



Plataforma JAVA

Especificaciones

Evolución

Enterprise Software

Date: Abril 2007
Autores: Jaime Cid
Revisión: 1.0



*Este documento es propiedad de Sun Microsystems.
Este documento tiene por finalidad la descripción de características técnicas para su mejor entendimiento, pero no es un documento oficial del producto, por tanto puede contener información inexacta, que se haya malinterpretado por parte de los autores.*

En caso de aclaración se puede contactar con los autores del mismo.

El documento completo podrá distribuido a terceros sin el consentimiento de Sun Microsystems.

Asimismo, alguna parte del documento puede ser copiada, fotografiada, fotocopiada, transmitida electrónicamente, almacenada en un sistema de gestión documental o reproducida mediante cualquier otro mecanismo sin la autorización previa de Sun Microsystems.

Asimismo el contenido de este documento no tiene ningún tipo de validez legal o contractual.

Índice de Contenidos

1	JAVA 5	5
1.1	Lenguaje	5
1.1.1	Genericos	5
1.1.2	Bucle for mejorado	6
1.1.3	Autoboxing y Unboxing	6
1.1.4	Enumeraciones con tipado seguro	6
1.1.5	Número de argumentos variables	6
1.1.6	Importación de miembros estáticos	6
1.1.7	Anotaciones	6
1.1.8	Menor tiempo en la fase de desarrollo	7
1.2	Entorno de ejecución	7
1.2.1	Tiempo de arranque mejorado	7
1.2.2	Rendimiento mejorado	7
1.2.3	Rendimiento mejorado en 64 bits	8
1.2.4	Parámetros de auto-configuración	8
1.2.5	Compatibilidad con J2SE 1.4.2	9
1.2.6	Soporte optimizado para sistemas de misión crítica	9
1.2.7	JMX 9	
1.2.8	Jconsole	10
1.3	Plataforma JAVA	10
1.3.1	16 Nuevas especificaciones JSR	11
1.3.2	JSR-013 Decimal Arithmetic	11
1.3.3	JSR-028 SASL	12
1.3.4	JSR-114 JDBC Rowsets	12
1.3.5	JSR-174 JVM Monitoring & Mgmt	12
2	Evolución y Versiones de JAVA	13
2.1	Nomenclatura de Versiones	13
2.2	Relación entre Java SE y Java EE	14
2.3	Política de EOL	14
2.4	Recomendación de versiones JAVA para 2007	14

1 JAVA 5

La versión 4, J2SE 1.4, supuso un gran avance en muchos aspectos en el lenguaje y la plataforma JAVA, pero la versión 5 (J2SE 5.0) disponible desde 2004, es aún mejor y en 2007 es el estándar de facto para la mayor parte de proveedores de software que utilizan JAVA como plataforma base.

Todas las nuevas soluciones empresariales de Sun en tecnologías de integración y SOA tienen como requisito mínimo la versión 5 de JAVA.

A continuación se detallan las mejoras en el lenguaje, entorno de ejecución y plataforma (APIs).

1.1 Lenguaje

La versión 5 de JAVA Standard Edition es la primera versión que introduce grandes novedades en el lenguaje, preparando y allanando el camino a algunas de las grandes mejoras en facilidades de desarrollo de la versión Java Enterprise Edition 5 (Java EE 5).

Debido a estas novedades en el lenguaje, que tienen como objetivo facilitar el desarrollo con JAVA, muchos nuevos productos, frameworks, librerías o entornos de desarrollo tienen JAVA 5 o superior como requisito.

En general los proveedores de software para soluciones de SOA, Web Services, BPEL, AJAX, Web 2.0 y otras categorías consolidadas en los últimos años siempre tendrán la plataforma Java 5 como requisito.

1.1.1 Genericos

Comprobación de tipos en tiempo de compilación no siendo necesario por tanto los castings en tiempo de ejecución. La sintaxis del código queda más limpia. Esta nueva funcionalidad es particularmente útil en el manejo de Colecciones. La nueva sintaxis usando Generics ahorrará esfuerzos en desarrollo y depuración especialmente cuando la complejidad de la aplicación se incrementa.

1.1.2 Bucle for mejorado

Facilidad en iteraciones sobre arrays y colecciones que minimiza los errores de sintaxis.

1.1.3 Autoboxing y Unboxing

Elimina la conversión manual entre tipos primitivos y los tipos envoltorio correspondientes, por ejemplo entre int e Integer. Ahora la conversión entre tipos primitivos y objetos es automática si necesidad de código de conversión.

Esto facilita la integración con las herramientas de desarrollo.

1.1.4 Enumeraciones con tipado seguro

Tipos enumerados orientados a objetos sin necesidad de patrones. La comprobación de valores válidos se produce en tiempo de compilación. Esta es una característica que los programadores Java todavía no tenían frente a C y C++.

1.1.5 Número de argumentos variables

Métodos con un número de argumentos variables sin necesidad de crear un array de objetos.

1.1.6 Importación de miembros estáticos

Acceso no calificado a miembros estáticos de un tipo sin heredar de él

1.1.7 Anotaciones

Posibilidad de crear anotaciones en el código fuente que son compiladas en los ficheros de clases JAVA de ejecución. Diferencia con los comentarios o los javadoc que nunca llegan a los ficheros .class de ejecución.

Las aplicaciones pueden acceder a la información de anotaciones en tiempo de ejecución, pudiendo tomar sofisticadas decisiones dinámicas.

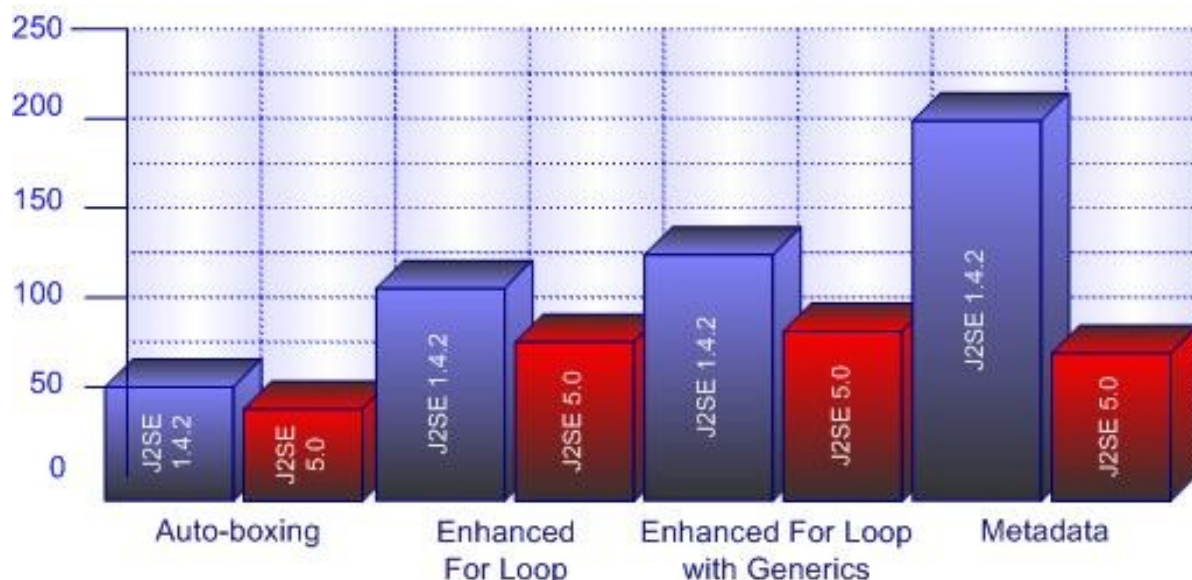
JAVA EE 5 hace uso extensivo de las anotaciones tanto para EJB 3.0 como para Web Services, y por tanto requiere J2SE 5.0.

Las nuevas arquitecturas SOA que tiene JAVA tecnología base están sacando partido de esta nueva funcionalidad de anotaciones, es por ello que la tecnología SOA de los diferentes fabricantes tenga como requisito Java 5 o superior.

1.1.8 Menor tiempo en la fase de desarrollo

Todas estas mejoras del lenguaje implican un menor tiempo de desarrollo, lo que se traduce en menores costes y agilidad en el mantenimiento de las aplicaciones.

Se puede ver un gráfico comparativo entre J2SE 1.4.2 y J2SE 5.0 que muestra las mejoras de la nueva plataforma:



1.2 Entorno de ejecución

Además de las mejoras en el lenguaje para conseguir un ciclo de desarrollo optimizado, la plataforma JAVA 5 proporciona un entorno de ejecución mejorado en diferentes áreas como rendimiento, tiempo de arranque, configuración, gestión y administración.

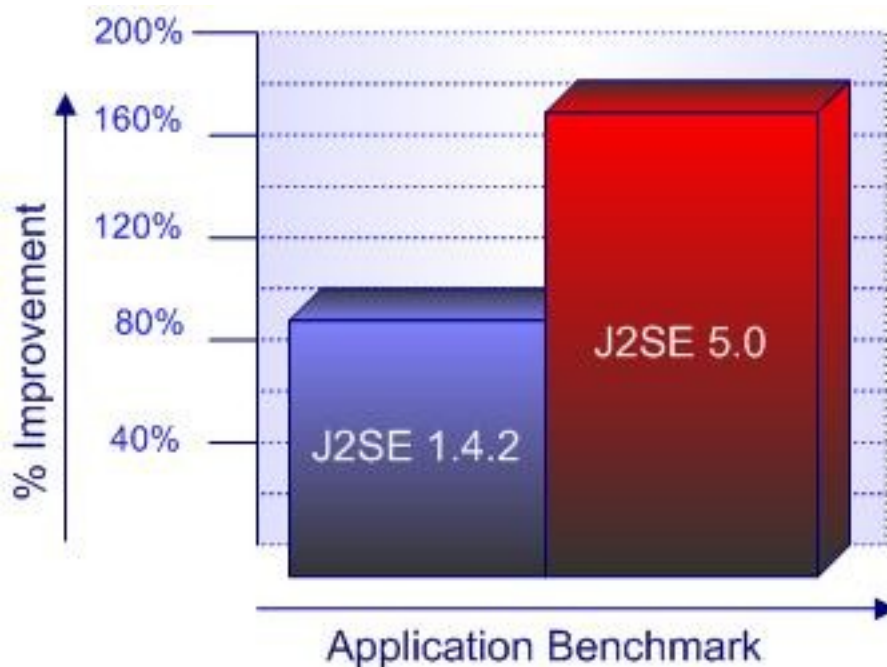
1.2.1 Tiempo de arranque mejorado

En Java 5 existe un nuevo mecanismo de compartición de clases que permiten un arranque mucho más rápido del orden del 30% tanto en aplicaciones java cliente como en servidor.

1.2.2 Rendimiento mejorado

Sólo con utilizar la plataforma JAVA 5 sin necesidad de cambiar el código ya se obtienen rendimientos mejorados y por tanto una optimización de los recursos Hardware disponibles.

Las mejoras de rendimiento habituales pueden superar el 60% como se muestra en el gráfico.



1.2.3 Rendimiento mejorado en 64 bits

Las arquitecturas de 64 bits han pasado de ser la excepción a ser la regla. Java 5 ha sido desarrollado simultáneamente a la consolidación de estas arquitecturas, y por tanto existen optimizaciones para sacar el máximo partido de las arquitecturas de 64 bits.

1.2.4 Parámetros de auto-configuración

En la versión 5 de JAVA la máquina virtual (JVM) por defecto incorpora capacidades de auto-configuración y auto-parametrización con objeto de sacar el máximo partido del Hardware disponible. Estas capacidades ahorran tiempo a la hora de optimizar las arquitecturas JAVA y obtener fácilmente el máximo rendimiento posible sin tener que modificar el código desarrollado ni tener que crear costosas pruebas de carga exhaustivas para obtener la mejor parametrización.

1.2.5 Compatibilidad con J2SE 1.4.2

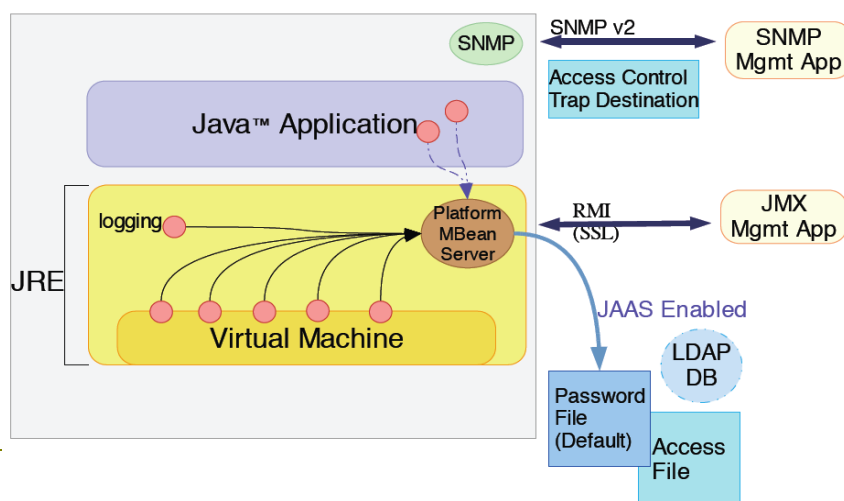
En la gran mayoría de los casos la plataforma J2SE 5.0 tiene compatibilidad binaria con J2SE 1.4.2, incluso sin tener que recompilar el código. De esta manera la inversión en código desarrollado con J2SE 1.4.2 queda protegida. Para asegurar esta compatibilidad el número de pruebas de compatibilidad se ha duplicado respecto a versiones anteriores, estando ahora en un número cercano a 200.000.

1.2.6 Soporte optimizado para sistemas de misión crítica

La versión 5 de Java ofrece grandes mejoras en las áreas de escalabilidad, calidad, y despliegue. Esta versión incluye nuevas librerías de concurrencia que facilitan la programación multi-hilo, incluye soporte de nuevas arquitecturas de 64 bits y multicore, que permiten optimizar la ejecución de Java en sistemas de misión crítica de más de 4GB de memoria RAM. Respecto a la calidad se han realizado extensivas pruebas en populares servidores de aplicaciones y aplicaciones cliente. Los tiempos medios entre fallos han pasado a niveles que superan los requeridos por los sistemas de misión crítica más exigentes. Respecto a las capacidades en el despliegue y ejecución de aplicaciones, ahora es posible extraer gran cantidad de datos de monitorización de la máquina virtual que deja de ser en muchos casos una caja negra. Incluso se pueden utilizar herramientas de terceros que soporten los nuevos estándares como JMX. La configuración y parametrización está ahora diseñada para soportar de una manera muy eficiente las demandas de los sistemas de misión crítica.

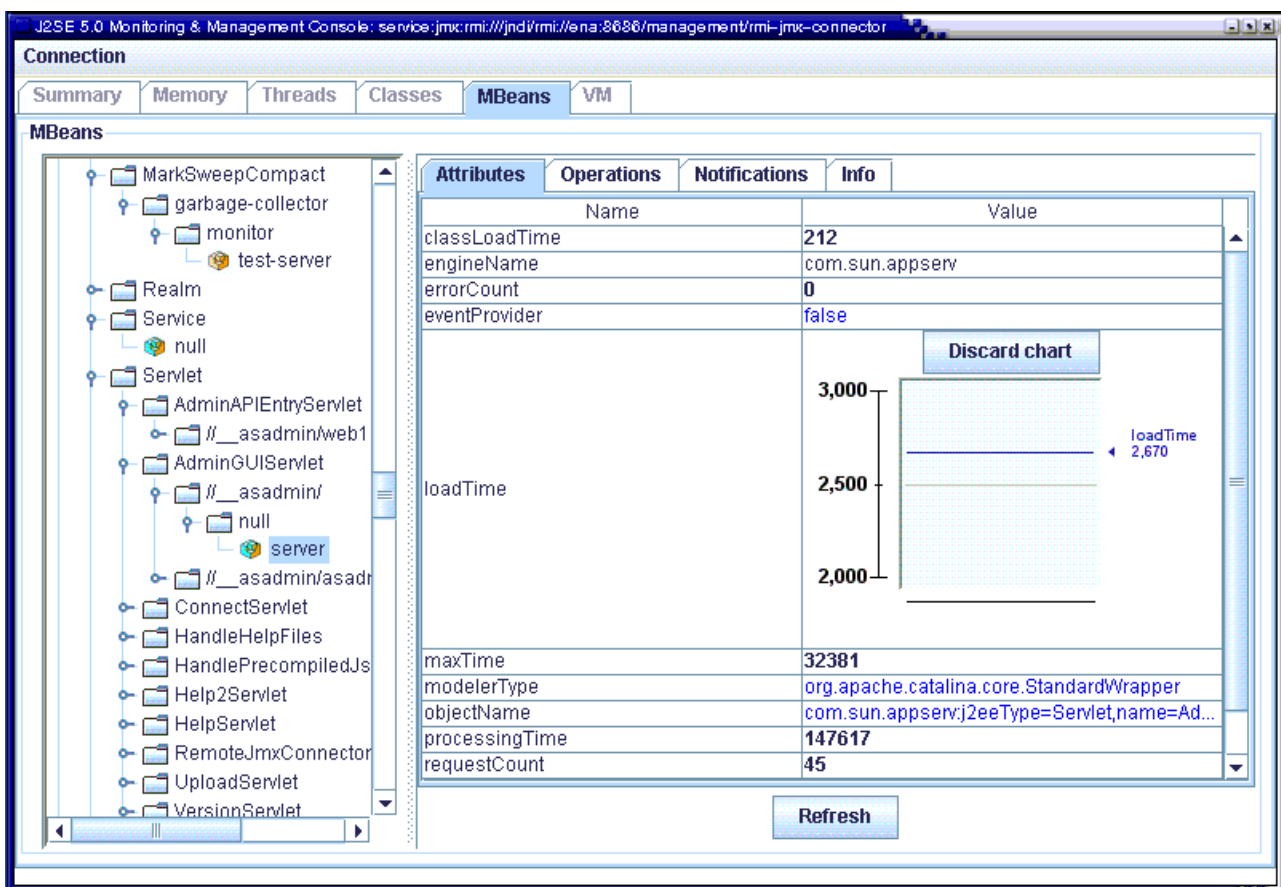
1.2.7 JMX

La plataforma Java 5 incorpora el estándar JMX para facilitar la gestión y administración de la plataforma. Se proporciona la herramienta Jconsole como interfaz de usuario de la interfaz JMX, aunque existen otras herramientas que también se pueden conectar mediante JMX. El estándar permite consultas, actualizaciones y notificaciones.



1.2.8 Jconsole

La plataforma Java 5 ofrece Jconsole como herramienta básica de gestión y monitorización JMX . Jconsole permite consultar en tiempo real el metadata de componentes gestionables y administrables incluyendo la propia máquina virtual, hilos, clases, objetos y en general cualquier objeto que implemente la interfaz Mbean.



1.3 Plataforma JAVA

Una de las claves de la plataforma JAVA es el gran número de APIs estándar que dan acceso a las funcionalidades de la plataforma. Los APIs de la plataforma JAVA se especifican mediante los grupos de trabajo multiproveedor conocidos como JCP (Java Community Process) donde participan los principales proveedores

tecnológicos de la industria del software como IBM, BEA, ORACLE, SAP, Jboss, RedHat, Google y otros. Las mejoras en el lenguaje y algunas del entorno de ejecución de JAVA 5 se han detallado anteriormente. Además existen otras. Todas ellas forman parte de los grupos de trabajo del JCP que generan especificaciones JSR (Java Specification Requests)

1.3.1 16 Nuevas especificaciones JSR

Java 5 (J2SE 5.0) incorpora 16 nuevas especificaciones:

- 003 JMX Mgmt API
- 013 Decimal Arithmetic
- 014 Generic Types
- 028 SASL
- 114 JDBC Rowsets
- 133 New Memory Model
- 160 JMX Remote API
- 163 Profiling API
- 166 Concurrency Utilities
- 174 JVM Monitoring & Mgmt
- 175 Metadata
- 200 Pack transfer format
- 201 Language updates
- 202 Classfile format
- 204 Unicode surrogates
- 206 JAXP 1.3

1.3.2 JSR-013 Decimal Arithmetic

El paquete `java.math` contiene modificaciones, como la adición en la clase `BigDecimal` de soporte para cálculo en coma flotante de precisión fija. Clases como `Math` o `StrictMath` además incluyen soporte para senos y cosenos hiperbólicos, raíces cúbicas o logaritmos en base 10. Por último, también se da soporte al uso de números hexadecimales con coma flotante.

1.3.3 JSR-028 SASL

Java 5 incorpora importantes mejoras en el área de seguridad. Se da soporte a más estándares de seguridad como SASL, OCSP, TSP y hay mejoras en la escalabilidad a través de SSLEngine, en criptografía y otras.

1.3.4 JSR-114 JDBC Rowsets

En Java 1.4 se introdujo la interfaz RowSet, que permitía pasar datos entre componentes. En Java 1.5 se han desarrollado 5 implementaciones de dicha interfaz, para cubrir 5 posibles casos de uso:

- JdbcRowSet: para encapsular un ResultSet o un driver que utilice tecnología JDBC
- CachedRowSet: desconecta de la fuente de datos y trabaja independientemente, salvo cuando esté obteniendo datos de dicha fuente o volcando datos en ella
- FilteredRowSet: hereda de la anterior, y se utiliza para obtener un subconjunto de datos
- JoinRowSet: hereda de CachedRowSet y se emplea para unir múltiples objetos RowSet
- WebRowSet: hereda de CachedRowSet para procesar datos XML.

Los Rowset facilitan la creación de aplicaciones basadas en Base de Datos especialmente la integración con entornos de desarrollo.

1.3.5 JSR-174 JVM Monitoring & Mgmt

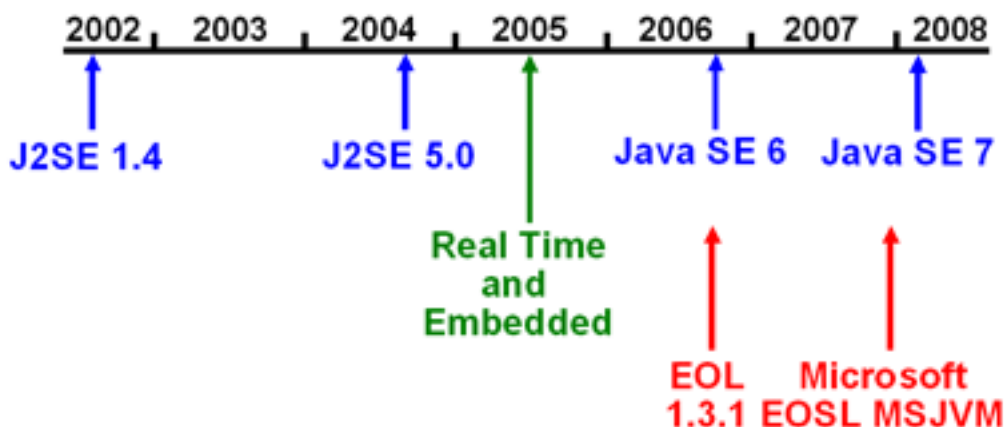
JPDA es un conjunto de interfaces utilizadas para depurar en entornos y sistemas de desarrollo. En la versión 1.5 se tienen nuevas mejoras y funcionalidades de esta herramienta. Muchas de estas nuevas funcionalidades se han añadido para adaptar las nuevas características de la versión 1.5 de Java, como los argumentos variables, imports estáticos, etc.

JVMTI (Java Virtual Machine Tool Interface) es una nueva interfaz de programación nativa para utilizar en herramientas de desarrollo y monitorización. Forma parte de las interfaces contenidas en JPDA para depuración. Permite chequear el estado y controlar la ejecución de una aplicación.

2 EVOLUCIÓN Y VERSIONES DE JAVA

La plataforma JAVA tradicional, el JDK o Standard Edition (SE), sigue un ritmo de evolución constante con la aparición de nuevas versiones aproximadamente cada dos años. La versión 6, ahora llamada Java SE 6, está disponible desde el 11 de Diciembre de 2006.

Java SE Platform Roadmap



2.1 Nomenclatura de Versiones

Lo primero que llama la atención es el cambio de nomenclatura, que en muchos casos causa confusión. Hemos pasado del nombre tradicional de J2SE 1.4.x a J2SE 5.0 (también conocido como J2SE 1.5 o JDK 1.5), y a partir de la versión 6 a Java SE 6 (equivalente a J2SE 1.6, JDK1.6 o JDK 6). Se pueden consultar algunos detalles adicionales en la [entrevista con Jean Elliot, Directora de Marketing de producto de Java SE](#).

2.2 Relación entre Java SE y Java EE

Los Servidores de Aplicaciones certificados J2EE 1.4 necesitan como mínimo J2SE 1.4, pero también pueden funcionar con J2SE 5.0. Igualmente los nuevos Servidores J2EE certificados Java EE 5, necesitan como mínimo J2SE 5.0, pero también pueden funcionar con Java SE 6. Por tanto suele ser una buena idea, en vez de migrar de un solo paso, hacerlo en dos. Primero migrar la plataforma J2SE, aprovecharse de las ventajas de las nuevas versiones y luego, sin cambiar de JDK, migrar la plataforma J2EE.

2.3 Política de EOL

A nivel de plataforma SE (Standard Edition) [la política de Sun](#) es tener activas 3 versiones, y por tanto en 2007 la versiones soportadas son J2SE 1.4.2, J2SE 5.0 y Java SE 6. [La versión 1.3.1 completó el proceso de EOL](#) cuando salió la versión 6, el 11 de Diciembre de 2006. Por tanto la versión 1.4.2 será la próxima en entrar en EOL y todos los proyectos en producción con esta versión deberían tener contemplada una actualización en 2007 o principios de 2008.

2.4 Recomendación de versiones JAVA para 2007

Evidentemente, todos los proyectos que comiencen ahora, en 2007, como mínimo deben utilizar Java SE 5.0, no sólo por las muchas novedades y mejoras técnicas, sino simplemente porqué la versión 1.4.2 quedará fuera de soporte próximamente. Otro argumento adicional es que la última versión de J2SE 1.4.2 es de 2003, y desde entonces sólo han salido parches, ninguna innovación, y estamos en el 2007. Por tanto es una plataforma que no parece muy recomendable para los productos comerciales que han salido en 2005, 2006 y 2007 y que suelen estar probados y soportados como mínimo con J2SE 5.0. Quedarse sin soporte de productos de terceros por causa de la plataforma Java puede convertirse en un serio problema. Algunos argumentos más técnicos sobre las ventajas de J2SE 5.0 se pueden encontrar en:

<http://java.sun.com/developer/technicalArticles/J2SE/5reasons.html>

<http://java.sun.com/developer/technicalArticles/releases/j2se50/MigrateToTiger.html>

Por otra parte, la versión Java SE 6 ya empieza a ser una opción a considerar seriamente. Según Java pasa a formar parte de la plataforma de aplicaciones de misión crítica, los departamentos de sistemas piden más y más herramientas y capacidades de administración y monitorización. La versión 5 ya introdujo importantes novedades como JMX y JConsole, pero la

versión 6 las supera con nuevas herramientas como jstat, jmap, jhat, y jstack que no necesitan arrancar Java en ningún modo especial. Adicionalmente se soportan plugins para JConsole y también integración con DTrace en caso de usar solaris 10. Se pueden ver detalles de estas nuevas capacidades en la [entrevista con Vasanthan Dasan](#) y en el [blog de Mandy Chung](#). Otras novedades de Java SE 6 se pueden leer en el [blog de Danny Coward](#).

[Sun Java SE 7 va a ser la primera versión construida a partir del código de OpenJDK](#), el proyecto al que Sun ha contribuido su implementación de Java, y está prevista para 2008. Además de mejoras generales en rendimiento, estabilidad y calidad de cada nueva versión, también se espera soporte directo de XML en el código, mejoras en empaquetamiento de librerías y clases (superpackages) y muchas otras que ahora mismo están en la fase de discusión pública. Aun así, lo más importante de esta versión será que al tener [licencia adicional GPL](#) podrá formar parte del core de distribuciones libres de linux como Debian, Ubuntu o Fedora lo que contribuirá a una rápida popularización.

Más información:

[Sun JDK 6 Project Website \(dev.java.net\)](#)

[Java SE 6: Top Ten Features \(Danny Coward\)](#)

[Sun JDK 7 Project Website \(dev.java.net\)](#)

[Java ya es totalmente libre: OpenJDK incluirá licencia GPL \(Jaime Cid\)](#)

[OpenJDK Project Website \(dev.java.net\)](#)

[JDK6, JDK7 and OpenJDK relationship \(sun.com/software/opensource/java/FAQ\)](#)

[OpenJDK Community Official Website](#)

[OpenJDK Presentation \(Mark Reinhold\) at FOSDEM 2007 \[PDF\]](#)