

Multiple Module Maven Projects

Produced
by

Eamonn de Leastar (edelestar@wit.ie)

Dr. Siobhán Drohan (sdrohan@wit.ie)

Department of Computing, Maths & Physics
Waterford Institute of Technology

<http://www.wit.ie>

<http://elearning.wit.ie>



Waterford Institute of Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE



Maven™

The logo for Maven features the word "Maven" in a bold, black, sans-serif font. The letter "v" is replaced by a stylized feather with a gradient of colors from purple at the base to orange at the tip. A small "TM" trademark symbol is positioned to the upper right of the "n".

Modular Approach

Assignment Rubric for Assignment 1

Standard	Core Features [30%]	Presentation [20%]	Tests [30%]	Build Systems [20%]
Baseline	Users/Activities/Locations (lius, la, du)	Plain	basic API tests	none
Good	Start DateTime (la sortBy:) Persistence - XML (l, s)	Pretty	full API tests	maven (build)
Excellent	Persistence -JSON (cff)	Tabular	UI Tests	maven (test)
Outstanding	Persistence - YAML OR Extra Reports	Enhanced	accurate coverage report submitted	maven (modular approach)

POM Relationships

- A module (and their POMS) are potentially related in three ways:
 - (1) Dependency
 - (2) Inheritance
 - (3) Aggregation
- We have covered the Dependency relationship to date, so we will now look at Aggregation.



1. Dependency Relationship

1. Dependency

- Module A requires module B, i.e. module B is to be on Module A's classpath.
- Module A's POM will have an entry specifying a dependency on Module B.
- Used when module is using another project's API, developed in-house or by a third-party.

```
<dependencies>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-remoting</artifactId>
    <version>2.0-rc2</version>
    <type>jar</type>
    <scope>compile</scope>
  </dependency>
  <dependency>
    <groupId>cglib</groupId>
    <artifactId>cglib</artifactId>
    <version>2.1</version>
    <type>jar</type>
    <scope>runtime</scope>
  </dependency>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.7</version>
    <scope>test</scope>
  </dependency>
</dependencies>
```

Maven™

2. Inheritance Relationship

2. Inheritance

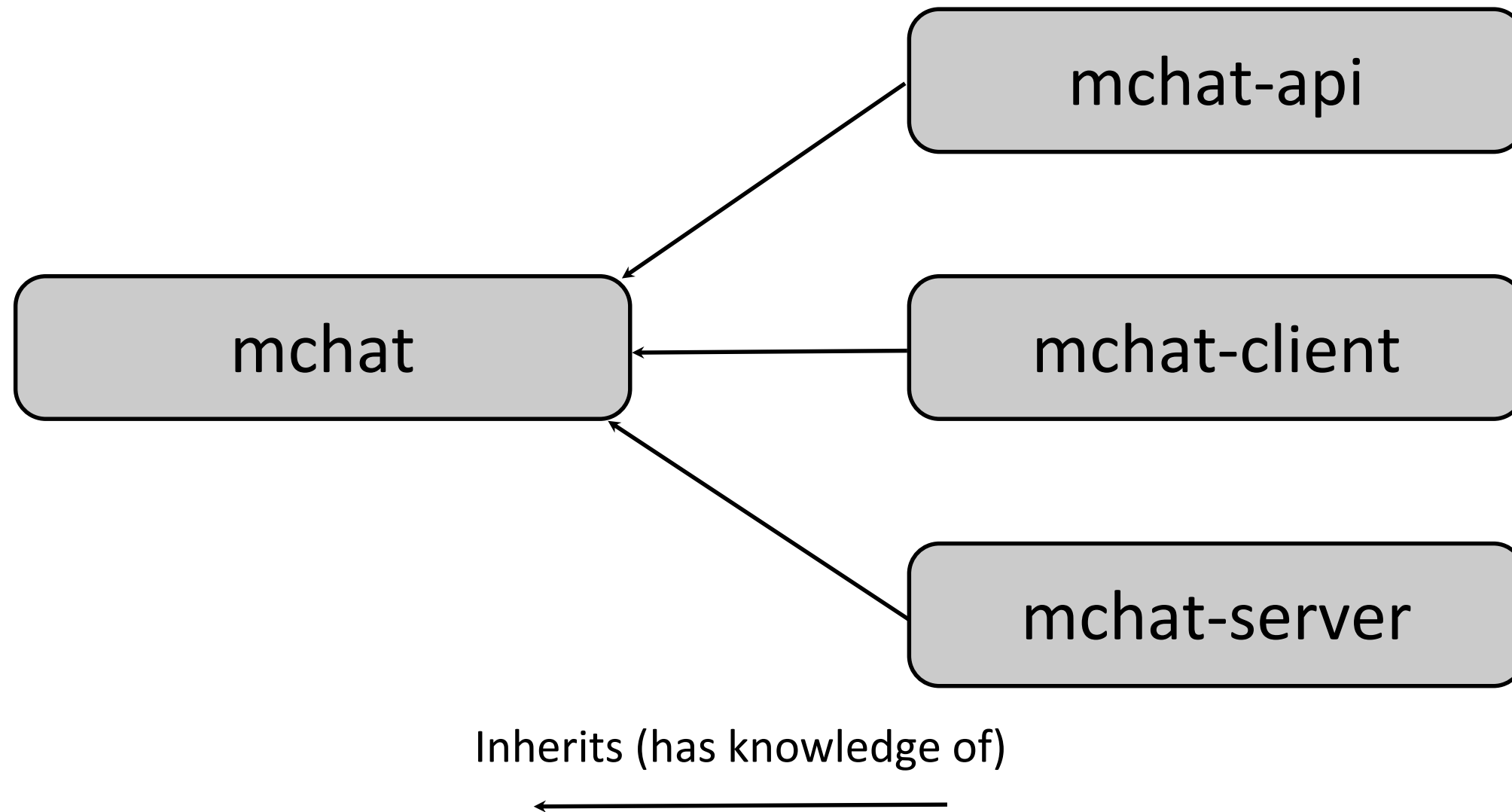
- Applies to general project configuration, and not necessarily explicit dependency management.
- Module A's POM inherits module B's POM:
 - this automatically acquires all of Module B's settings, configurations and dependencies.

2. Inheritance – Super POM

- All maven POMs have an implicit “parent”
- Called the Super POM is Maven's default POM (somewhat like the “cosmic” base class Object in java).
- All POMs extend the Super POM unless explicitly set, meaning the configuration specified in the Super POM is inherited by the POMs you created for your projects.
- Located in the maven.jar file, it is not editable or necessarily intended to be human readable
- Can viewed by entering: `mvn help:effective-pom`

2. Inheritance - Hierarchies

- You can define your own POM inheritance hierarchy.



Parent

```
<groupId>msccomm.mchat</groupId>  
<artifactId>mchat</artifactId>  
<packaging>pom</packaging>  
<version>1.0</version>  
<name>mchat</name>  
<url>http://vle.wit.ie</url>
```

- *packaging - pom* indicates a parent project – i.e. does not have code of its own
- The Super POM is its parent, by default.

Child

```
<parent>  
  <artifactId>mchat</artifactId>  
  <groupId>msccomm.mchat</groupId>  
  <version>1.0</version>  
</parent>  
<artifactId>mchat-server</artifactId>  
<version>1.0</version>  
<name>mchat-server</name>
```

- mchat-server inherits from mchat

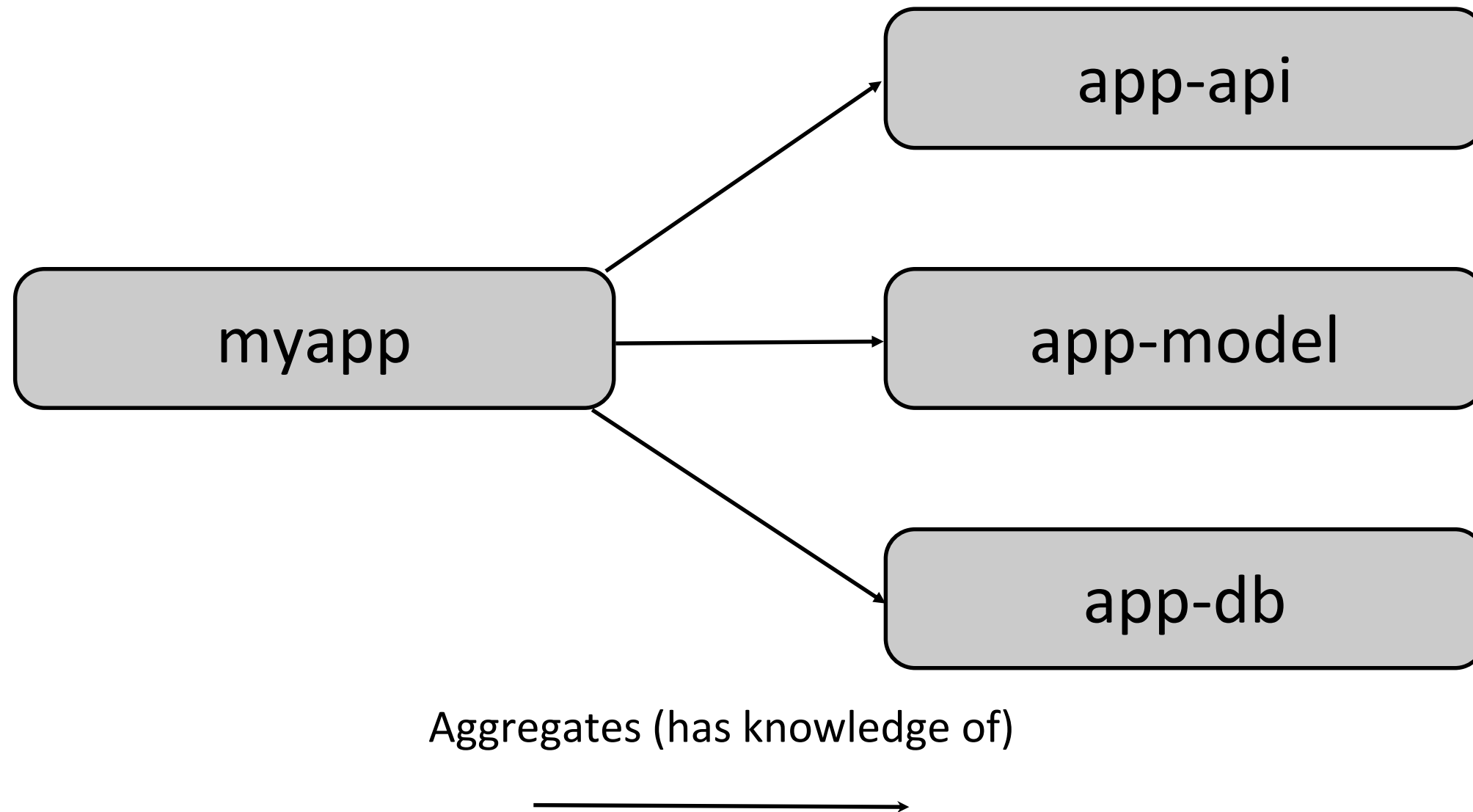


3. Aggregation Relationship

3. Aggregation

- Recall the '*single artifact per project*' principle.
- Non-trivial projects should be broken into modules (sub-projects).
- Facilitates reusability of code, manageability of projects.
- Modules are maven projects (with their own POM), perhaps listed in a parent POM, and executed as a set.

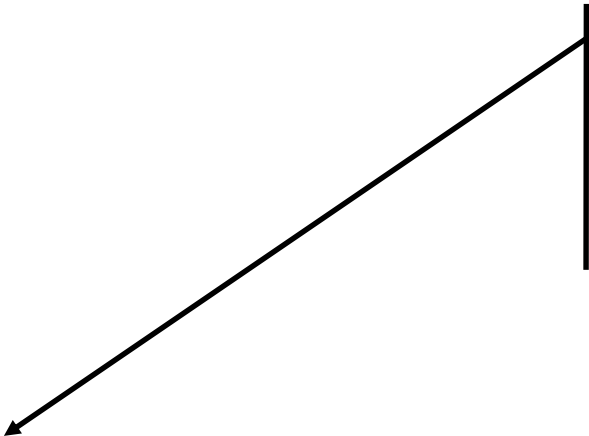
3. Aggregation – high level example



POM (Project) relationships – Aggregation.

- Typically the parent POM's folder has a subfolder for each module.
- Building from the parent folder will cause a build of each modules

```
<modelVersion>4.0.0</modelVersion>
<groupId>msccomm.mchat</groupId>
<artifactId>mchat</artifactId>
<packaging>pom</packaging>
<version>1.0</version>
<name>mchat</name>
<url>http://vle.wit.ie</url>
<modules>
  <module>mchat-api</module>
  <module>mchat-client</module>
  <module>mchat-server</module>
</modules>
```



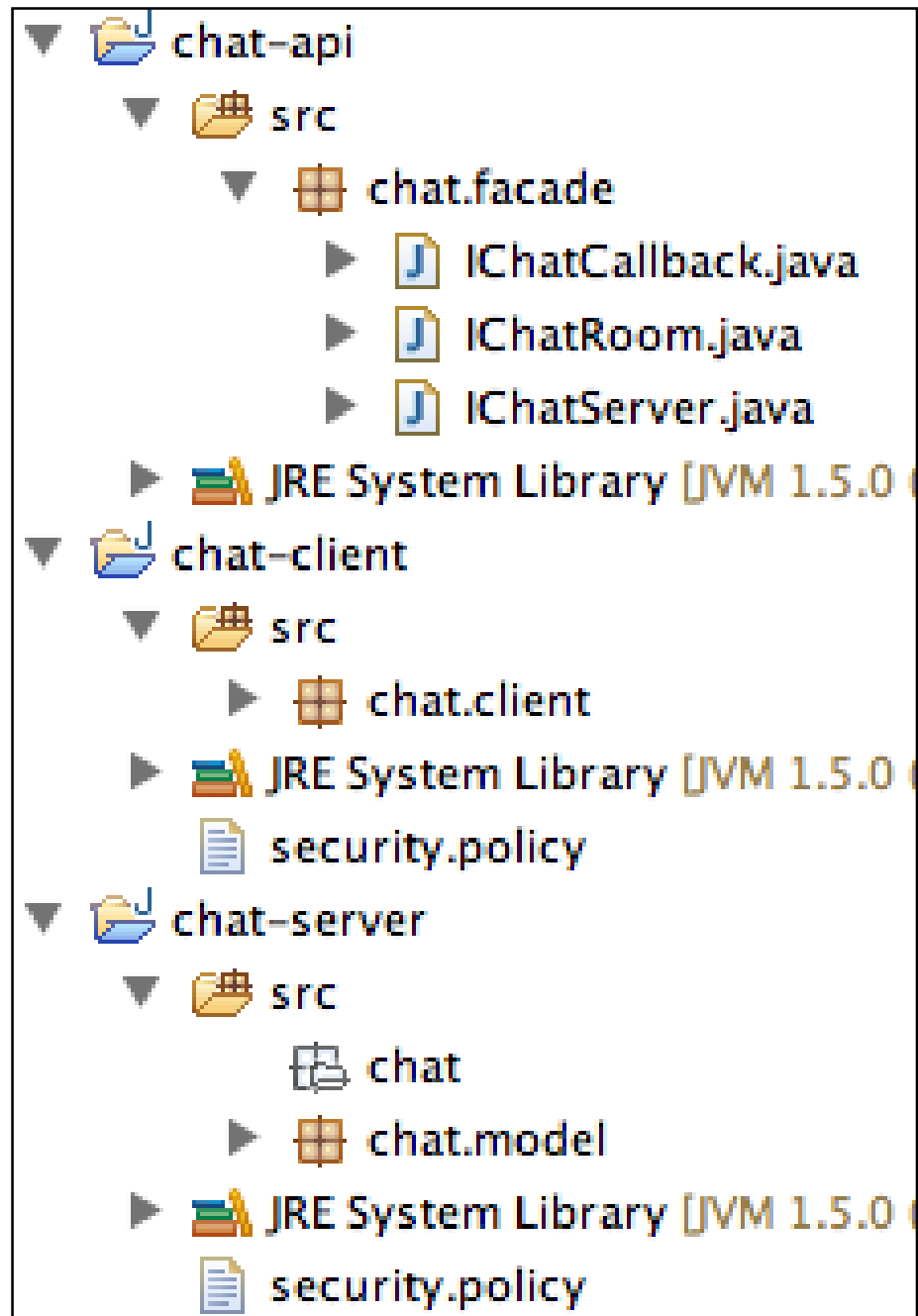
Names of folders housing
the module projects

Maven™

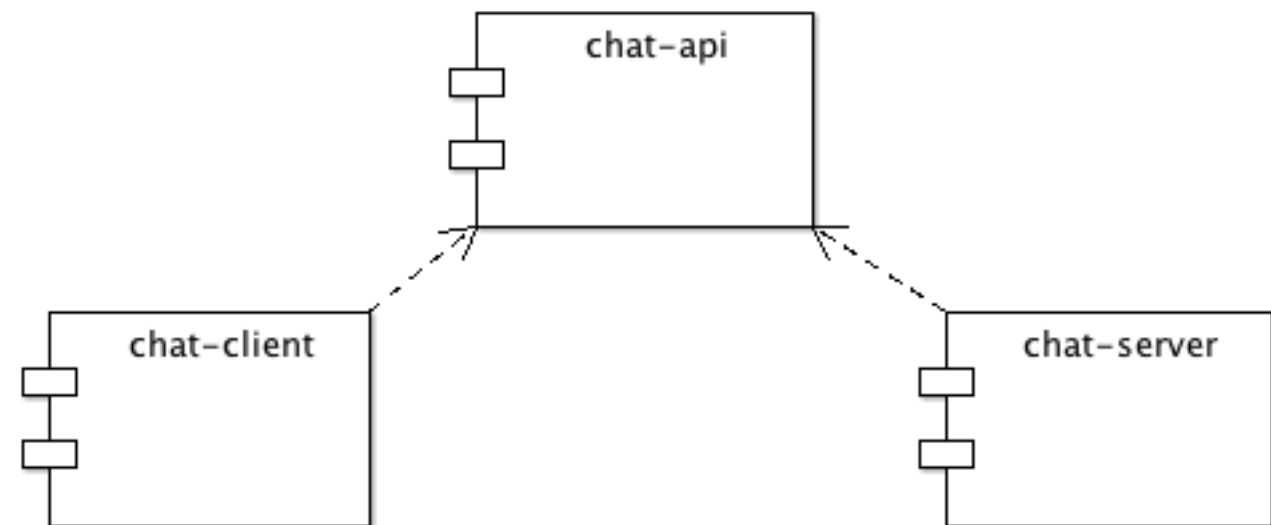
The logo for Maven, featuring the word "Maven" in a bold, black, sans-serif font. The letter "v" is replaced by a stylized feather with a gradient of colors from purple at the base to orange at the tip. A small "TM" trademark symbol is positioned to the upper right of the "n".

Example – Multi-Module Approach

A "Chat" Project

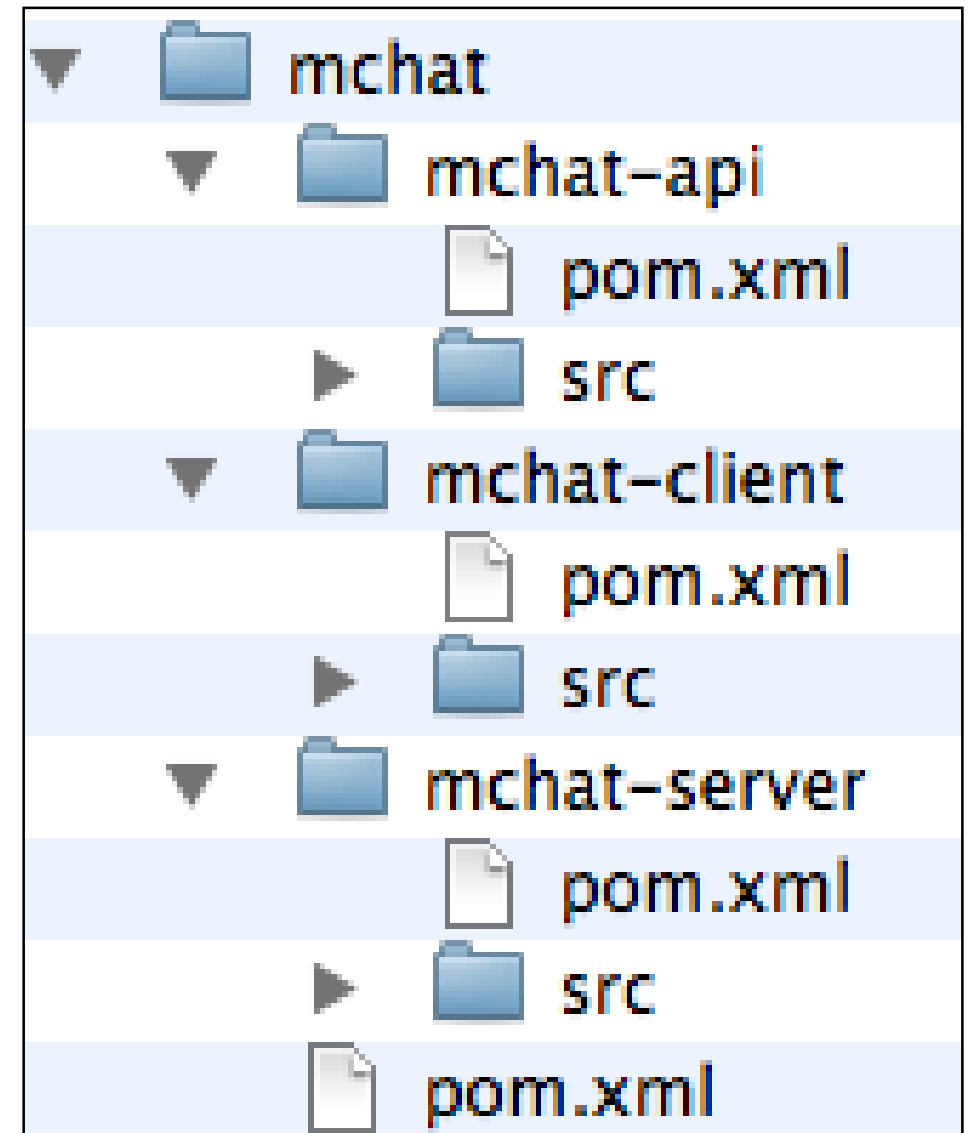


- Dependency Structure:

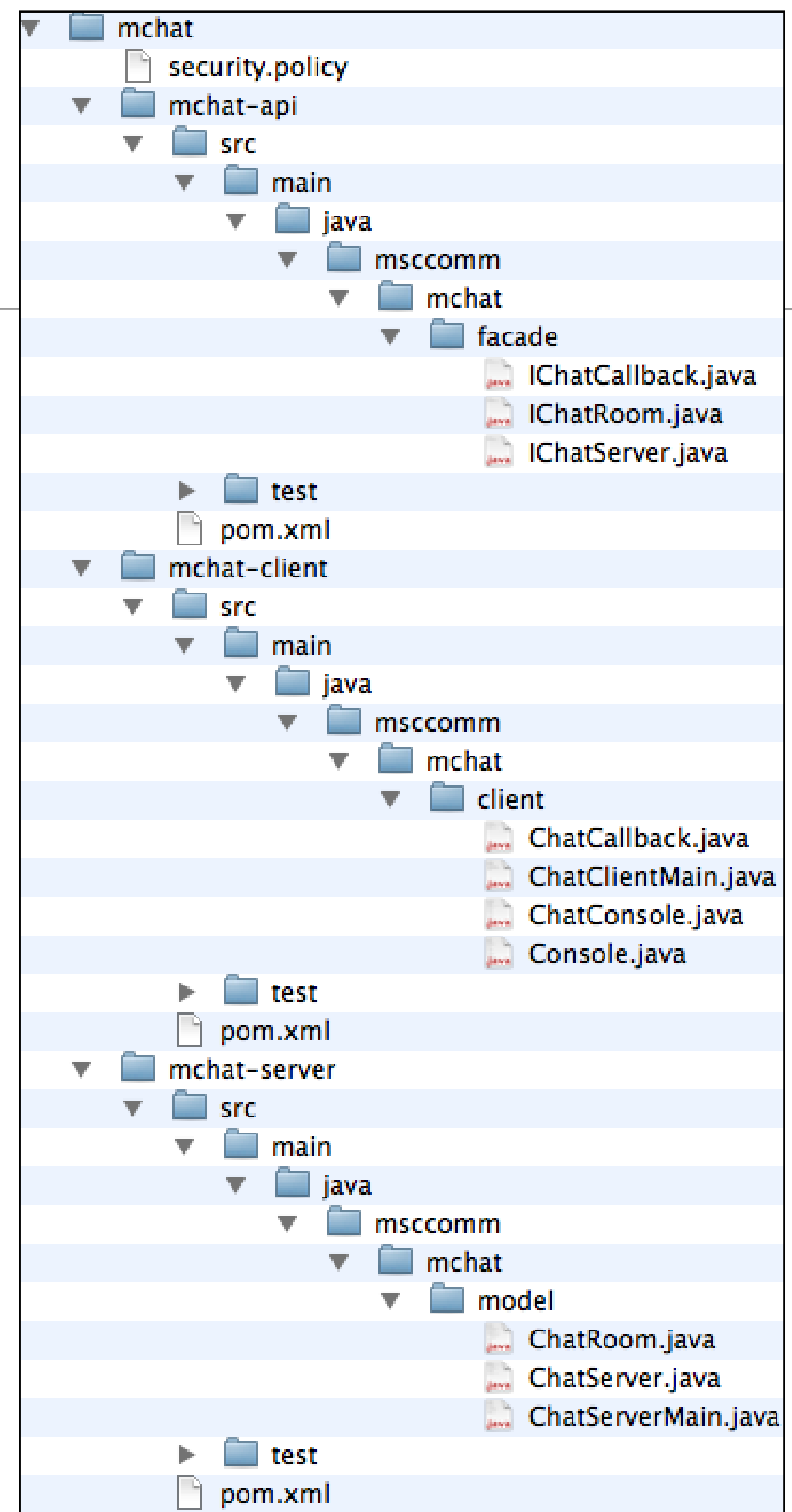
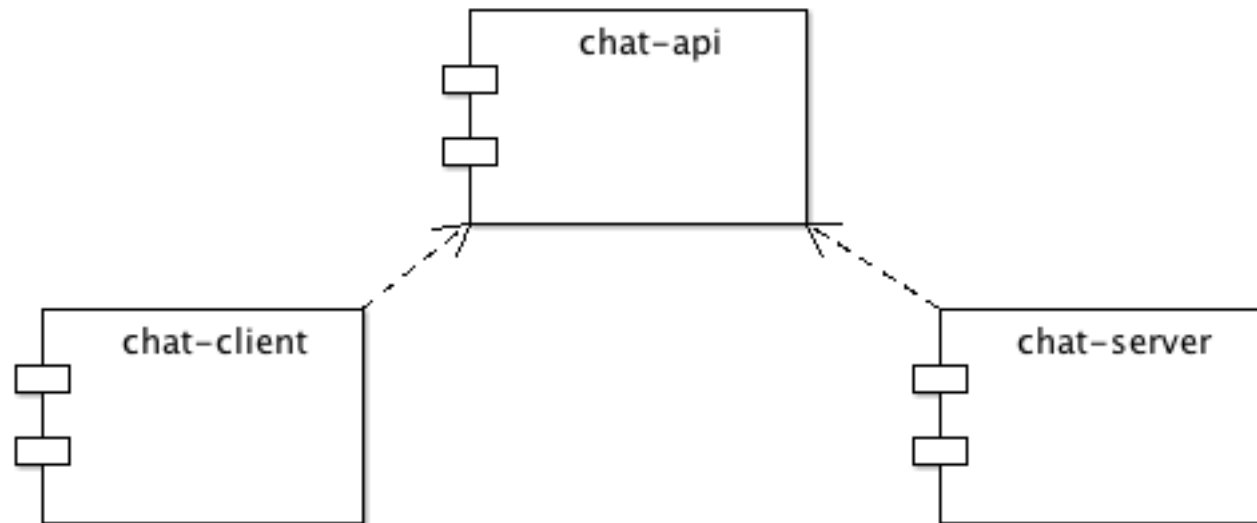


Dependencies in Maven

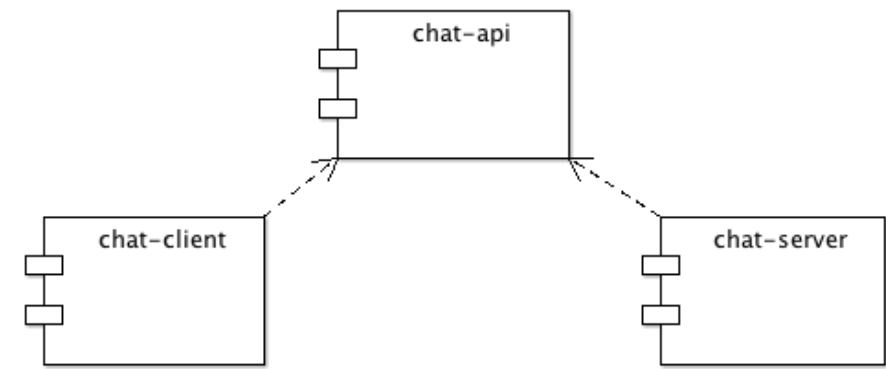
- Define Parent POM for mchat “group” of projects.
- Have mchat-api, mchat-client and mchat-server as children of this parent.
- All stored hierarchy in subfolders



mchat group



mchat (parent)

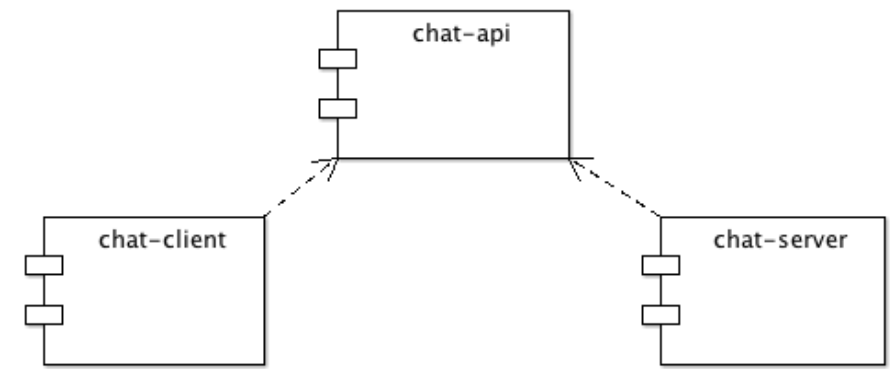


- Defines
 - groupId
 - version
 - base dependencies
 - build options

... all of which will be inherited by child modules.
- Specifically indicates modules that it aggregates (modules tag)

```
<project xmlns="..."
  <modelVersion>4.0.0</modelVersion>
  <groupId>msccomm.mchat</groupId>
  <artifactId>mchat</artifactId>
  <packaging>pom</packaging>
  <version>1.0</version>
  <name>mchat</name>
  <url>http://vle.wit.ie</url>
  <dependencies>
  </dependencies>
  <modules>
    <module>mchat-api</module>
    <module>mchat-server</module>
    <module>mchat-client</module>
  </modules>
</project>
```

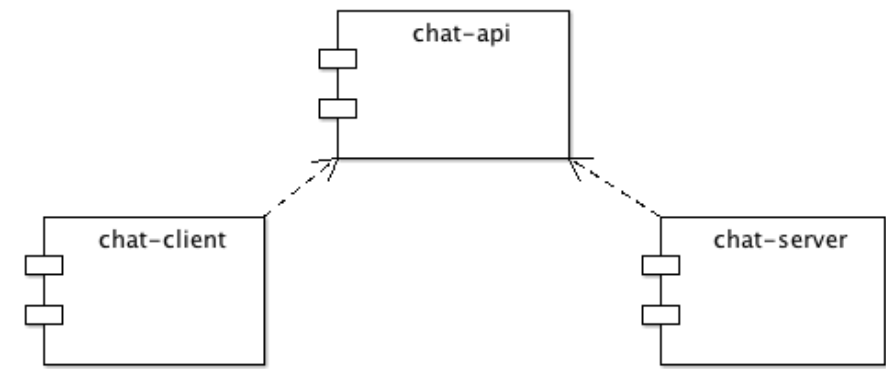
mchat-api



- Standalone module, without any explicit dependences (apart from those inherited from parent).
- Also inherits parent version, parent groupId and web site.

```
<project>
  <parent>
    <artifactId>mchat</artifactId>
    <groupId>msccomm.mchat</groupId>
    <version>1.0</version>
  </parent>
  <modelVersion>4.0.0</modelVersion>
  <artifactId>mchat-api</artifactId>
  <name>mchat-api</name>
</project>
```

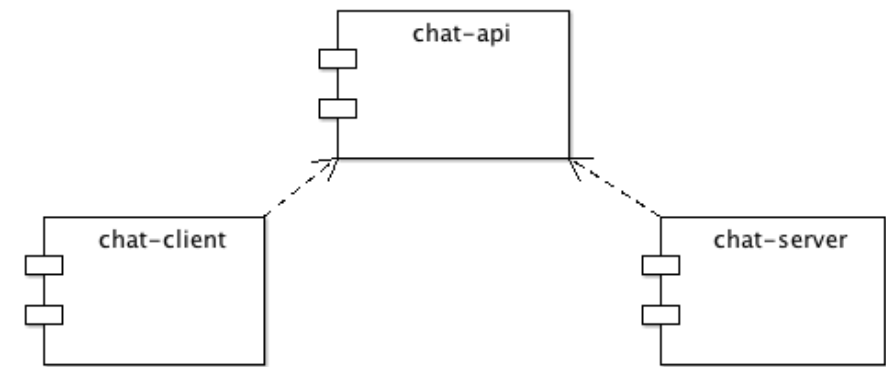
mchat-client



- Inherits same properties as mchat-api
- Augments with explicit dependency on the mchat-api module

```
<project>
  <parent>
    <artifactId>mchat</artifactId>
    <groupId>msccomm.mchat</groupId>
    <version>1.0</version>
  </parent>
  <modelVersion>4.0.0</modelVersion>
  <artifactId>mchat-client</artifactId>
  <name>mchat-client</name>
  <version>1.0</version>
  <dependencies>
    <dependency>
      <groupId>msccomm.mchat</groupId>
      <artifactId>mchat-api</artifactId>
      <version>1.0</version>
    </dependency>
  </dependencies>
</project>
```

mchat-server



- Similar to mchat-client, inherits key properties from parent.
- Explicit dependency on mchat-api

```
<project>
  <parent>
    <artifactId>mchat</artifactId>
    <groupId>msccomm.mchat</groupId>
    <version>1.0</version>
  </parent>
  <modelVersion>4.0.0</modelVersion>
  <artifactId>mchat-server</artifactId>
  <name>mchat-server</name>
  <dependencies>
    <dependency>
      <groupId>msccomm.mchat</groupId>
      <artifactId>mchat-api</artifactId>
      <version>1.0</version>
    </dependency>
  </dependencies>
</project>
```

Aggregation

- Because mchat (the parent) explicitly aggregates the three modules, we can operate on the mchat parent for common operations.

mvn clean

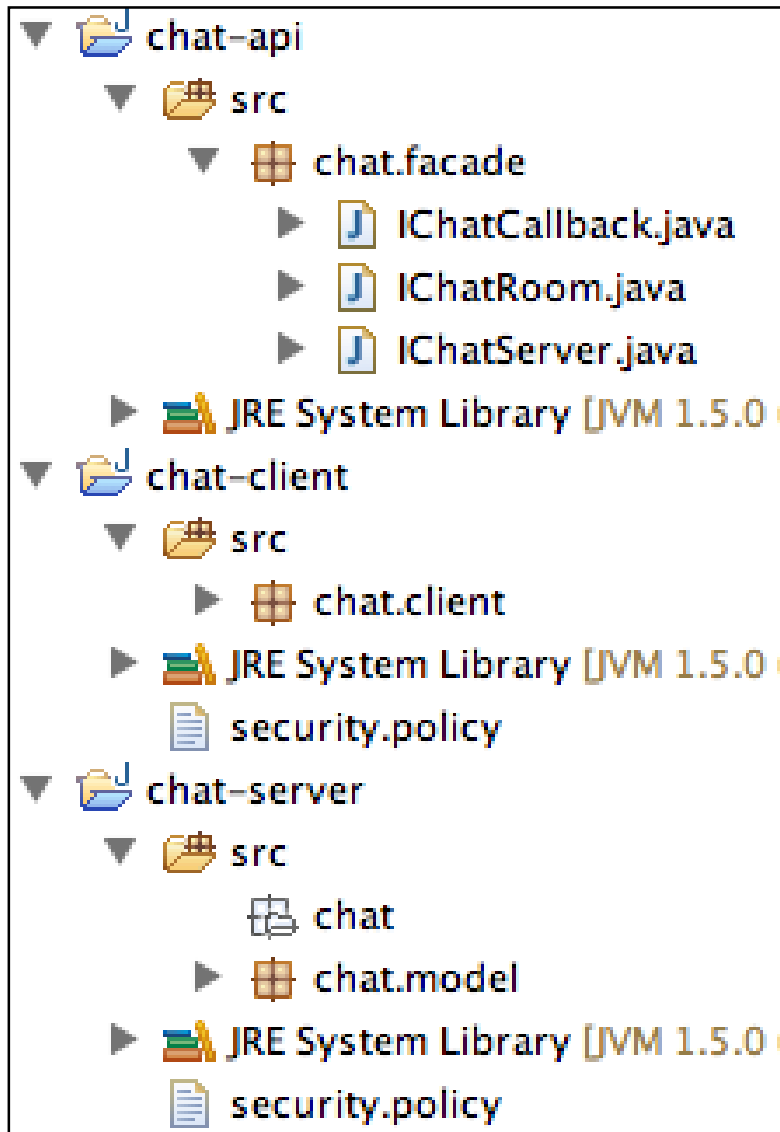
mvn package

mvn install

- ...will all perform these goals on each component.

```
<groupId>msccomm.mchat</groupId>
<artifactId>mchat</artifactId>
<packaging>pom</packaging>
...
<modules>
  <module>mchat-api</module>
  <module>mchat-server</module>
  <module>mchat-client</module>
</modules>
...
</project>
```

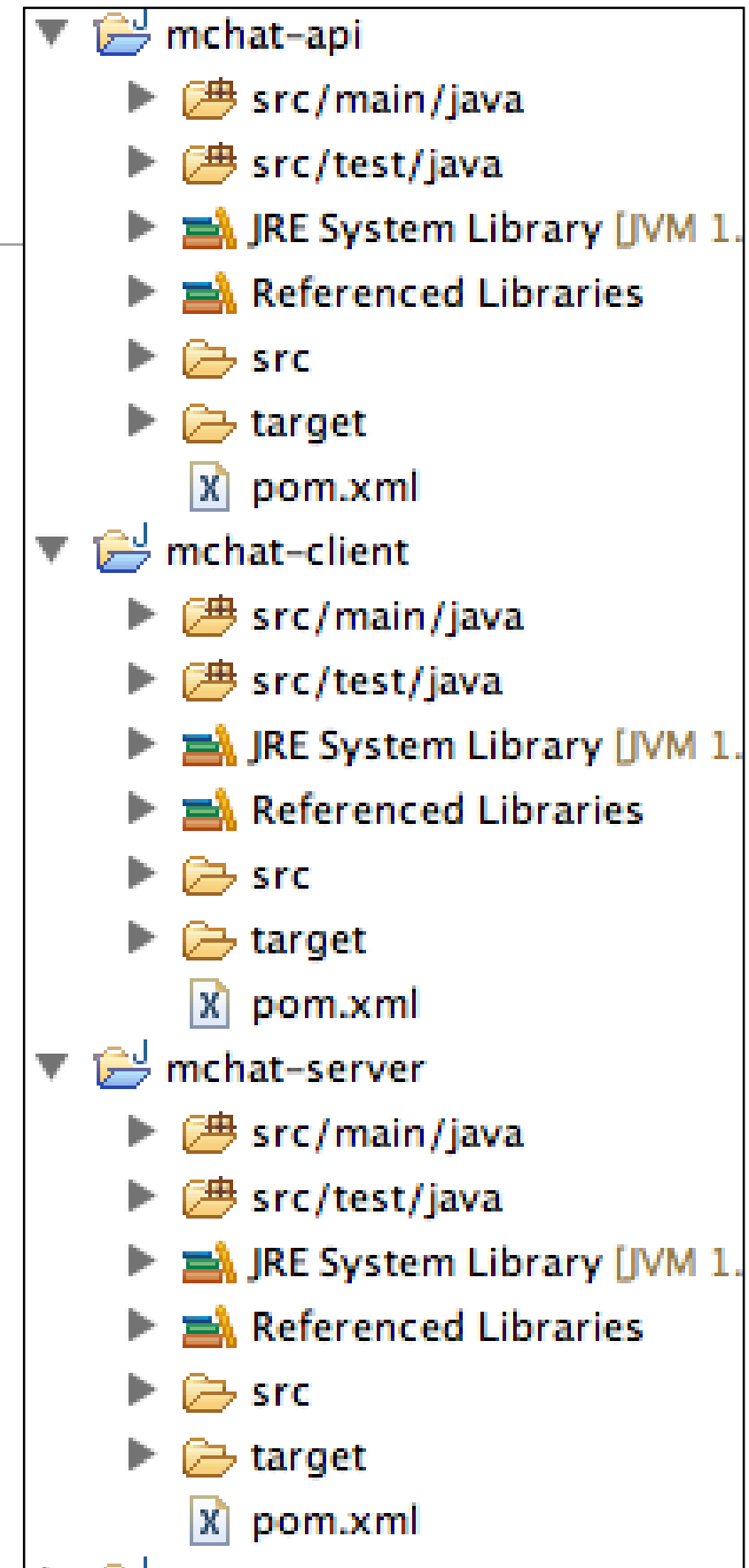
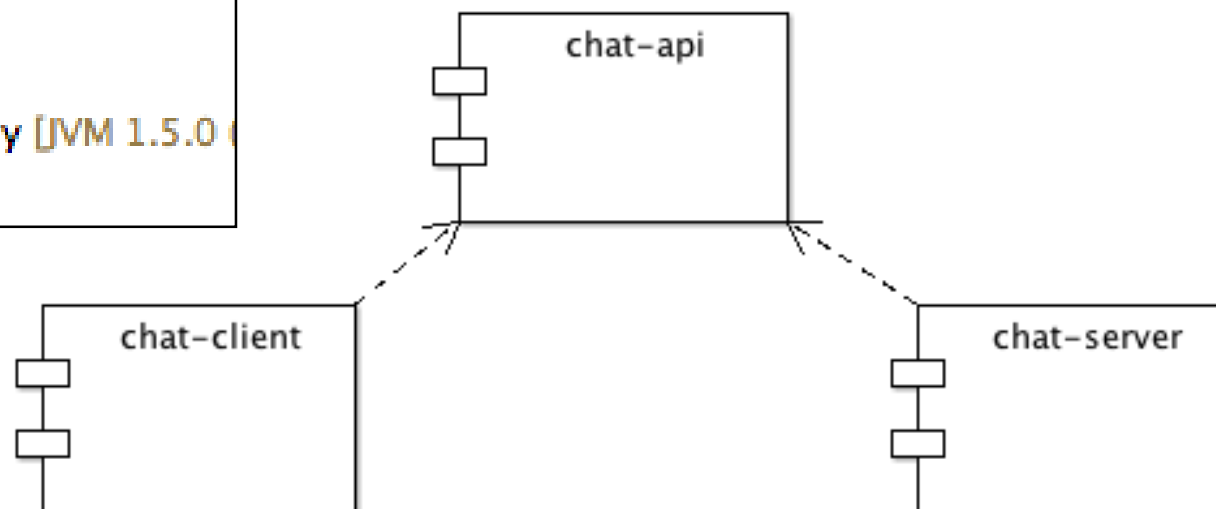

Eclipse View



- The Eclipse projects generated by:

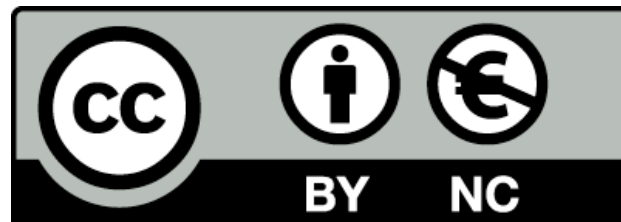
`mvn eclipse:eclipse`

- ...will generate projects with the inter-module dependencies intact



More Reading

- <https://maven.apache.org/plugins/maven-assembly-plugin/examples/multimodule/module-source-inclusion-simple.html>



Except where otherwise noted, this content is licensed under a [Creative Commons Attribution-NonCommercial 3.0 License](http://creativecommons.org/licenses/by-nc/3.0/).

For more information, please see <http://creativecommons.org/licenses/by-nc/3.0/>

