Scrum According to Rational Unified Process

Christina Skaskiw
Consultant, REAL Solutions
christina.skaskiw@realsolutionsuk.com

IBM Rational Software Development Conference 2008

WHERE TEAMS ARE R-HEROES

IBM

Rational. software

Why Consider Scrum?

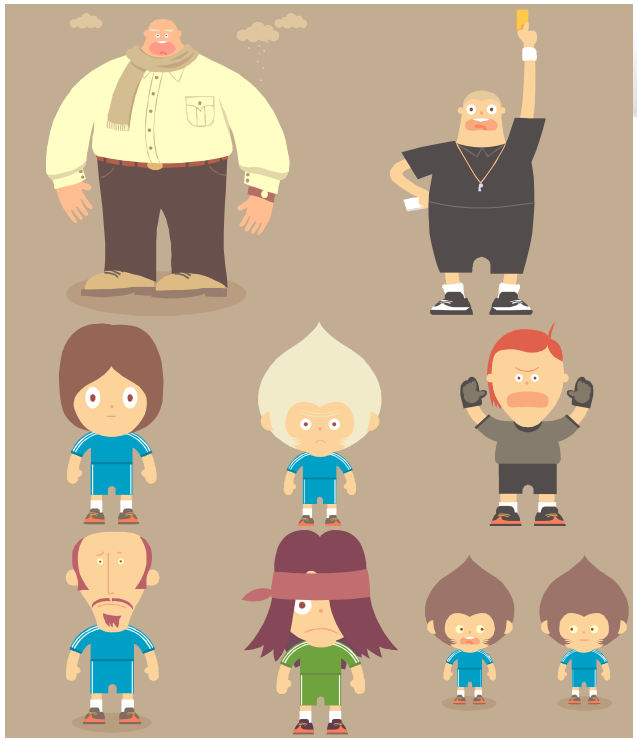
- Phenomenal productivity
- User satisfaction
- Success where waterfall was stumped
- Scales linearly

How can RUP be tailored to reap similar productivity gains?

- Has the foundational iterative approach
- Needs “sharpening” and filling in

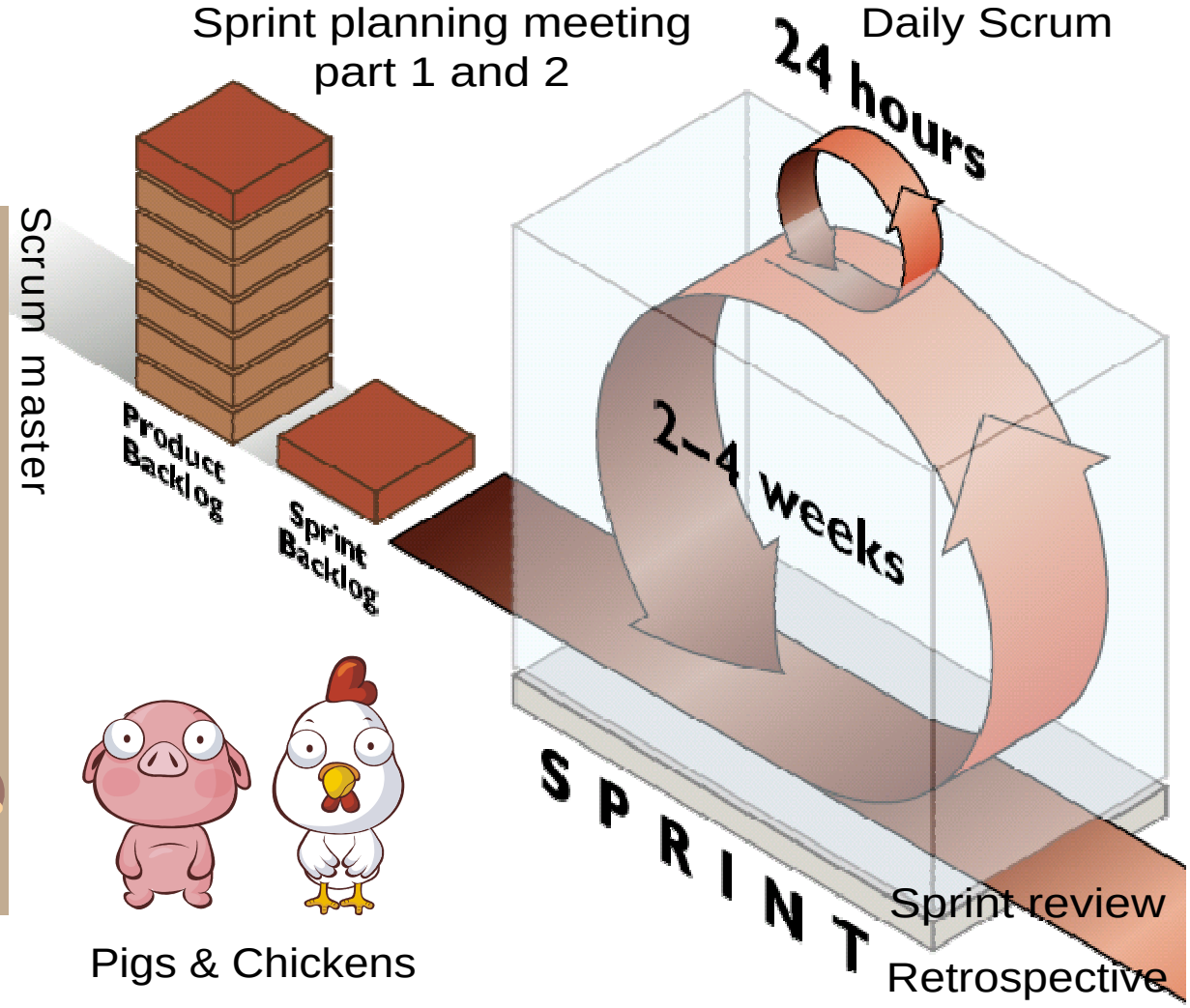
What is Scrum?

Product owner



Scrum master

Self-managed,
cross-functional team



Pigs & Chickens

picture from “Scrum in five minutes” by **SOFTHOUSE**

Why does Scrum work?

- Empirical process
  - handles complexity
- Stakeholder involvement
  - insight into emerging system for course correction
- Self-managed teams
  - commits to what it selects in the Sprint scope
- Right-sizes the process
  - continual improvement through retrospectives

Why does Scrum work?

Defined process control:

The defined model of process control exercises a *predictable process* producing a *predictable result*, based on plans and predictions, i.e. *feed-forward*. [wikipedia]

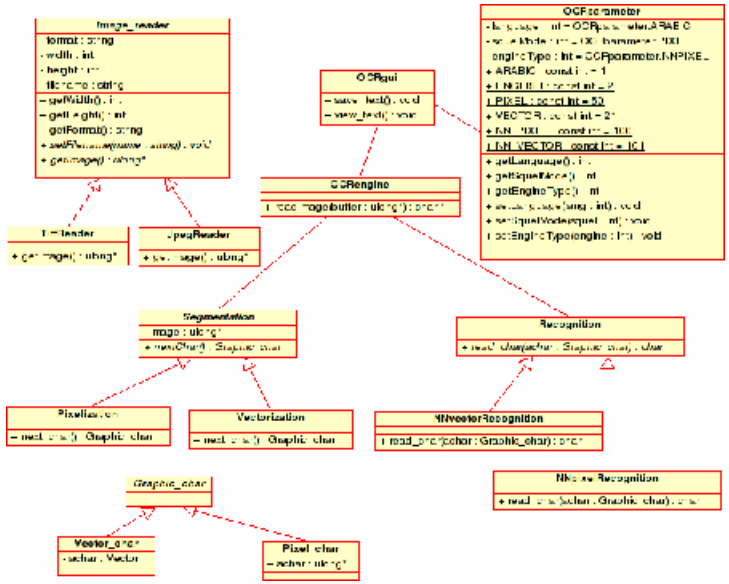
Why does Scrum work?

Empirical process control:

The empirical model of process control provides and exercises control through *frequent inspection and adaptation* for processes that are *imperfectly defined* and generate *unpredictable and unrepeatable outputs*. [wikipedia]

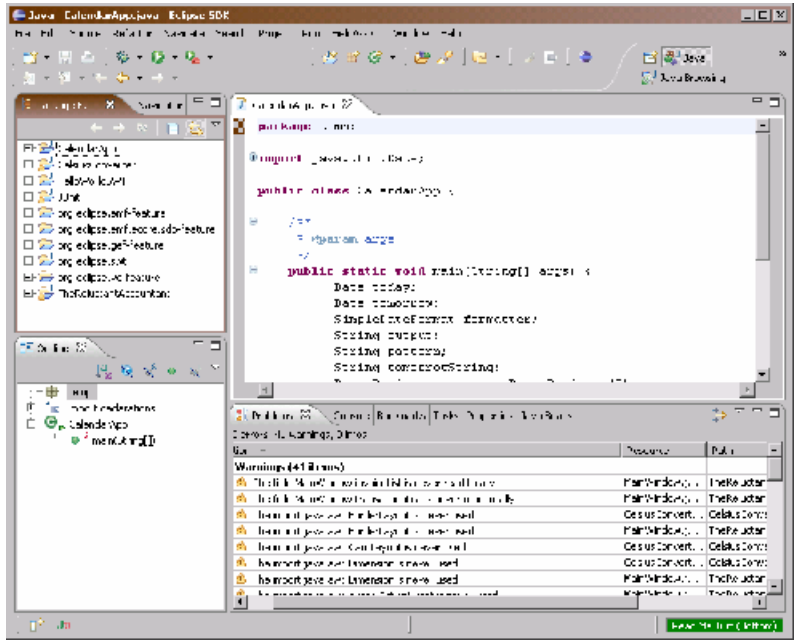
...i.e. *feedback*.

Blueprints = Design



Bridge = Code

Blueprints = Code



Bridge = Executable

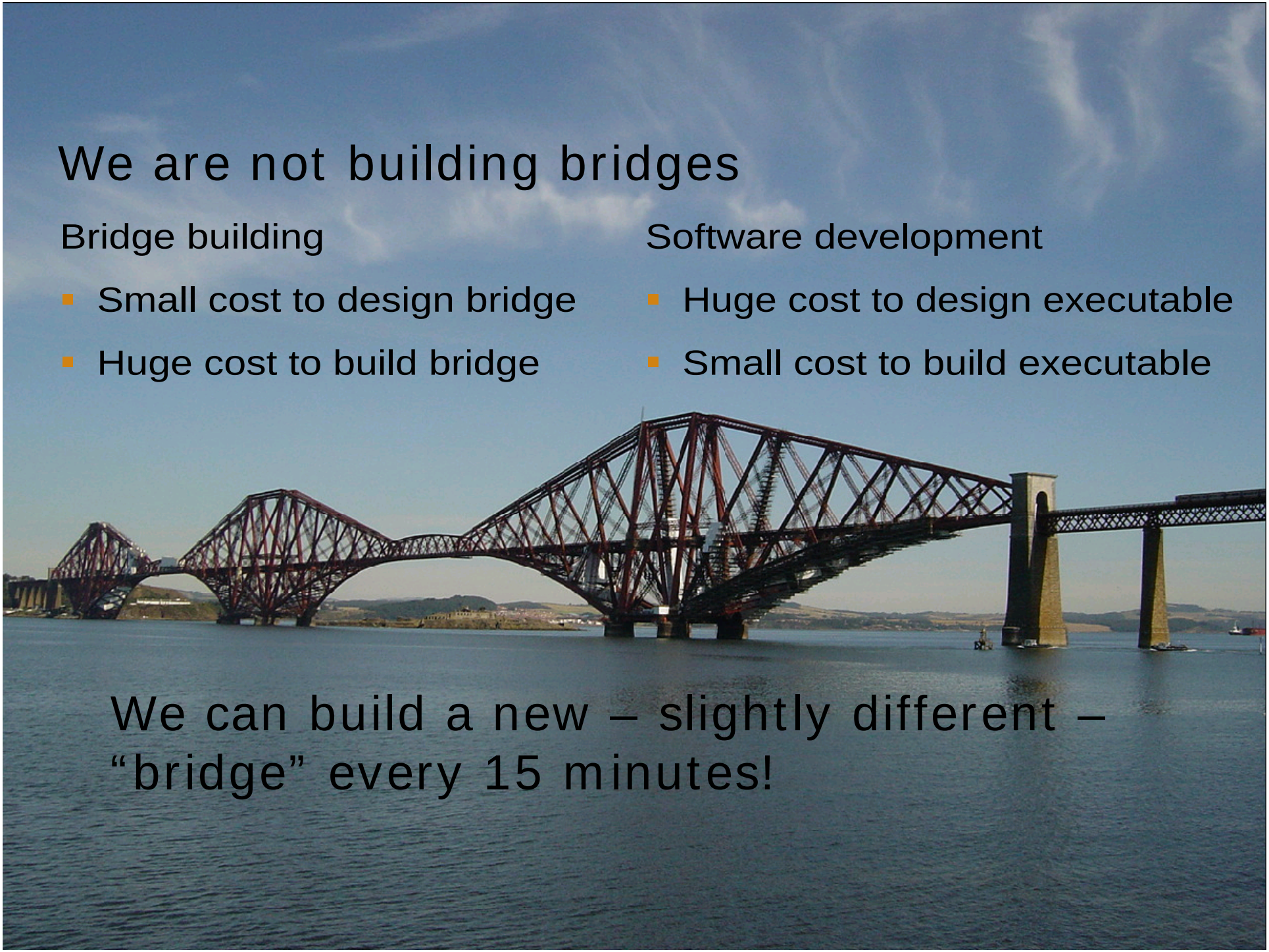
We are not building bridges

Bridge building

- Small cost to design bridge
- Huge cost to build bridge

Software development

- Huge cost to design executable
- Small cost to build executable



We can build a new – slightly different – “bridge” every 15 minutes!

Why does Scrum work?

Cybernetic principle:

In order to create simplicity amidst complexity, your process must be equally complex. The corollary to that would be that if you're trying to manage something very complex with too simple a process, it will over-complexify it!

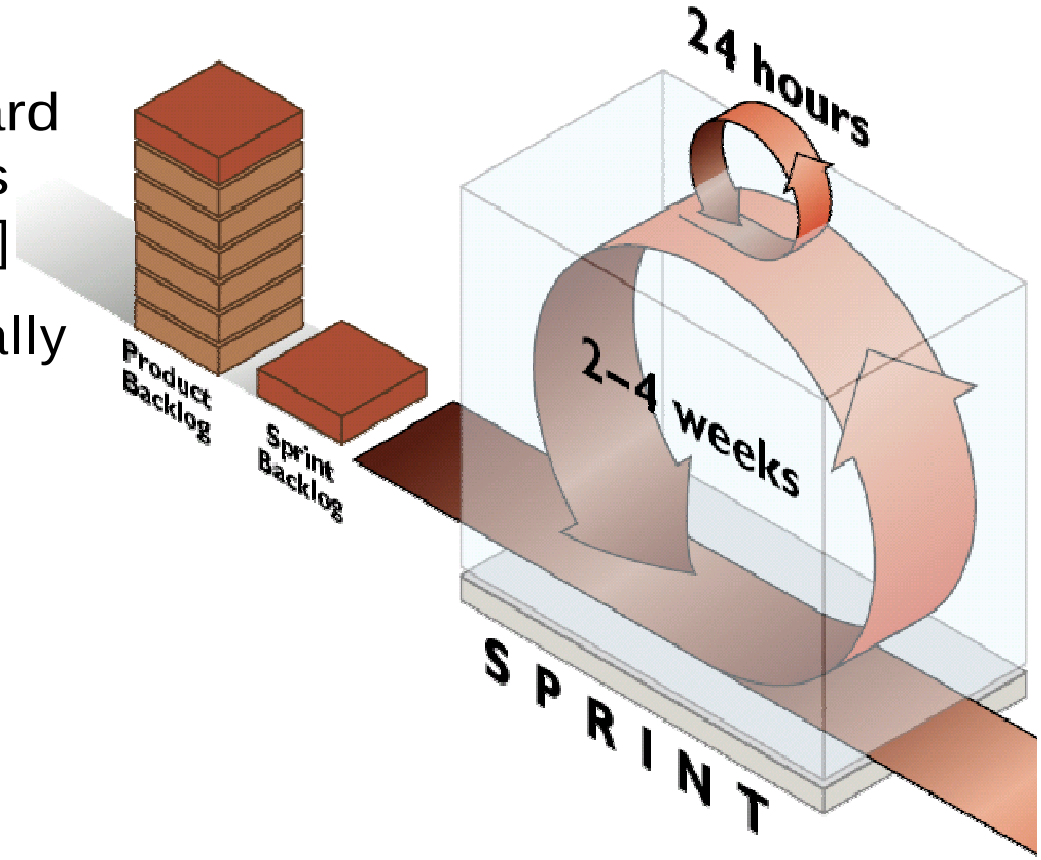
Why does Scrum work?

That is: using a defined process for software development will over-complexify it, because a defined process will always be too simple.

inefficient and expensive micro-management

Why does Scrum work?

- Feedback is more basic and dependable than feed-forward (acting on the basis of plans or predictions). [G&H, 2004]
- Feedback control is specifically intended to cope with disturbances. [G&H, 2004]
- Positive feedback [...] is the condition to change, evolution, growth... [wikipedia]



Empirical process implications:

- Clear goals on all levels
  - requirement focus
  - no work breakdown structure
- Inspections on all levels
  - testing
  - reviewing
  - demonstrating
- Risk management

Empirical process implications:

- Good practices & tools
  - Modelling
  - Coding standards
  - Refactoring
  - Documentation
  - Tracing tests to requirements
  - Test automation

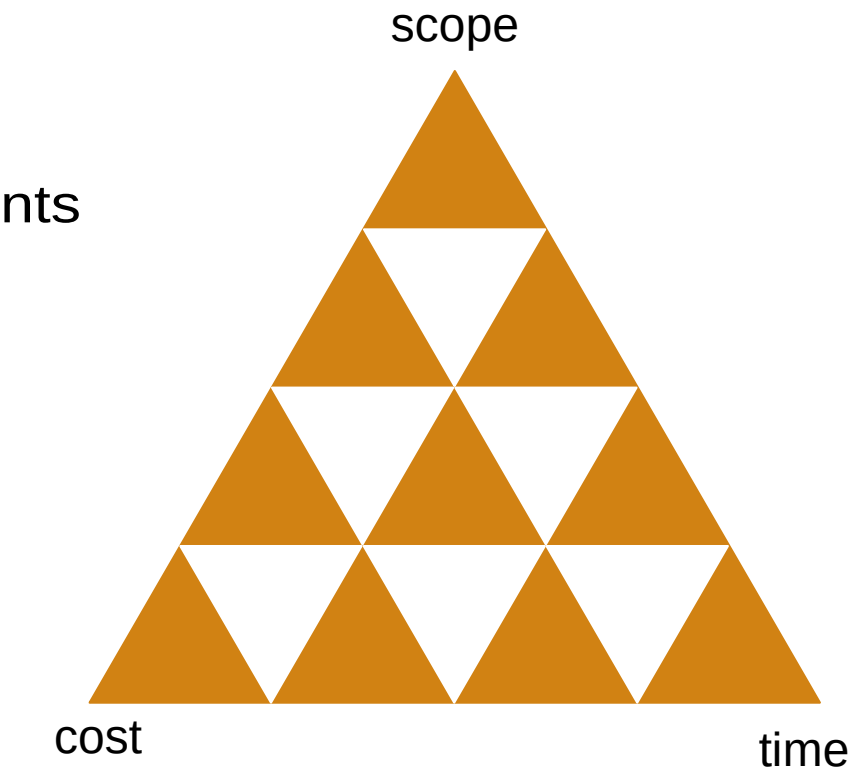
Less than production level quality will cause drag and give incorrect feedback

Why does Scrum work?

- Enables stakeholder involvement
  - Don’t know what they need till they see it
  - Allows for emerging requirements
  - Adapts to changing requirements
  - Fosters communication and collaboration between stakeholders and developers

Why does Scrum work?

- Self-managed teams
  - Product Owner rules requirements and their ranking
  - Team rules scope for sprint
    - team commitment
      - they are the Pigs
  - Undisturbed for a Sprint
    - management involvement to remove impediments
  - Flow
    - goal + frequent feedback [csik, 1990]

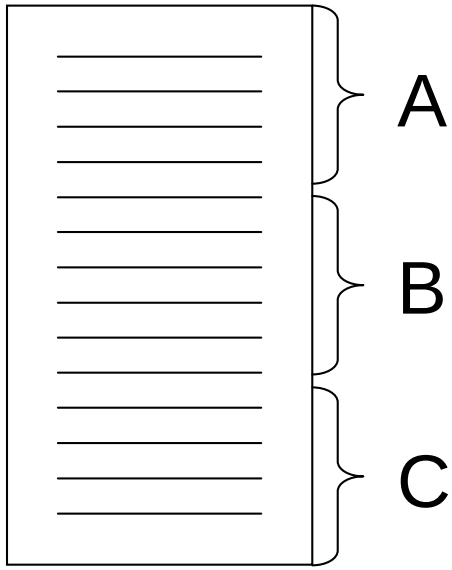


Scrum more operates from a fixed cost and time perspective with flexibility in scope

Why does Scrum work?

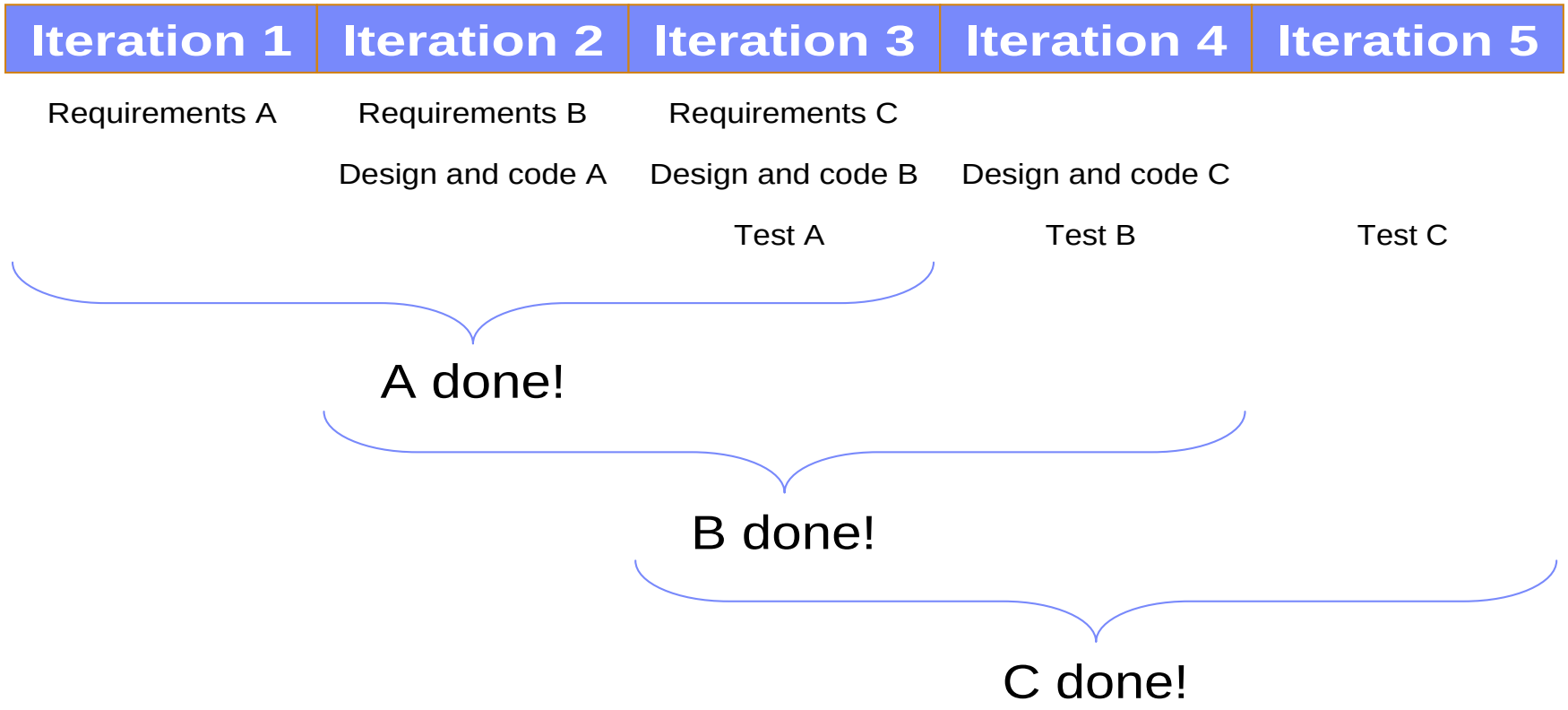
- Right-sizing the process
  - Sprint retrospectives
  - Identified improvements are implemented in the following Sprints

Aside: How NOT to do iterative development

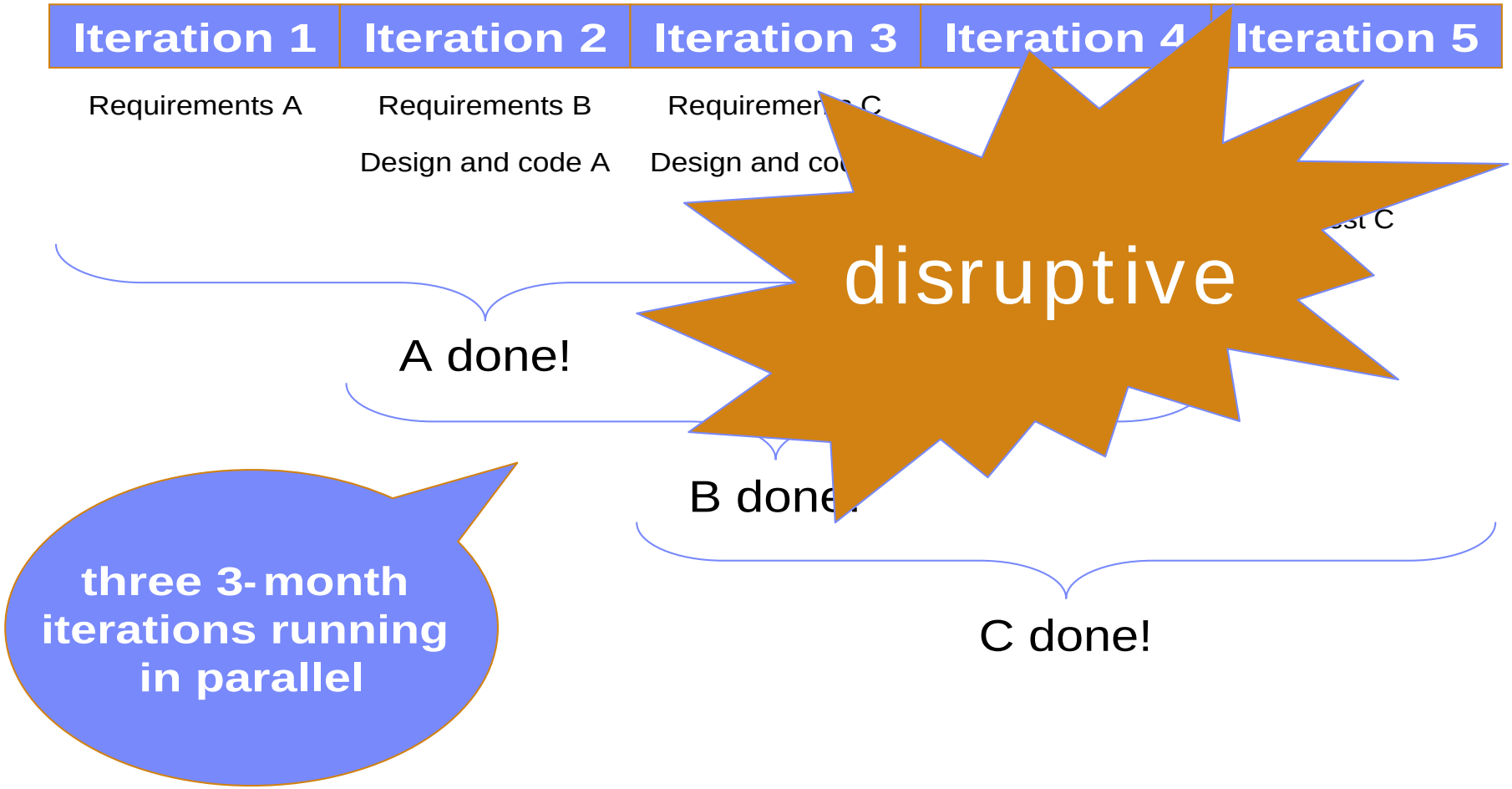


Product backlog

Aside: How NOT to do iterative development



Aside: How NOT to do iterative development



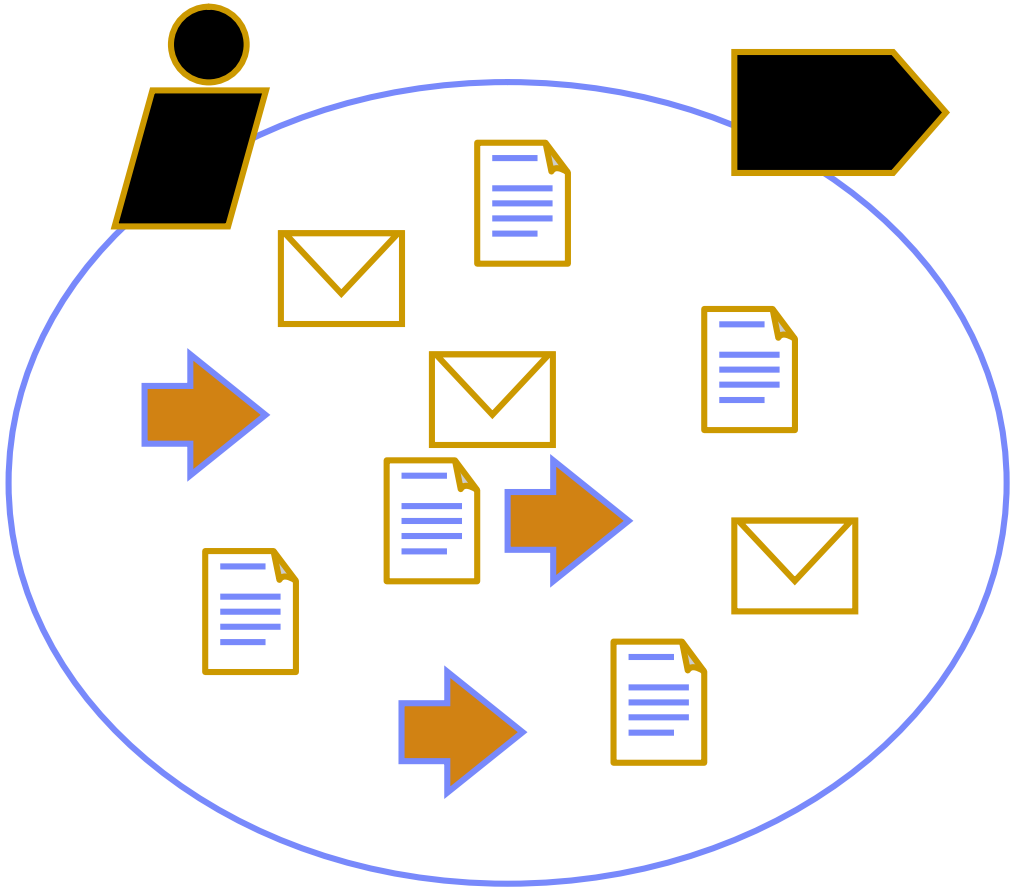
Rational Unified Process in a nutshell

- Key principles
- Disciplines with work products, roles and activities
- Phases and iterations
- Architecture first
- Risk-driven development
- Use cases
- Object-orientated

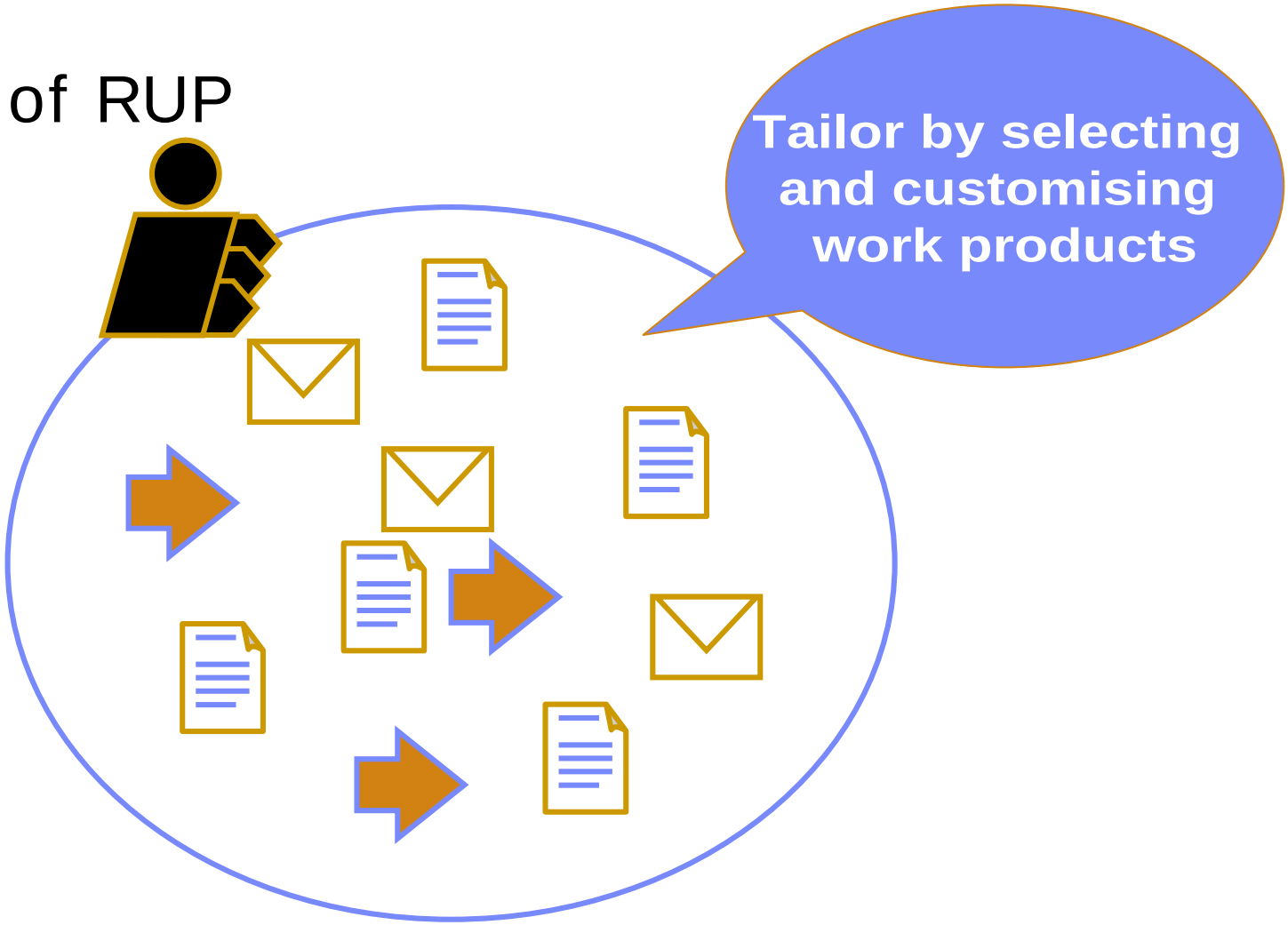
RUP Key Principles vs. Scrum

- Adapt the process
- Balance competing stakeholder priorities
- Collaborate across teams
- Demonstrate value iteratively
- Elevate levels of abstraction
- Focus continuously on quality
- Retrospectives
- Product backlog
- Cross-functional teams, scrum-of-scrums
- Sprint review
- Good engineering practices
- Production quality code in each Sprint

Structure of RUP



Structure of RUP



Product Backlog vs. RUP Artefacts

- Feature list in Vision – RANKED

or

- Use cases and scenarios, identified + Non-functional features – RANKED
- Part of Software Development Plan

Sprint Backlog and Tasks vs. RUP Artefacts

- Iteration plan
- Work orders

Product Owner vs. RUP Roles

- Work products
  - Vision
  - Business case
  - Use cases and scenarios at identification level
  - Stakeholder requests
  - Change requests
  - Coarse grain plans
  - (Business models)
- Project Manager (part)
- Stakeholder
- Management Reviewer
- System Analyst
- Change Control Manager
- (Business Analyst)

New Project Manager

- Has Product Owner responsibilities
- Represents management
- Overview role, not managing daily activities
- Responsible for
  - overall project plans
  - Product backlog
- Supported by
  - Stakeholders, including “management”
  - System Analyst
  - Change Control Manager

Scrum Master vs. RUP Roles

- Work products
  - Risk list
  - Issues list

[BC]

- Project Manager's role pertaining to
  - removing impediments
  - making team as productive as possible
  - enforcing the process
- Not the team's manager
- Introducing new role "Team Coach"

Self-Managed, Cross-Functional Teams vs. RUP Roles

- Work products
  - Detailed requirement work products
  - All development work products
  - All test work products
  - Iteration plans and assessments
- Requirements Specifier
- All developer roles
- All test roles
- Technical Reviewer
- 7 or so team members
  - Team members will be playing more than one role

Support Teams on Large Projects

- Independent test team for entire integrated system
- CM team

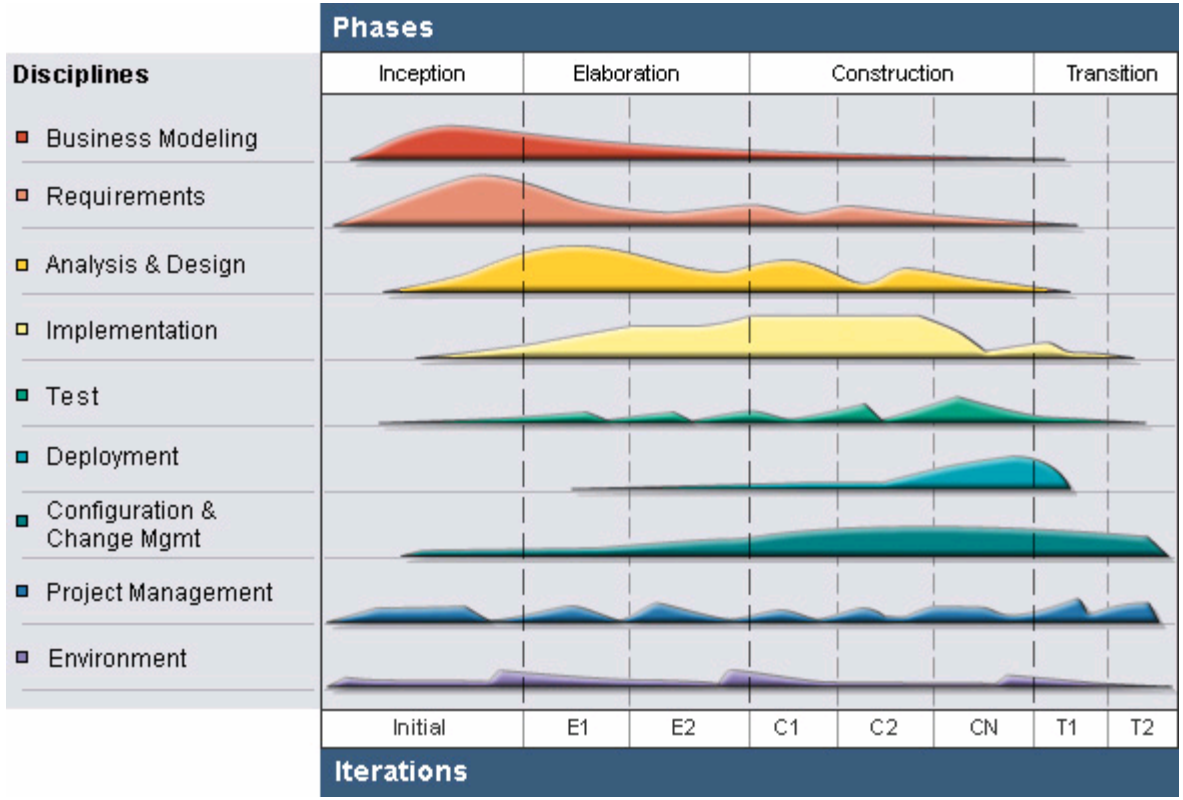
- Other support functions serving multiple development teams
  - Deployment Manager
  - System Administrator
  - Process Engineer
  - Course Developer
  - Graphic Artist
  - Tech Writer
  - Tool Specialist

Scrum Activities vs. RUP Activities

- Sprint Planning Meeting
  - Requirements Workshop and Iteration Planning rolled in one
- Daily Scrum Meeting
  - Monitor Project Status
  - Project Manager can listen in (as Chicken) to MONITOR
- Sprint Review
  - Iteration Assessment of product
    - Feedback to be incorporated into Product Backlog
    - Plans updated
  - Project Manager's responsibility

- Retrospectives
  - Iteration Assessment of work practices
    - Conclusions to be acted on in the next iteration
  - Team's responsibility

Disciplines, Phases & Iterations



Sprints vs. Iterations

- Delivering increments of potentially shippable functionality at the end of each iteration
  - Always part of RUP iterations
- Sprint Planning Meeting results in the Iteration plan
- Iteration Assessment covers
  - Sprint Review
  - and Retrospective

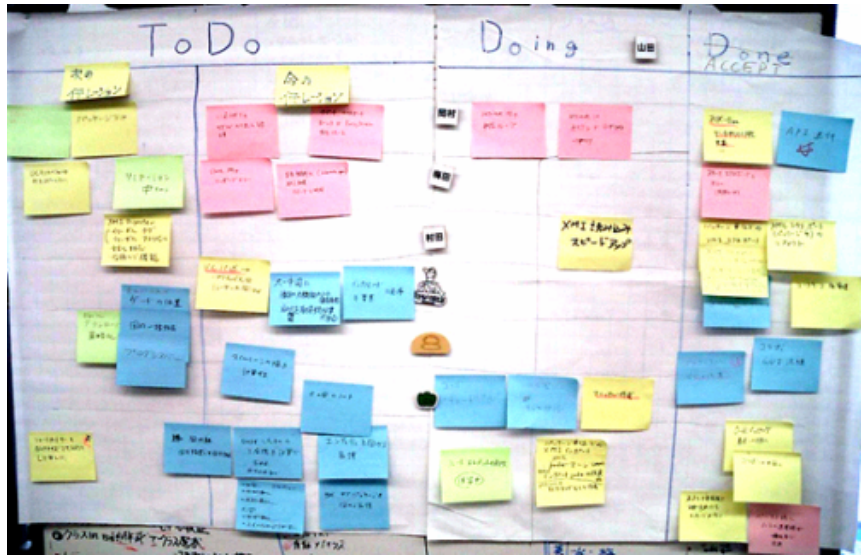
Sprint Review / Iteration Assessment

- Feedback loop at product or project level
- Clear goal: system release
- Inspection: iteration demo's or partial releases
- Change control management

month

Daily Meeting

- Feedback loop at team level
- Clear goal: iteration target
- Inspection: continuous build, automated regression testing, integration test, systems test
- Defect tracking



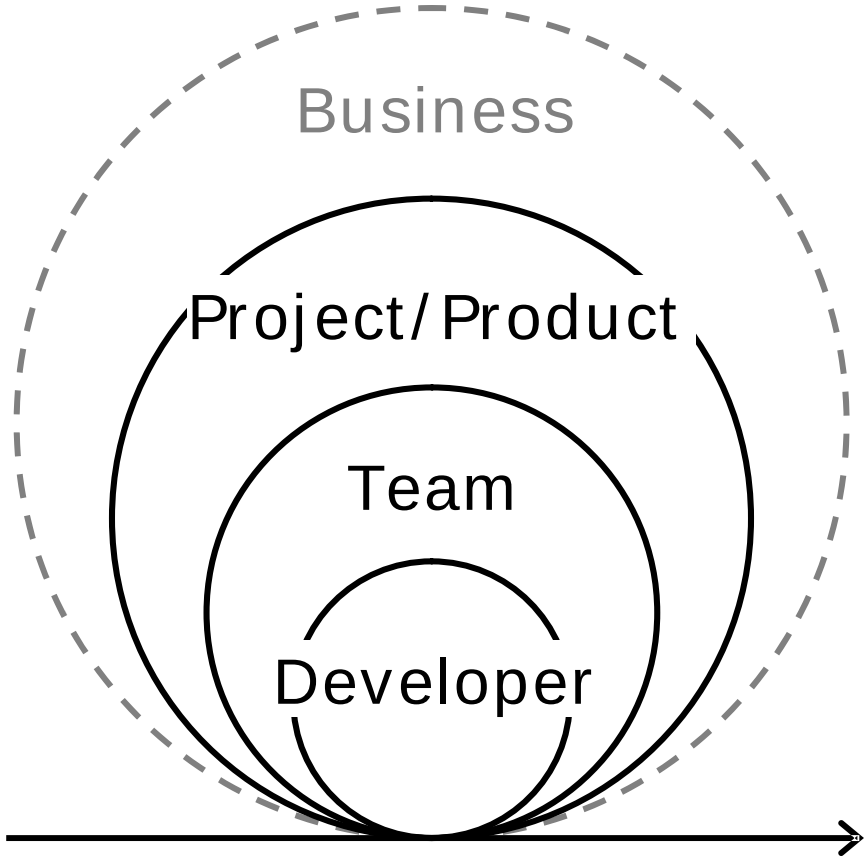
day

Edit – Compile – Debug

- Feedback loop at the level of an individual developer
- Clear goal: task, requirement or piece of function
- Inspection: local build, unit test

minutes – hours

Loops within loops



The tighter the loops,
The more agile the business

Adding Phases

- Inception:
  - building the initial product backlog (identifying use cases and non-functional features, ranking them, estimating effort, forecasting potential iterations)
- Elaboration:
  - focus on architecture, main risks resolved, velocity reasonably well established
- Construction:
  - remaining increments, can we release?
- Transition:
  - release activities

Questions?
Reflections?

IBM Rational Software Development Conference 2008

WHERE TEAMS ARE R-HEROES

IBM

Rational. software

Summary of applying the Scrum to Rational Unified Process

Elements of Scrum	Tailoring of RUP
Product backlog	Ranked list of use case scenarios and non-functional requirements maintained as part of the Software Development Plan Responsibility of Project Manager
Sprint backlog	Tailor Artefacts: Iteration plan, Work orders to capture Sprint backlog, for example as Task board Responsibility changes to that of Team
Impediment list	Same as Artefact: Issues list Responsibility of Scrum Master / Team Coach
Product Owner	Tailor Role: Project Manager Responsible for high-level management artefacts
Scrum Master	Introduce Role: Team Coach (or Scrum Master) Responsible for Issues and Risk lists
Self-managed, cross-functional teams	Adopt Scrum's team structure Responsible for Iteration plan and Work orders

Summary of applying the Scrum to Rational Unified Process

Elements of Scrum	Tailoring of RUP
Sprint planning meeting	Adopt Sprint planning meeting as the Activity: Plan for next iteration (The meeting could be called Iteration planning meeting)
Daily Scrum meeting Scrum-of-Scrums	Adopt Daily Scrum meetings (and Scrum-of-Scrums as needed) as the Task: Monitor project status Responsibility of Team and Scrum Master / Team Coach and also Project Manager as chicken (operative verb: monitor, does not imply steering)
Sprint review	Adopt Sprint review as the Task: Assess iteration 1st purpose: Determine success or failure of the iteration Responsibility of Project Manager Detailed requirements reviewing by stakeholders is dropped
Retrospective	Adopt Retrospective as the Task: Assess iteration 2nd purpose: Capture lessons learned to modify the project or improve the process Responsibility of Team and Scrum Master / Team Coach

Summary of applying the Scrum to Rational Unified Process

Requirements Elicitation

The System Analyst (member of Project Management Team) does all the high-level requirements elicitation. Use Case diagrams, brief descriptions, use case outlines and non-functional features are input to the Development Team's detailed requirements. The Development Team only needs that which is relevant to the upcoming Iteration.

High-level requirements are handed over to the team during the first part of the Iteration / Sprint Planning Meeting.

Because requirement documentation is not reviewed (replaced by Sprint reviews), the main readers of requirement descriptions are developers and testers, and the descriptions should meet their needs.

References / Sources of inspiration

- [Csik, 1990]
  - Csikszentimihalyi, M. (1990). *Flow: The Psychology of Optimal Experience*, Harper Perennial.
- [G&H, 2004]
  - Gershenson C. & F. Heylighen (2004). How can we think the complex? in: Richardson, K. (ed.) *Managing the Complex* Vol. 1 (Institute for the Study of Coherence and Emergence/Information Age Publishing)
- [Schwa, 2004]
  - Schwaber, Ken (2004), *Agile Project Management with Scrum,* MicroSoft Press
- [H&V, 2008]
  - Heylighen, F. & Vidal, C. (2008). *Getting Things Done: The Science behind Stress-Free Productivity* (http://pespmc1.vub.ac.be/Papers/GTD-cognition.pdf)
- [BC]
  - B.C. by Johnny Hart, http://www.arcamax.com/bc/s-394353-732345