Importing Managed Dependencies

Context

In many environments controlling the exact version of dependencies being used is required. Maven provides support for this through its dependency management support in project descriptors. However, as projects grow in size and begin to incorporate components from other development organizations the current implementation is increasingly difficult to use. This is largely due to the limitation that the only way to incorporate managed dependency declarations from other projects is to incorporate those projects via inheritance.

As an example, consider the following projects:

1. ThirdParty 1.0 - contains declarations of third party jars.
2. Project A 1.0 - A project that uses declarations in ThirdParty and produces some number of versioned artifacts.
3. Project B 1.0 - A project unrelated to Project A. It also uses declarations in ThirdParty and produces some number of versioned artifacts.
4. Project C 1.0 - Has dependencies on ThirdParty, Project A, and Project B.

In Maven 2 it is not possible for Project C to use the managed dependencies from both A and B since neither has any direct relationship to the other.

Solution

One possible solution would be to simply allow multiple inheritance. However, the negative consequences of allowing that could introduce more problems than it would solve. For example, there might be conflicting report lists or plugin lists in the two parents that would somehow need to be resolved.

A better solution is to allow only the managed dependencies of other projects to be imported as managed dependencies in the current project. For example,

```
<project>
  <modelVersion>4.0.0</modelVersion>
  <groupId>org.test</groupId>
  <artifactId>ThirdParty-BOM</artifactId>
  <version>1.0</version>
  <packaging>pom</packaging>
  <dependencyManagement>
    <dependencies>
      <dependency>
        <groupId>commons-beanutils</groupId>

<artifactId>commons-beanutils</artifactId>
```

```
        <version>1.7.0</version>
      </dependency>
    </dependencies>
  </dependencyManagement>
</project>

<project>
 <modelVersion>4.0.0</modelVersion>
 <groupId>org.test</groupId>
 <artifactId>A-BOM</artifactId>
 <packaging>pom</packaging>
 <name>A-BOM</name>
 <version>1.0</version>
 <dependencyManagement>
   <dependencies>
     <dependency>
       <groupId>org.test</groupId>
       <artifactId>a</artifactId>
       <version>1.0</version>
     </dependency>
   </dependencies>
 </dependencyManagement>
</project>

<project>
  <parent>
    <artifactId>A-BOM</artifactId>
    <groupId>org.test</groupId>
    <version>1.0</version>
  </parent>
 <modelVersion>4.0.0</modelVersion>
```

```
  <groupId>org.test</groupId>
  <artifactId>A</artifactId>
  <packaging>pom</packaging>
  <name>A</name>
  <version>1.0</version>
  <dependencyManagement>
    <dependencies>
      <dependency>
        <groupId>org.test</groupId>

<artifactId>ThirdParty-BOM</artifactId>
        <version>1.0</version>
        <type>pom</type>
        <scope>import</scope>
      </dependency>
    </dependencies>
  </dependencyManagement>
</project>

<project>
  <modelVersion>4.0.0</modelVersion>
  <groupId>org.test</groupId>
  <artifactId>B-BOM</artifactId>
  <packaging>pom</packaging>
  <name>B-BOM</name>
  <version>1.0</version>
  <dependencyManagement>
    <dependencies>
      <dependency>
        <groupId>org.test</groupId>
        <artifactId>b</artifactId>
```

```
        <version>1.0</version>
      </dependency>
    </dependencies>
  </dependencyManagement>
</project>

<project>
  <parent>
    <artifactId>B-BOM</artifactId>
    <groupId>org.test</groupId>
    <version>1.0</version>
  </parent>
  <modelVersion>4.0.0</modelVersion>
  <groupId>org.test</groupId>
  <artifactId>B</artifactId>
  <packaging>pom</packaging>
  <name>B</name>
  <version>1.0</version>
  <dependencyManagement>
    <dependencies>
      <dependency>
        <groupId>org.test</groupId>

<artifactId>ThirdParty-BOM</artifactId>
        <version>1.0</version>
        <type>pom</type>
        <scope>import</scope>
      </dependency>
    </dependencies>
  </dependencyManagement>
</project>
```

```
<project>
 <modelVersion>4.0.0</modelVersion>
 <groupId>org.test</groupId>
 <artifactId>C-BOM</artifactId>
 <packaging>pom</packaging>
 <name>C-BOM</name>
 <version>1.0</version>
 <dependencyManagement>
   <dependencies>
     <dependency>
       <groupId>org.test</groupId>
       <artifactId>c</artifactId>
       <version>1.0</version>
     </dependency>
   </dependencies>
 </dependencyManagement>
</project>

<project>
  <parent>
    <artifactId>C-BOM</artifactId>
    <groupId>org.test</groupId>
    <version>1.0</version>
  </parent>
 <modelVersion>4.0.0</modelVersion>
 <groupId>org.test</groupId>
 <artifactId>C</artifactId>
 <packaging>pom</packaging>
 <name>C</name>
 <version>1.0</version>
```

```
 <dependencyManagement>
   <dependencies>
     <dependency>
       <groupId>org.test</groupId>

<artifactId>ThirdParty-BOM</artifactId>
       <version>1.0</version>
       <type>pom</type>
       <scope>import</scope>
     </dependency>
     <dependency>
       <groupId>org.test</groupId>
       <artifactId>A-BOM</artifactId>
       <version>1.0</version>
       <type>pom</type>
       <scope>import</scope>
     </dependency>
     <dependency>
       <groupId>org.test</groupId>
       <artifactId>B-BOM</artifactId>
       <version>1.0</version>
       <type>pom</type>
       <scope>import</scope>
     </dependency>
```

```
    </dependencies>
  </dependencyManagement>
</project>
```

Now when project C is constructed it will incorporate all the managed dependencies declared in the ThirdParty, A, and B bill of materials poms. No other behaviors from these projects will be inherited. If one assumes that each of ThirdParty, A and B are declaring several artifacts that are being constructed in each of those projects it makes it fairly easy for project C to specify the exact versions of the artifacts it needs without having to copy the definitions from the other projects.

Votes

Please provide comments with your votes.
+1: Ralph Goers
+0:
-1: