Software Engineering Theory and Practice

School of Computing	
Title	Software Engineering Theory and Practice
Module Coordinator	Steven Ossont
Email	steven.ossont@port.ac.uk
Code	M30819
Moodle	https://moodle.port.ac.uk/course/view.php?id=11429

U30819: Software Engineering Theory and Practice

Continuous Integration

https://moodle.port.ac.uk/course/view.php?id=11429

Software Configuration Management

Configuration management involves four activities:

1. Version management

   Keeping track of the multiple versions

2. **System building**

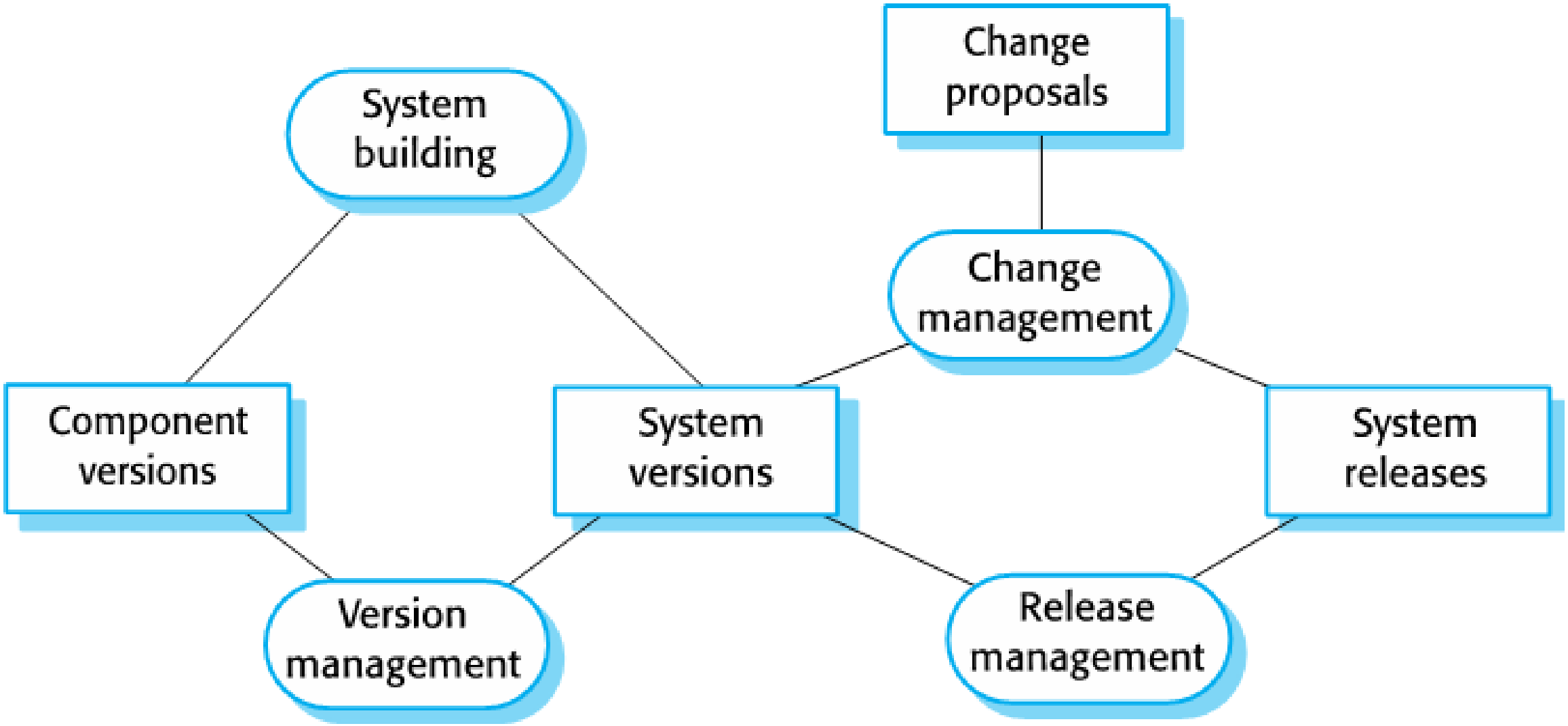
   The process of assembling components; data; libraries.

3. Change management

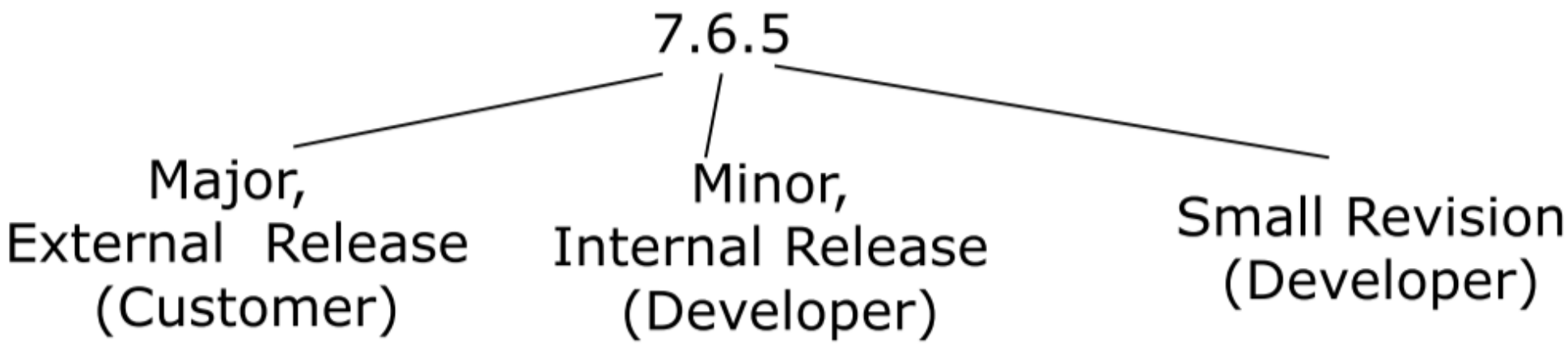
   Keeping track of requests for changes / which changes should be implemented.

4. Release management

   Preparing software for release


Change proposals
System building
Change management
Component versions
System versions
System releases
Version management
Release management

Terminology -- Baseline naming schemes



Software Configuration Management - Versioning

> Free book: https://git-scm.com/book/en/v2

Components for release

As well as the the executable code of the system, a release may also include:

- configuration files
- data files
- an installation program
- documentation
- packaging and publicity

Continuous Integration

- You are working as part of a team on developing a new software system for an estate agency
- You are 1 of 25 developers and working closely with the 15 testers on the project
- The client has decided to add a new search criteria for the property search feature of the system
- They have informed the team leader who has assigned this development task to you

Continuous Integration

- Step 1: Take a copy of the current source code - integrated and working!
  - Check out the repository as a whole
- Step 2: Change the code as necessary
- Step 3: Build the system locally
  - Compile the new version locally [, run all tests automatically]
  - Generate an executable version of the system
- Step 3.1: If build successful:
  - Consider committing the change back to the server
- Step 3.2 If build fails:
  - Go to Step 2

> https://martinfowler.com/articles/continuousIntegration.html

Continuous Integration

Meanwhile, other developers have made changes to the repository!

Continuous Integration

- Step 5: Update the local working copy with the other changes made
- Step 6: Rebuild system locally
  - Step 6.1: If rebuild fails locally: Your responsibility to fix it; Change code until build successful
  - Step 6.2: If rebuild successful locally: Commit changes to server
- Step 7: Rebuild system on server

> Job done only when the changes committed to the server build successfully on the server!

Why Build Twice?

> A successful build in **your** environment (local), can **fail** on the server

- Changing the environment configuration leads to build failing even if the code works
  - Programming language version
  - Operating system
  - Libraries version

> How do you make sure the system builds in a different environment?

Continuous Integration

- *Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily - leading to multiple integrations per day. Each integration is verified by an automated build (including test) to detect integration errors as quickly as possible.* https://martinfowler.com/articles/continuousIntegration.html
- In software engineering, continuous integration (CI) is the practice of merging all developers' working copies to a shared mainline several times a day. https://en.wikipedia.org/wiki/Continuous_integration

> **Each** integration is accompanied by an automated build

How to put this in practice?

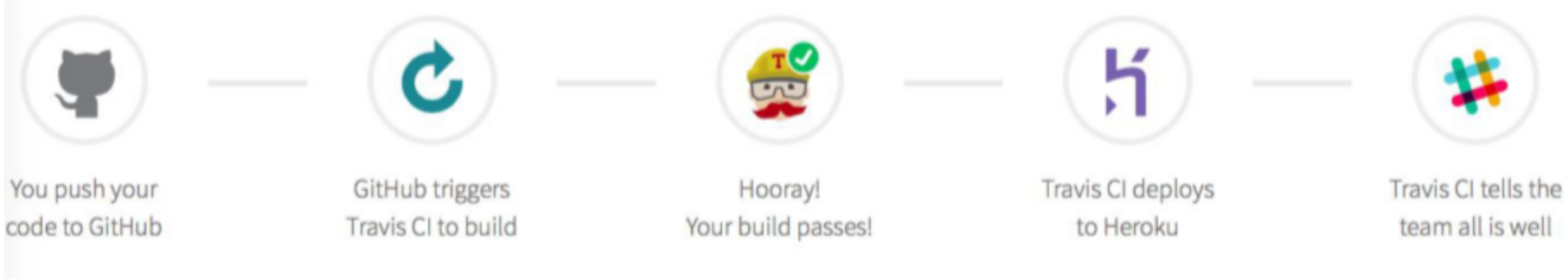
- Maintain a single repository for the project
- Automate the build - TravisCl
  - Include everything in the build!
- Everyone commits every day [, every change]
- Every commit must build on the server
- Fix broken builds immediately

> Be clever with branches

Reasons

- Time is best used to write code not running tests
- Local variable (env) -- system as a whole

Tools for Continuous Integration



https://travis-ci.com

https://www.heroku.com

> How is this different from running the build yourself? **Automation!**

Questions?

- https://www.youtube.com/watch?v=Uft5KBimzyk
- https://www.youtube.com/watch?v=aUW5GAFhu6s
- https://git-scm.com/book/en/v2

> Ensure your are familiar with this excellent book!