**Présentation de Maven**

Apache Maven est un outil "Open Source" utilisé pour gérer la production de projets logiciels Java de manière général. Maven est un projet de l'organisation Apache Software Foundation et était historiquement une branche de l'organisation Jakarta Project. L'objectif de cet outil peut être comparé au système Make sous Unix, à savoir : produire un logiciel à partir de ses sources, en optimisant les tâches réalisées à cette fin et en garantissant le bon ordre de fabrication. Il est également comparable à l'outil Ant, mais fournit des moyens de configuration plus simples, eux aussi basés sur le format XML. Ant, Make et bon nombre d'outils similaires s'appuient sur une approche procédurale, pour laquelle on décrit les opérations à accomplir pour construire le logiciel ou éxecuter des tâches annexes. Cela se traduit donc par une suite de commandes, qui prendra d'une façon ou d'une autre la forme suivante :

* Initialisation : Préparation des répertoires de travail.
* Compilation : Invocation du compilateur "javac".
* Assemblage : Invocation de l'archiveur "jar".

Cette approche fonctionne très bien mais elle nécessite :

* de répéter pour chaque projet une liste de tâche très similaires.
* de gérer une liste de dépendances entre les étapes clés.

Maven choisit une approche différente, fondée sur le constat suivant : tous les projets Java vont suivre à peu près le même schéma. Ainsi, les développeurs de Maven considèrent qu'il est plus simple de décrire en quoi un projet est différent de ce "scénario type" que de répéter des commandes très comparables d'un projet à l'autre. Maven exploite donc le concept de "Convention over configuration".

Maven fait le choix d'une approche déclarative, dans laquelle on indique les particularités du projet et non la manière de le construire. On précise l'emplacement des fichiers sources, les bibliothèques qui sont nécessaires, plutôt que la ligne de commande du compilateur. Ainsi, il ne s'agit plus de définir les options de "javac", mais de décrire une structure plus générale du projet, qui pourra être exploitée dans un autre contexte. Elle sera, par exemple, utilisée pour s'intégrer dans un IDE(Integrated Development Environment) comme Eclipse ou par les outils d'analyse de code.

Une des nombreuses conventions proposées par Maven est la structure prédéfinie des répertoires du projet. La logique est plutôt simple : on trouve à la racine le fichier [POM](http://igm.univ-mlv.fr/~dr/XPOSE2010/Apache_Maven/pom.html) qui gère toute la gestion Maven du projet. L'ensemble des sources est placé dans un répertoire *src*, tandis qu'un répertoire *target* sert de zone temporaire pour toutes les opérations réalisées sur le projet. Cela permet de faciliter grandement la configuration de la gestion du code source. Il suffit d'exclure *target* (en plus des fichiers spécifiques de l'IDE) pour être sûr de ne pas inclure des fichiers de travail qui n'ont pas à être partagés.

Sous le répertoire des sources, Maven effectue un découpage explicite entre ce qui fait partie du projet et ce qui sert d'outillage de test. Deux sous-répertoires, *main* et *test*, marquent cette distinction.

Enfin, dans chacune de ces branches, un dernier niveau de répertoires sépare les fichiers sources par language :

* *"java"* pour le code source des classes Java.
* *"resources"* pour les fichiers de ressources (configuration XML, fichiers de propriétés...).
* *"webapp"* pour les fichiers statiques d'une application web.