Mike Griffiths

Speed-bumps and Potholes on the Road from Projects to Products

Presenter Background

Consultant, Author and Trainer

- >30 years IT experience on defense, finance, digital products
- 20 years agile consulting, coaching, training

Agile Experience

- Helped create Agile approach DSDM in 1994
- 25 years agile project experience (DSDM, FDD, XP, Scrum)
- Board director of Agile Alliance and APLN
- Trainer and presenter Agile Conference 2001-2019
- Author and co-author of several agile books

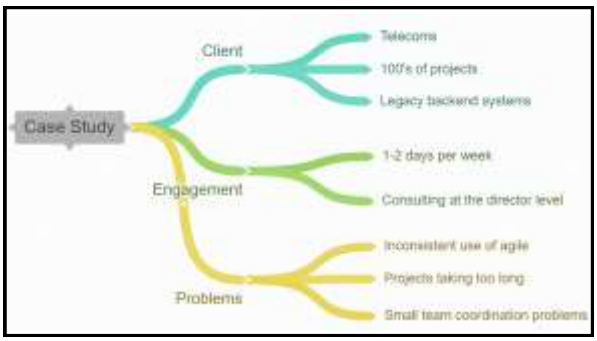
Overview and Context

Resources

Quantifying Dependency Problems

Visualizing Dependency problems

Product vs. Project Development



Why Small Teams?

Physical Work vs Knowledge Work

	Physical Work	Knowledge Work
Examples:	Picking apples, Moving dirt, Welding pipeline	Developing software, Designing new products, Filmmaking, Educational design
Nature	Tangible, Visible, Atoms (physical)	Intangible, Invisible, Bits (informational)
Ability to Scale	Easy	Difficult
Adding more people: Time	Work completes earlier	Work completes later
Adding more people: Costs	Costs increase linear	Costs increase non-linear
Economies of Scale	Positive	Negative

Physical Work has economies of scale

Knowledge Work has diseconomies of scale

Why Small Teams?

1. Diseconomy of scale
   - Adding more people does not make it go faster
   - More people adds communications overhead
   - More people just makes it cost more
2. Knowledge work based on solving problems and sharing information
3. Knowledge work is driven by Communication

Why Small Teams?

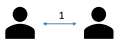
Cockburn, Alistair, *Agile Software Development*, Pearson Education, Inc. 2002.

Face-to-Face communication:
- Fast
- Cheap
- Allows for Q&A
- Conveys body language and emotions

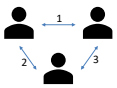
- Does not scale well

Why Small Teams?

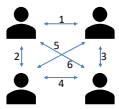
Number of communication channels = N(N-1)/2



2 people, 1 comm. line



3 people, 3 comm. lines



4 people, 6 comm. lines

Why Small Teams?

Team of 8 people, 28 comm. lines

Team of 16 people, 120 comm. lines

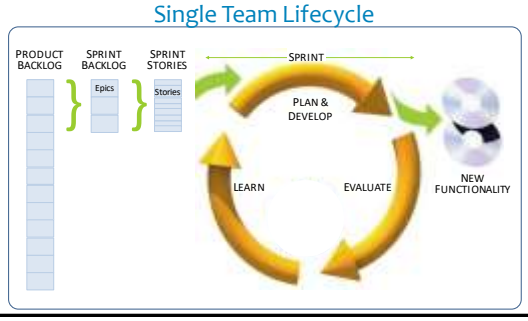
Two Pizza Teams
5-9 people – Miller's Law

Local Optimization

Goldilocks Team Size
- Not too big – Smooth communications
- Not too small – Few handoffs

Optimized for team performance

Single Team Lifecycle

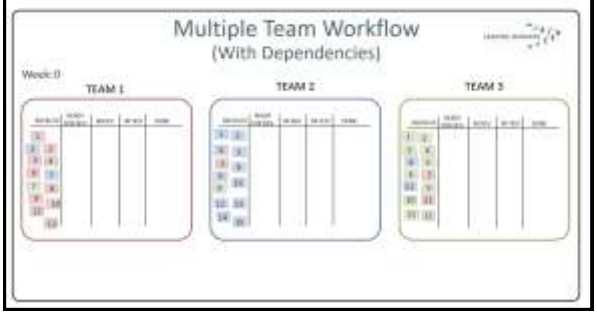


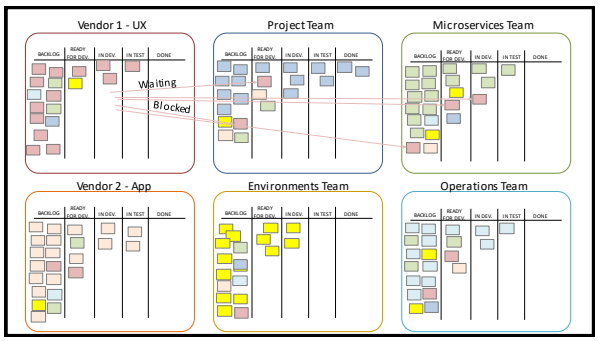
Single Team Workflow

Agenda

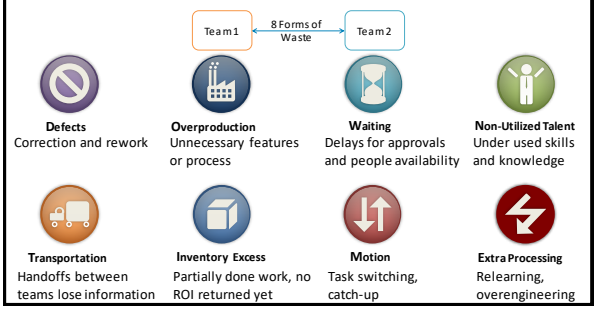
1. Single Team Flow ✓
2. Multi-Team Dependency and Handoff Issues
3. Solving Dependency Issues

Multiple Team Workflow (With Dependencies)

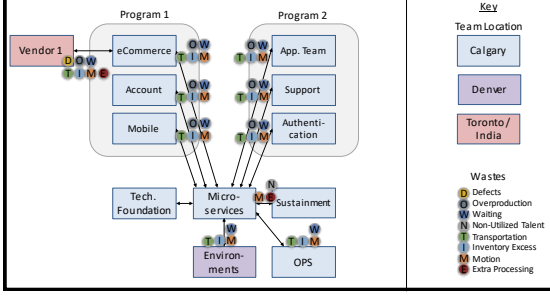




Handoffs – 8 SW Wastes - D.O.W.N.T.I.M.E.



Case Study:



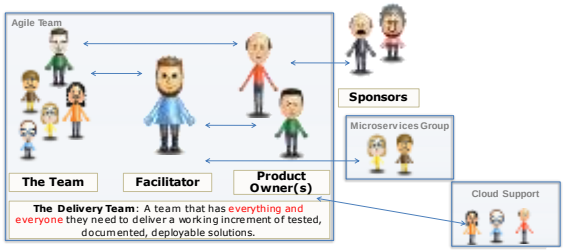
Agile Team Structures

Agile promotes moving from Silos to Holistic Teams of Generalizing Specialists

Holistic Teams: All skills necessary to develop, test, deploy and get work accepted

Generalizing Specialists: Possessing both a core competence and ability in supporting roles. Sometimes called "T" shape people.

Agile Teams Using Cloud and Microservices



Solving Dependency Problems

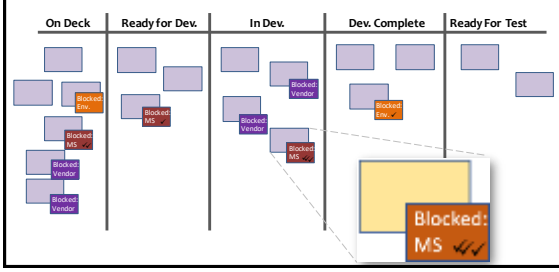
1) **Visibility**: Make dependencies, blockers and queues visible
2) **Metrics**: Measure # of dependencies, durations of blocked items, lengths of queues
3) **Expedite**: Assign people to unblock stuck work, focus on items with most dependencies

} **Easy but Limited**

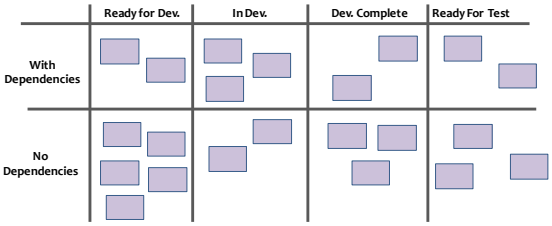
4) **Merge teams:** Consider fewer, larger teams to improve workflow
5) **Move from Projects to Products**: Restructure for long term value delivery
   - Major reduction in handoff waste
   - Knowledge is retained and developed
   - Investment focused on long term employees
   - Vendor costs minimized

} **Difficult but Better**

Make Visible: Show Dependencies and Queues



Make Visible: Show Dependencies and Queues



Metrics:

Tracked and Reported:

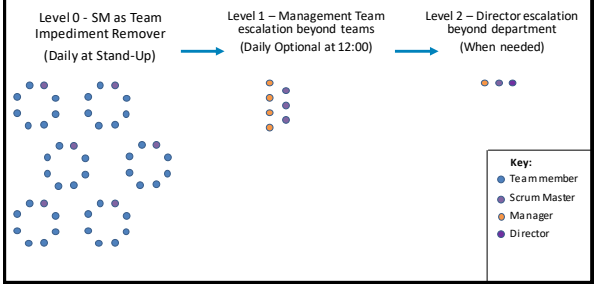
- Measure # of dependencies
- Durations of blocked items
- Lengths of queues

Expedite Blocks:

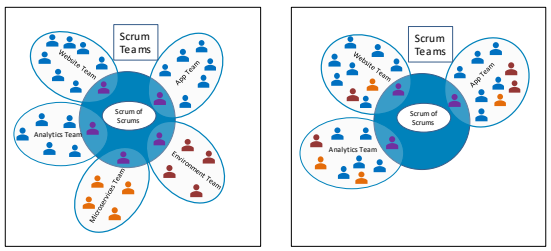
Assign people to unblock stuck work, focus on items with most (critical) dependencies

- Need sufficient authority and connections to resolve problems
- Have a clear escalation path

Expedite Blocks:



Merge Teams:



Switch from Projects to Products

1. Organizational Changes
2. Business Engagement
3. Budgeting and Planning

Current: Separation by Function

Sponsors	Vendors	Project Teams		Supporting Teams	Maintenance
Business	Vendor 1	eCommerce	Marketing Cloud	Micro-services	Sustainment
	Vendor 2	Website	App	OPS	
	Vendor 3	Single Sign On	eCare	Env.	

Future: Align Around Products. Bring Business and Sustainment Work to Product Teams

Sponsors	Vendors	Product Teams	Supporting Teams	Maintenance
Business	Vendor 1	Website		Sustainment
	Vendor 2	Apps		
	Vendor 3	Product 3		
		OPS	Env.	

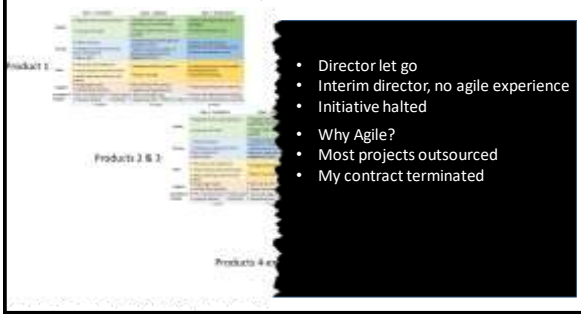
An iterative and Incremental Approach to Transitioning to Agile Product Development

	Step 1 - Foundation	Step 2 - Adoption	Step 3 - Development
People	• Organize teams around products • Co-locate Calgary staff	• Support teams towards self organizing and self managing • Vendor staff onsite as much as possible	• Teams self organizing • Vendor staff onsite only
Process	• Define Products • Establish foundational metrics (roles, behaviours) • Robust DoD	• Create product ownership and architecture roles • Metrics based on number of deliveries, experiments, & engagement levels	• Teams running frequent experiments and adapting process • Product Roadmaps in place
Tools	• Set up Jira and Confluence • Create product and team boards • Agree reporting structures and process	• Integrated Product roadmaps • Report manually	• Implement product and portfolio road-mapping tools • Automate reporting
Support	• Assign Agile coach • Provide Team training	• More SM and PO training • Support and development	• Ongoing coaching and mentoring
Acceptance Criteria	• FTE, co-located roles • Tools ready • Products defined • Coaching	• Teams self organizing • Supporting roles • Metrics in place	• Teams self organizing & managing • Productive retrospectives occurring
	4 weeks	4 weeks	4 weeks

Time

Incremental Rollout – One Product First, then a couple more, then the remainder

Incremental Rollout – One Product First, then a couple more, then the remainder



- Director let go
- Interim director, no agile experience
- Initiative halted

- Why Agile?
- Most projects outsourced
- My contract terminated

This experience did not work out. Yet if the problems sound familiar:

- Think global, not just local optimization
- Reduce Handoffs / Dependencies
- Visualize work and issues
- Sometimes bigger teams might be better
- Think products not projects

Next Steps, Questions

Get in touch:

Email: Mike@LeadingAnswers.com
Website: www.LeadingAnswers.com
Twitter: @AgileMikeG
LinkedIn: https://www.linkedin.com/in/mikegriffiths/

Visualizing Dependency Problems

Product *vs.* Project Development

LEADING ANSWERS