IIBA Nottingham, May 2010

More important than ever: The Business Analysts' role in Agile software development

Allan Kelly

allan@allankelly.net

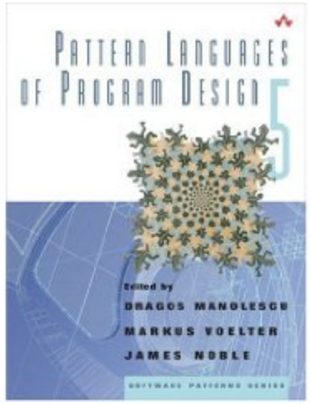
http://www.allankelly.net

Software Strategy

http://www.softwarestrategy.co.uk

Allan Kelly, BSc, MBA

- Consulting, Training & Coaching for Agile adoption and deepening
- Author:
  - *Changing Software Development: Learning to be Agile*, Wiley 2008.



97 Things Every Programmer Should Know, Henney, 2010

Context Encapsulation in *Pattern Languages of Program Design* volume 5, 2006

Collective Wisdom from the Experts
97 Things Every Programmer Should Know
O'REILLY
Edited by Kevlin Henney

33 Business Strategy Patterns for Software Creators

Agile

- Everyone familiar?
  - and Lean?

What is Agility?

Jim Highsmith, 2002

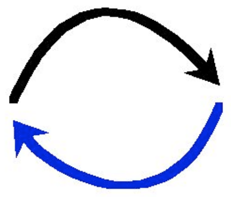
"Agility is the ability to both create and respond to change in order to profit in a turbulent business environment."

"Agile processes promise to react flexibly to changing requirements, thus providing the highest business value to the customer at any point in time"

Jutta Eckstein 2004

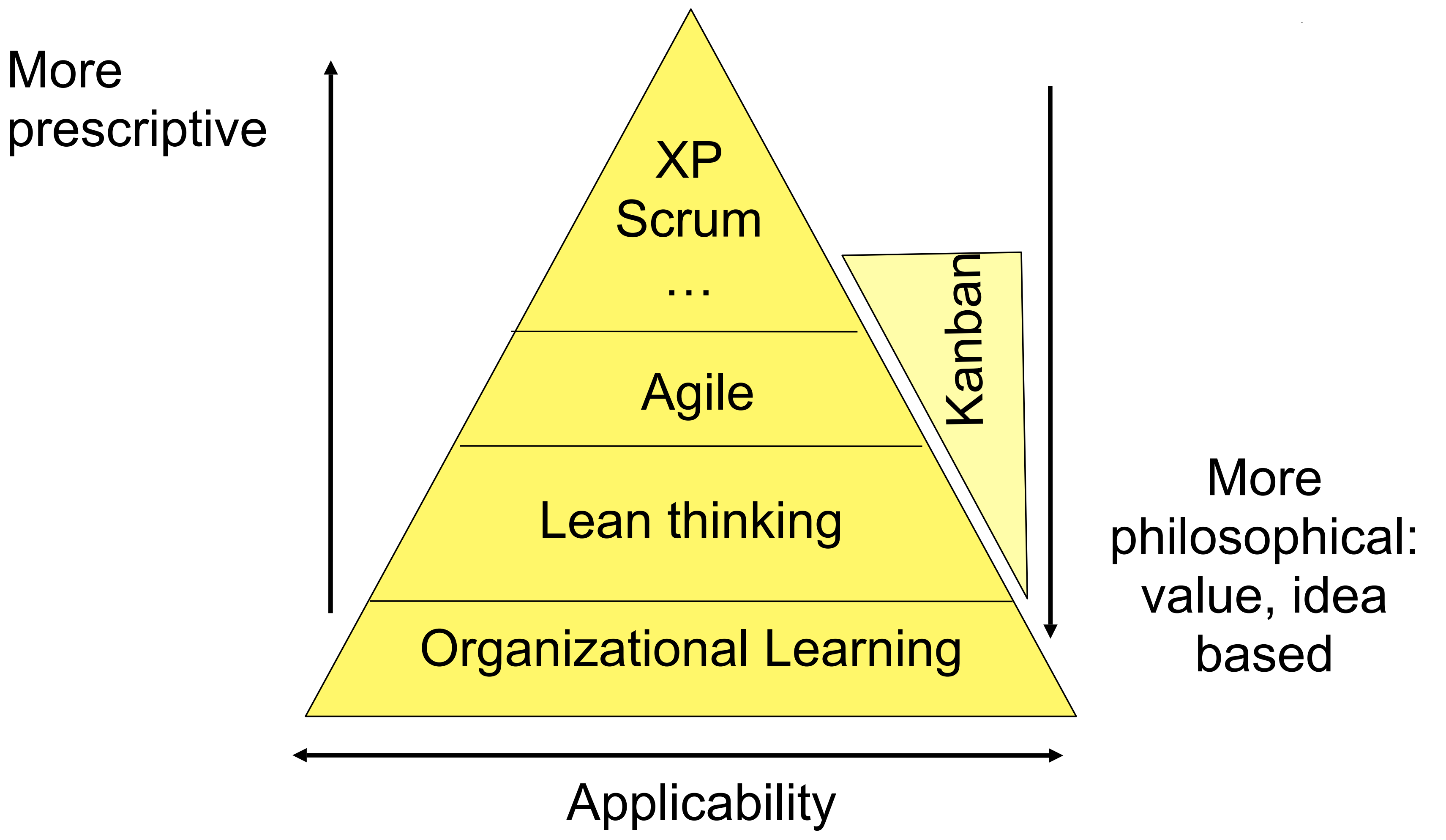
- Today: *Agile as Better*
  - Respond to changing (business) environment
  - Faster, more productive, higher quality
- Tomorrow: Agile creates new business models
  - Opportunities for those not confined by traditional IT

Agile



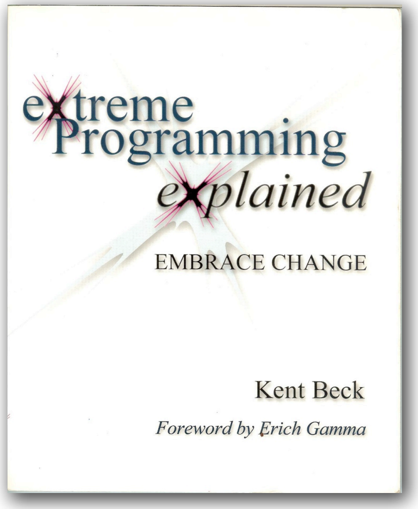
- Its the business need, *stupid*

Agile in context



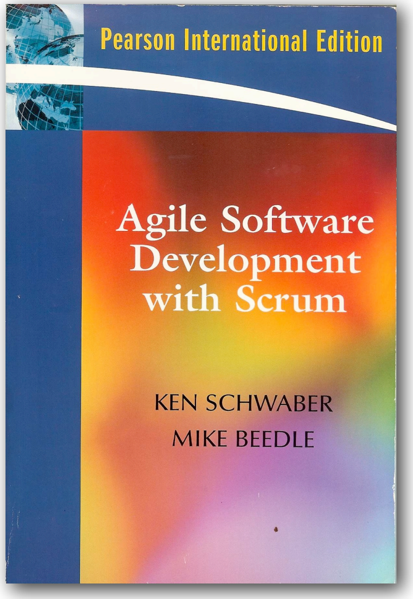
1999-2004: Agile = XP

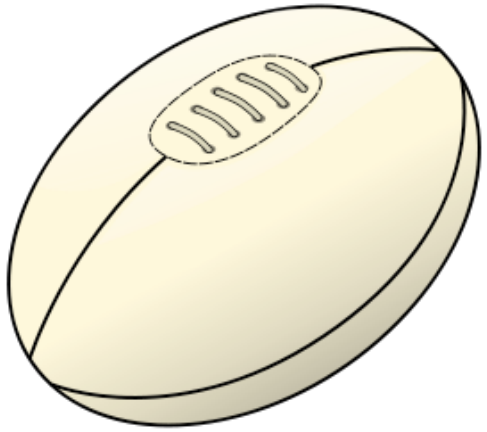
- Extreme Programming
  - First Agile method to gain popularity
  - Developer centric practices and literature
- Business need from *onsite Customer*
  - Customer on C3 was a Business Analyst
- “Customer” view too simplistic
  - Short sighted
  - Assume customer knows
  - No discussion on how the customer knows



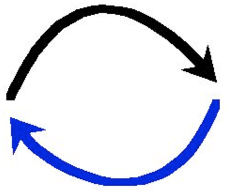
2005-today: Agile = Scrum

- Scrum
  - A project management method without a project manager
- *Product Owner* specifies need
- Scrum silent on how the Product Owner knows


Pearson International Edition
Agile Software Development with Scrum
KEN SCHWABER
MIKE BEEDLE



Who is the Product Owner?

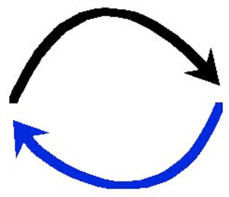


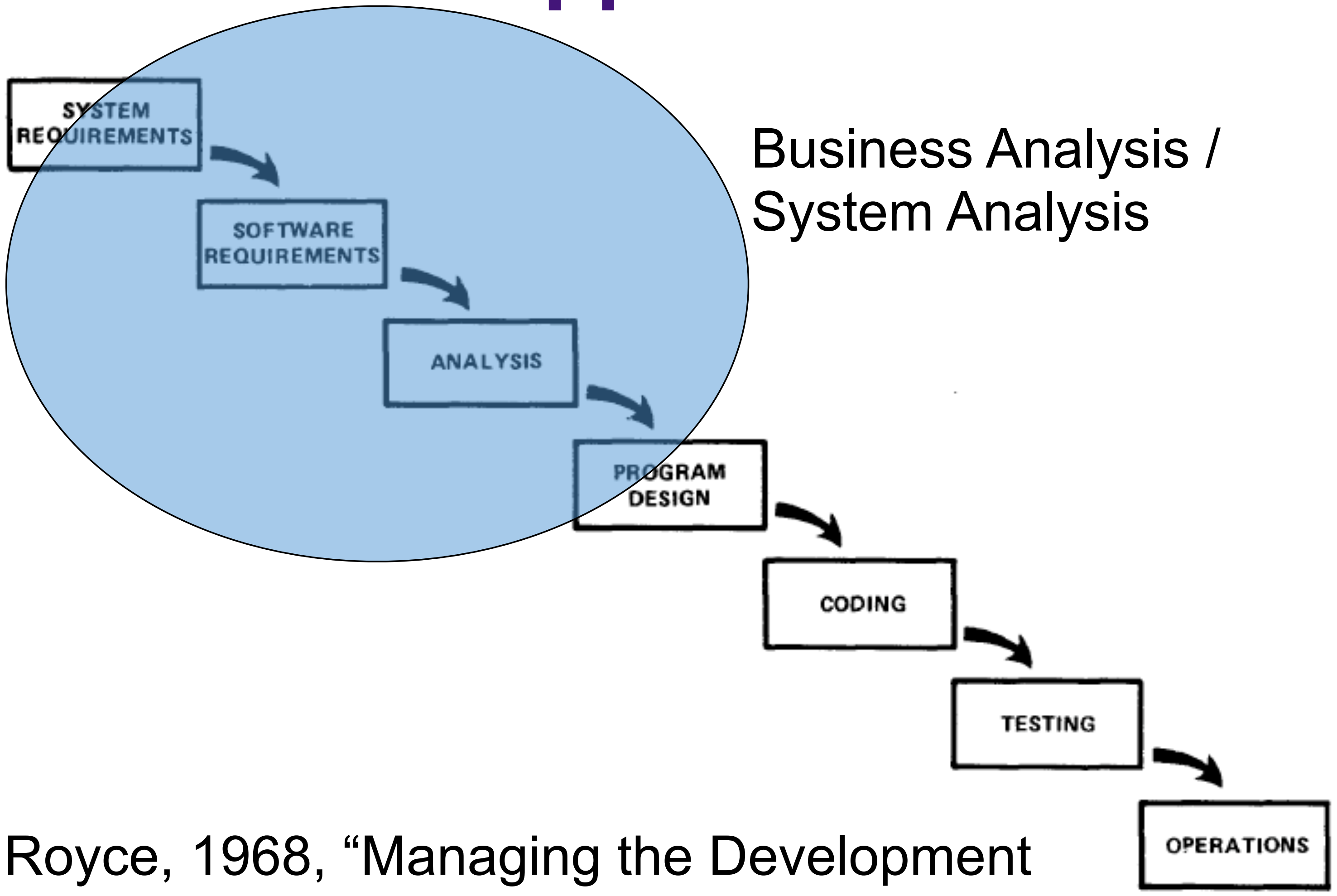
Subject Matter / Domain Expert

Product Manager

Business Analyst

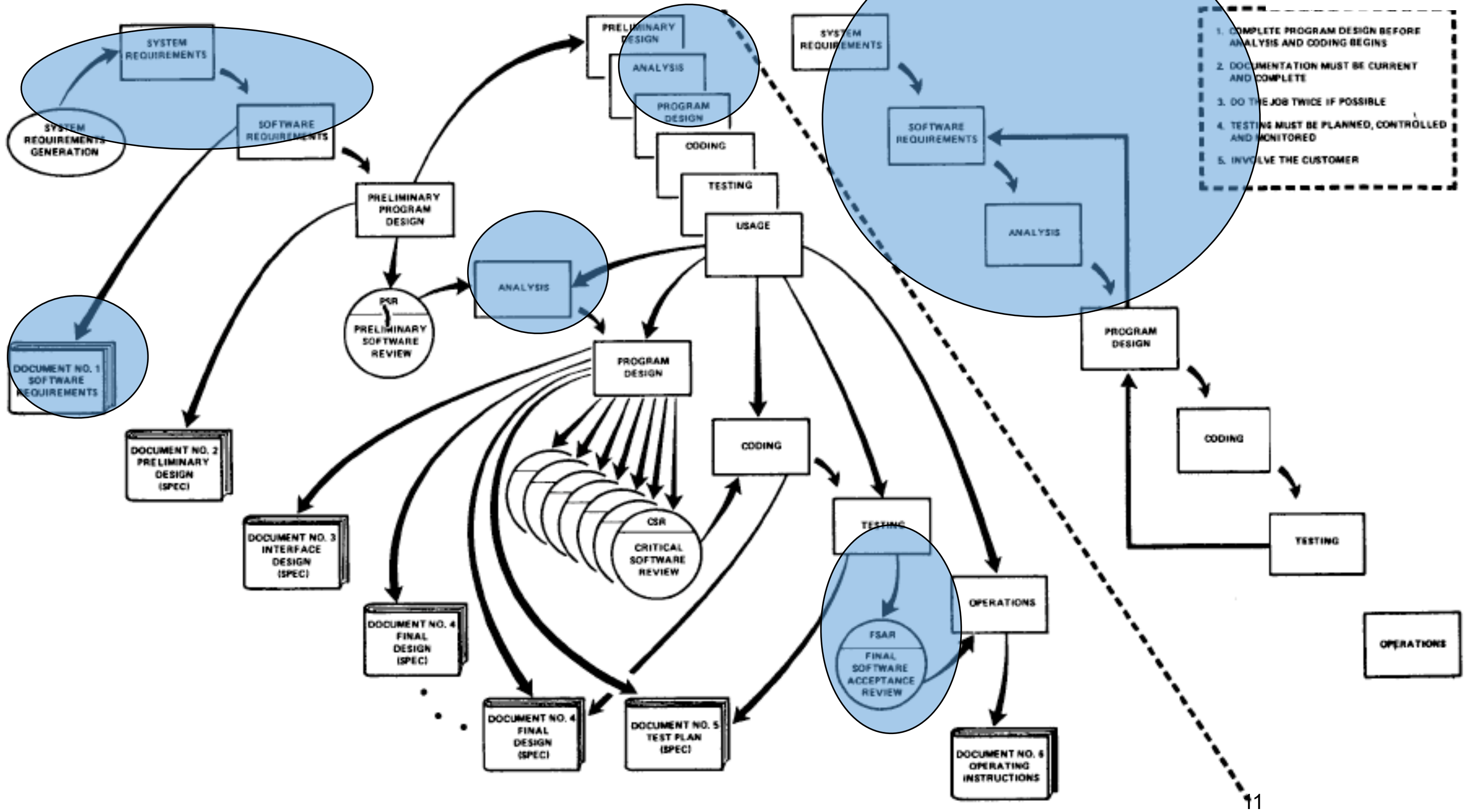
Traditional approach





Business Analysis / System Analysis

Royce, 1968, “Managing the Development of Large Software Systems”


SYSTEM REQUIREMENTS GENERATION
SYSTEM REQUIREMENTS
SOFTWARE REQUIREMENTS
DOCUMENT NO. 1 SOFTWARE REQUIREMENTS
PRELIMINARY PROGRAM DESIGN
DOCUMENT NO. 2 PRELIMINARY DESIGN (SPEC)
PSR PRELIMINARY SOFTWARE REVIEW
ANALYSIS
PRELIMINARY DESIGN
ANALYSIS
PROGRAM DESIGN
CODING
TESTING
USAGE
PROGRAM DESIGN
DOCUMENT NO. 3 INTERFACE DESIGN (SPEC)
DOCUMENT NO. 4 FINAL DESIGN (SPEC)
DOCUMENT NO. 4 FINAL DESIGN (SPEC)
CSR CRITICAL SOFTWARE REVIEW
CODING
DOCUMENT NO. 5 TEST PLAN (SPEC)
TESTING
FSAR FINAL SOFTWARE ACCEPTANCE REVIEW
OPERATIONS
DOCUMENT NO. 6 OPERATING INSTRUCTIONS
SYSTEM REQUIREMENTS
SOFTWARE REQUIREMENTS
ANALYSIS
PROGRAM DESIGN
CODING
TESTING
OPERATIONS

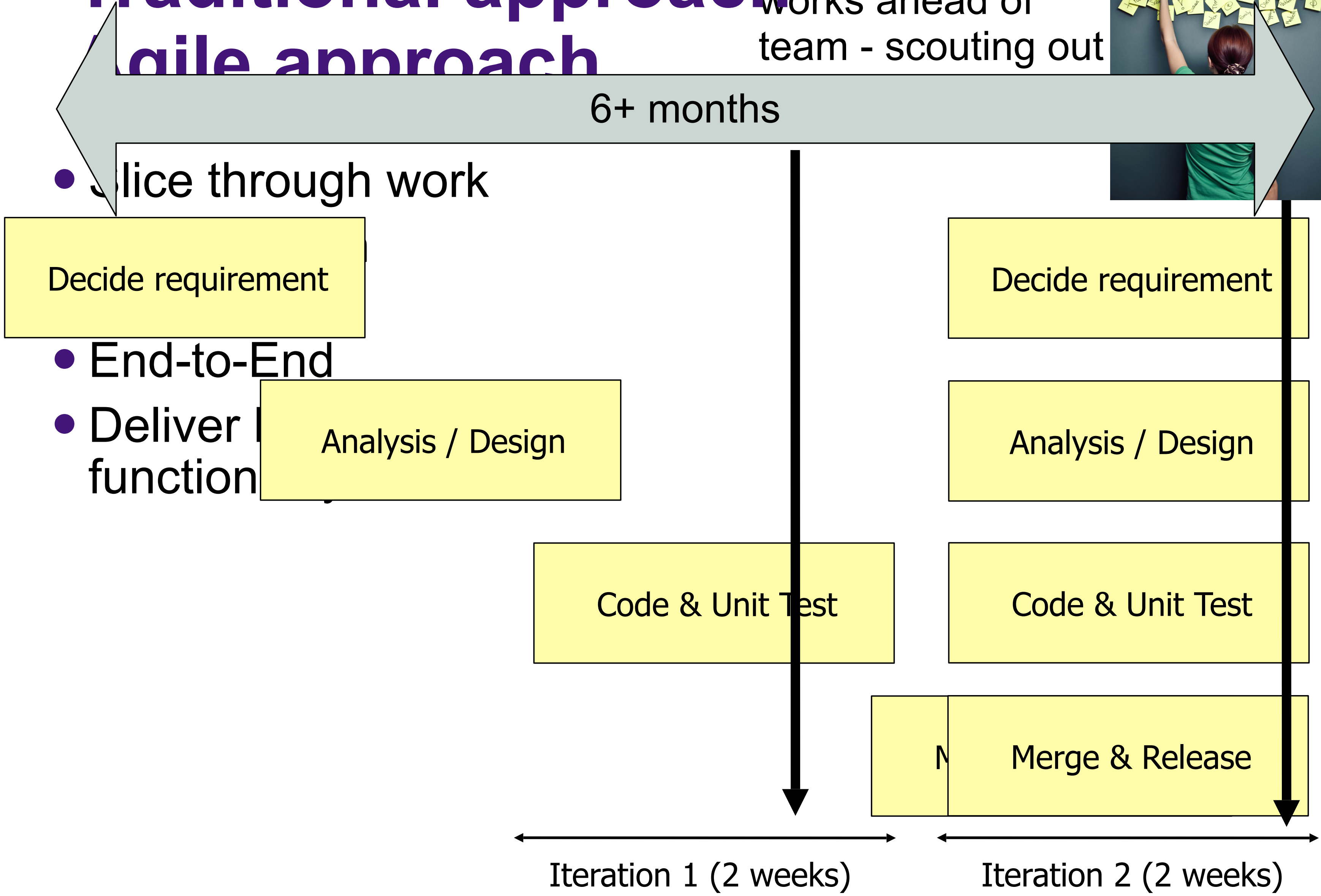
1. COMPLETE PROGRAM DESIGN BEFORE ANALYSIS AND CODING BEGINS
2. DOCUMENTATION MUST BE CURRENT AND COMPLETE
3. DO THE JOB TWICE IF POSSIBLE
4. TESTING MUST BE PLANNED, CONTROLLED AND MONITORED
5. INVOLVE THE CUSTOMER

Traditional approach

Agile approach

BA/Product Owner works ahead of team - scouting out

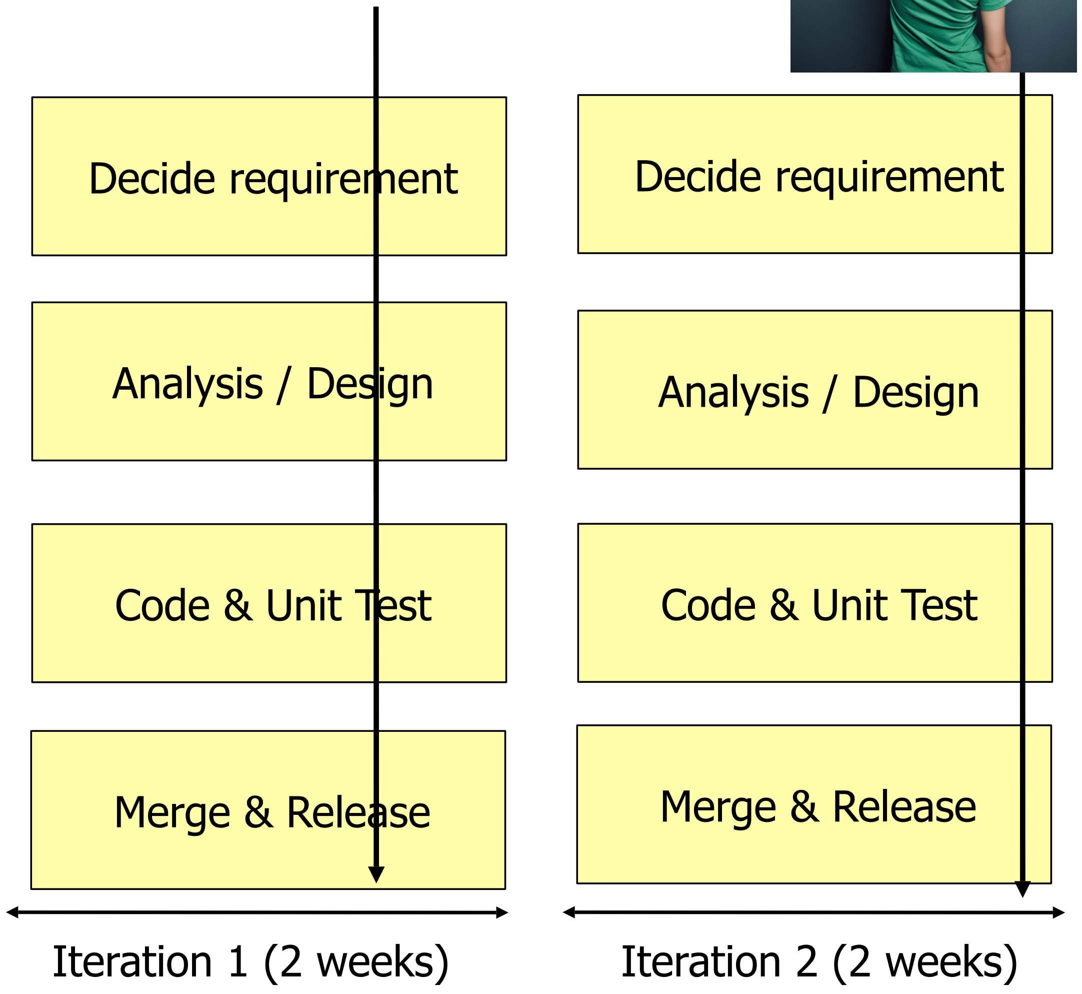
- Slice through work
- End-to-End
- Deliver functionality



Agile approach

- Slice through work
- Everything in iteration
- End-to-End
- Deliver business functionality

BA/Product Owner works ahead of team - scouting out requirements



Close quarters requirements

- Goals and objectives
  - replace *Big Requirements Documents*
  - under continual review
- Requirements gathering is ongoing process
  - rather than only at the start
- BA needs to stay involved
  - rather than leave after initial stages
- Delivered functionality changes and evolves
  - in direction of the goal and objective
- More to it than requirements gathering
  - Dialogue over document

Less (software) is more

Potentially 80% of software development work is waste

- Better requirements can reduce demand by 80%

If 30+% of requirements change then

- Why bother doing work on them in the first place?

Solution:

- **Just In Time Requirements**
- Identify, implement, deliver in quick succession

Only about 20% of features & functions in typical custom software are used

We often encounter requirements churn of 30% to 50%

Mary & Tom Poppendieck
Implementing Lean Software Development 2007

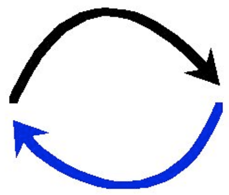
But....

There is a time and a place for everything

....

Requirements come second <u>when changing</u> to Agile

The Alignment Trap

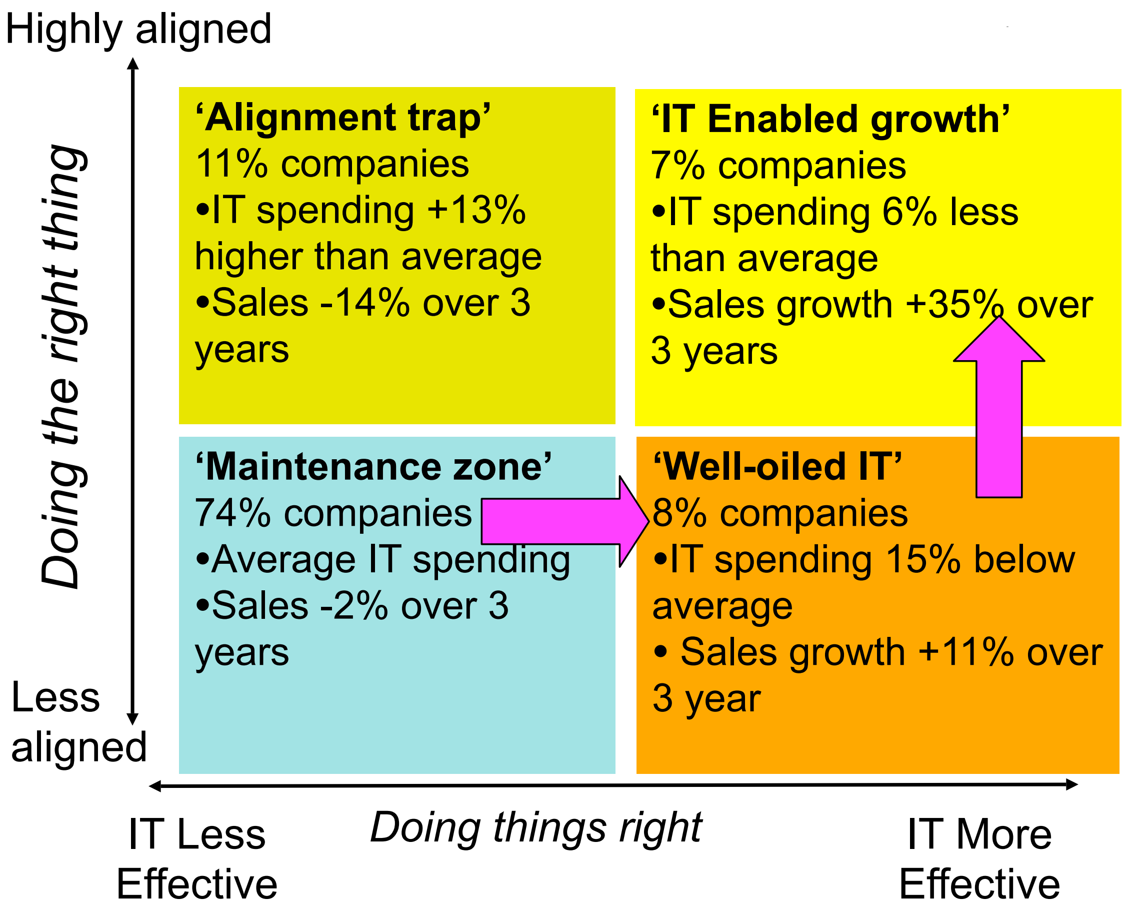


Challenge 1:

- Get Agile
- From *Maintenance* to *Well-oiled*
- Delivery focus

Challenge 2:

- From Well-oiled
- To Growth
- Requirements focus



Source: Shpilberg, Berez, Puryear, Shah: MIT Sloan Review, Fall 2007

When adopting Agile

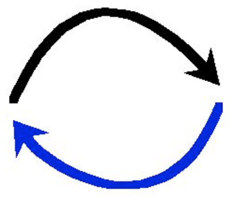
Sequence the changes

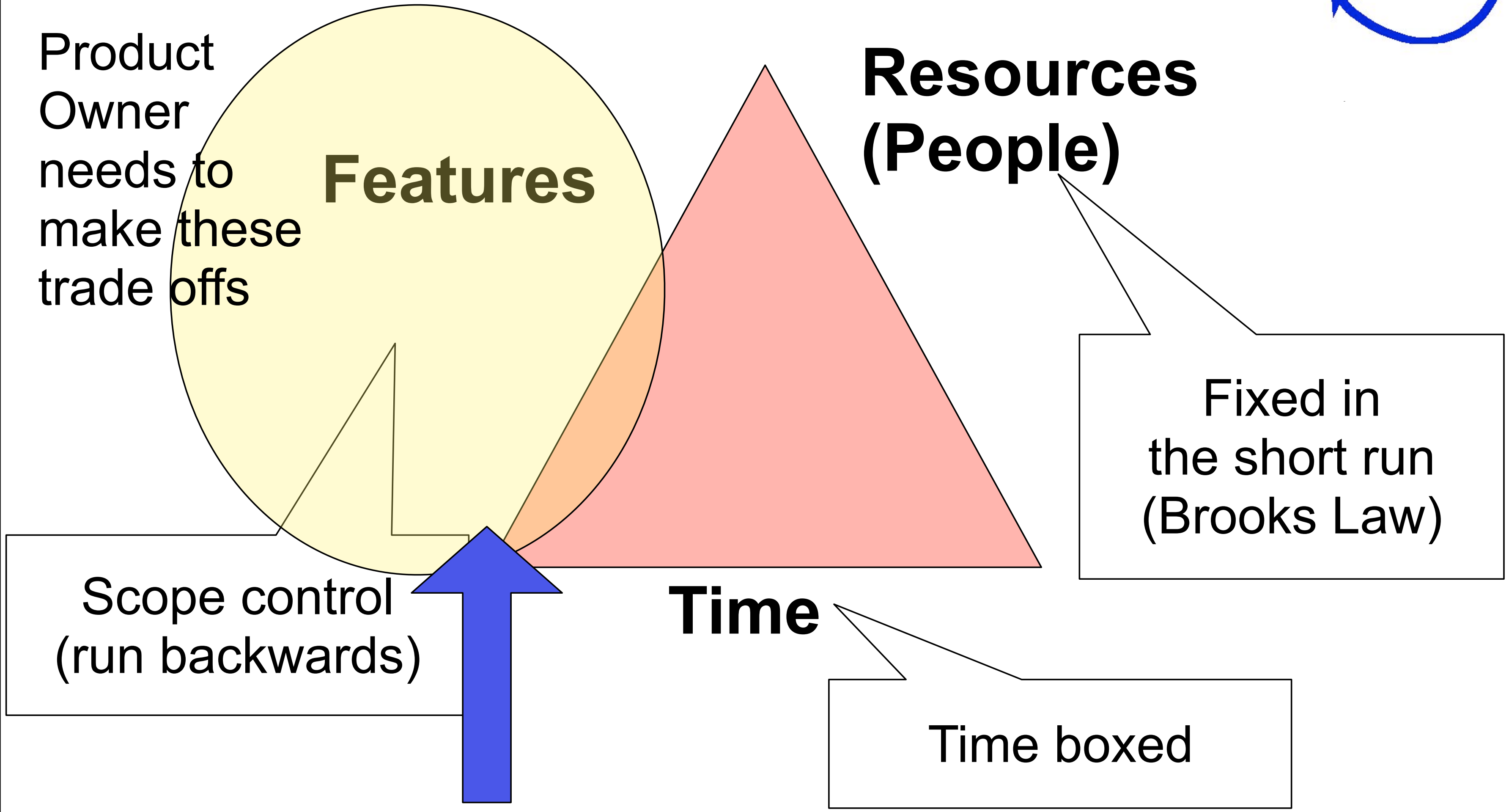
1. First ***Do it right***
   - Management focus on the development team
2. Do not emphasis requirements or BA role
3. Get developers more effective

Then

4. ***Do the right thing***
   - Focus on the what
5. Long term benefits in BA role

Project constraints



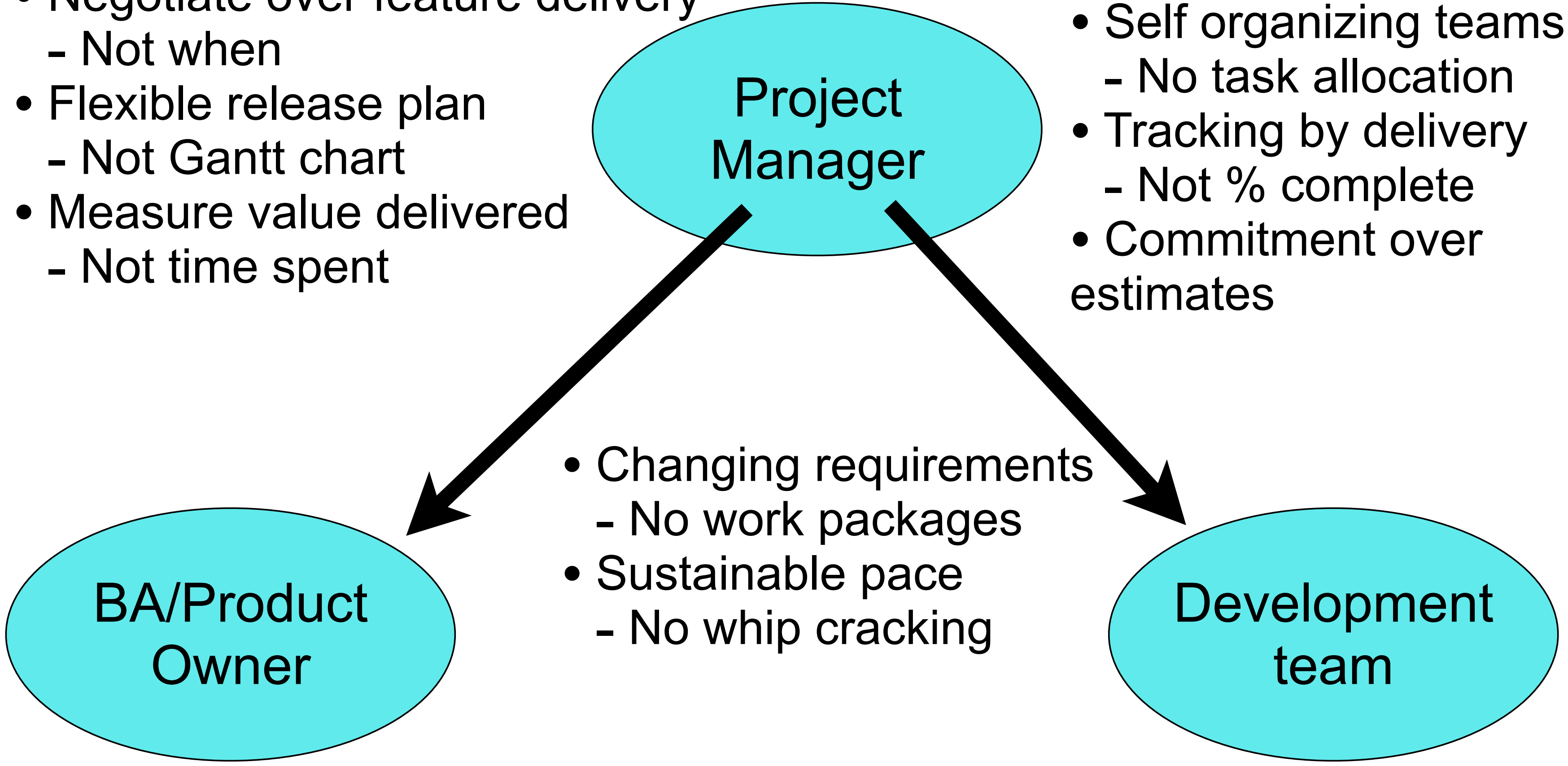

Product Owner needs to make these trade offs
Features
Resources (People)
Fixed in the short run (Brooks Law)
Scope control (run backwards)
Time
Time boxed

Agile projects negotiate over requirements rather than resources or time

More work for Product Owners Less work for Project Managers

- Negotiate over feature delivery
  - Not when
- Flexible release plan
  - Not Gantt chart
- Measure value delivered
  - Not time spent

- Self organizing teams
  - No task allocation
- Tracking by delivery
  - Not % complete
- Commitment over estimates

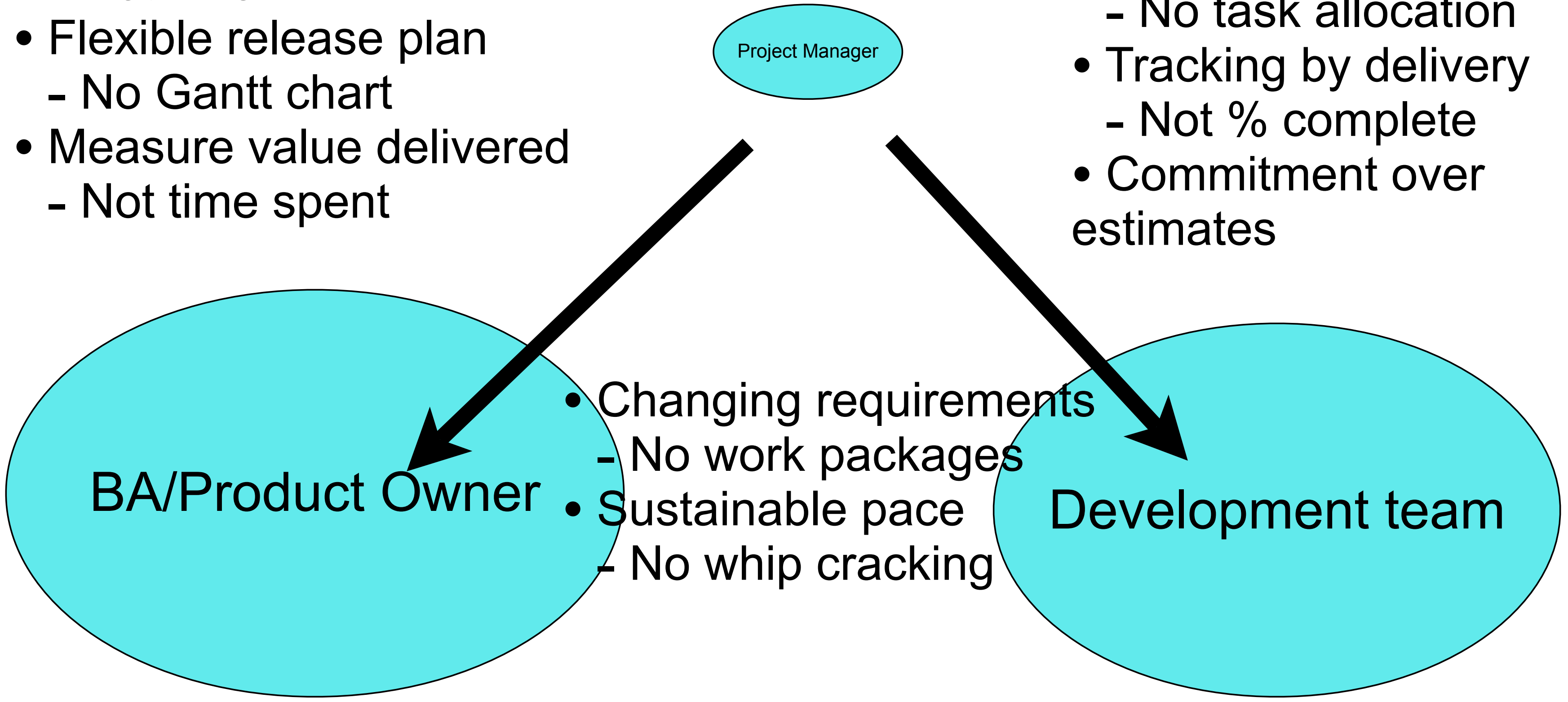


- Changing requirements
  - No work packages
- Sustainable pace
  - No whip cracking

More work for Product Owners
Less work for Project Managers

- Negotiate over feature delivery
  - Not when
- Flexible release plan
  - No Gantt chart
- Measure value delivered
  - Not time spent

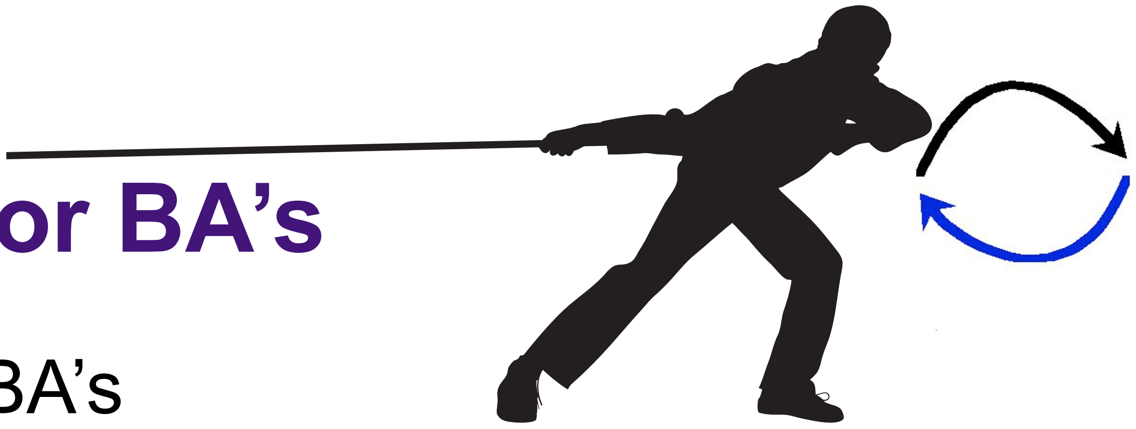
- Self organizing teams
  - No task allocation
- Tracking by delivery
  - Not % complete
- Commitment over estimates



- Changing requirements
  - No work packages
- Sustainable pace
  - No whip cracking

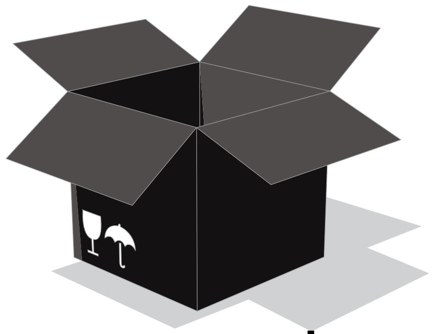
x

More work for BA’s



- More work for BA’s
  - More/better analysis can reduce work load in time
  - More responsible for value delivered
  - More conversations with Developers
  - Writing/Creating acceptance tests
  - Slack for *Just in time requirements* (Queuing theory)
- Move from ***requirements push*** to ***needs pull***
- Therefore... 1 BA for every 3 to 7 developers
  - Stable product: 1 BA -> 7 developers
  - Rapid change: 1 BA -> 3 developers

Take aways



1. Being Agile means delivering business needs
2. Product Owner is often a BA
   - Agile process does not remove need for needs
3. BA take a back seat in early transition
   - Step forward as team becomes effective
   - Key in reducing work to be done
4. Product Owner role is larger than BA role
   - Need greater staffing
   - Shift from *Requirements Push* to *Need Pull*

Thank you

allan@allankelly.net

http://www.allankelly.net

http://blog.allankelly.net

http://www.softwarestrategy.co.uk