

¿Qué ofrece Autentia Real Business Solutions S.L?

Somos su empresa de **Soporte a Desarrollo Informático**.
Ese apoyo que siempre quiso tener...

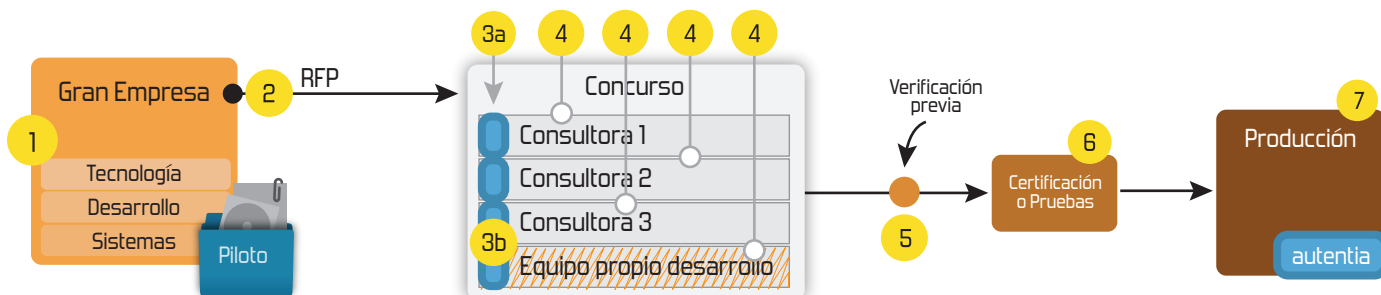
1. Desarrollo de componentes y proyectos a medida



2. Auditoría de código y recomendaciones de mejora

3. Arranque de proyectos basados en nuevas tecnologías

1. Definición de frameworks corporativos.
2. Transferencia de conocimiento de nuevas arquitecturas.
3. Soporte al arranque de proyectos.
4. Auditoría preventiva periódica de calidad.
5. Revisión previa a la certificación de proyectos.
6. Extensión de capacidad de equipos de calidad.
7. Identificación de problemas en producción.



4. Cursos de formación (impartidos por desarrolladores en activo)

Spring MVC, JSF-PrimeFaces /RichFaces,
HTML5, CSS3, JavaScript-jQuery

Gestor portales (Liferay)
Gestor de contenidos (Alfresco)
Aplicaciones híbridas

Tareas programadas (Quartz)
Gestor documental (Alfresco)
Inversión de control (Spring)

Control de autenticación y
acceso (Spring Security)
UDDI
Web Services
Rest Services
Social SSO
SSO (Cas)

JPA-Hibernate, MyBatis
Motor de búsqueda empresarial (Solr)
ETL (Talend)

Dirección de Proyectos Informáticos.
Metodologías ágiles
Patrones de diseño
TDD

BPM (jBPM o Bonita)
Generación de informes (JasperReport)
ESB (Open ESB)



[Home](#) | [Quienes Somos](#) | [Empleo](#) | [Tutoriales](#) | [Contacte](#)



CoNcept


Lanzado

TNTConcept versión 0.6 (12/07/2007)

Desde [Autentia](#) ponemos a vuestra disposición el software que hemos construido (100% gratuito y sin restricciones funcionales) para nuestra gestión interna, llamado TNTConcept (auTeNTia).

Construida con las últimas tecnologías de desarrollo Java/J2EE (Spring, JSF, Acegi, Hibernate, Maven, Subversion, etc.) y disponible en licencia GPL, seguro que a muchos profesionales independientes y PYMES os ayudará a organizar mejor vuestra operativa.

Las cosas grandes empiezan siendo algo pequeño Saber más en: <http://tntconcept.sourceforge.net/>

<p>Tutorial desarrollado por: Alejandro Perez García 2003-2007 Alejandro es Socio fundador de Autentia y nuestro experto en J2EE, Linux y optimización de aplicaciones empresariales.</p> <p>Si te gusta lo que ves, puedes contratarle para impartir cursos presenciales en tu empresa o para ayudarte en proyectos (Madrid).</p> <p>Contacta: alejandropg@autentia.com</p>	<p>NUEVO CATÁLOGO DE SERVICIOS DE AUTENTIA (PDF 6,2MB)</p> <p>www.adictosaltrabajo.com es el Web de difusión de conocimiento de www.autentia.com</p> <p> autentia real business solutions</p> <p>Catálogo de cursos</p>
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Descargar este documento en formato PDF [xmlBeans.pdf](#)

[Firma en nuestro libro de Visitas](#) <-----> [Asociarme al grupo AdictosAlTrabajo en eConozco](#)



Fecha de creación del tutorial: 2007-08-20

XMLBeans, una forma de mapear un XML en objetos Java

Creación: 18-08-2007

Índice de contenidos

- [1. Introducción](#)
- [2. Entorno](#)
- [3. Instalando XMLBeans](#)
- [4. Generando las clases Java a partir del esquema \(XSD\)](#)
 - [4.1. Pasar del DTD al XSD](#)
 - [4.2. Pasar del XSD a las clases Java](#)
 - [4.3. Definiendo el paquete para las clases generadas](#)
- [5. Ejemplo de uso de las clases generadas](#)
- [6. Conclusiones](#)
- [7. Sobre el autor](#)

1. Introducción

Hay varias formas de trabajar con documentos XML. Una de ellas es mapear el XML en objetos Java, de forma que en vez navegar por el XML con un DOM o con XPath, directamente usamos objetos con sus getters y sus setters; lo cual puede resultar muy cómodo para trabajar con un XML dentro de nuestra aplicación.

Dentro de esta técnica podríamos decir que hay dos grandes posibilidades o tecnologías (hay más de dos, pero entre todas podríamos destacar estas por ser las más extendidas y estándar):

- JAXB (Java Architecture for XML Binding): El es estándar de Java (lo podemos encontrar en el JEE 5). JSR 222 (<http://jcp.org/en/jsr/detail?id=222>).
- XMLBeans: Podríamos decir que es el estándar de "facto" por estar altamente extendido. En un principio fue un proyecto de BEA, que posteriormente donaron a la comunidad Apache (<http://xmlbeans.apache.org/>).

Ambas tecnologías son muy parecidas y hacen prácticamente lo mismo. En este tutorial no voy a entrar en el debate de cual de las dos es mejor. Para eso no tenéis más que buscar en Google "xmlbeans vs jaxb" y encontraréis multitud de referencias.

En este tutorial simplemente vamos a ver una introducción a XMLBeans para ver como podemos obtener, a partir de un DTD, las clases Java que procesan los XML que cumplen ese DTD.

Vamos a hacer un ejemplo práctico donde generaremos las clases Java para procesar los XMLs que genera el Bugzilla como resultado de una búsqueda. Estas clases nos podrán servir para "alimentar" otras aplicaciones con los datos que genera el Bugzilla.

Dejaremos para otro tutorial hacer el mismo ejemplo con JAXB (a ver si alguien se anima y lo hace ;)

2. Entorno

El tutorial está escrito usando el siguiente entorno:

- Hardware: Portátil Asus G1 (Core 2 Duo a 2.1 GHz, 2048 MB RAM, 120 GB HD).
- Sistema Operativo: GNU / Linux, Debian (unstable), Kernel 2.6.22, KDE 3.5
- Java 1.5
- XMLBeans 2.3.0

3. Instalando XMLBeans

Nos descargamos la última release de <http://www.apache.org/dyn/closer.cgi/xmlbeans/binaries>.

Y la descomprimos, por ejemplo, en el directorio /opt:

```
$ cd /opt
$ tar -xzf /directorio_donde_hicimos_la_descarga/xmlbeans-2.3.0.tgz
```

Si queremos poder ejecutar desde línea de comandos las utilidades que vienen con XMLBeans, sería conveniente poner el directorio /opt/xmlbeans-2.3.0/bin en el PATH del sistema. Por ejemplo, en Debian podemos hacer:

```
$ export PATH=$PATH:/opt/xmlbeans-2.3.0/bin
```

A partir de este momento, en la consola donde hemos ejecutado este comando, podremos invocar las utilidades que vienen con XMLBeans sin necesidad de escribir su ruta completa.

4. Generando las clases Java a partir del esquema (XSD)

Sí, has leído bien, **XMLBeans sólo es capaz de generar las clases Java a partir de un esquema (un fichero .xsd)**. Esto puede ser un problema si lo que tenemos es un DTD (como en el ejemplo que hemos planteado).

Aún así no hay que desesperar ;) lo que vamos a hacer es pasar del DTD al XSD y del XSD a las clases Java.

4.1. Pasar del DTD al XSD

El DTD que utiliza Bugzilla lo podéis ver [aquí](#).

En vez de hacer la conversión a mano, lo mejor es usar alguna herramienta que podamos encontrar por Internet. Yo he probado con estas y parece que funcionan correctamente:

- http://www.hitsw.com/xml_utilites/ - Se trata de una página web donde, de forma gratuita, le indicamos el fichero y nos devuelve el XSD equivalente. Es la opción más cómoda, ya que no tenemos que instalar nada, y lo tendremos resuelto en un momento. Esta es la opción que he seguido yo.
- <http://www.syntext.com/products/index.htm#Dtd2Xs> - Se trata de una herramienta tanto para Windows como para Linux. El XSD que genera esta aplicación es prácticamente idéntico a la primera opción que hemos comentado.

El XSD obtenido con la primera opción lo podéis encontrar [aquí](#).

Si os fijáis en el XSD, se declaran todos los elementos como tipos complejos (como en un DTD no se puede especificar el tipo de los elementos, la herramienta a intentado escoger la solución más genérica). Esto no me convence porque es una complejidad innecesaria, así que he editado a mano el fichero generado para cambiar estos tipos complejos por tipos "xs:string" es decir, por cadenas de caracteres

(para el ejemplo me va bien, pero podríais especificar el tipo que mejor se ajuste a vuestras necesidades).

El XSD simplificado lo podéis encontrar [aquí](#).

4.2. Pasar del XSD a las clases Java

Para pasar del XSD a las clases Java vamos usar el comando `scomp`, que viene con XMLBeans. Haremos algo como:

```
scomp -src src -javasource 1.5 -out bugzilla-3.0-xmltypes.jar bugzilla-3.0-simple.xsd
```

- `-src src` : con esto le estamos diciendo que queremos que nos genere el código fuente (los `.java`) en el directorio `src`. Esta opción no es obligatoria, es sólo por si luego tenéis curiosidad de ver el código que genera XMLBeans. Si no ponemos esta opción se limitará a generar un `.jar` con las clases que necesitamos.
- `-javasource 1.5` : con esta opción le estamos indicando que tipo de código Java queremos que genere. Podemos elegir entre 1.4 o 1.5.
- `-out bugzilla-3.0-xmltypes.jar` : estamos indicando el nombre del jar (por defecto `xmltypes.jar`) que se va a crear con las clases generadas y compiladas.
- `bugzilla-3.0-simple.xsd` : el nombre del fichero XSD que queremos usar para generar las clases Java.

Si todo funciona correctamente deberíamos ver en la consola algo como:

```
Time to build schema type system: 0.782 seconds
Time to generate code: 0.811 seconds
Time to compile code: 6.144 seconds
Compiled types to: bugzilla-3.0-xmltypes.jar
```

Y deberíamos tener el jar `bugzilla-3.0-xmltypes.jar` con todas las clases necesarias para procesar los XMLs generados por el Bugzilla.

4.3. Definiendo el paquete para las clases generadas

Si nos fijamos en las clases generadas (no hay más que entrar en el directorio `src`), veremos que las ha puesto en el paquete `noNamespace`. Esto no es nada conveniente, y **nunca debemos generar clases en el paquete `noNamespace`**; ya que si en un mismo proyecto tenemos mas clases generadas con XMLBeans, pertenecientes a XMLs diferentes, todas estarían dentro del mismo paquete y podrían entrar en conflicto.

El hecho de que el paquete utilizado sea `noNamespaces`, se debe a que nuestro esquema no tenía definido el `targetNamespace`. Por ejemplo, si nuestro XSD hubiera tenido el siguiente `targetNamespace`:

```
<xs:schema targetNamespace="http://autentia.com/xmlbeans/bugzilla"
  xmlns:cr="http://autentia.com/xmlbeans/bugzilla"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  ...
```

las clases se hubieran generado en el paquete:

```
com.autentia.xmlbeans.bugzilla
```

Otra forma de especificar el paquete para las clases Java, bien si no tenemos definido el `targetNamespace` o bien porque queremos generarlas en otro paquete, sería usando un fichero para configurar como se debe hacer la generación de código. Este fichero debe tener la extensión **`.xsdconfig`**.

En nuestro caso, por ejemplo, si no quisiéramos añadir el `targetNamespace` a nuestro XSD, podríamos usar el siguiente fichero de configuración (`bugzilla-3.0.xsdconfig`):

```
<?xml version="1.0" encoding="UTF-8"?>
<xb:config xmlns:xb="http://xml.apache.org/xmlbeans/2004/02/xbean/config">
  <xb:namespace uri="##any">
    <xb:package>com.autentia.xmlbeans.generated.bugzilla30</xb:package>
  </xb:namespace>
</xb:config>
```

Nótese como en el atributo `uri` del elemento `xb:namespace`, se ha puesto el valor `##any`. Este indica que, en ausencia de `targetNamespace`, se debe usar el paquete indicado en el elemento `xb:package`.

Si en vez de `##any` hubiéramos especificado un namespace, lo que estaríamos haciendo es cambiar un nombre de paquete por otro.

Para más información sobre este fichero de configuración se puede consultar: <http://dev2dev.bea.com/lpt/a/8>

Para lanzar la generación usando el fichero de configuración, lo añadiremos como último parámetro del comando `scomp`. Por ejemplo:

```
scomp -src src -javasource 1.5 -out bugzilla-3.0-xmltypes.jar bugzilla-3.0-simple.xsd bugzilla-3.0.xsdconfig
```

5. Ejemplo de uso de las clases generadas

Ahora que ya tenemos el jar con las clases generadas, ya podemos usarlo en nuestros proyectos.

Para leer uno de estos XMLs generados por el Bugzilla haríamos algo como:

```
...

// Leemos el XML y lo convertimos en objetos
BugzillaDocument doc = BugzillaDocument.Factory.parse(new File(file));

// Ahora podemos trabajar con los objetos
final List<Bug> bugs = doc.getBugzilla().getBugList();

...
```

Como se puede ver, es muy sencillo, ya que basta con ir llamando a los métodos de las clases generadas por XMLBeans.

6. Conclusiones

Gracias a XMLBeans hemos visto como en pocos minutos podemos generar un jar que nos permite operar con XMLs que sigan un determinado esquema, tanto para leerlos, como para escribirlos, ...

El conocimiento de la existencia de este tipo de herramientas nos puede ahorrar mucho tiempo y quebraderos de cabeza.

7. Sobre el autor

Alejandro Pérez García, Ingeniero en Informática (especialidad de Ingeniería del Software)

Socio fundador de Autentia (Formación, Consultoría, Desarrollo de sistemas transaccionales)

<mailto:alejandropg@autentia.com>

Autentia Real Business Solutions S.L. - "Soporte a Desarrollo"

<http://www.autentia.com>



This work is licensed under a [Creative Commons Attribution-NonCommercial-No Derivative Works 2.5 License](https://creativecommons.org/licenses/by-nc-nd/2.5/).
[Puedes opinar sobre este tutorial aquí](#)



Recuerda

que el personal de [Autentia](#) te regala la mayoría del conocimiento aquí compartido ([Ver todos los tutoriales](#))

¿Nos vas a tener en cuenta cuando necesites consultoría o formación en tu empresa?

¿Vas a ser tan generoso con nosotros como lo tratamos de ser con vosotros?

info@autentia.com

Somos pocos, somos buenos, estamos motivados y nos gusta lo que hacemos

Autentia = Soporte a Desarrollo & Formación



[Autentia S.L.](#) Somos expertos en:
J2EE, Struts, JSF, C++, OOP, UML, UP, Patrones de diseño ..
 y muchas otras cosas

Nuevo servicio de notificaciones

Si deseas que te enviemos un correo electrónico cuando introduzcamos nuevos tutoriales, inserta tu dirección de correo en el siguiente formulario.

Subscribirse a Novedades	
e-mail	
	<input type="button" value="Enviar"/>

Otros Tutoriales Recomendados ([También ver todos](#))

Nombre Corto	Descripción
JOX Mapping entre JavaBeans y XML	En este tutorial se realiza una presentación de una de las muchas APIS que nos facilitan esta tarea, de mapeo entre un documento XML y un JavaBean
Schemas XML. Introducción esquemas XML	Los esquemas XML (schemas XML) son una evolución natural de las DTDs. Os mostramos como empezar con esta tecnología.
XML y XSL en Cliente	En este tutorial os enseñamos como formatear documentos XML directamente en vuestro navegador a través de Plantillas XSL. En cursos sucesivos veremos como hacerlo en el servidor, para no crear dependencias con el navegador del cliente.
XML básico	Si quieres ver de un modo visual como crear un documento XML, este es tu tutorial. Este es el primero de un conjunto de tutoriales que iremos publicando sobre esta fascinante y amplia tecnología
Procesamiento XML en Java con JAXB y WSDP 1.6	Os mostramos como instalar la versión 1.6 de WSDP y como procesar los ficheros XML con uno de sus componentes, JAXB
Transformación de XML y XSL en JSPs	Os mostramos como poder utilizar XML y XSL en JSPs, combinado con el Patrón MVC
Soporte XML en Eclipse con X-MEN	Alejandro Perez nos enseña como potenciar el entorno eclipse para facilitarnos el trabajo con ficheros xml, gracias al plugin X-MEN
XMLEncryption en Java	En este magnífico tutorial, Alberto Carrasco nos enseña los fundamentos y un ejemplo práctico de XMLEncryption.

Nota: Los tutoriales mostrados en este Web tienen como objetivo la difusión del conocimiento.

Los contenidos y comentarios de los tutoriales son responsabilidad de sus respectivos autores.

En algún caso se puede hacer referencia a marcas o nombres cuya propiedad y derechos es de sus respectivos dueños. Si algún afectado desea que incorporemos alguna reseña específica, no tiene más que solicitarlo.

Si alguien encuentra algún problema con la información publicada en este Web, rogamos que informe al administrador rcanales@adictosaltrabajo.com para su resolución.

[Patrocinados por enredados.com Hosting en Castellano con soporte Java/J2EE](#)

