

# Maven 3 - Paving the desire lines

About the author:

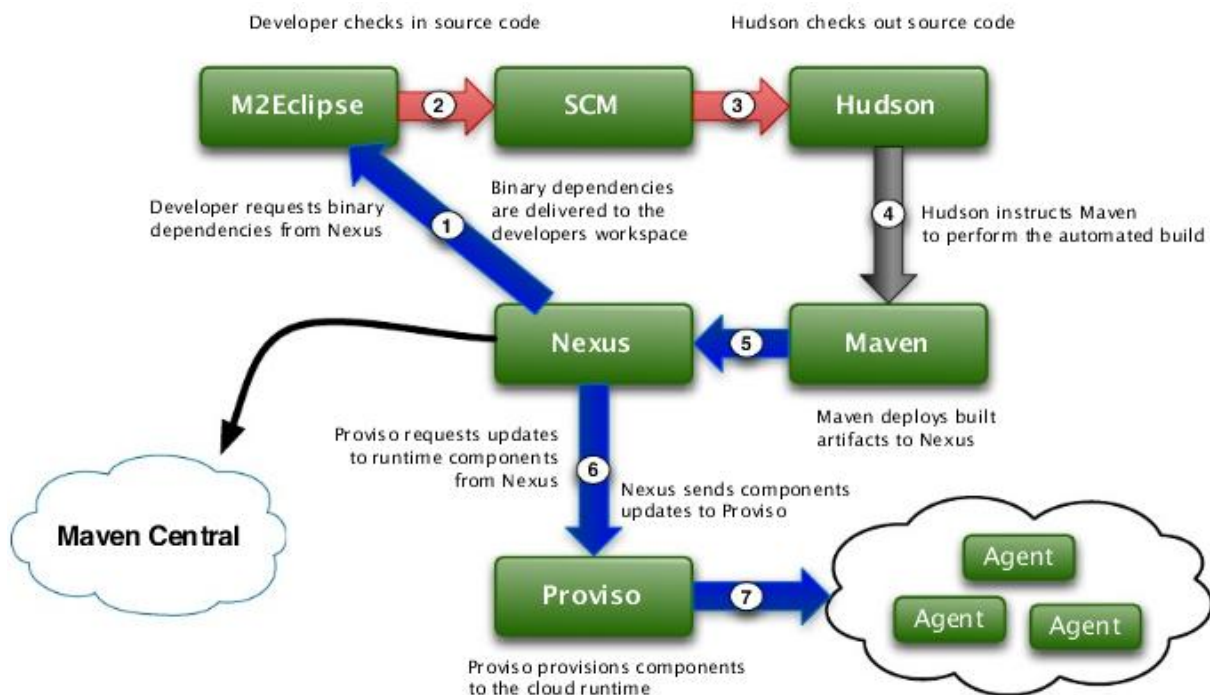
Aino Andriessen is consultant Enterprise Java en Oracle ADF en Expertise Lead ALM voor AMIS Services.

## Samenvatting

Hoewel al lang aangekondigd, binnenkort kunnen we na ruim vijf jaar eindelijk Maven versie 3 verwachten. De architectuur van Maven is flink op de schop gegaan omdat versie twee de nodige tekortkomingen kent waarmee nieuwe ontwikkelingen niet mogelijk zijn en omdat Maven soms op een andere manieren gebruikt wordt dan het bedacht is. Maven 3.0 betreft feitelijk het *bestraten van de wandelpaden*, oftewel **'Paving the desire lines'**, naar het voorbeeld van de campus van de Berkeley Universiteit in de jaren zestig, waar de paden pas daadwerkelijk werden aangelegd nadat de gebruikers van de campus zelf de optimale routes hadden uitgesleten in het grasveld. Versie 3.0 zal vrijwel geheel backwards compatibel zijn met versie 2, maar wel volgens de nieuwe architectuur en zal daarom als drop-in replacement kunnen fungeren. Nieuwe functionaliteit kunnen we verwachten vanaf versie 3.1.

## Maven landschap

De laatste jaren heeft het gebruik van Maven een grote vlucht genomen. Werd het in het begin voornamelijk ingezet als een 'script' tool voor het maken van deliverables als wars en jars, nu wordt het inzet als een multi-inzetbare tool voor alle aspecten van het build en delivery proces en heeft het zijn weg gevonden naar de grote enterprises. Rondom Maven is een uitgebreid landschap ontstaan van tooling en producten ter ondersteuning van het deliveryproces als Artifact Repository Management, IDE ondersteuning, Continuous Integration, Quality Management, Provisioning etc. Figuur 1 geeft een weergave van de verschillende tools en hun relatie.



Figuur 1 Maven landschap volgens Sonatype.

**Artifact Repository Management** betreft het opslaan en beschikbaar stellen van artifacten, oftewel deliverables als applicatie releases en libraries, in een centrale, bedrijfsbrede, repository. Het biedt faciliteiten als security, zoeken, staging en workflow ter ondersteuning van het delivery en development proces. Daarnaast fungeert het ook als een eenduidige (proxy) toegang tot ander repositories als Maven Central en CodeHaus

waardoor allerhande lokale configuraties tot het verleden behoren. Het is een onmisbare component in een ontwikkelstraat. De belangrijkste producten zijn momenteel Nexus, van Sonatype, en Artifactory, van JFrog. Beide kennen het model van een gratis open source versie en een betaalde Professional versie met extra functionaliteit.

De toenemende populariteit van de repository managers en het daardoor veranderende artifact / dependency management van Maven is één van de drijvende krachten geweest achter de reorganisatie van de Maven architectuur en de ontwikkeling van het eerder genoemde *Aether*.

De belangrijkste *IDE's* hebben ondersteuning voor Maven. Momenteel gebruiken ze allemaal al embedded Maven 3 (beta) voor stabielere, consistentere en beter presterende Maven integratie. De build lifecycle is daardoor een intrinsiek onderdeel geworden van de IDE en niet meer iets wat alleen maar afgetrapt wordt. Het geïntegreerde artifact management maakt het gebruik van externe (en interne) libraries veel transparanter en overzichtelijker. De reporting en code metrics plugins bieden niet alleen in de IDE maar ook in andere omgevingen eenvoudig en duidelijke toegang tot kwaliteitsrapportages. Dit alles verhoogt de transparantie en kwaliteit van het ontwikkelproces, niet alleen voor ontwikkelaars maar ook projectleiders en andere betrokkenen. Een interessante ontwikkeling die momenteel gaande is, is 'developer onboarding', waarmee de ontwikkelomgeving in zeer korte tijd klaar gezet kan worden voor gebruik.

De **build server** is de spin in het web van het ontwikkel proces. In essentie is het een scheduler voor het uitvoeren van (onderdelen van) de build lifecycle maar is in de meeste gevallen uitgebreid met reporting en delivery faciliteiten. Hiervoor zijn de laatste jaren verschillende producten ontwikkeld als Continuum, Hudson, Bamboo, TeamCity etc. Al deze producten ondersteunen naast tools als ant ook Maven en zijn reporting mogelijkheden

**Provisioning** De aandacht heeft zich de laatste jaren vooral gericht op Build Servers, Artifact Management en Kwaliteits Management. Provisioning, oftewel de delivery van een applicatie naar de run-time omgeving is nog een relatief onontwikkeld gebied waarvoor veelal homegrown oplossingen worden ontwikkeld. Proviso, de laatste loot aan de producten tak van Sonatype, richt zich op het provisioning proces, oftewel installatie en deployment van applicaties op en van de run-time omgeving(en). De aandacht ligt in eerste instantie op Tomcat, Jetty en JBoss. Later zal ook database en LDAP provisioning ondersteund worden.

### *Maven 3.0*

Het mag duidelijk zijn dat vijf jaar geleden bij het uitkomen van Maven 2 uitgegaan werd van de toenmalige situatie en dat veel ontwikkelingen niet voorzien konden worden. Maven is verder ontwikkeld en aangepast aan de wensen en eisen van gebruikers maar soms werden de beperkingen en tekortkomingen pijnlijk duidelijk en konden aanpassingen alleen nog met veel kunst en vliegwerk gedaan worden en soms ook helemaal niet. Het werd steeds meer duidelijk dat er radicalere en grotere ingrepen nodig waren om de nieuwe ontwikkelingen te kunnen volgen; wat uiteindelijk geleid heeft tot Maven3.

Onder de motorkap van Maven zijn grote veranderingen uitgevoerd. De architectuur is verbeterd en de functionaliteit is in afzonderlijke onderdelen ondergebracht. De Maven core is daardoor fors opgeschoond en de plugin API is aangepast. De drijvende kracht achter al deze veranderingen is de wens om Maven op verschillende manieren en in complexere situaties in te kunnen zetten. Niet alleen vanaf de command-line, maar ook in IDE's, build servers en andere omgevingen. Dat vereist een goede componenten-architectuur en consistent gedrag. Maven 2 was daartoe niet geschikt en vooral voor het artifact-management, repository interactie en reporting moesten de internals fors op de schop. Het resultaat is een consistente, stabiele en beter presterende architectuur. Een heel belangrijk uitgangspunt is om Maven 3.0 volledig backwards compatibel te houden opdat de veranderingen voor de gebruikers minimaal zijn en ze tegelijkertijd optimaal kunnen profiteren van de verbeteringen. Om de backwards compatibiliteit te garanderen is het aantal integratie testen uitgebreid van 50 naar meer dan 600. Maven 3 voldoet zelfs beter aan de specificaties dan enige Maven 2 versie.

**Aether** is het resultaat van de reorganisatie op het gebied van artifact management. Het is een stand-alone library voor artifact resolutie, dependency management en repository interactie. Het is een standaard Java library, dus toepasbaar in alle Java omgevingen, hoewel de aandacht in eerste instantie uitgaat naar Maven en Maven repositories, maar is daar zeker niet toe beperkt; de ant tasks zijn momenteel al in ontwikkeling. Het is dé component die verantwoordelijk is voor alle dependency analyses, dependency graphs, interactie met repositories, artifact transport, versie resolutie etc. Op dit moment kent het een beperkt security mechanisme, maar dat zal worden uitgebreid met het SAT4J framework. Het is het vervolg van een eerder Maven project, Mercury dat nu opgeheven is.

**Dependency Injection** is één van de peilers van Maven. Hiervoor is indertijd de Plexus Inversion of Control container ontwikkeld. Voor de gewenste nieuwe functionaliteit voldoet Plexus echter niet meer en zou drastisch moeten worden aangepast. Uiteindelijk is Plexus vervangen door een andere IoC container. Google Guice bleek echter zowel functioneel als structureel het beste aan te sluiten bij Plexus en bij Maven en zal daarom Plexus gaan vervangen. Natuurlijk bleek dit niet zonder slag of stoot te gaan en is er de nodige code ontwikkeld om de backwards compatibility te garanderen. Het is momenteel nog onbekend of de vervanging al in 3.0 of pas in 3.1 zal plaatsvinden.

Dit zijn allemaal hele grote veranderingen. Er is dan ook veel zorg gedragen dat zowel de bestaande API's als de nieuwe API's blijven werken. Er wordt verwacht dat het merendeel van de projecten zonder problemen Maven 2 kunnen vervangen door Maven 3. Eigenlijk zullen alleen custom plugins die gebruik maken van Maven 2 artifact management niet meer ondersteund worden. In de praktijk bleek dat dit ook maar zeer beperkt wordt gedaan. Dat betekent echter niet dat een migratie compleet zonder issues zal zijn. In Maven 3 is de *pom validatie* namelijk een stuk strikter geworden. De eerste keer met Maven 3 zal je waarschijnlijk geconfronteerd worden met de nodige warnings dat de pluginversie ontbreekt (Figuur 2). Deze requirement bestond al in Maven 2 maar werd nooit afdgedwongen en daardoor ook vaak niet toegepast. Eigenlijk is het wel logisch dat je voor een plugin dependency een versie op moet geven, zoals dat ook voor de andere dependencies geldt. Sowiezo kan het ontbreken van een versie tot onverwachte problemen leiden, omdat de plugin resolutie op zijn minst enigszins typisch was: bij het ontbreken van een versienummer wordt in versie 2 de laatste release óf de laatste snapshot versie gebruikt, afhankelijk van welke recenter is. In Maven 3.0 zal dat altijd de laatste release zijn. De plugin versie verplichting is vanwege backwards compatibility nog niet aanwezig.

```
[WARNING] Some problems were encountered while building the effective model for nl.amis.demo:one:jar:1.0-SNAPSHOT
[WARNING] 'build.plugins.plugin.version' for org.apache.maven.plugins:maven-compiler-plugin is missing. @ line 18, column 15
[WARNING] It is highly recommended to fix these problems because they threaten the stability of your build.
[WARNING] For this reason, future Maven versions might no longer support building such malformed projects.
[WARNING]
```

**Figuur 2** Warning dat de plugin versie ontbreekt.

Behalve de interne veranderingen van maven zijn er ook de nodige functionele verbeteringen doorgevoerd en nieuwe features ontwikkeld zoals embedded Maven, errormeldingen, pom validatie, performance verbeteringen, Maven PloyGlout en de Maven Shell,.

**Embedded Maven** is een belangrijke feature voor vooral IDE's en build environments. De integratie tussen de IDE en Maven is daardoor fors verbeterd waardoor Maven gecontroleerder en ook sneller kan worden uitgevoerd, onder andere door middel van incremental builds. De IDE kan nu letterlijk luisteren naar het buildproces en er onderdeel van uitmaken in plaats van alleen maar opstarten. Veel van de interne wijzigingen van Maven zoals lifecycle extension points en incremental builds zijn doorgevoerd ten bate van embedded Maven. Alle IDE's zijn al overgestapt naar Maven 3 Beta en gebruiken al actief embedded Maven.

**Error and Integrity Reporting** is verbeterd. In plaats van lange stacktraces wordt er zinvolle informatie getoond met links naar Wiki pagina's met meer uitleg (Figuur 3). Deze informatie kan direct in de embedded Maven opgehaald worden. De *Queryable Lifecycle* maakt de uitvoering van de gehele build (dus inclusief modules) beschikbaar als een soort executie plan voordat de build daadwerkelijk wordt gedaan. Daardoor is alle relevante informatie van de build al aan het begin beschikbaar en kan deze proactief gecontroleerd worden op eventuele problemen zoals het ontbreken van informatie uit een niet geactiveerd profile.

```
C:\TEMP\one>mvn clean install
[INFO] Scanning for projects...
[ERROR] The build could not read 1 project -> [Help 1]
[ERROR]   The project <C:\TEMP\one\pom.xml> has 1 error
[ERROR]     Non-parseable POM C:\TEMP\one\pom.xml: end tag name </propertie> must match start tag name <properties> from line 13 (position: TEXT seen ...</project.build.sourceEncoding>\n </propertie> ... @15:15) @ line 15, column 15 -> [Help 2]
[ERROR]
[ERROR] To see the full stack trace of the errors, re-run Maven with the -e switch.
[ERROR] Re-run Maven using the -X switch to enable full debug logging.
[ERROR]
[ERROR] For more information about the errors and possible solutions, please read the following articles:
[ERROR] [Help 1] http://cwiki.apache.org/confluence/display/MAVEN/ProjectBuildingException
[ERROR] [Help 2] http://cwiki.apache.org/confluence/display/MAVEN/ModelParseException
C:\TEMP\one>
```

**Figuur 3** Gehele error message bij een fout in de pom. Merk op dat de stacktrace niet meer getoond wordt.

De **performance** van Maven 3 is fors verbeterd. Het gebruikt minder resources en ook de execution tijd is een stuk sneller geworden. De *queriable lifecycle* biedt de mogelijkheid om te bepalen wat er daadwerkelijk uitgevoerd moet worden. Dat zorgt al voor een forse performance verbetering. Daarnaast biedt *Incremental build* support met name in embedded omgevingen als M2Eclipse de mogelijkheid om zeer specifiek alleen op gewijzigde resources de benodigde acties uit te voeren. Een andere performance boost is *parallelisatie*, waarmee tegelijkertijd verschillende onderdelen van een build uitgevoerd wordt. Dit is vooral van toepassing voor multi-module projecten. Het bevindt zich nog in de begin fase maar is al onderdeel van Maven 3.0. Met bijvoorbeeld `mvn -T 4 clean install` wordt de build uitgevoerd in 4 threads. Meer informatie is te vinden op de Maven wiki.

**Maven Reporting** en de **site plugin** kenden teveel verwevenheid met de Maven core, onder andere met betrekking tot report plugin dependencies. Alle reporting functionaliteit is uit de core gehaald en wordt ondergebracht in de site plugin. Dit geeft een flexibeler reporting systeem dat meer op individuele reports is gebaseerd dan op een gehele site waardoor het beter geschikt is voor tools als Hudson en Sonar. Waarschijnlijk zal Maven 3 voor het genereren van rapporten niet meer gekoppeld zijn aan Doxia en vervangen worden door een ander systeem. Maar ook zal het mogelijk worden om daarvoor je eigen oplossing te gebruiken.

De **Maven Shell** is een nieuwe tool in het Maven landschap. Dit is een Command Line Interface (CLI), een shell, voor het uitvoeren van Maven commando's. Het is een apart te downloaden product dat niet alleen standaard shell functionaliteit biedt zoals code completion, syntax highlighting, history, aliases, variables of properties, preferences etc., maar ook functionaliteit die specifiek op Maven is gericht zoals een caching mechanisme voor pom files, voor een forse performance boost, interactie met Nexus, archetype integratie en log level configuratie. Ook kunnen we ondersteuning voor verwachten voor Hudson zoals het aanmaken en runnen van een job. Op MacOS is het aangesloten op het Growl notificatie mechanisme (Figuur 4). Het zal de aanbevolen manier zijn voor het uitvoeren van command-line Maven.



**Figuur 4** Growl notification.

**Polyglot Maven** is een extensie, een aparte download, voor Maven waarmee je in andere Domain Specific Languages (DSL) talen de build configuratie, pom file, kan doen. Momenteel ondersteunt het Scala (Figuur 5), Groovy, YAML, Clojure en Ruby. Hoewel dit nog een beperkte toepassing is, lijkt het slechts het begin te zijn van de mogelijkheden omdat de DSL's ook toegang hebben tot Maven's internals en daardoor kunnen ingrijpen op de Maven lifecycle. Het is nog onduidelijk welke richting dit zich zal ontwikkelen.

```
project("myGroupId:myArtifactId:1.0-SNAPSHOT") { proj =>
    //...adds scala library dependency and configures Maven Scala compiler plugin...
    proj includesScalaSourceCode "2.7.7"

    //...adds normal Maven artifact dependency...
    proj dependency "apache-httpclient:commons-httpclient:3.0.1"
}
```

**Figuur 5** Voorbeeld van een scala 'pom' file.

**Tycho** is een set Maven plugins en extensions voor het maken van Eclipse plugins en OSGI bundles, voor de interactie met P2 en Maven (Nexus) repositories en voor Eclipse PDE target platforms.

Hoewel de pom en settings versies gelijk zijn gebleven, respectievelijk 4.0.0 en 1.0.0 zijn er wel enkele **non-backwards incompatibilities**, waarmee stabielere en consistentere gedrag van Maven wordt bereikt.

- Het is niet meer mogelijk om externe profiles, buiten de pom of settings op te nemen.
- Legacy style repositories worden niet meer ondersteund. Je zult een repository manager moeten gebruiken om er toch toegang tot te hebben.
- RELEASE en LATEST kunnen niet meer als plugin versie gebruikt worden.

- Striktere pom validatie zoals op het ontbreken van een plugin versie nummer en duplicate dependencies.
- Deployment van snapshots levert altijd een uniek versie nummer op. Het uniqueVersion element voor de snapshotRepository configuratie werkt niet meer.
- De site plugin versie 2.x werkt niet meer omdat die gebruik maakt van code in Maven-core die daaruit gehaald is.
- Plugin configuratie zonder versie wijst altijd naar een release. Momenteel is een versienummer voor plugin nog niet verplicht, maar dat zal in de toekomst veranderen.
- In Maven 2 hadden extensies een globaal effect. Een extensie die in de ene module werd geladen was ook in een andere module beschikbaar. In Maven 3 kan dit niet meer en zal iedere module zelf zijn extensie moeten configureren.
- De native transport protocollen zijn http, https en file. Alle andere, zoals scp, moeten expliciet gedeclareerd worden.
- Maven3 style project werkt nog niet in Hudson. De Maven plugin voor Hudson is onder ontwikkeling en zal ook gebaseerd worden op embedded Maven. Tot die tijd kan je een freestyle project gebruiken. Voor een migratie is het verstandig om de compatibility notes op de Maven wiki te raadplegen.

### Maven 3.1

Hoewel Maven 3.0 onder de motorkap op de schop gegaan is zijn er geen grote nieuwe features toegevoegd. Daarvoor zullen we moeten wachten op 3.1, die aan het einde van het jaar verwacht wordt. Daarin kunnen we nieuwe versies, en dus nieuwe functionaliteit, van het pom model en de settings.xml verwachten.

De nieuwe features voor de **pom.xml** zijn:

- **Mixins.** Eén van de meest interessante nieuwe features zijn mixins waarmee je herbruikbare stukjes pom file kan maken. Bijvoorbeeld met een configuratie voor deployment naar een applicatie server of de bedrijfsspecifieke configuratie van het release proces. Voorheen kon dit alleen via overerving van een parent pom.
- Version-less parent elements. In multi-module projecten hoef je niet meer de versie van de parent op te geven. Deze kan namelijk eenvoudig afgeleid worden.
- Global Excludes bieden de mogelijkheid om eenmalig aan te geven dat bepaalde dependencies, zoals bijvoorbeeld commons-logging, überhaupt niet gebruikt moeten.
- Plugin extension points (zie hieronder)

De nieuwe features voor de **settings.xml** zijn:

- Ondersteuning voor security management.
- Ondersteuning voor repository management in plaats van de 'mirror hack' die momenteel nodig is.
- Ondersteuning voor meerdere omgevingen, bijvoorbeeld voor klant en kantoor, door onder andere de mirror in een profile op te kunnen nemen.

Ook zullen er onder de motorkap weer de nodige veranderingen plaatsvinden. Voor plugin ontwikkelaars is er de *nieuwe plugin API* waarin Java 5 annotaties gebruikt kunnen worden in plaats van de 'JavaDoc style' annotaties en waarmee *extension points* kunnen worden gedefinieerd om de functionaliteit van een plugin aan te passen in plaats van een nieuwe plugin te maken. Een voorbeeld hiervan betreft de war plugin met een extension point waarmee project specifieke aanpassingen aan de web.xml uitgevoerd kunnen worden in plaats van de ontwikkeling van een eigen custom war plugin.

### Conclusie

Zo'n vijf jaar geleden heeft Maven 2 een ware revolutie ontketend en heeft het build proces de aandacht gekregen die het verdient. Hoewel versie 2 in sommige aspecten tekortschoot heeft het gebruikersgemak velen overtuigd en is Maven bijna de de-facto standaard geworden. Met de toenemende populariteit bleek ook op een aantal punten de tekortkomingen en bleek dat Maven soms ook anders werd gebruikt dan het oorspronkelijk bedoeld was. Het motto van Maven 3.0 is 'Paving the desirables' en het is de versie waarmee de interne architectuur is aangepast op de gebruikers, het is voorbereid op de toekomst én het consistent zal zijn in zijn gedrag. Bovendien komt het tegemoet aan vele kritieken op de (plugin) documentatie en overcomplexiteit van de pom. Het is vrijwel volledig backwards compatibel zodat een migratie een relatief eenvoudige exercitie zal zijn. Met deze versie is Maven en dus ook het ontwikkelproces klaar voor een mooie toekomst.

## Referenties

- Maven wiki: <https://cwiki.apache.org/confluence/display/MAVEN>
- Aether: <https://docs.sonatype.org/display/AETHER/Home>
- Sonatype blog: <http://www.sonatype.com/people/>
- Maven shell: <http://shell.sonatype.org/> en <https://docs.sonatype.org/display/MVNSH/Home>
- Polyglot Maven: <http://polyglot.sonatype.org/>