best practices - multi-project builds - plug-in inheritance

This is the mail of Chris:

Greetings,
I am looking for guidance on POM inheritance. I know that I can set up a build system where all common functionality is pulled into a Parent POM, so that Child POMs can be remarkably small. What I'm uncertain on is how I override some of the plugins – leaving the others as is. I see that I can override properties and customize the common plugin use that way. But can I completely override a particular plugin?? Or perhaps add another plugin?? I guess I'm looking for the rules of inheritance...

I.e can I override a plugin like this?? Is there a better way to supply different configuration sets??

Parent POM:

```
<project>
....
<build>
<plugins>
<plugin>
....
<artifactId>my-plugin</artifactId>
<configuration>
<something>val1</something>
</configuration>
</plugin>
```

Child POM

```
<project>
....
<parent>....</parent>
<build>
<plugins>
<plugin>
....
<artifactId>my-plugin</artifactId>
<configuration>
<something>val2</something>
<somethingelse>val3</somethingelse>
</configuration>
</plugin>
```

Thanks,
– Chris

And his own follow up mail:

I can now answer this myself.
It appears that m2 does "the right thing" – i.e. what you'd expect
All plugins are inherited from the Parent. To override any Plugin, simply

declare it in the Child. Those you don't declare remain unchanged. All <configuration> in the overriding Plugin declaration is inherited. And you can override any config properties or add new ones where necessary. Powerful, good stuff. This, plus the ability to parameterize with <properties>, makes it really easy to create a malleable "build system" good job maven guys 🙂
Cheers,
– Chris

JIRA reference