Herramienta de software para la gestión y construcción de proyectos: maven, gradle, ant, ivy.

Luis Fernando Vásquez Fabián

Universidad de la Sierra Sur

19 de abril de 2022

1. Introducción

Una de las grandes dificultades que deben afrontar los gerentes de proyectos de construcción es la falta de in tegración entre las metodologías, filosofías, herramien tas e instrumentos de gestión que se han desarrollado a lo largo de la historia. Aunque en la industria pocas empresas implementan diferentes metodologías de gerencia al tiempo, como la propuesta en la guía para gestión de proyectos del PMI y los postulados de Lean Construction (construcción sin pérdidas), Sánchez 1 ha identificado aspectos de compatibilidad entre las herramientas de estas dos metodologías y propone un método que permite la implementación de estas.

Por otra parte, la existencia de un gran número de herramientas de gestión desarrolladas tanto por la in dustria como por la academia y las comunidades de expertos, cada una con un enfoque muy propio de su naturaleza teórica o práctica, genera una barrera bas tante difícil de superar en la medida en que según la encuesta practicada se pudo identificar que una de las barreras en la implementación de métodos de gerencia es que los gerentes no cuentan con criterios suficientes para identificar, seleccionar e implementar el grupo de herramientas más adecuado para realizar la gestión que demanda la magnitud del proyecto

2. maven

Maven es una herramienta que usa una arquitectura diseñada en plugins, puede ejecutarse nativamente en el computador o también puede integrarse en la mayoría de IDEs1 , una de las características importantes de Maven es que tiene la capacidad de trabajar en red, permite la creación y gestión de software con librerías incluidas, compilación y empaquetado del código dentro de la estructura del JAR, los proyectos desarrollados en Maven se identifican en un archivo XML, el motor que Maven tiene incluido en su núcleo permite descargar los plugins necesarios de un repositorio. Una vez que el proyecto ha sido desarrollado pasa a la fase de validación y compilación pero si surgen errores regresa al área de desarrollo para las respectivas correcciones, si pasa la etapa de validación pasa a la fase de pruebas pero si existen errores en esta fase nuevamente regresa al área de desarrollo, una vez corregido los errores el proyecto se empaqueta en un archivo WAR y se despliega en el servidor de pruebas y pasa nuevamente a la fase de pruebas de integración y aquí también pueden surgir errores de ser el caso regresará nuevamente al área de desarrollo, una vez solventado los errores el archivo WAR sube al servidor de producción, una vez terminado este proceso se procederá a la creación de la página de documentación.(Ulloa Campoverde,)

Maven surge después de Ant para aportar mayor facilidad y claridad de uso, permitiendo mayor automatización de ciertas tareas y una mejor gestión de dependencias. No obstante, su uso en este proyecto se reducirá a utilizar los comandos propios de gestión de un repositorio Maven2, no así a especificar los proyectos de acuerdo a este software como se ha explicado anteriormente. Su instalación no se realiza por medio de yum sino que se descarga la aplicación de la zona de descargas del sito de Maven [CSL02], y se extrae sobre un directorio que especifique el usuario. Posteriormente se actualizan las variables de entorno adecuadas, en este caso, $\text{MAVEN}_HOMEyPATH : MAVEN_HOME =$

$/usr/local/mavenPATH$ =MAVEN$_{H}OME/bin$:PATH export PATH MAVEN$_{H}OMEMavenesunaherramientasmásan$
$tieneunastareasdefinidaspordefectoypermitemayoruniformidadenladefiniciónd elprocesode$(González Sánchez,)

2.1. Instalación y configuración de Maven

Para poder empezar a trabajar con Maven es necesario preparar el ambiente de trabajo, es decir, se comenzará con la descarga de Maven 3.3.9 que está disponible para Windows (.zip) o para Linux (.tar) que es una versión estable, para esto debemos descargarnos el paquete de Maven de http://maven.apache.org/download.htmlinstallation (es recomendable elegir un mirror para la descarga),

3. gradle

Gradle es una herramienta que facilita la construcción de código, no solo permitiendo indicar fácilmente cómo (mediante Groovy[4]), sino basándose en la estructura existente de Maven para la gestión de dependencias. De esta forma, se facilita la separación en distintos proyectos más pequeños, además de eliminar la necesidad de guardar las librerías en GitHub o de obligar al usuario a buscarlas y enlazarlas por su cuenta.(Mena Nieto,)

3.1. Implementación de la base del motor

La implementación básica del motor se basa en dos partes: primero, la extensibilidad del motor; segundo, la base del motor que buscará las extensiones , las añadirá, y comenzará la ejecución. Además, implementa la posibilidad de cargar fácilmente mapas creados con la herramienta gratuita y de código abierto Tiled.(Andreu López,)

3.2. Requisitos funcionales

La base del motor es el núcleo sobre el que se centra el proyecto y desde el que van a partir el resto de extensiones, por ello debe cumplir los siguientes requisitos funcionales:

• Debe ser capaz de cargar extensiones en tiempo de ejecución e interactuar con ellas.
• Debe mantener un tamaño mínimo indispensable, puesto que será una parte no modificable por las extensiones.
Sin embargo, el motor no se compone unicamente de la base. Puesto que para obtener la máxima flexibilidad es necesario mantener la base en el tamaño mínimo posible, la mayor parte de requisitos propios del motor se implementarán en la extensión principal que se explicará más adelante.

4. ant

es una herramienta usada en ambientes de desarrollo software para la realización de tareas mecánicas y repetitivas, normalmente durante la fase de compilación y construcción. Esta herramienta, hecha en Java, tiene la ventaja de no depender de las órdenes de la línea de comandos o shell de cada sistema operativo, sino se basa en archivos de configuración XML y clases Java para la realización de las distintas tareas, siendo idónea como solución multi-plataforma.(Gómez,)

En Ant es necesario definir todas las operaciones que se quieren realizar, incluso las habituales, Estas tareas se definen en un fichero con estructura XML llamado build.xml y su tamaño suele ser elevado en proyectos grandes. Además, si se requiere especificar propiedades de compilación, puede crearse un archivo build.properties que se utilizará cuando se compile un proyecto. Un ejemplo de tarea que borra el contenido de un directorio podría ser: ¡target name=çlean» ¡delete dir=”build/¿¡/target¿ Para compilar un proyecto, debe introducirse en el terminal de comandos y situado sobre el directorio del proyecto, que a su vez contendrá el fichero build.xml ant Y para ejecutar tan sólo una tarea concreta: ant ¡nombretarea¿ Para más detalles, se remite al lector al manual de Ant [CSL01c] Su uso como herramienta única de compilación cada vez es menor. Se requieren otras herramientas que hagan una mejor gestión de la información del proyecto, de modo que especificar las tareas sea un

proceso mucho más rápido. No obstante, al ser muy extensible la comunidad ha generado una cantidad muy grande de plugins de modo que prácticamente se ha convertido en una shell más, con la peculiaridad de que usa XML como lenguaje. Ant es la primera aplicación que surgió como complemento del lenguaje Java para facilitar la compilación multiplataforma. La instalación con yum yum install ant deja el entorno de modo que el ejecutable ant está accesible para cualquier usuario y la variable $ANT_HOME apuntaal directorio/usr/share/ant$(González Sánchez,)

5. ivy

. Ivy es el software que se utilizará en este proyecto para especificar las dependencias. Es dependiente de Ant y su instalación es aún más sencilla. Simplemente se requiere descargar el programa del sitio oficial [CSL05a], extraer su contenido en cualquier directorio, y seleccionar después uno de los dos archivos .jar –el archivo cuyo nombre sea ivy-¡version¿-incubating.jar– y copiarlo al directorio lib de la aplicación Ant. Siguiendo la lógica de directorios que se ha empleado en este documento, supondría copiar este fichero sobre el directorio usr/share/ant/lib. A partir de aquí, si al ejecutar ant sobre un directorio concreto se encuentra en el fichero build.xml una referencia a un fichero ivy.xml, Ant se encargará de llamar a Ivy y resolver las dependencias. Esta herramienta era anteriormente parte de la empresa Jayasoft, pero ahora forma parte de Apache. Con el cambio se ha producido un proceso de adaptación de Ivy que ha provocado que en este momento –julio de 2008– sólo haya disponibles dos versiones beta. Su uso no es autónomo, y va asociado a Ant. De este modo, en cada proyecto se requerirá un archivo build.xml como el visto anteriormente, y un nuevo fichero ivy.xml que gestione las dependencias. Para hacer llamadas desde el fichero de Ant al de Ivy y resolver las dependencias(González Sánchez,)

5.1. Aplicación práctica: Manejo de dependencias Ivy con repositorio Maven2

Se trata de crear un repositorio de artefactos remoto y utilizado por Ivy durante la compilación, conectando la aplicación con uno o más servidores y permitiendo además la gestión de artefactos tanto por línea de comandos como mediante un interfaz web (Artifactory). En primer lugar, hay que reseñar que los repositorios locales y de ibiblio vienen dados por defecto. Sólo es necesario definir los repositorios que vayamos a crear específicos para nuestro entorno de trabajo, en este caso el repositorio de los artefactos de la red en un servidor virtual creado para el efecto. Por ello, dado que se va a utilizar un repositorio Maven2, crearlo es el primer paso, aunque no hay una instrucción específica asociada pues no se crea ninguna estructura de directorios específica, es más un contenedor "ordenado" de artefactos. Estos pasos se harán con Artifactory, que ayuda a gestionar el almacenamiento de artefactos e información asociada, y que aporta un interfaz web para realizar las opciones más habituales –ver ficheros, búsqueda de artefactos, cargar y descargar artefactos, permitir distintos accesos en función de usuario y realizar backups entre otras–. Una vez puesto en marcha el servidor artifactory, ya se podrá abrir en un navegador la siguiente dirección: http://repos-server:8080/artifactory/ Tras el intercambio de credenciales se verán unos subdirectorios creados por defecto, para que los se utilicen si se quiere mantener los nombres habituales de los repositorios –libs-releases, libs-snapshots, etc–. Se puede crear uno nuevo, o configurar la aplicación con uno de estos. En este caso se va a instar a Ivy a que utilice el repositorio remoto indicado por: http://repos-server:8080/artifactory/libs-releases Si se accede a esta dirección desde un navegador, se obtiene un error porque no hay plantilla asociada –se supone que para visualizar los artefactos, está la opción Browse repository–. Pero el repositorio concreto está configurado y se podrá indicar en los ficheros que utilice Ivy. En primer lugar, se edita el fichero que aparecerá en el directorio de instalación de Maven, en ./maven/conf/settings.xml, para incluir la información del servidor asociado y los datos de acceso, dentro de las etiquetas ¡servers¿:

¡server¿¡id¿myArtifRepository¡/id¿¡username¿admin¡/username¿¡password¿password¡/password¿¡/server¿(González Sá
)

5.2. Conclusiones

En conclusión, el uso de Ant, Maven e Ivy supone una ayuda considerable para el proceso de compilación, aunque el tener que utilizar parte de las tres herramientas puede llevar a cierto desconocimiento

inicial de los procesos a seguir y a ciertas dudas por parte de las personas que tengan que crear los ficheros en ant + ivy, utilizando comandos de maven si se desean subir los archivos manualmente. No obstante el proceso de configuración de las herramientas implicadas no ha conllevado demasiadas complicaciones, más allá de establecer las primeras conexiones entre el servidor y las herramientas, por lo que superada esa fase el resto de procesos son mecánicos y en su mayoría se hacen a través de un interfaz web, con lo que sólo quedaría como proceso manual escribir los scripts de dependencias de Ivy

Referencias

Andreu López, L. (2018). *Desarrollo de una aplicación web en java usando"gwt material design", para el control y mantenimiento de plantas solares fotovoltaicas* (Tesis Doctoral no publicada). Universitat Politècnica de València.

Gómez, O. S. G. (2008). Integración continua en componentes ejb. *vol. Versión*, *1*, 17.

González Sánchez, C. (2008). Generación, gestión y distribución de artefactos java con técnicas de integración continua y software libre.

Mena Nieto, E. (s.f.). Jage (just another game engine), creación de un motor de videojuegos 2d multiplataforma de código abierto.

Ulloa Campoverde, G. A. (2019). *Estudio de la herramienta maven como gestor de proyectos spring mvc con el caso de uso aplicación para comercio electrónico* (B.S. thesis).