Software Engineering : Design Patterns 2CB105
Design Patterns for Games 2CB106

02a – Agile Methodology

Dr Andy Guest

a.guest@yorksj.ac.uk

Room 104 44 Lord Mayor’s Walk

Agile

- A response to traditional software engineering
- To address the problems and difficulties
- Agile is philosophy, a set of values, rather than a process

Agile Manifesto

- **Individuals and interactions** over process and tools
- **Working software** over comprehensive documentation
- **Customer collaboration** over contract negotiation
- **Responding to change** over following a plan

Agile Umbrella

- Scrum, Crystal, Kanban, FDD, XP, DSDM, RUP
- Range in complexity/the number of rules to follow
  - 0 – Do whatever!
  - 3 – Kanban
  - 9 – Scrum
  - 13 – XP
  - 120+ - RUP

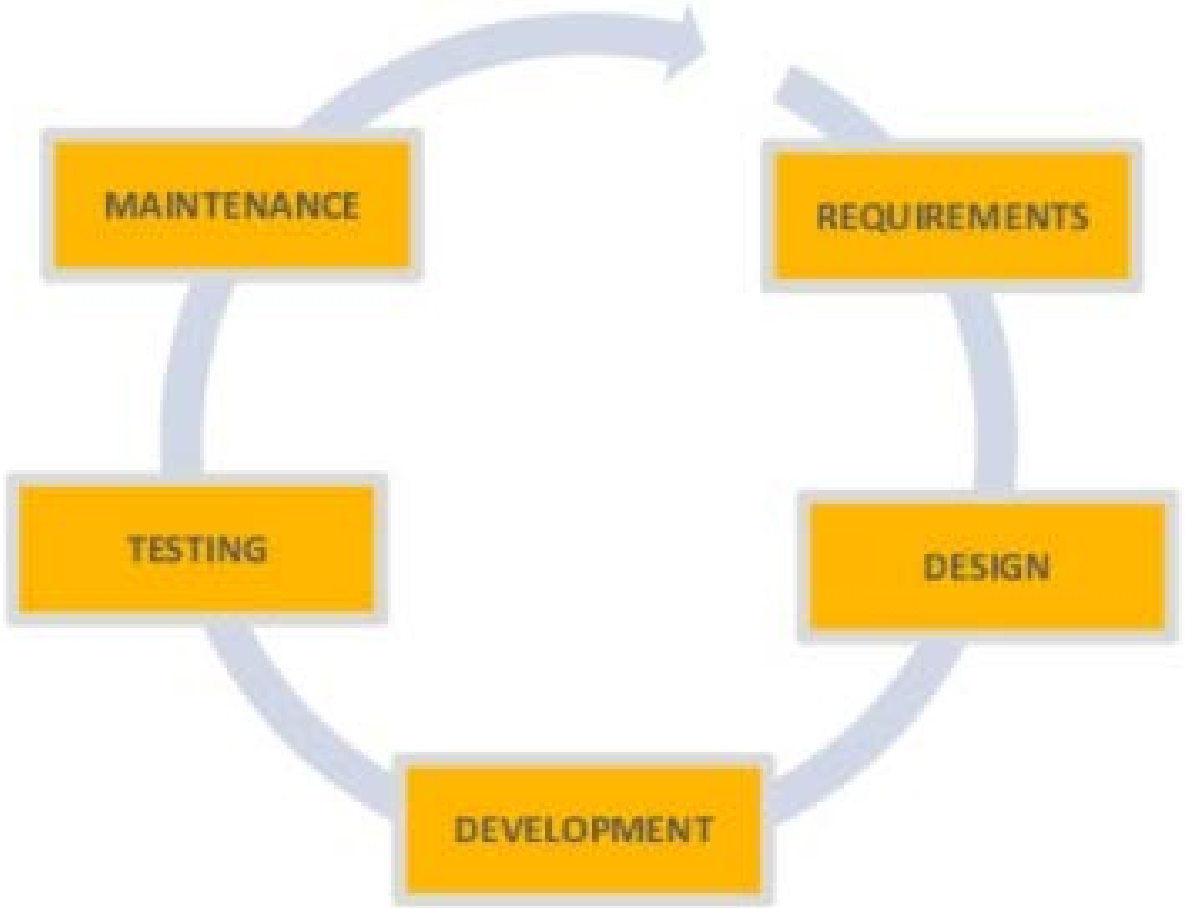
Scrum

- A **light-weight agile process tool**
- **Splits your organization** into small, cross-functional, self-organising teams
- **Split your work** into a list of small, concrete deliverables. Sort the list by priority and estimate the relative effort of each time
- **Split time** into short fixed-length iterations/sprints (usually 2-4 weeks), with potentially shippable code demonstrated after each iteration

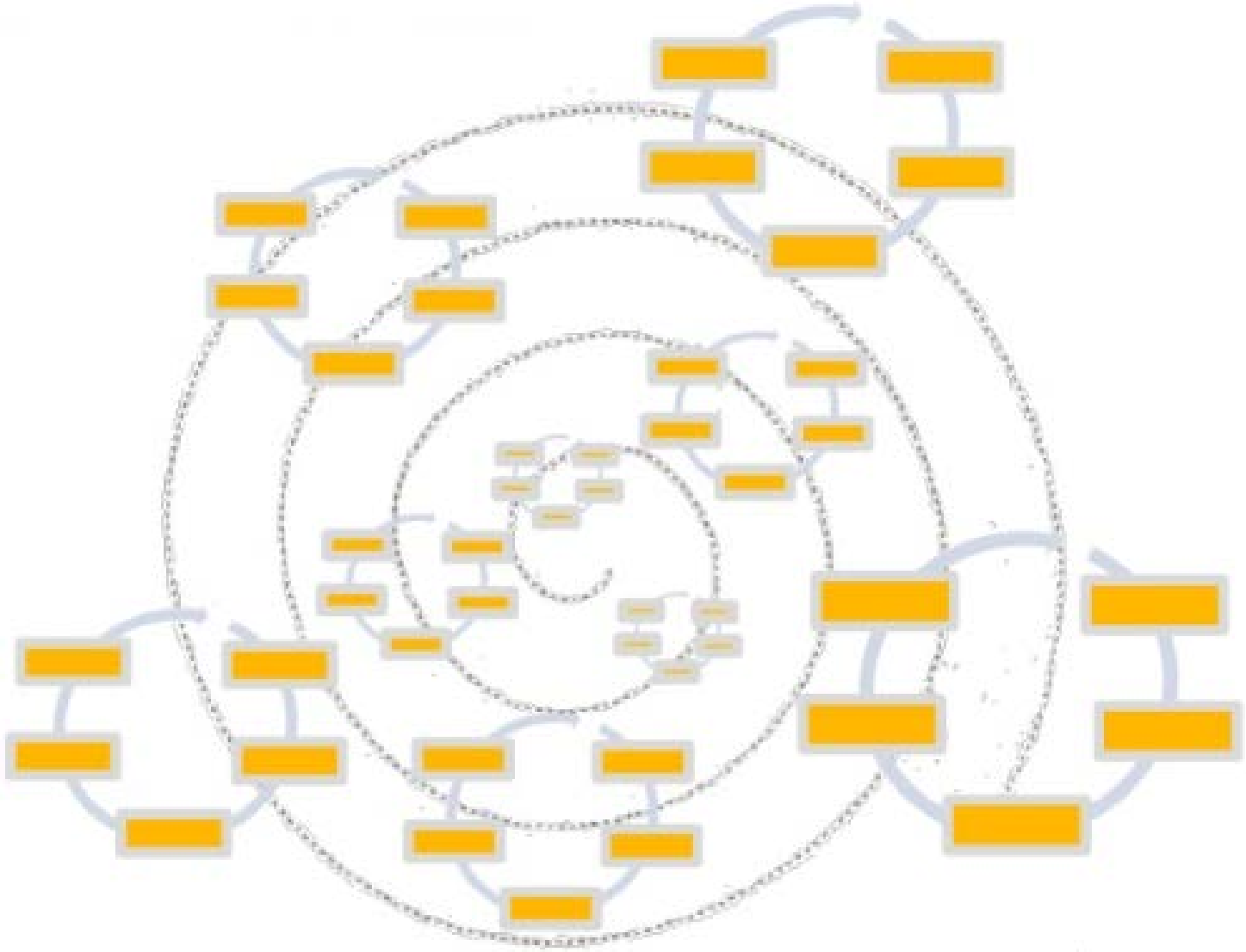
Scrum

- **Optimize the release plan** and update priorities in collaboration with the customer, based on insights gained by inspecting the release after each iteration
- **Optimize the process** by having a retrospective after each iteration

Scrum vs Waterfall



Iterative Scrum



Scrum Terminologies

- The project/product is described as a list of features: the **backlog**
- The features are described in terms of **user stories**
- The scrum team **estimates** the **work** associated with each story
- Features in the backlog are **ranked** in order of importance
- Result: a **ranked** and **weighted** list of product features, a **roadmap**
- **Daily scrum meeting** to discuss *What did you do yesterday? What will you do today? Any obstacles?*

Scrum in a nutshell

- Instead of a **large group** spending **a long time** building a **big thing**, we have a **small team**, spending a **short time** building a **small thing**, but **integrating regularly** to see the whole.

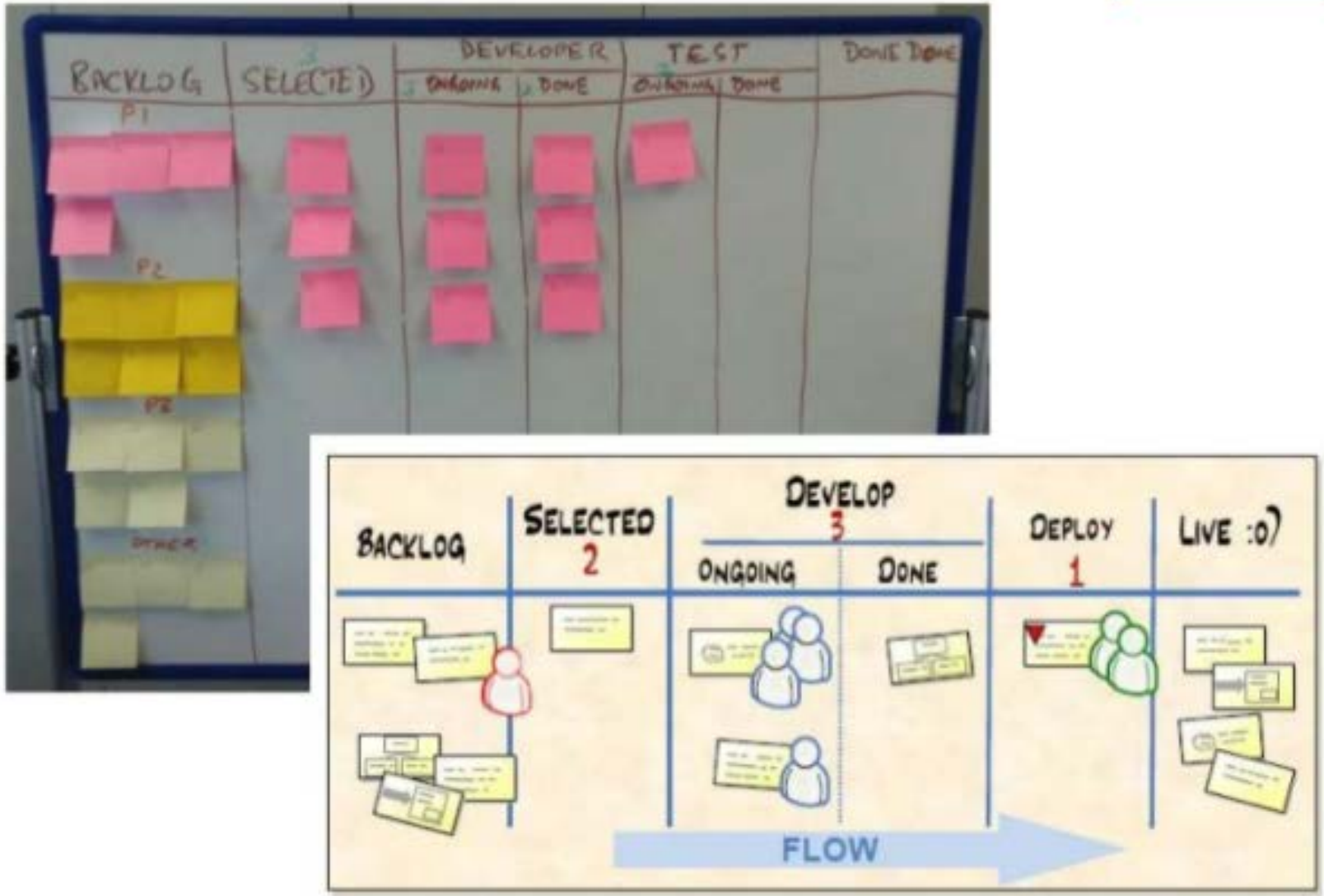
Kanban

- **Lean approach** to agile development
- Similar to Scrum in the sense that you **focus on features as opposed to groups of features** – however Lean takes this one step further again.
- You **select, plan, develop, test and deploy one feature** (in its simplest form) **before you select, plan, develop, test and deploy the next feature**
- **Aim** is to *eliminate 'waste'* wherever possible

Kanban

- **Visualise the workflow**
  - Split the work into pieces, write each item on a card and put on the wall
  - Use named columns to illustrate where each item is in the workflow
- **Limit WIP** (work in progress)
  - Assign explicit limits to how many items may be in progress at each stage
- **Measure the lead time** (average time to complete one item, sometimes called “cycle time”)
  - Optimise the process to make lead time as small as predictable as possible

Kanban Board



Kanban Board