THE UNIVERSITY OF WESTERN AUSTRALIA

Agile Development

CITS3403 Agile Web Development

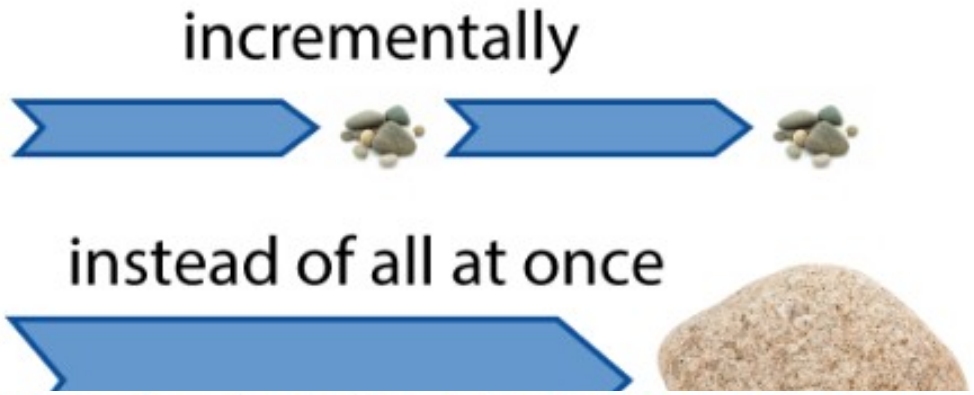
From Agile in a Nutshell, by Jonathan Rassmusson

Further reading: The agile handbook

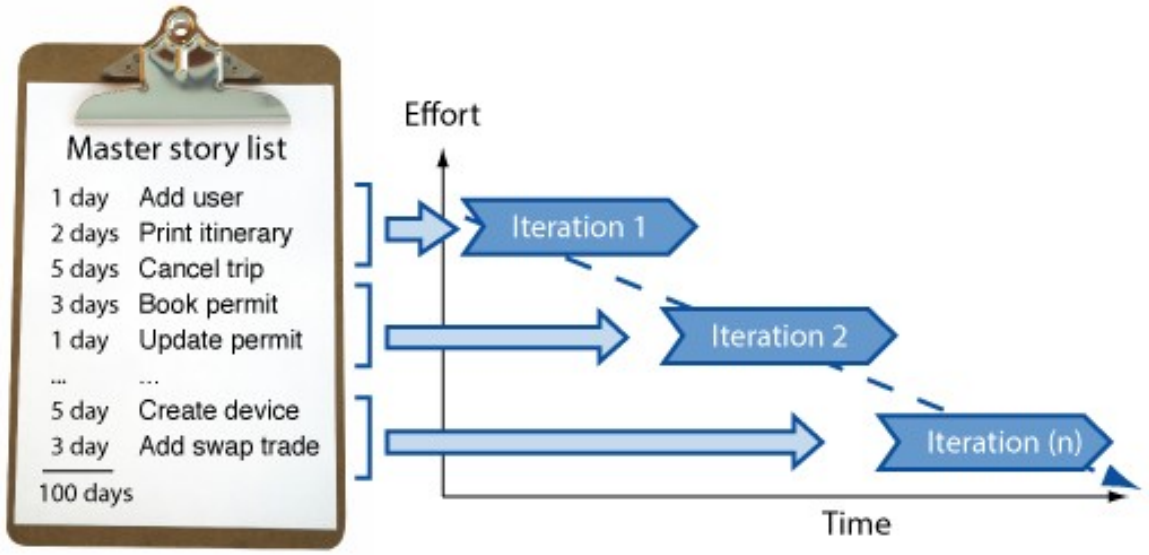
Semester 1, 2019

What is Agile

Agile is a time boxed, iterative approach to software delivery that builds software incrementally from the start of the project, instead of trying to deliver it all at once near the end.

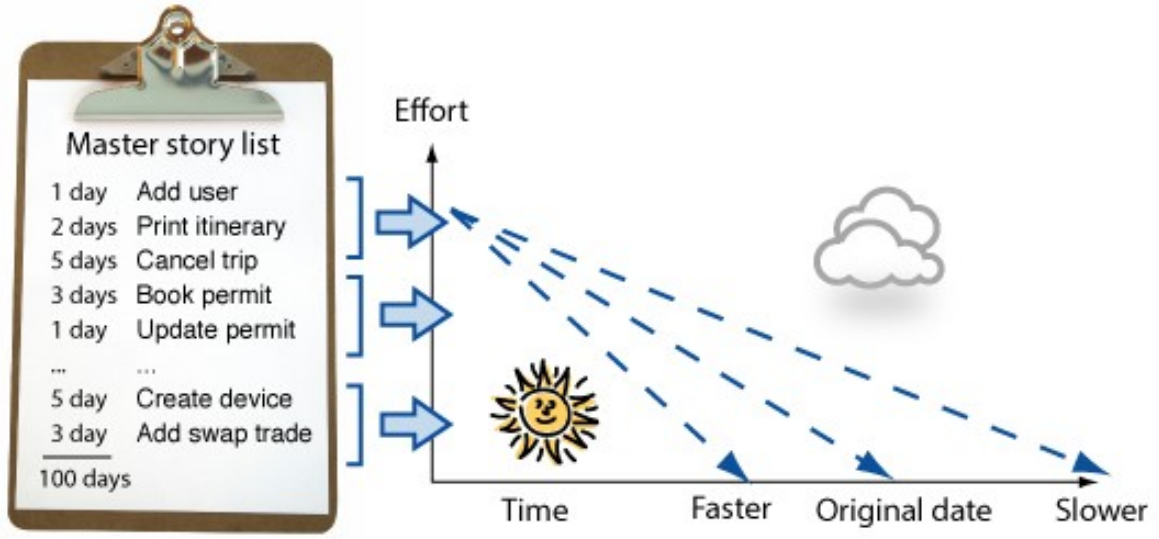


It works by breaking projects down into little bits of user functionality called user stories, prioritizing them, and then continuously delivering them in short two week cycles called iterations.

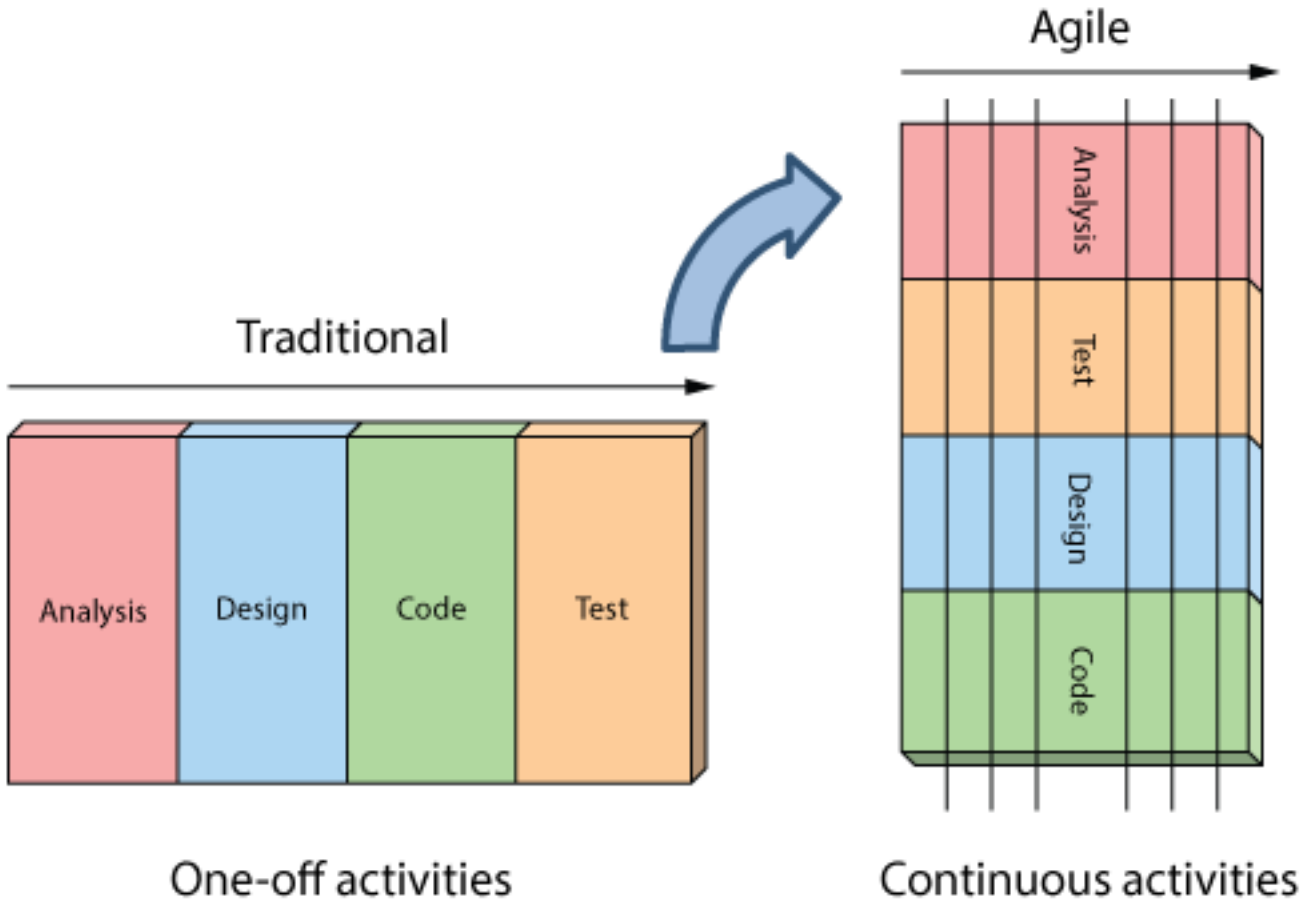


How does it work:

- You make a list
- You size things up
- You set priorities
- You start executing
- You update the plan as you go…

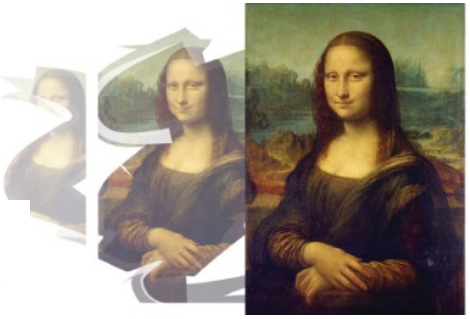


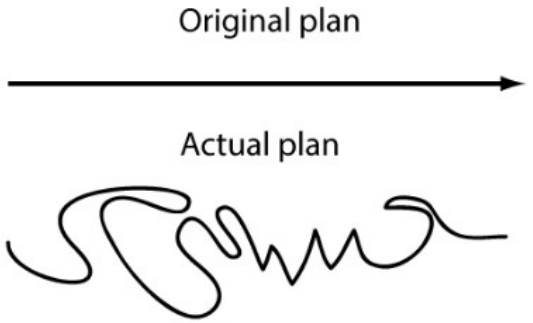
Analysis, design, coding testing are continuous

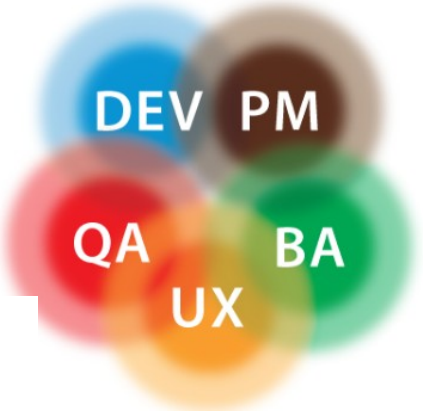


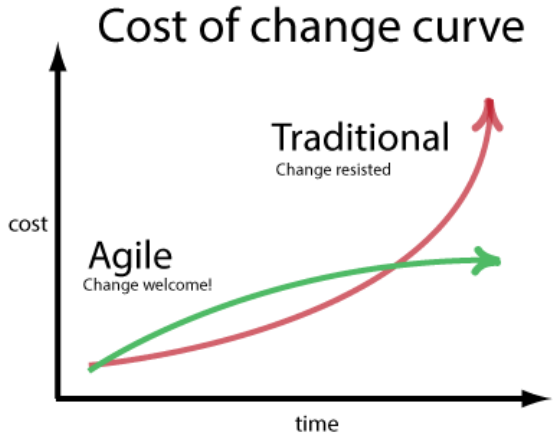
How is Agile different

- Development is iterative
- Planning is adaptive
- Roles blur
- Requirements change
- Working software



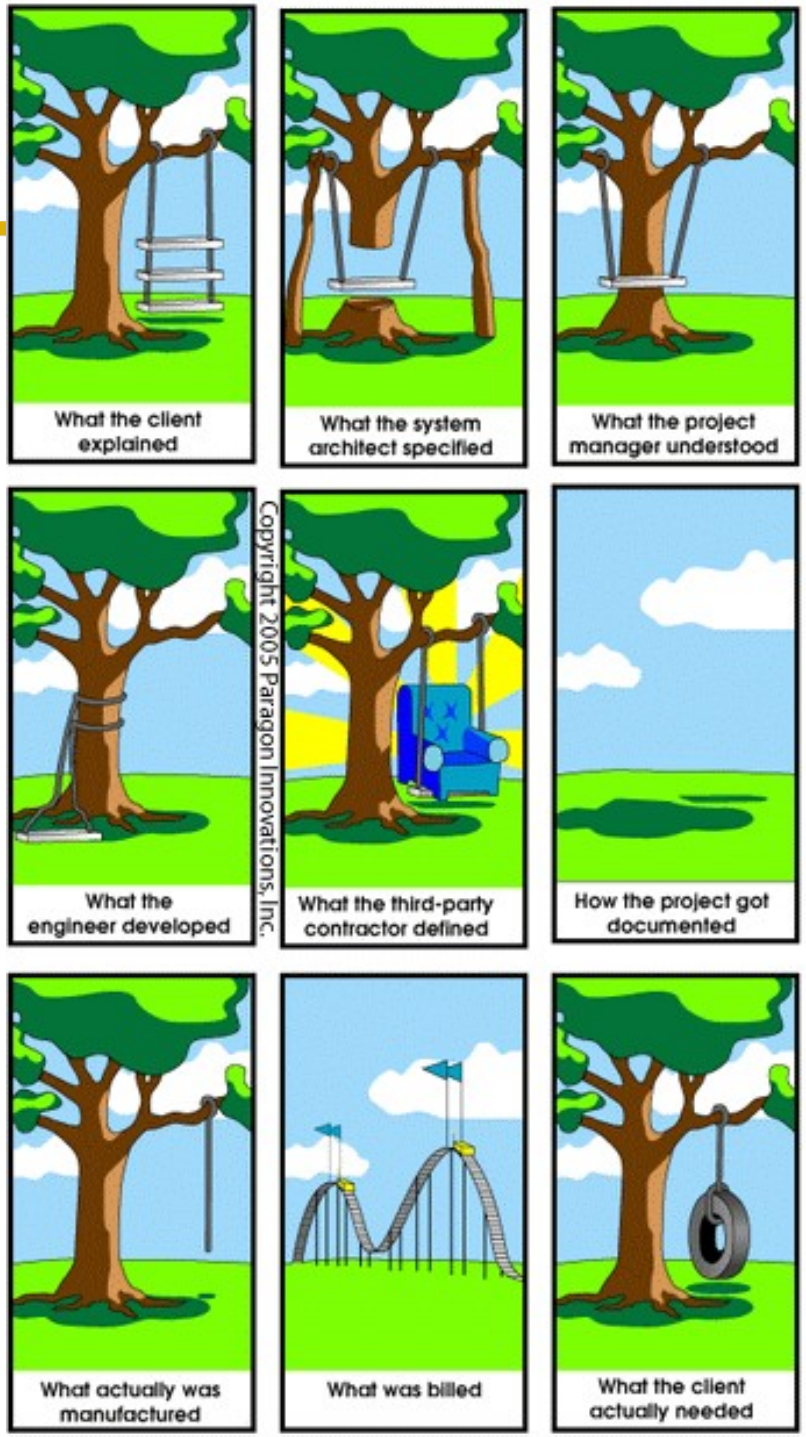






Agile myths

- Agile is a silver bullet
- Agile is anti-documentation
- Agile is anti-planning
- Agile is undisciplined
- Agile requires a lot of rework
- Agile is anti architecture
- Agile doesn’t scale



Fundamental approaches: User stories

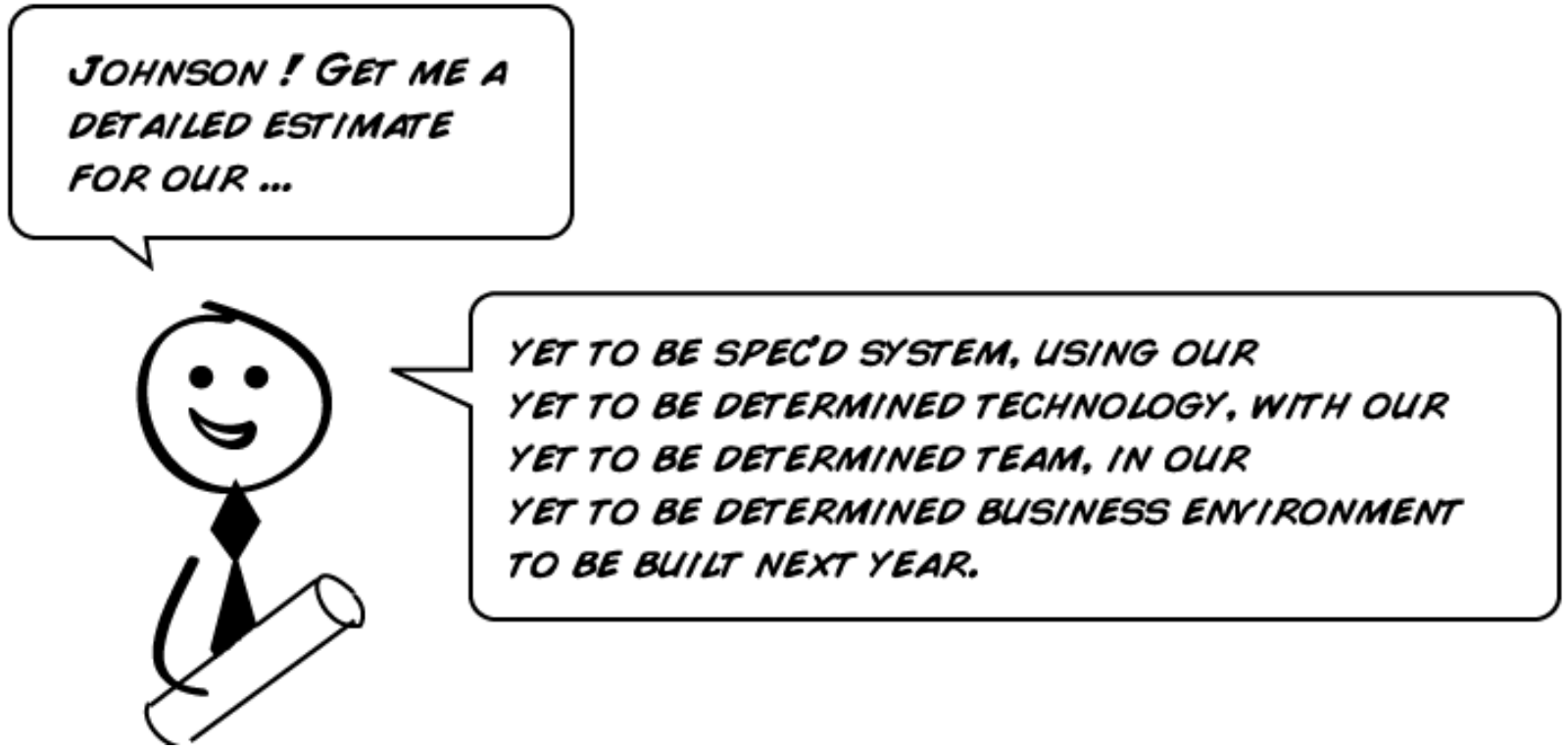
THE UNIVERSITY OF WESTERN AUSTRALIA

- User stories describe features
- They are told from the end user point of view.
- These features can be deliver in short units of work.
- They are often written on cards to facilitate communication

#	Backlog Item (User Story)	Story Point
1.	As a Teller, I want to be able to find clients by last name, so that I can find their profile faster	4
2.	As a System Admin, I want to be able to configure user settings so that I can control access.	2
3.	As a System Admin, I want to be able to add new users when required, so that...	2
4.	As a data entry clerk, I want the system to automatically check my spelling so that...	1

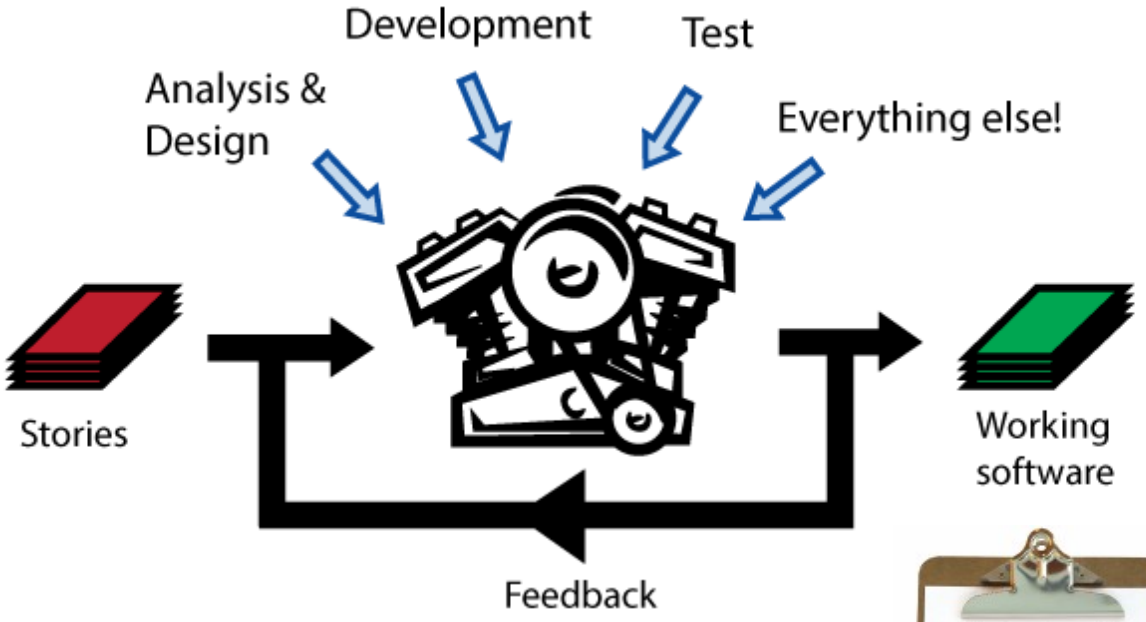
Fundamental approaches: estimation

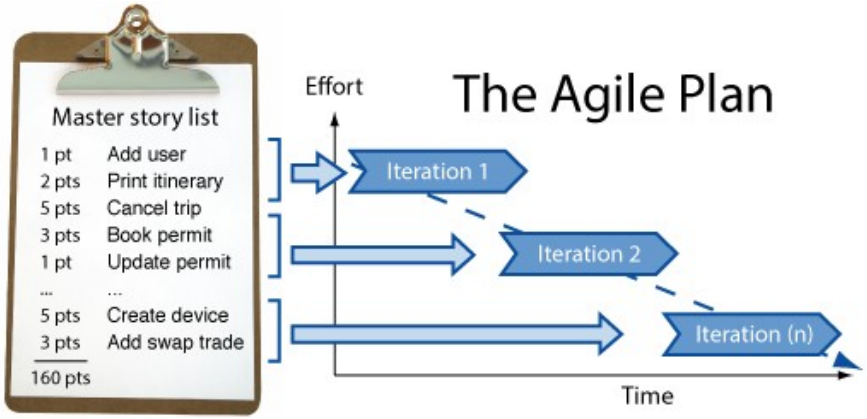
- Estimation is difficult but essential.
- You should always practice estimating the amount of time development will take.



Fundamental Approaches: Iterations

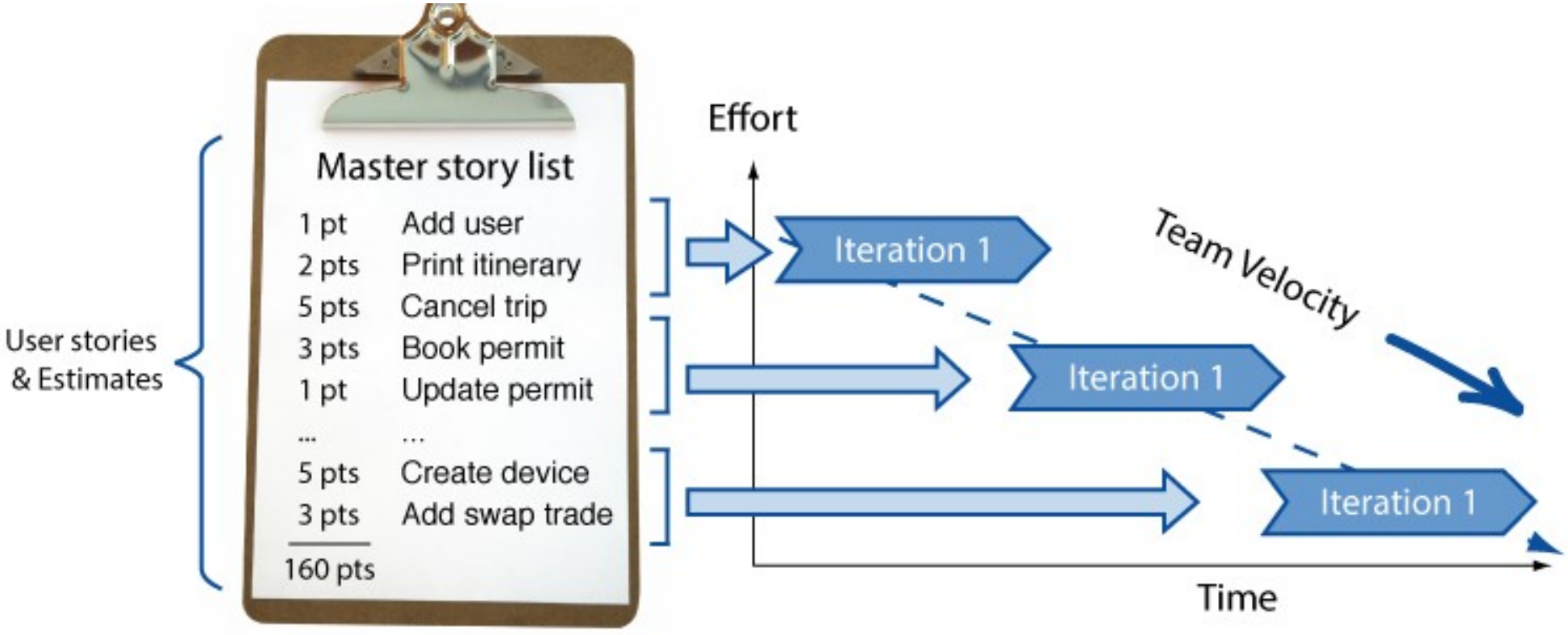
- Iterations are the core of software develpoment.





Fundamental approaches: planning

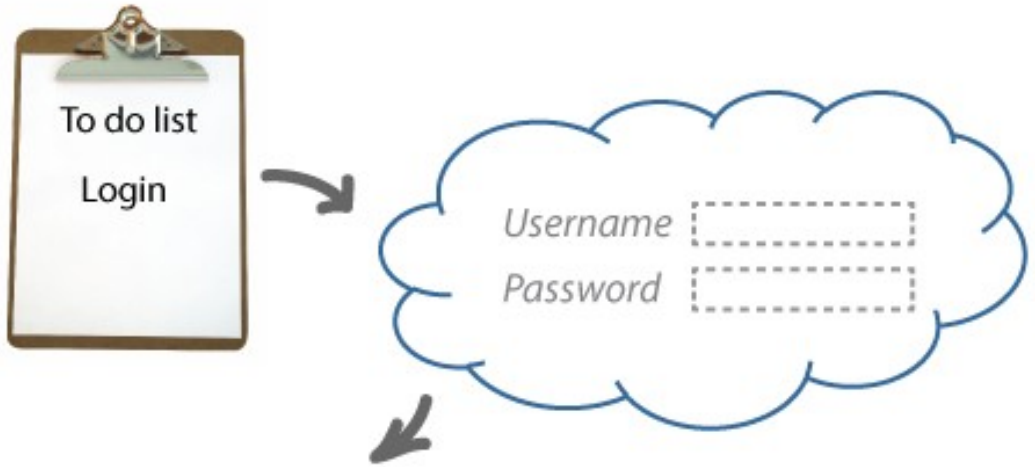
- Combines the user stories and estimations to build a feasible plan for delivery.



Unit Testing

THE UNIVERSITY OF WESTERN AUSTRALIA

Unit tests are snippets of test code developers write to prove to themselves that what they are developing actually works. Think of them as codified requirements.

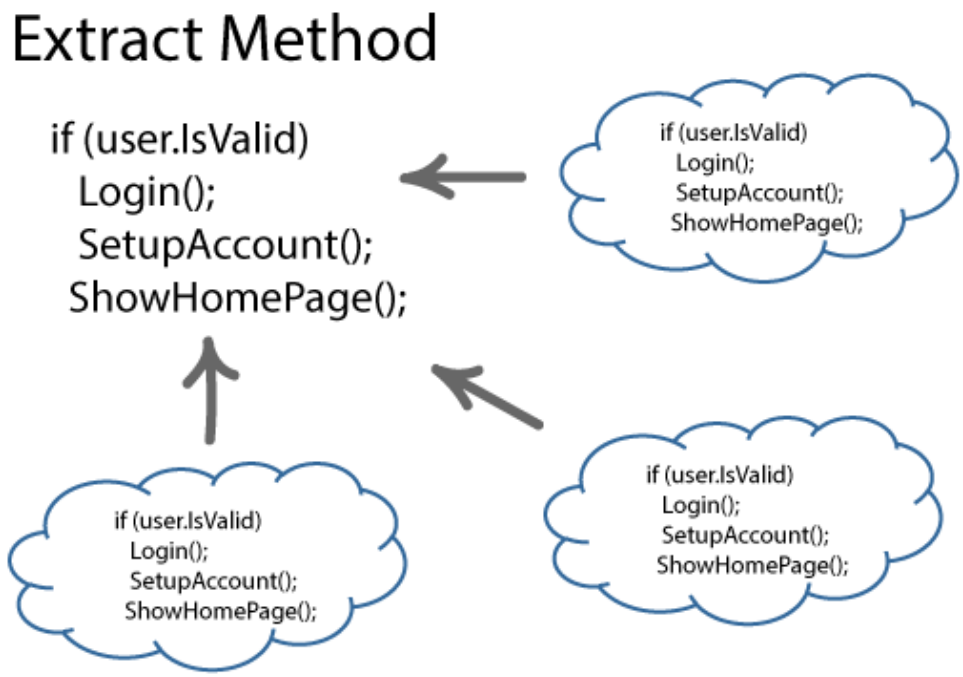


They are powerful because when combined with a continuous integration process they enable us to make changes to our software with confidence.

Refactoring

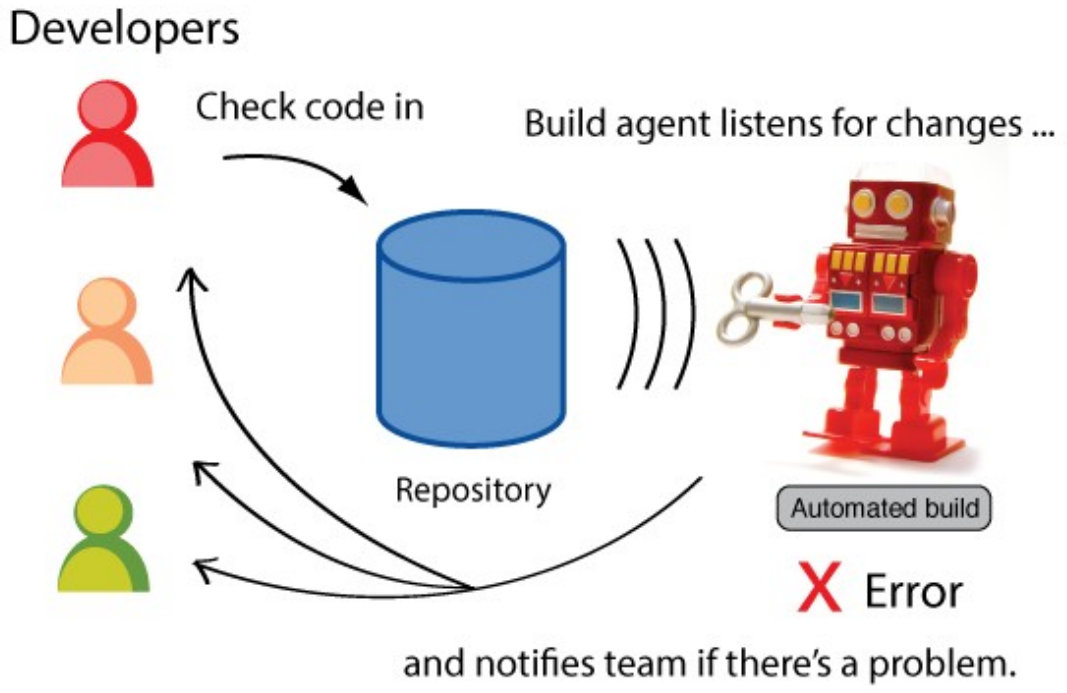
- To maintain a design and functionality, we must be prepared to refactor code.

- Organise code into manageable modules.
- Don't repeat yourself (DRY)



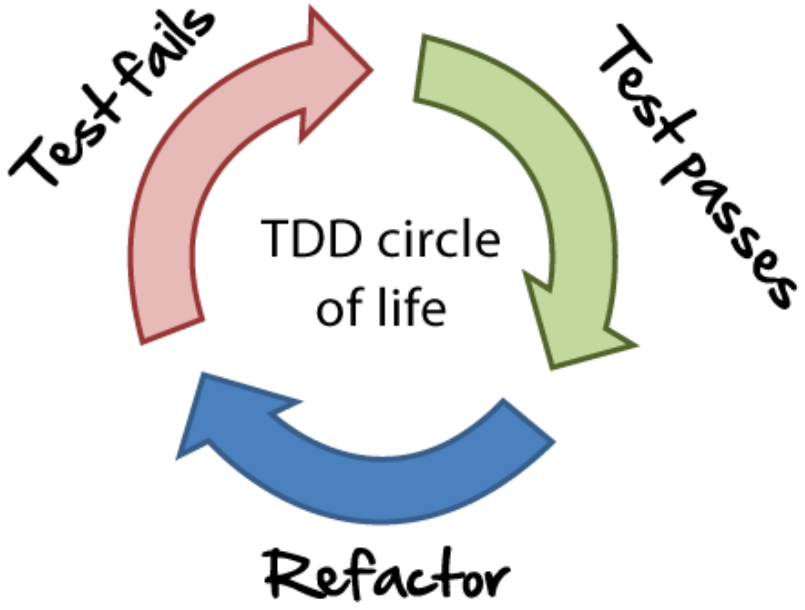
Continuous Integration

- Continuous integration keeps the code in a repository that is automatically maintained and everyone works on at the same time.

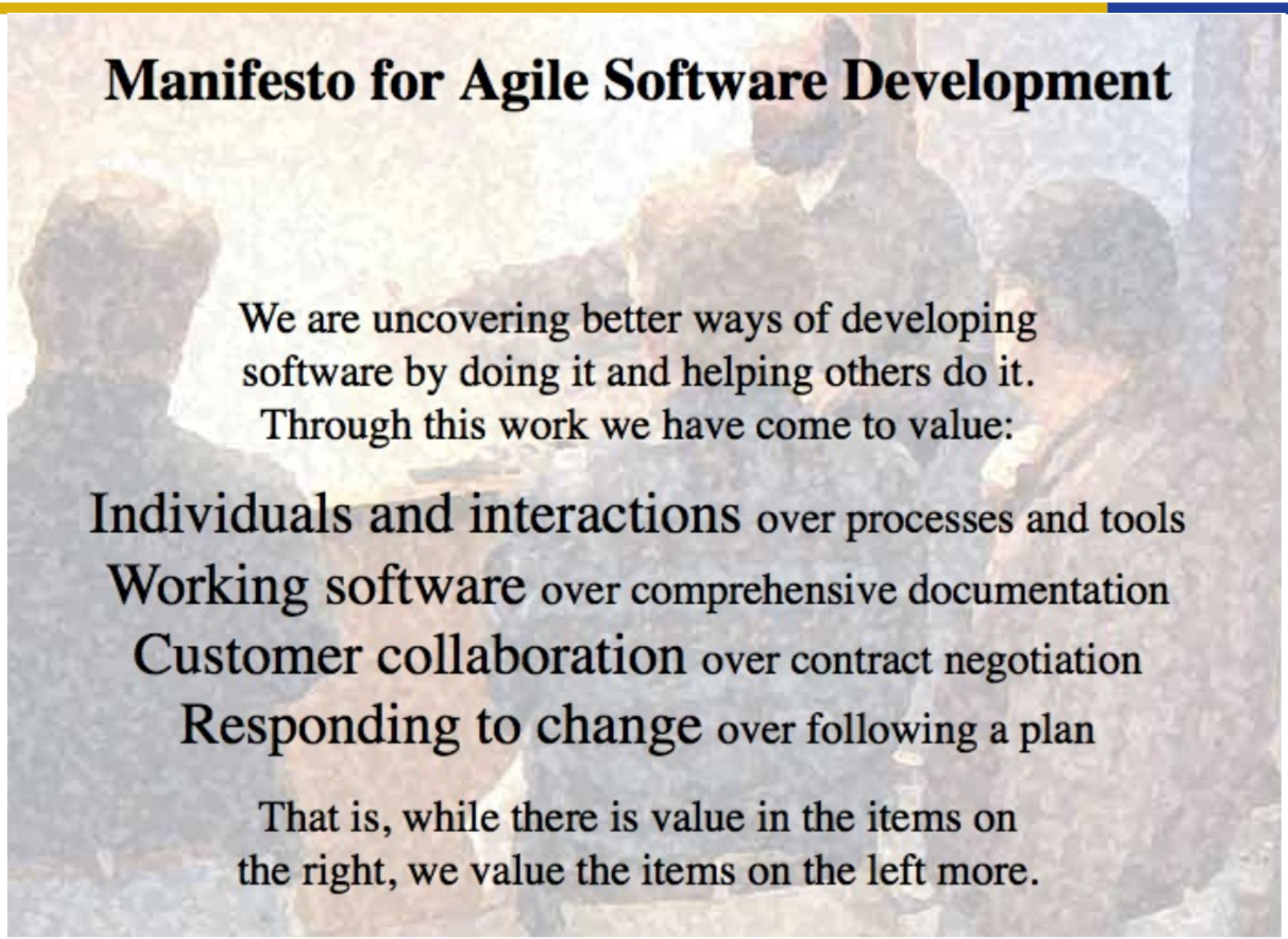


Test Driven Development

- Write tests at the start and then write code to pass the tests.
- The tests become the defacto documentation for the system.



Types of Agile:

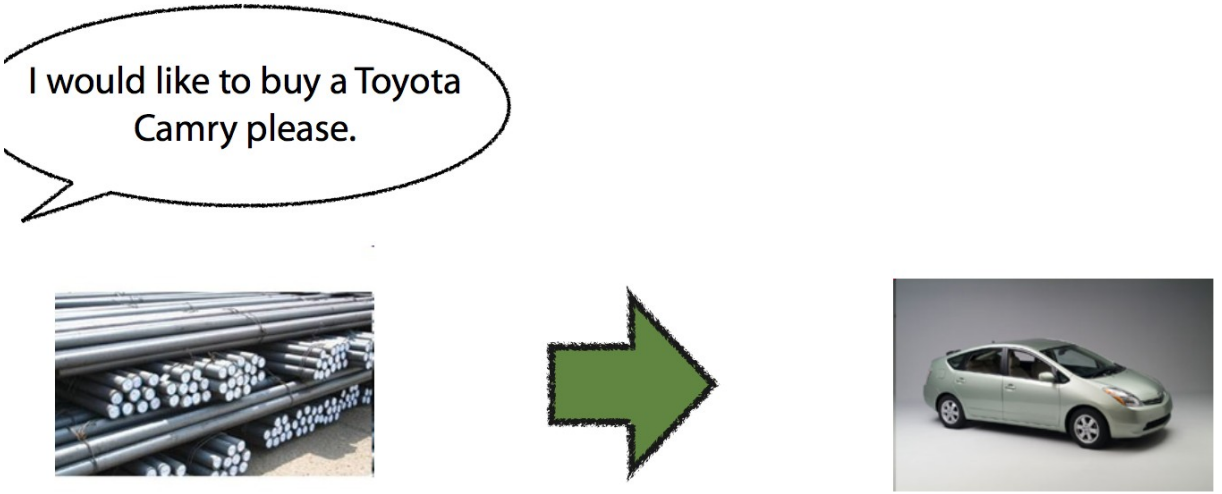


Flavours of Agile

- Good high organisation
- But not IT specific

Lean

Toyota's ultra-lean manufacturing process.

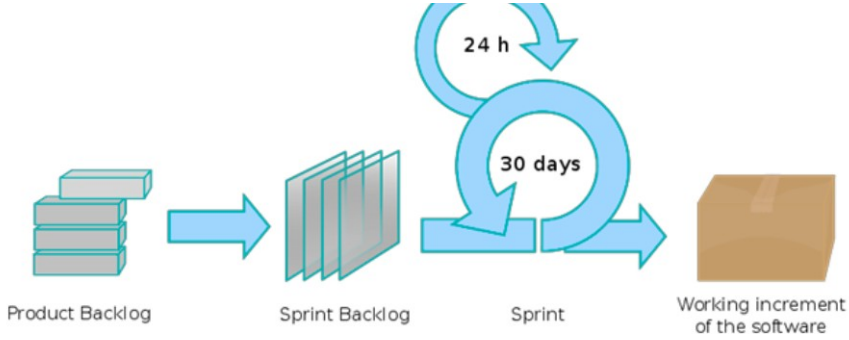


Eliminating waste

Flavours of Agile

- Easy to understand and start
- Very popular
- Not much engineering

Scrum



- A project management wrapper for incremental delivery of projects, independent of technology or business vertical.
- Can be used in non-IT projects.

Flavours of Agile

THE UNIVERSITY OF WESTERN AUSTRALIA

- Detailed engineering practices
- IT focussed
- Popular with developers

Extreme Programming

- Popularized software engineering practices necessary for agile development
- Emphasizes
  - –upfront testing
  - –automation
  - –evolutionary design
  - –continuous integration

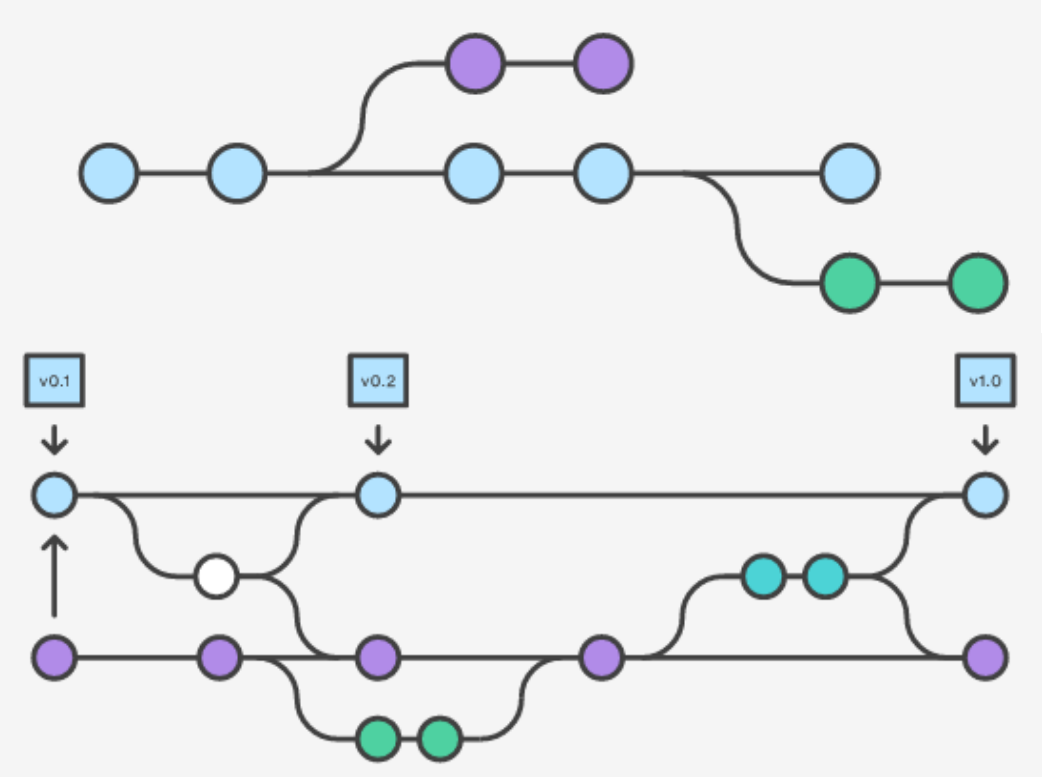
GIT

THE UNIVERSITY OF WESTERN AUSTRALIA

- Git is a distributed version control system
- Developed in 2005 by Linus Torvalds
- Now the most widely used version control system in the world.
- Git is able to manage different branches of a development, allowing teams to work on the latest branch, roll back changes, or develop independent features.

Git Theory

- Git has a decentralised structure. Everyone on the project has a copy of the history of the project.
- The history of the project is structured as a graph. Each commit can be undone and replayed
- Git tracks changes in the current working directory.
- Changes are *added*, then *committed*, then *pushed* to a branch.
- The new code is then *pulled* to other spaces.



Git CheatSheet

THE UNIVERSITY OF WESTERN AUSTRALIA

learn more about git the simple way at *rogerdudler.github.com/git-guide/*
cheat sheet created by Nina Jaeschke of *ninagrafik.com*

create & clone

create new repository	*git init*
clone local repository	*git clone /path/to/repository*
clone remote repository	*git clone username@host:/path/to/repository*

add & remove

add changes to INDEX	*git add <filename>*
add all changes to INDEX	*git add **
remove/delete	*git rm <filename>*

commit & synchronize

commit changes	*git commit -m "Commit message"*
push changes to remote repository	*git push origin master*
connect local repository to remote repository	*git remote add origin <server>*
update local repository with remote changes	*git pull*

branches

GIT CheatSheet

THE UNIVERSITY OF WESTERN AUSTRALIA

branches

create new branch	*git checkout -b <branch>* *e.g. git checkout -b feature_x*
switch to master branch	*git checkout master*
delete branch	*git branch -d <branch>*
push branch to remote repository	*git push origin <branch>*

merge

merge changes from another branch	*git merge <branch>*
view changes between two branches	*git diff <source_branch> <target_branch>* *e.g. git diff feature_x feature_y*

tagging

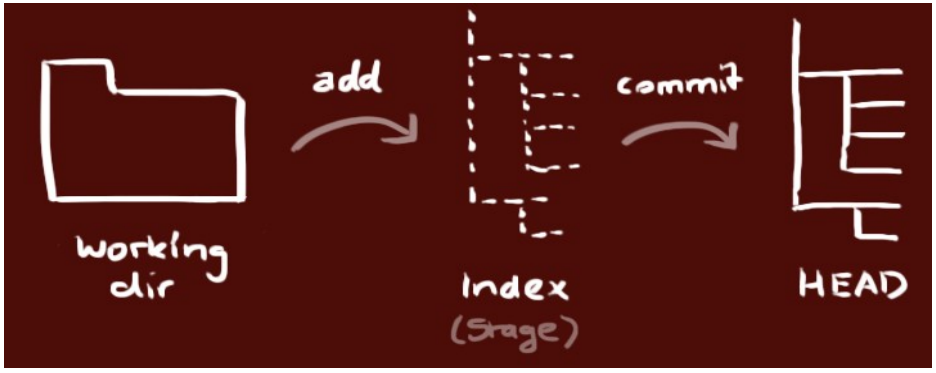
create tag	*git tag <tag> <commit ID>* *e.g. git tag 1.0.0 1b2e1d63ff*
get commit IDs	*git log*

restore

replace working copy with latest from HEAD	*git checkout -- <filename>*

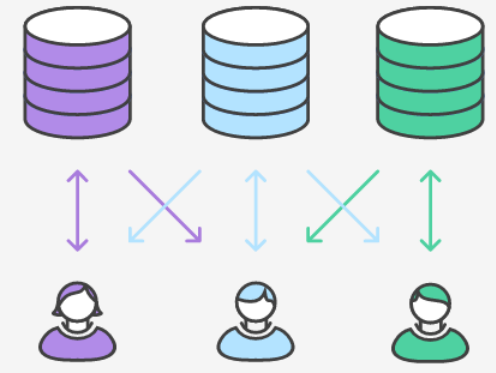
GIT operations

- Create a new repository: `git init`
  - adds files monitoring changes, preferences etc.
- Checkout a repo: `git clone usr@url:[path]`
  - copies files and history to a local copy
- The local repo has three trees:
  - the *working directory* (the actual files)
  - the *index* (a set of changes that is ready to commit)
  - the *head* (the last commit you made)



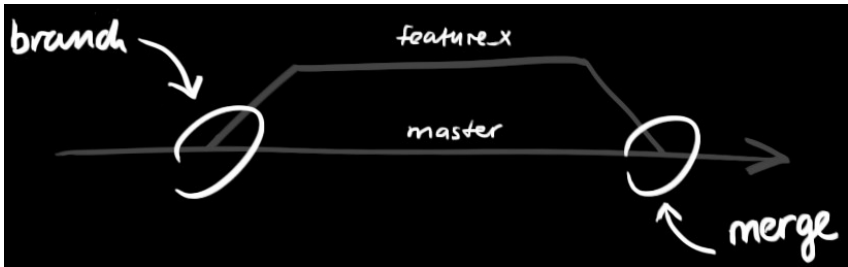
GIT add, commit, push, pull

- Add new files: `git add <filename>`
  - adds file to index
- Commit changes to Head: `git commit –m "msg"`
  - commits changes in index to the head with the msg.
- Push changes to repository:
  - `git remote add origin <server>`
  - `git push origin master`
  - adds remote server, and pushes latest commit to it.
- Get latest commit from repo: `git pull`



GIT branching

- Create a new branch: `git checkout -b b1`
  - creates a new branch that can change independently.
- Switch back to master: `git checkout master`
  - changes to the master branch.
- Delete Branch: `git branch -d b1`
- Push new branch: `git push origin <branch>`
- Merge into your branch: `git merge <branch>`
  - merging is automatic, but there may be conflicts.
- To see the differences between two branches:
  `git diff <b1> <b2>`



GIT utilities

THE UNIVERSITY OF WESTERN AUSTRALIA

- Get a log of commits: `git log`
- Tag a version: `git tag 1.0.0 <commit-id>`
  - gives a version number to a commit tag.
- Rollback changes: `git checkout -- <file>`
  - returns local file to last commit.
- Delete Branch: `git branch -d b1`
- Undo all changes and commits: `git fetch origin`
  - then `git reset -hard origin/master`

- Lots of GUIs, environments exist: GitHub, BitBucket, GitKraken

GitHub

THE UNIVERSITY OF WESTERN AUSTRALIA

- GitHub is a service that hosts Git repositories.
- You can develop colaboratively, and use GitHub as a remote Repo.
- **Note:** Free GitHub repositories are public, so anyone can see your code.
- Students are able to get free education accounts, which allow private repositories.
  - https://education.github.com/pack

- Bitbucket is a similar service

Coordinating Dev Environments

THE UNIVERSITY OF WESTERN AUSTRALIA

- Git is a great way to link development and deployment environments:
  - Work on your local machine (a laptop) with all the latest features and branches
  - Push commits to GitHub or some central repo
  - Pull changes from GitHub to a server environment for test or deployment
  - You can incorporate testing, documenting and reporting into these workflows.