Large Scale Programming with Modules

Concepts of Programming Languages
19 December 2019

Johannes Weigend (QAware GmbH)
University of Applied Sciences Rosenheim

Enterprise Programming

- Large codebases
- Conflicting dependencies
- Stable dependencies
- Compatiblity requirements

What is a Module?

- A module is an unit of Programming, Testing, Distribution, Versioning and Visibility
- Languages have diffent interpretations and focus
- Decomposition into modules is a well understood

"The major advancement in the area of modular programming has been the development of coding techniques and assemblers which (l) allow one module to be written with little knowledge of the code in another module, and (2) allow modules to be reassembled and replaced without reassembly of the whole system."

On the Criteria To Be Used in Decomposing Systems into Modules, David Parnas 1972

(https://www.win.tue.nl/~wstomv/edu/2ip30/references/criteria_for_modularization.pdf)

Visibility

- Makes sure that certain content of a module is private and can not be accessed be other modules
- The benefits are the same like OO encapsulation but at larger scale
- Other modules can only depend on the public (API) part of a module
- Modules hide complexity from clients (information hiding)

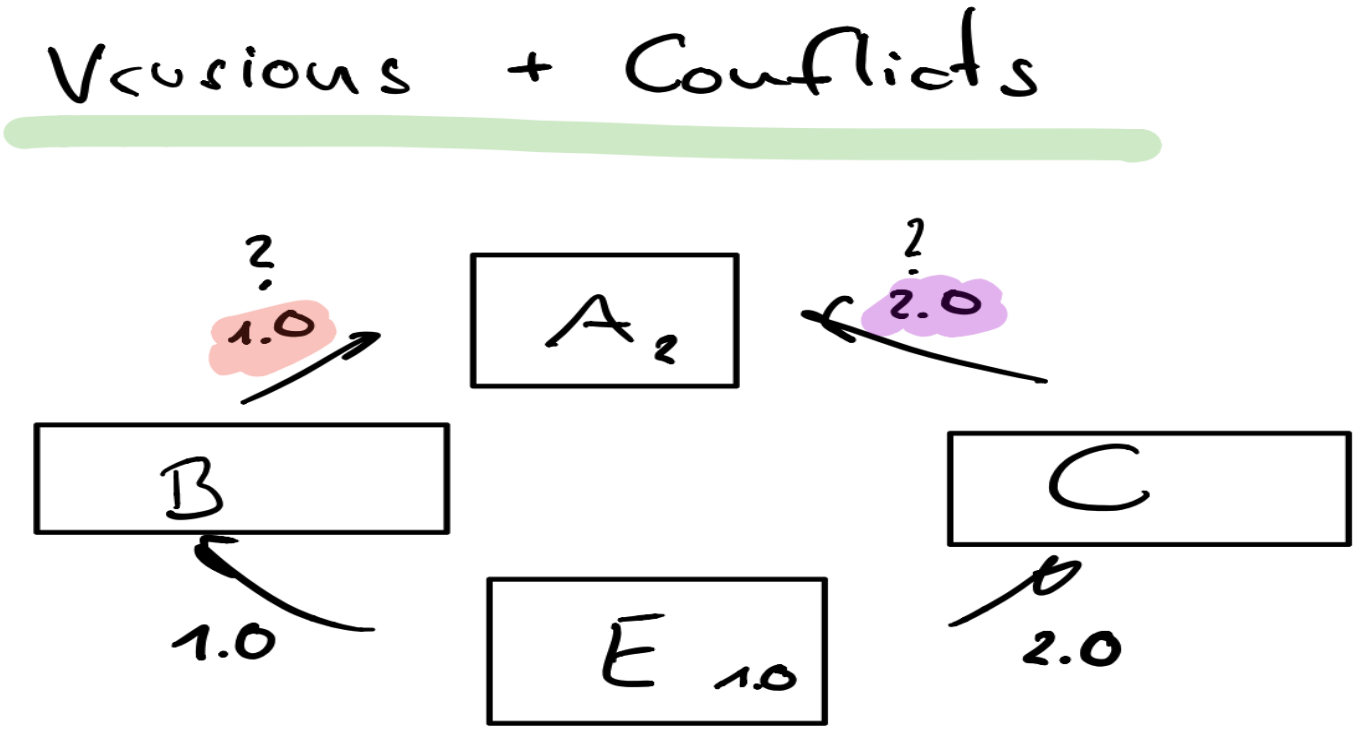
Distribution and Replaceability

- Modules are monolithic deployment units
- Modules could be separate developed and released
- Languages differ in the way an application is assembled from modules (static or dynamic linking)
- Modules can be exchanged without affecting the whole system

Versioning

- Modules should be versionized to guarantee stable dependencies
- The version should make the module immutable, so a module can be clearly identified by its version
- Languages differ greatly on that issue (From support of naming conventions to build tools (maven))

Versions and Conflicts



Module System in Java

- Java knows Modules since Java SE 9 (Project Jigsaw)
- Java Modules focus on Distribution and Visibility but not on Versioning!
- Java need 3rd party tools to implement versionized dependencies (Maven, Gradle ...)

www.sigs-datacom.de/uploads/tx_dmjournals/weigend_JS_05_16_6n6J.pdf (https://www.sigs-datacom.de/uploads/tx_dmjournals/weigend_JS_05_16_6n6J.pdf)

The Go Module System

- There is an evolution in the Go module systems
- Before 1.11: Go dep was the tool of choice to implement safe, versionized dependencies
- Since 1.11: Go modules is the concept for the future

The Go Module System (https://github.com/golang/go/wiki/Modules)

The Principles of Versioning in Go

Repeatability
Make sure that build results never change over time !!!

Compatibility
Make sure that released code stays compatible. Make incompatible changes explicit visible to users of your code !!!

Cooperation
Make sure your code ist stable with the latest minor version of a dependent library

Semantic Versioning

- Distinguish MAJOR, MINOR and PATCH version numbers
- A version number is of the form: <MAJOR>.<MINOR>.<PATCH> (e.g 1.2.0, 0.9.0-alpha)
- Semantic versioning defines several rules how and when to change the version number

semver.org/ (https://semver.org/)

The Design of the Go Module System

Russ Cox (@_rsc, rsc@swtch.com)
c/o Google
5 Cambridge Center
Cambridge, MA 02142

www.youtube.com/watch?v=F8nrpe0XWRg (https://www.youtube.com/watch?v=F8nrpe0XWRg)

The Go Module System in Action

youtu.be/V8FQNen4WAA (https://youtu.be/V8FQNen4WAA)

youtu.be/aeF3l-zmPsY (https://youtu.be/aeF3l-zmPsY)

Hands On

- Write a Go module mail with API and impl
- The Mail implementation should log a message with logrus (https://github.com/sirupsen/logrus)
- Change your module dependency to a former logrus version
- Write a second Go module client which uses the mail module
- Make a incompatible change to your mail API and increase the major version number

Thank you

Johannes Weigend (QAware GmbH)
University of Applied Sciences Rosenheim
johannes.weigend@qaware.de (mailto:johannes.weigend@qaware.de)
http://www.qaware.de (http://www.qaware.de)
@johannesweigend (http://twitter.com/johannesweigend)