STEVENS
INSTITUTE of TECHNOLOGY
THE INNOVATION UNIVERSITY

SSW-555: Agile Methods for Software Development

Final Review
Week 13

Instructor: Prof. Lu Xiao
School of Systems and Enterprises
Stevens Institute of Technology

Thanks to Prof. James Rowland for his slides

Software Development before Agile

1960s:

Original software crisis leads to "software engineering"

1970s:

Adoption of traditional engineering methods, such as the waterfall model

1980s:

Many attacks on complexity (OO, RUP, CASE tools, formal methods, iterative process models, process maturity)

"No silver bullet" published by Brooks

1990s:

WWW and Internet time

Backlash against heavy process

Agile Manifesto

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over processes and tools

Working software over comprehensive documentation

Customer collaboration over contract negotiation

Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

The DevOps Manifesto expands the Agile Manifesto to include DevOps issues

Principles of Agile Methods

Satisfy the customer through frequent delivery

Welcome changing requirements

Business people and developers work together

Motivated people working at sustainable pace

Face-to-face communication

Continuous attention to technical quality

Simplicity

Self-organizing teams

Regular reflection

Principles of Agile Methods

Satisfy the customer through frequent delivery

Welcome changing requirements

Business people and developers work together

Motivated people working at sustainable pace

Face-to-face communication

Continuous attention to technical quality

Simplicity

Self-organizing teams

Regular reflection

Different Agile Methods

Extreme Programming

Scrum

Lean

DSDM

FDD

Crystal

SAF

Spotify

DevOps

(and many others)

Different practices

- User stories
- Testing: Unit, regression, TDD
- Pair programming
- Refactoring
- Continuous integration
- Code ownership
- Kanban boards
- JAD
- Inspections
- Crystal strategies and techniques
- (and many others)

Size matters

- Large systems cannot be created by small teams
  - Larger teams require planning, control
- Some practices, such as impromptu face-to-face communication, don't scale well
  - Need to find others ways to coordinate between teams
  - Need to allow cross-fertilization across teams
- Impossible to make all tasks independent
  - Dependencies must be managed

Other things matter, too

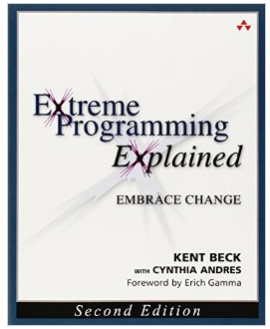
Most software development is maintenance
- Need to include operations and customer support
- Long-term planning is needed

Some systems have higher risks than others
- Special practices may be needed

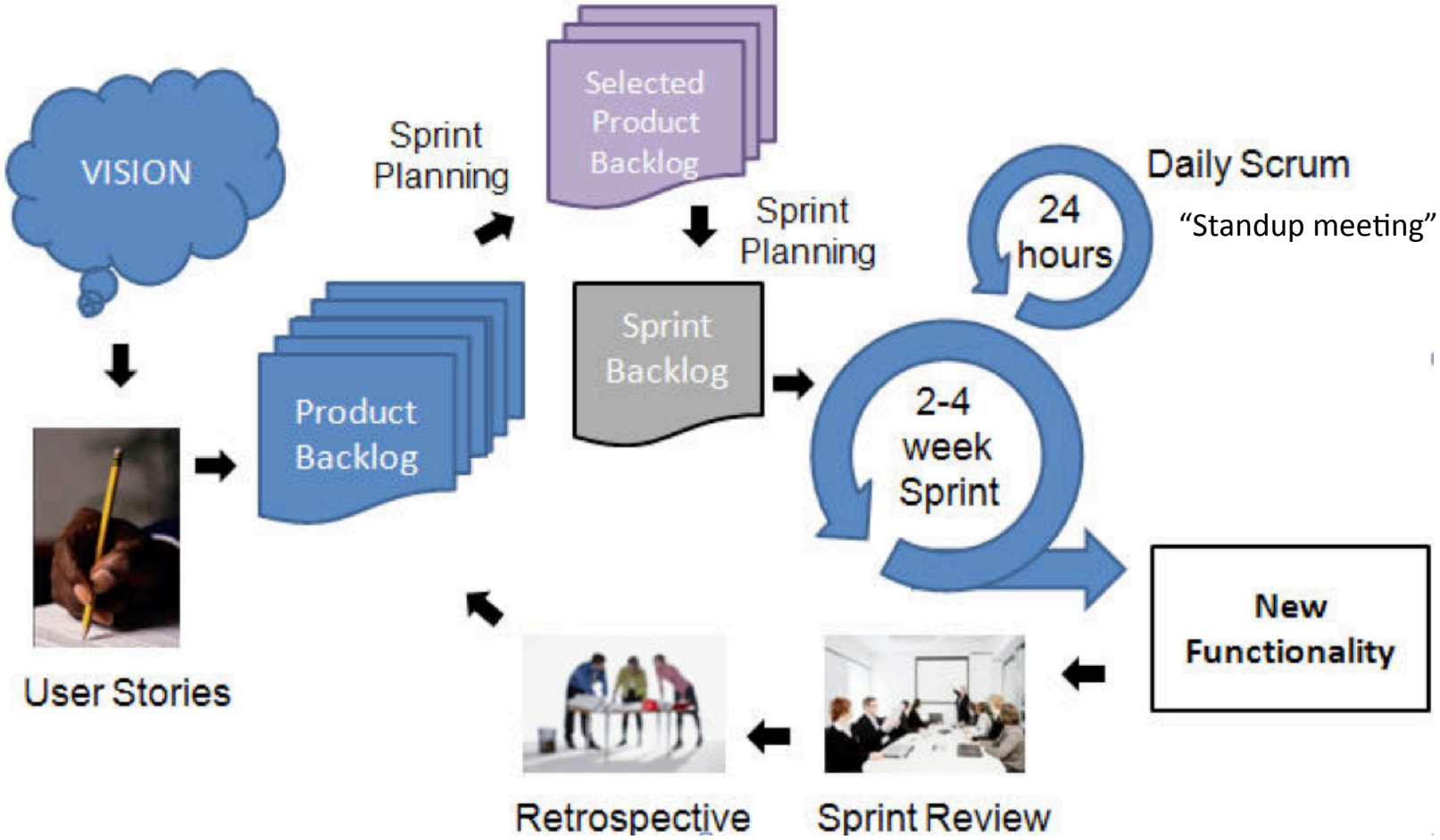
Starting and maintaining a software development project is as least as difficult as developing the software
- Need to establish relationships with all important stakeholders
- Need to understand the market
- Coordination of effort requires constant vigilance

Extreme Programming

- The Planning Game
- Small releases
- Metaphor
- Simple design
- Testing
- Refactoring
- Pair programming
- Collective ownership
- Continuous integration
- Sustainable pace
- Whole team
- Coding standards



Scrum Process



Source: http://ras-consulting.com/wp-content/uploads/2013/05/scrum-overview.jpg

Lean

Toyota's need for a better manufacturing process

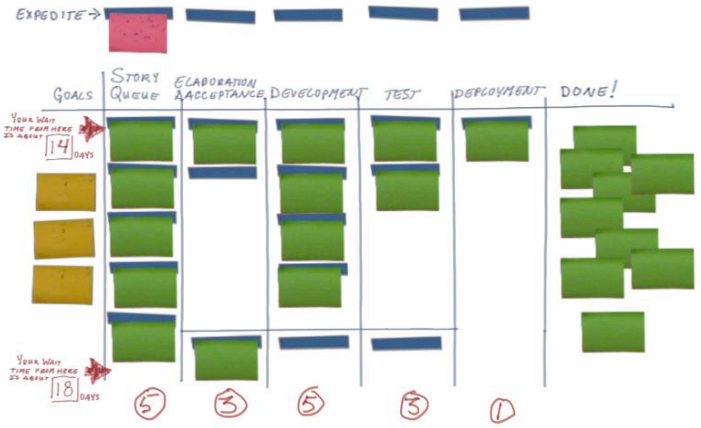
Toyota's 7 Principles of Lean

- Eliminate waste, Amplify learning,
- Decide as late as possible, Deliver as fast a possible,
- Empower the team, Build integrity in, See the whole

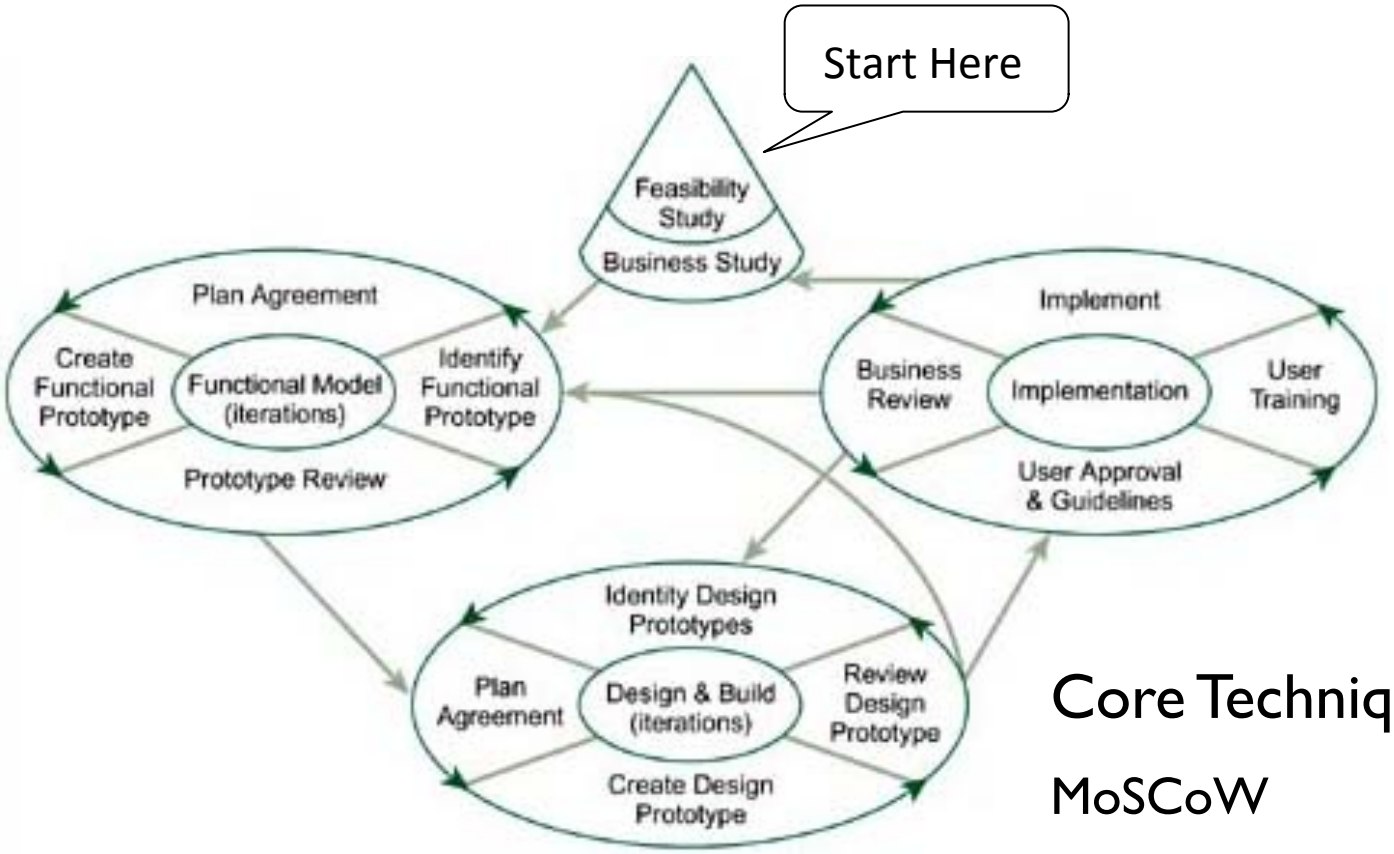
Lean practices

- Value stream mapping
- Set-based development
- Pull systems (Just In Time)

Kanban boards



RAD/DSDM Process



Core Techniques

MoSCoW

Time boxing

Evolutionary prototyping

Facilitated Joint Workshops (JAD)

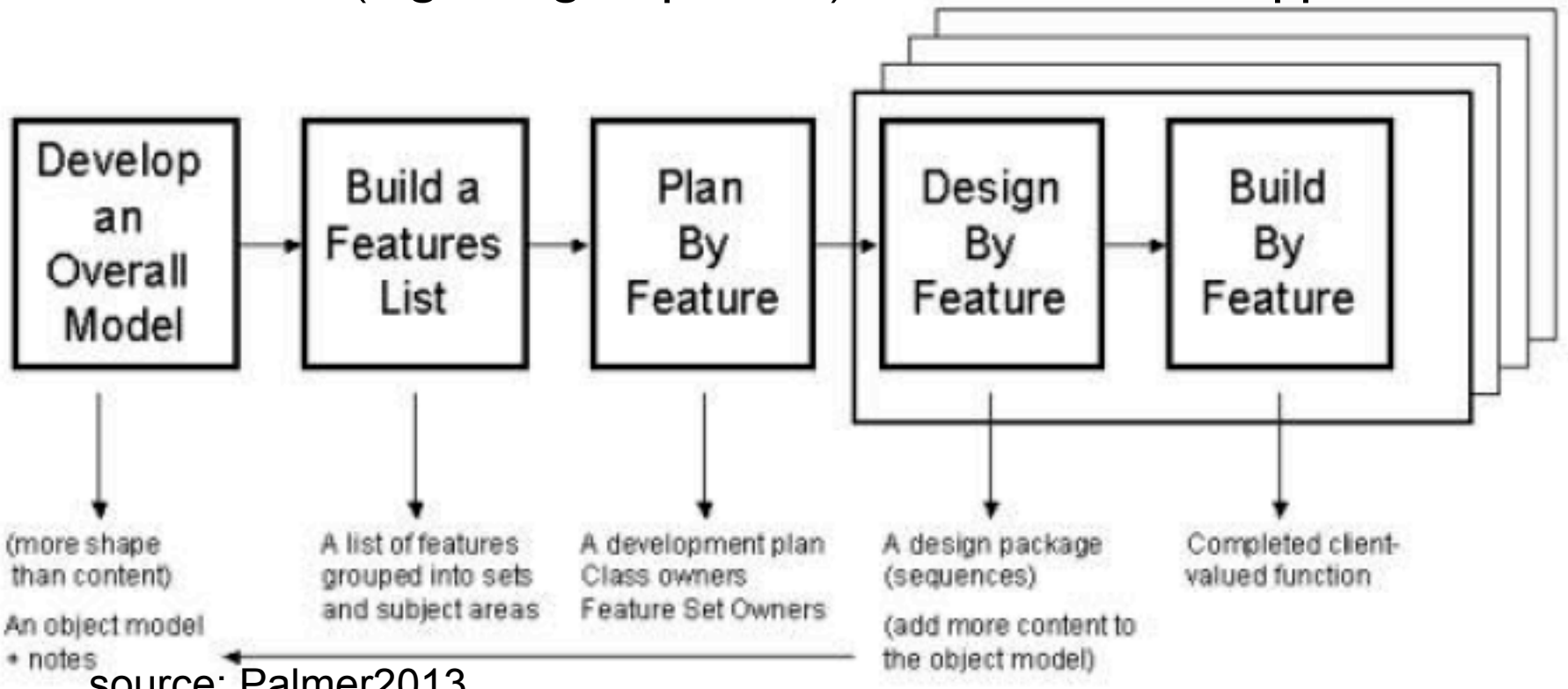
Source: Highsmith chapter 18

Feature Driven Development

Scale Agile to large projects

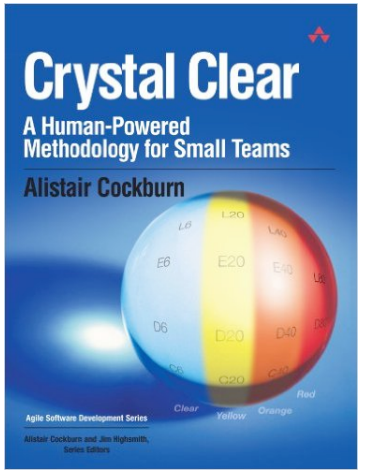
Jedi (Just Enough Design Initially) approach

Not BDUF (Big Design Up Front) as in Plan driven approaches



source: Palmer2013

Crystal

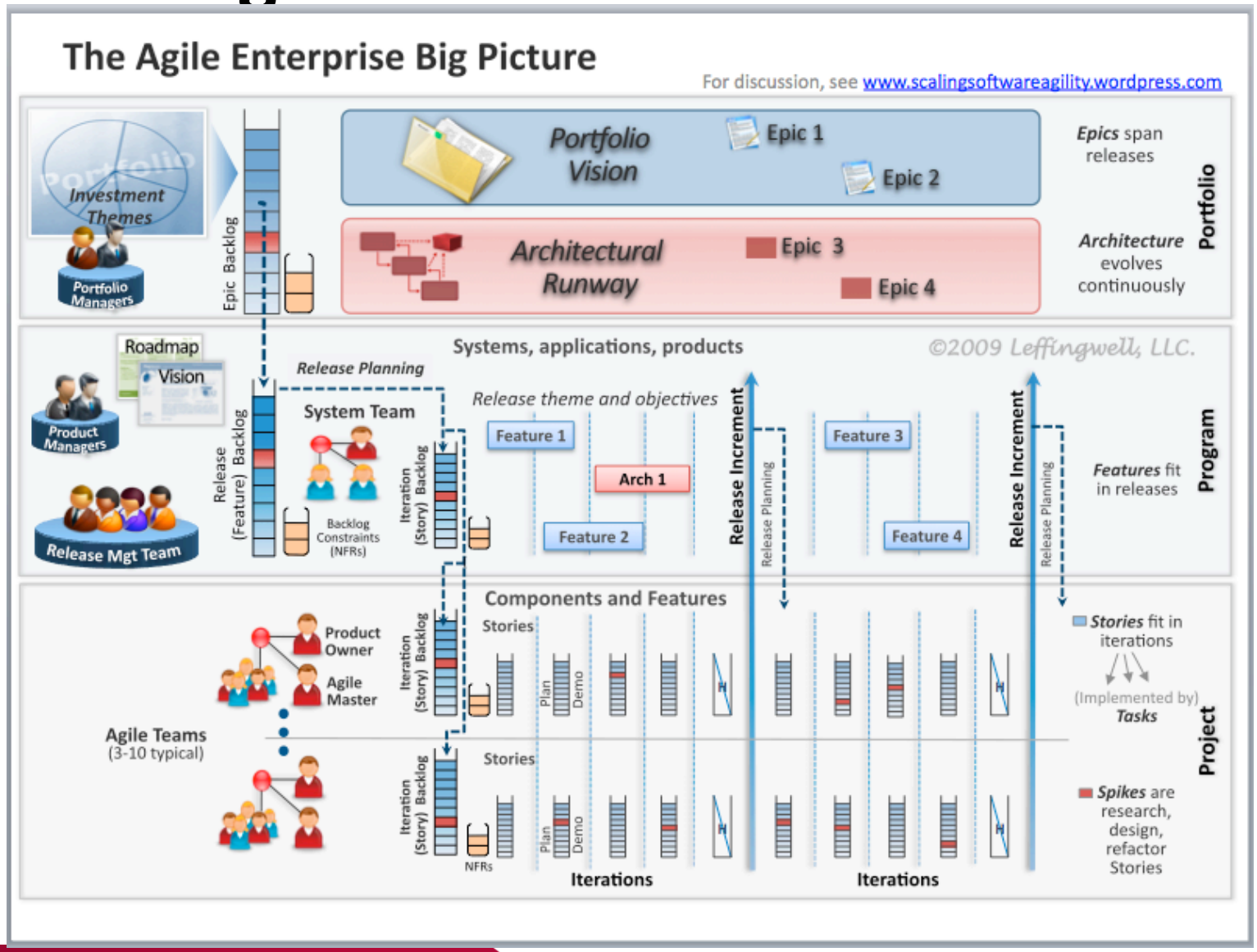


Alistair Cockburn's family of Crystal methods
- Add rigor to scale from small to large projects
- Crystal Clear, Yellow, Orange, Red

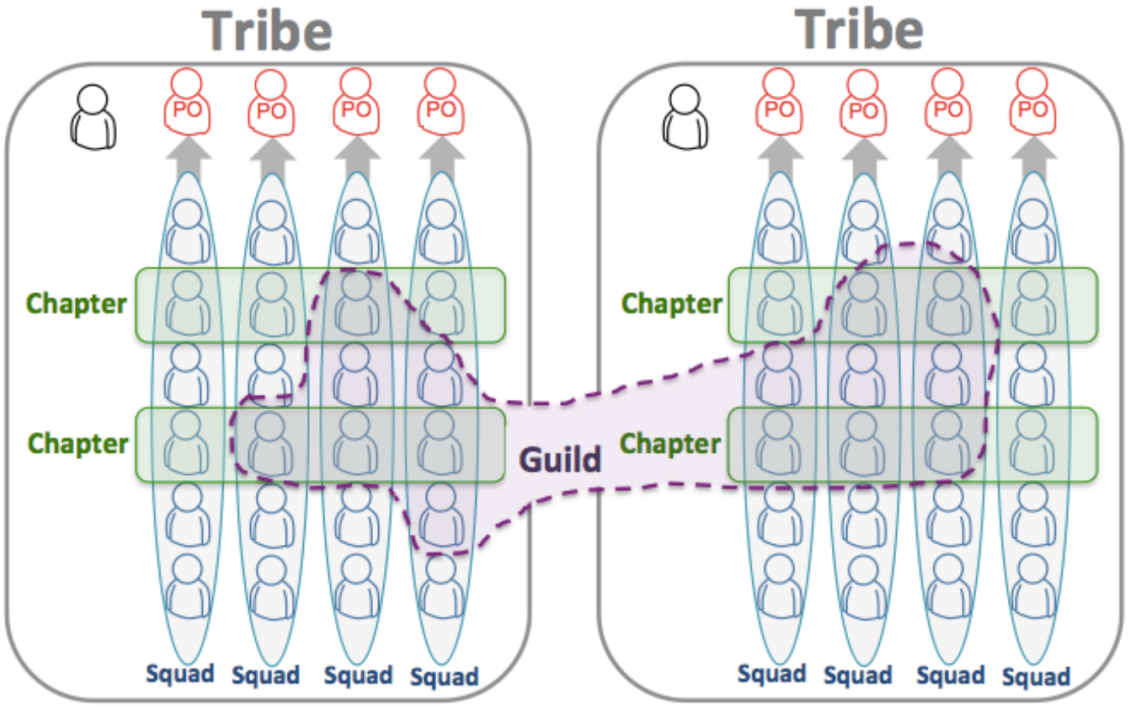
Properties, techniques, and strategies
- Borrowed from XP and adapted
- Osmotic communication, early victory, walking skeleton, …

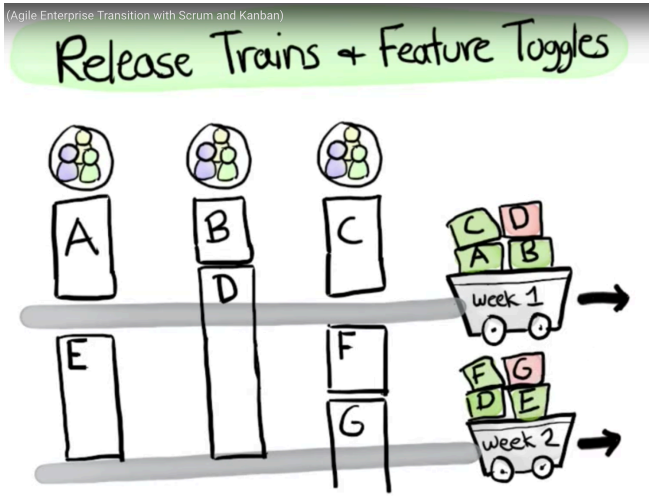
Roles and work products
- Sponsor, team, coordinator/manager
- Business expert, lead designer, tester, designer/programmer, writer

Scaled Agile Framework

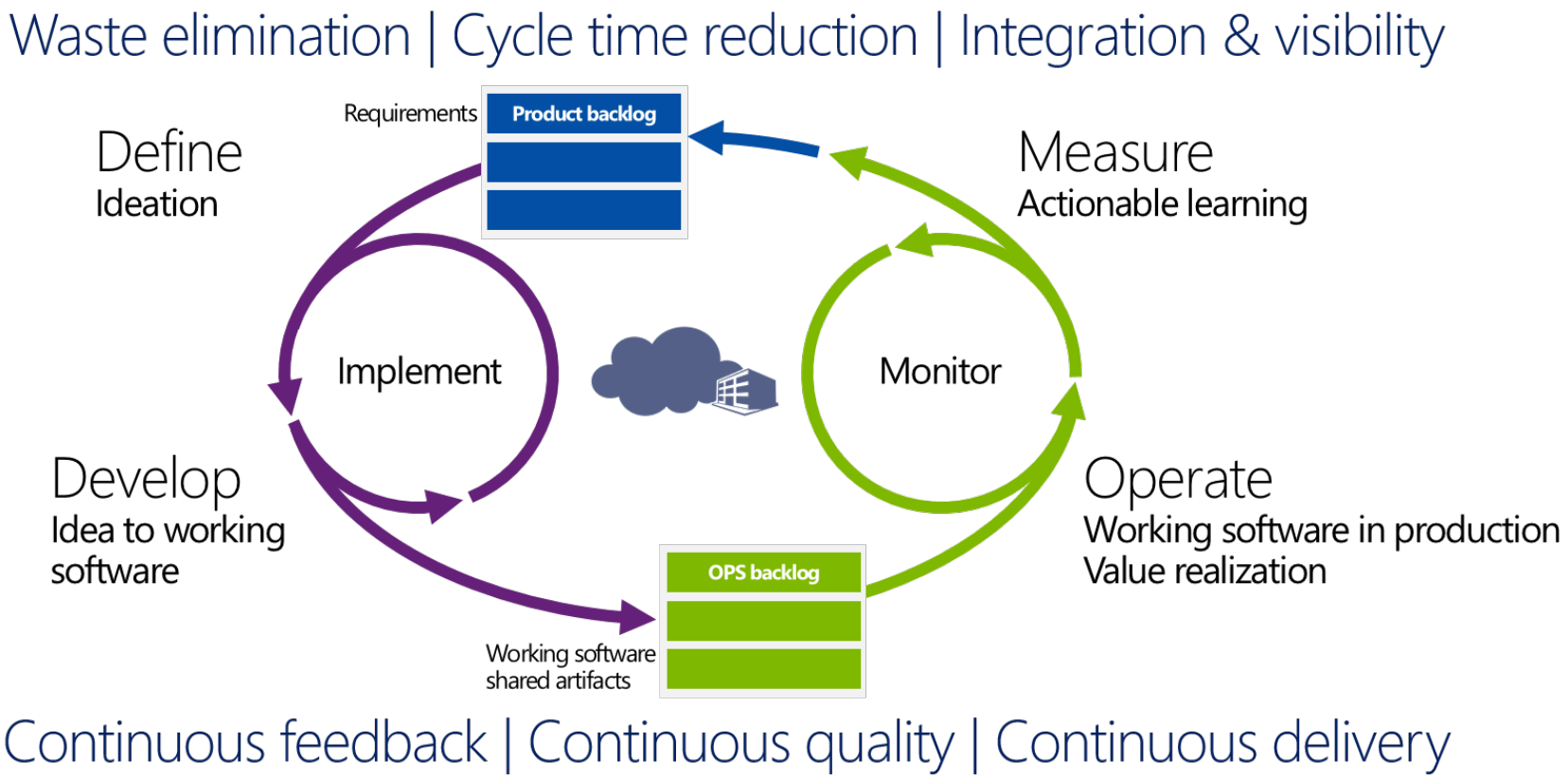


Agile Methods at Spotify



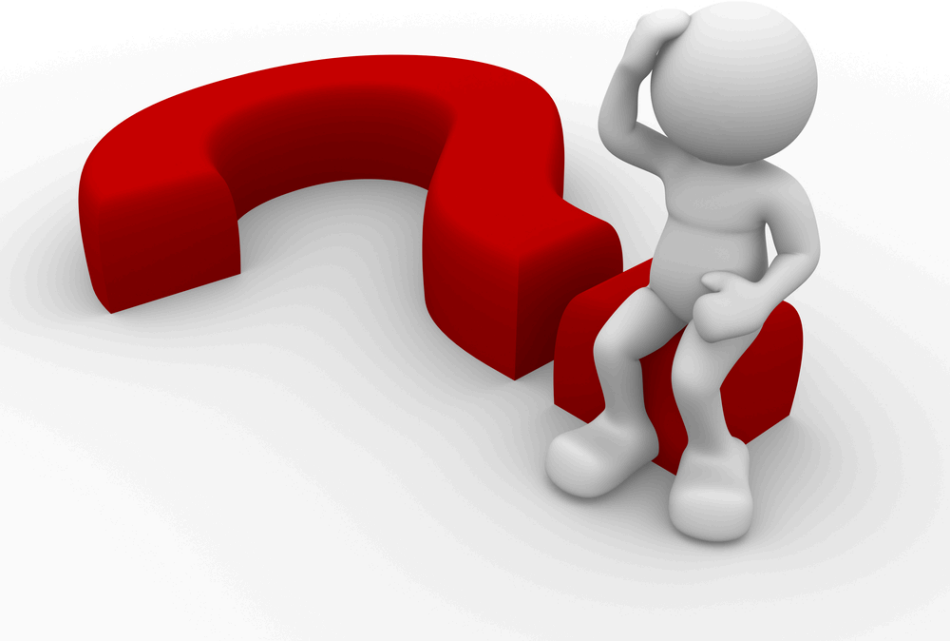


DevOps



Source: http://incyclesoftware.com/wp-content/uploads/2013/01/modern-release-cycle.png

Questions?



Good luck on the final exam!