PIGIER
Côte d'Ivoire
L'Université des métiers

ANNÉE ACADÉMIQUE : 2019-2020

UNION – DISCIPLINE – TRAVAIL

PRESENTATION DE MAVEN

CLASSE : MAGIR 2

Participant :

KOFFI Gnyssand Maëlle

Professeur :

M. KOUAO

I- PRESENTATION

Apache Maven est un outil open source permettant d'automatiser la gestion de projets Java. Maven est un projet de l'organisation Apache Software Foundation et a été créé comme outil d'aide au développement d'un autre projet jakarta : **Turbine**.

Il offre entre autres les fonctionnalités suivantes :

- Compilation et déploiement des applications Java (JAR, WAR)
- Gestion des librairies requises par l'application
- Exécution des tests unitaires
- Génération des documentations du projet (site web, pdf, Latex)
- Intégration dans différents IDE (Eclipse, JBulder , NetBeans)

Maven exploite le concept de "Convention over configuration".

L'objectif principale de cet outil est de permettre à un développeur de comprendre l'état complet d'un effort de développement dans les plus brefs délais. Afin d'atteindre cet objectif, Maven tente de traiter plusieurs domaines de préoccupation :

- ✓ Rendre le processus de construction facile
- ✓ Fournir un système de construction uniforme

Permet à un projet de se construire en utilisant son modèle d'objet de projet (POM) et un ensemble de plugins qui sont partagés par tous les projets utilisant Maven, fournissant un système de construction uniforme.

- ✓ Fournir des informations de qualité sur les projets

Fournit de nombreuses informations utiles sur le projet qui sont en partie extraites de votre POM et en partie générées à partir des sources de votre projet. Par exemple, Maven peut fournir :

- Document de journal des modifications créé directement à partir du contrôle de code source
- Sources croisées
- Liste des listes de diffusion gérées par le projet
- Liste des dépendances
- Rapports de tests unitaires, y compris la couverture

- ✓ Fournir des directives pour le développement des meilleures pratiques

Maven vise à rassembler les principes actuels pour le développement des meilleures pratiques et à faciliter l'orientation d'un projet dans cette direction.

Par exemple, la spécification, l'exécution et la génération de rapports sur les tests unitaires font partie du cycle de construction normal à l'aide de Maven. Les meilleures pratiques actuelles en matière de tests unitaires ont été utilisées comme lignes directrices :

- Conserver le code source du test dans une arborescence source distincte mais parallèle
- Utilisation des conventions de dénomination des cas de test pour localiser et exécuter des tests
- Ayant les cas de test configurer leur environnement au lieu de compter sur la personnalisation de la construction pour la préparation des tests

Maven vise également à aider au flux de travail du projet, comme la gestion des versions et des problèmes.

✓ Permettre une migration transparente vers de nouvelles fonctionnalités

Maven offre aux clients Maven un moyen simple de mettre à jour leurs installations afin qu'ils puissent profiter des modifications apportées à Maven lui-même.

II- LE POM

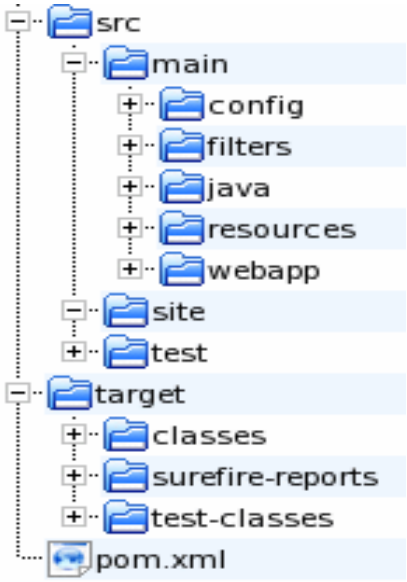
Le POM (Project Object Model) décrit un fichier XML mettant le projet en place. L'ensemble des sources est placé dans un répertoire src, tandis qu'un répertoire target sert de zone temporaire pour toutes les opérations réalisées sur le projet. Cela permet de faciliter grandement la configuration de la gestion du code source.

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 langage :

- "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.

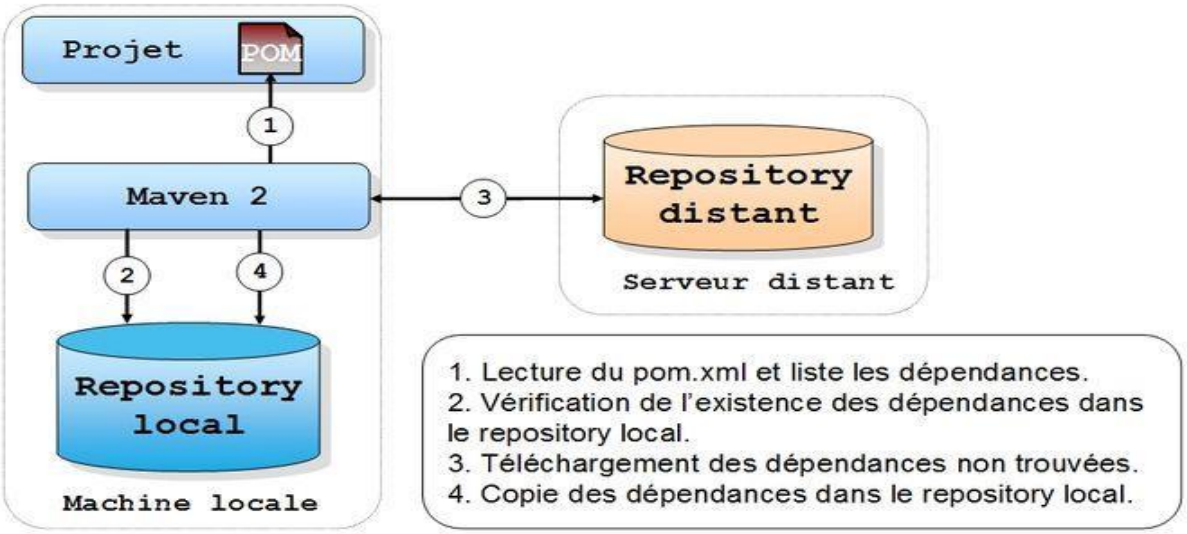
✓ Arborescence par défaut



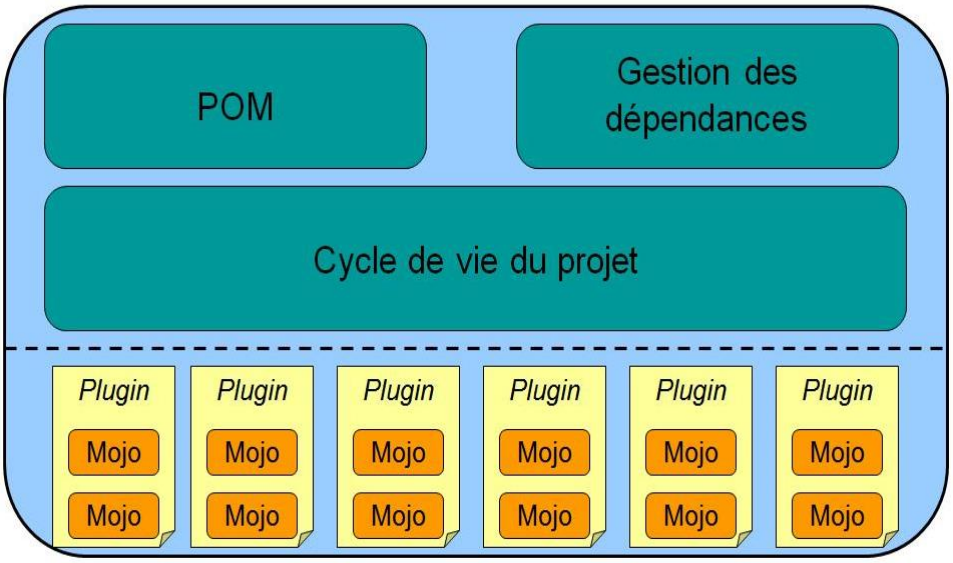
III- GESTION DES DEPOTS ET DEPENDANCES

Maven dispose de plusieurs référentiels à plusieurs niveaux. Le but du référentiel est de rendre disponible aussi bien les plugins utilisés ou envisagés de l'être que les projets générés par Maven. On peut bien sûr y installer des projets pour les utiliser (sans qu'ils ne soient générés par Maven). Il y a trois référentiels :

- Un au niveau de la machine du développeur, appelé repository local
- Un au niveau du site Maven qui contient l'ensemble des plugins
- Un troisième référentiel (facultatif) qui peut être déclaré au niveau de l'entreprise ou la structure utilisant Maven. Il joue l'intermédiaire entre les deux premiers référentiels.



IV- ARCHITECTURE DE MAVEN



process-resources	resources:resources
compile	compiler:compile
process-test-resources	resources:testResources
test-compile	compiler:testCompile
test	surefire:test
package	jar:jar
install	install:install
deploy	deploy:deploy

V- LES PRINCIPAUX PLUGINS

1- Maven-eclispe-plugin

- Génère les métadonnées eclipse à partir du POM (.classpath, .project)
- Compatible WTP (.components)
- Liaison avec les jars de sources
- Maven-netbeans-plugin

2- Maven-assambly-plugin

- Créé un artefact (zip, jar) pour déploiement et distribution
- Configuration par un descripteur XML
- Possibilité d'inclure les dépendances

3- Maven-site-plugin

- Génération d'un site de documentation à partir du POM et de données complémentaires
- Mise en page de site (moteur de template)
- Déploiement du site

4- Maven-release-plugin

- Gère la publication de versions du projet
- Enchaine automatiquement les taches nécessaires :
  - Contrôle par rapport au SCM
  - Mise en place des versions stables
  - Vérification du build
  - Tag sur le SCM
  - Repassage en version de développement
- Publication des packages à l'aide de la version tagguée