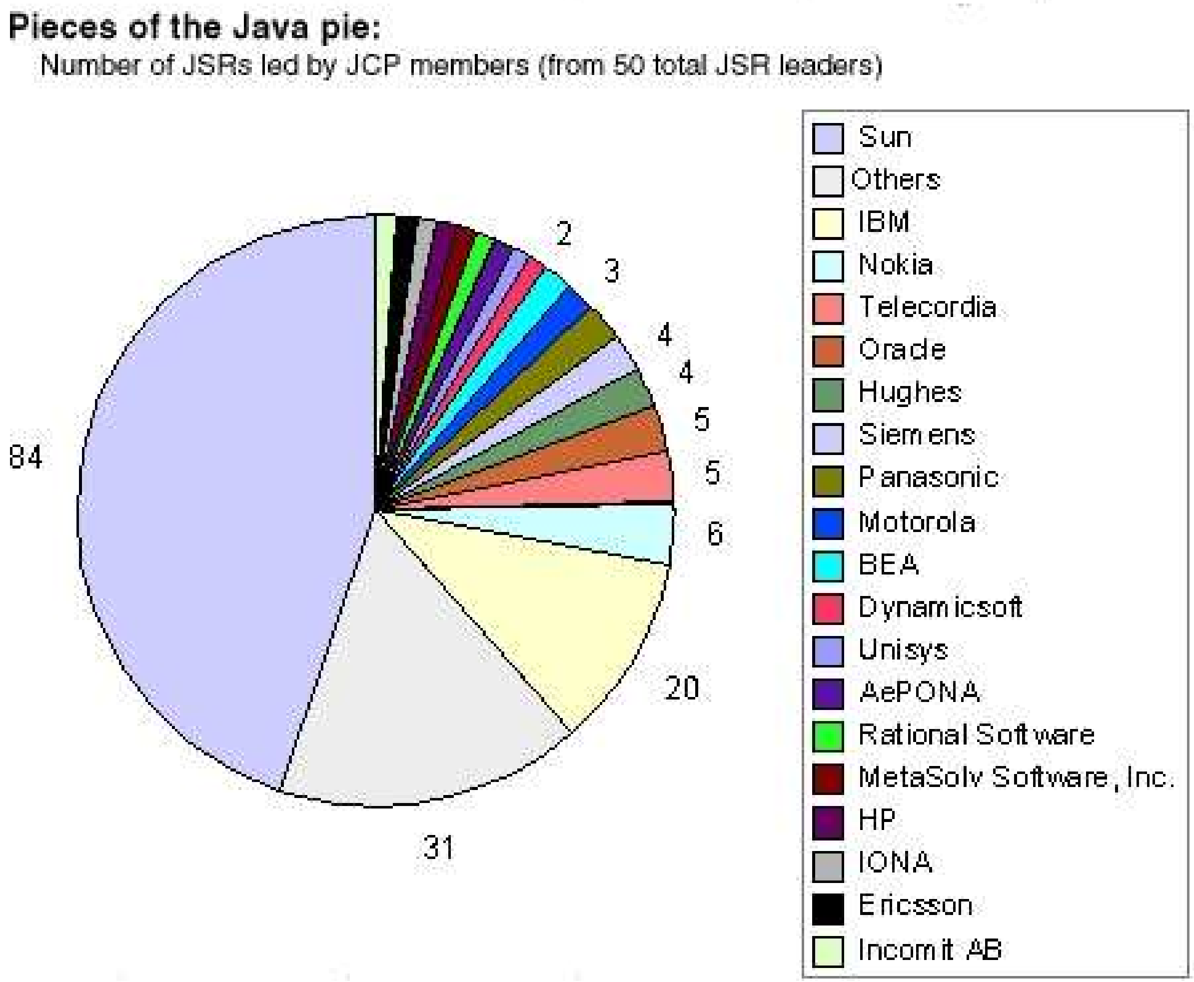


Gráficas cedidas por la revista JavaWorld para javaHispano

Miembros del JCP más activos



JSR's liderados por miembros del JCP



¿Es Java libre?

- No se puede responder a esta pregunta sin tener en cuenta Java como...
  1. Un lenguaje de programación.
  2. Una especificación de una máquina virtual.
  3. Una especificación de un formato binario (*bytecodes*).
  4. Un conjunto de clases y liberías.
- Para los tres primeros casos, **java es libre**.

- Para el último caso, aparecen varias alternativas:
  1. Utilizar el JDK de SUN. Este SDK no es libre. Su código fuente se distribuye bajo licencia SCSL.
  2. Utilizar un SDK de algún licenciatario de SUN. Es el caso del JDK de IBM. Estos JDK's tampoco son libres, ya que normalmente lo que se licencia es el código fuente.
  3. Utilizar una implementación libre. Java no es sólo un JDK, sino que también es una especificación de un lenguaje, máquina virtual y de un formato de código de bytes. Estas especificaciones son libres y cualquiera puede implementarlas. Este es el caso de gjc o Jikes.
     El único requisito es que no se llamen "Java".

Los peligros de una liberación

- Mucha gente cree que SUN debería liberar su kit de desarrollo.
- SUN opina que esta liberación pondría en compromiso a la plataforma Java. El motivo es simple: si lo hiciese, ¿cómo podría garantizar la condición de *Write Once, Run Anywhere*?.
- Muchos expertos opinan que no se podría, ya que se han dado casos como Visual J++, donde empresas como Microsoft han creado variaciones importantes del lenguaje que ponen en peligro la propiedad de multiplataforma.
- Sin embargo otros opinan que es importante la liberación del código.
- Danese Cooper, ejecutivo de SUN, dice que tras muchas reuniones entre miembros de SUN y líderes del mundo del software libre, ambos coinciden en una cosa: **la compatibilidad obligatoria y el software libre no pueden coexistir**.

- Este último concepto es muy interesante a la par que polémico.
- La licencia de los UJSR's, limita la aparición de implementaciones que se aparten de la especificación.
- No se pueden crear ni subconjuntos ni superconjuntos de dichas especificaciones.
- Liberar estos UJSR's permitiría crear ampliaciones o reducciones de lo marcado, y daría pie a que implementaciones devaluadas se hiciesen con el mercado aprovechándose de su situación (ej: Microsft).
- ¿Dónde está el equilibrio? La solución no es sencilla, y el tiempo lo dirá, aunque la opción más sensata parece el paso de la especificación del lenguaje dentro del JCP para que toda la comunidad participe en su evolución.

Ejemplo: un entorno totalmente libre

- Ejemplo de un entorno usando implementaciones libres en todas las aplicaciones.
- Supuesto: desarrollo de una aplicación web usando la plataforma empresarial, J2EE.

El entorno de desarrollo (IDE)

netBeans.org	`http://www.netbeans.org`
eclipse	`http://www.eclipse.org`

El JDK

Kaffe.org	`http://www.kaffe.org`
GNU Classpath	`http://www.gnu.org/software/classpath`
GNU Classpath Extensions	`http://www.gnu.org/software/classpathx`
gcj	`http://www.gnu.org/software/gcc/java`
Jikes	`http://www.ibm.com/developerWorks/oss/jikes/`

Modelado UML

ArgoUML	`http://argouml.tigris.org`

Servidor de aplicaciones

	http://www.jboss.org

Contenedor web

	http://jakarta.apache.org/tomcat

Sistema de desarrollo web (web frameworks)

Struts	`http://jakarta.apache.org/struts`
Velocity	`http://jakarta.apache.org/velocity`
cáñamo	`http://canyamo.sourceforge.net`

Sistema de desarrollo XML (xml frameworks)

Xerces	`http://xml.apache.org/xerces-j`
Xalan	`http://xml.apache.org/xalan-j`
JDOM	`http://www.jdom.org`
cocoon	`http://xml.apache.org/cocoon`

Otras herramientas

<APACHE ANT>	`http://ant.apache.org`
JUnit.org	`http://www.junit.org`

Dudas y preguntas

¿...?