Version number policy

Apache Maven components and plugins will follow a semver-ish version number policy.

- Version numbers will consist of three components `MAJOR.MINOR.INCREMENTAL`
- If the changes since the previous tag are only bug fixes, then the `INCREMENTAL` number will be increased.
- If the changes since the previous tag include improvements and/or new features which are backwards compatible with the previous tag, then the `MINOR` number will be increased and the `INCREMENTAL` number will be reset to 0.
- If the changes since the previous tag are not backwards compatible then the `MAJOR` number will be increased and the `MINOR` and `INCREMENTAL` numbers will be reset to 0.
- The basis for deciding on the changes since the previous tag are the issues detailed in the JIRA report for the release.
- In the event that there is a disagreement over whether specific JIRA issues are bug fixes or improvements/new features, the majority of committers that express an opinion will determine the classification. In the event of a tie, the bias is towards improvement/new feature (as it is safer to assume it is an improvement/new feature than assume it is a pure bug fix).

Version lines

In order to limit the size of the code base that we need to maintain, we will adopt the following policy for version lines. If there are issues identified in a Maven component then those issues will be fixed in active versions only. There are three types of version lines:

- Development - where new features, improvements and bug fixes will be delivered.
- Maintenance - where backwards compatible bug fixes will be back-ported from the development line. (Note: this does not mean that we will back-port any fixes, rather that we will consider back-porting fixes)
- Security only - where backwards compatible bug fixes addressing security related issues will be applied.

Maven Core

The development line is the highest `MAJOR.MINOR`

There can be at most two maintenance lines:

- The previous `MAJOR.MINOR`
- The highest `(MAJOR-1).MINOR` if less than 18 months since the first release of that line

The security only lines are any remaining `MAJOR.MINOR` that is within 18 months since the first release of that line.

The development line of Maven core should require a minimum JRE version that is no older than 18 months after the end of Oracle's public updates for that JRE version at the time that the first version of the development line was released, but may require a higher minimum JRE version if other requirements dictate a higher JRE version

- e.g. Maven core 3.2.1 (the first release of the 3.2.x line) was released in Feb 2014 and JRE 1.6 was EOL in Feb 2013 thus the minimum version of the JRE for the Maven 3.2.x line could be 1.6. Any new Maven release lines after Aug 2014 will have a minimum JRE requirement of 1.7 (or higher if other technical requirements dictate)

Maven Plugins

The development line is the highest `MAJOR.MINOR`

The maintenance line is the previous `MAJOR.MINOR`
The security only line is the previous MAJOR.MINOR to the maintenance line if within 18 months since the first release of that line.

The development line of a Maven plugin should require a version of Maven that is at most 36 months since the first release of that line but may require a newer version if the API/JRE requirements dictate a newer version.

Maven Shared Components

The development line is the highest MAJOR.MINOR

Shared components maintenance line needs are determined by the Maven Core and Maven Plugin lines that depend on the shared components.

i.e. if a shared component is used by both Maven Core and two Maven Plugins, the maintenance lines follow from the development and maintenance lines of those plugins.

e.g. Maven Wagon is used by Maven Core and the Maven Site Plugin. If

○ the Core development line is 4.5 and depends on Wagon 5.7,
○ the Core maintenance line(s) are 4.4 and 3.7 depending on Wagon 5.6 and 5.5 respectively,
○ the Site Plugin development line is 6.2 depending on Wagon 5.5 and
○ the Site Plugin maintenance line is 6.1 depending on Wagon 5.4

Then

○ Wagon 5.8 is the development line
○ Wagon 5.7 is a maintenance line (as Core is the corresponding development line)
○ Wagon 5.6 is a maintenance line (as Core has a maintenance line)
○ Wagon 5.5 is a maintenance line (as Core has a maintenance line and Site Plugin has a development line)
○ Wagon 5.4 is a maintenance line (as Site Plugin has a development line)

If a bug fix to Site Plugin 6.1 requires an upgrade of Wagon from the 5.4 line then that bug fix is out of scope for back-porting. The only changes to Site Plugin’s 6.1.x dependency on Wagon must stay in the range [5.4, 5.5) as moving out of this range would represent an increase in functionality and is thus out of scope for a maintenance line.

Shared component security only line needs are determined by the Maven Core and Maven Plugin lines that depend on the shared components.