

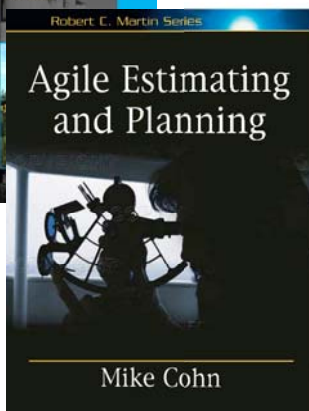
Succeeding With Agile: A Guide to Transitioning

Mike Cohn
December 4, 2007



1

Mike Cohn - background



- Agile coach and trainer**
- Founding member and director of Agile Alliance and Scrum Alliance
- Founder of Mountain Goat Software
- Ran my first Scrum project in 1995
- Typical programmer to manager etc. progression



2

Topics today...

1. Why transitioning to agile is hard
2. A framework for transitioning
3. Leadership's role
4. Patterns of agile adoption
5. Overcoming resistance



Why Transitioning
to Agile
Is Hard





1

Change is not top-down or bottom-up; it's both

- Two simplistic views of change:
 - Top down
 - Powerful leader shares a vision
 - Bottom-up
 - A team starts and everyone else sees the benefits of the new approach
- But, transitioning to agile is neither top-down nor bottom-up
 - It's everywhere, all together, all-at-once

© Mountain Goat Software, LLC

5

2

Best practices are tempting

- It is tempting to codify things that work in a given context into *best practices*
 - This leads to inflexible processes[†]
- Once we know what's "best" we stop adapting
 - Or even thinking about what we're doing
- Once we've stopped inspecting and adapting we're not agile, or won't be for long

[†]Source: Anderson, P. "Seven Layers for Guiding the Evolving Enterprise" in *The Biology of Business*.

© Mountain Goat Software, LLC

6

3

The transition process must be congruent with the development process

Part of the move to agile is a move to self-organizing teams

Moving to self-organization requires self-organization



© Mountain Goat Software, LLC

7

4

We cannot predict how an organization will respond to change

- How we traditionally view our organizations:
 - Behavior is highly predictable
 - Once set in motion, will continue in motion
- An organization change strategy can be mapped out:
 - Do this first, then that, then such and so
 - And we'll end up right where I predict



© Mountain Goat Software, LLC

8

“From a very early age, we are taught to break apart problems, to fragment the world. This apparently makes complex tasks and subjects more manageable, but we pay a hidden, enormous price. We can no longer see the consequences of our actions; we lose our intrinsic sense of connection to a larger whole. When we try to ‘see the big picture,’ we try to reassemble the fragments in our minds, to list and organize all the pieces. But, as physicist David Bohm says, the task is futile—similar to trying to reassemble the pieces of a broken mirror to see a true reflection. Thus, after awhile we give up trying to see the whole altogether.”

Peter Senge, *The Fifth Discipline*

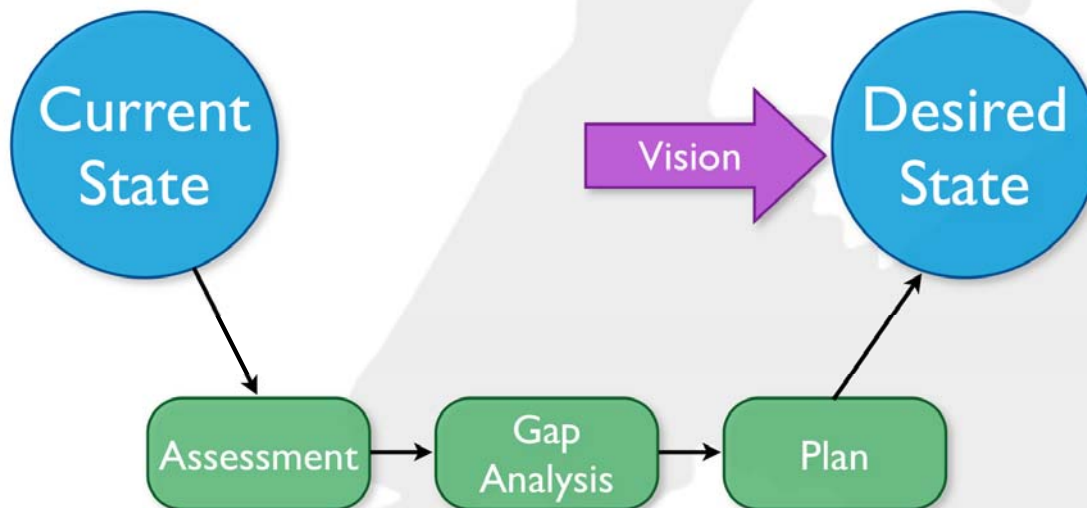


“This machine imagery [Newtonian view] leads to the belief that studying the parts is the key to understanding the whole. Things are taken apart, dissected literally or figuratively...and then put back together without any significant loss. The assumption is that the more we know about the workings of each piece, the more we will learn about the whole.”

~Margaret Wheatley
in *Leadership and the New Science*



The Newtonian view leads to thinking of change like this



© Mountain Goat Software, LLC

11

We need a different mental model

- The organization as a **C**omplex **A**daptive **S**ystem (CAS)

- A dynamic network of many agents
 - acting in parallel
 - acting and reacting to what other agents are doing
- Control is highly dispersed and decentralized
- Overall system behavior is the result of a huge number of decisions made constantly by many agents

John Holland in *Complexity: The Emerging Science at the Edge of Order and Chaos* by Mitchell Waldrop



© Mountain Goat Software, LLC

12

Differing views of success

Newtonian view

Success =
closing the gap with the
desired state

CAS view

Success =
achieving a good fit with
the environment

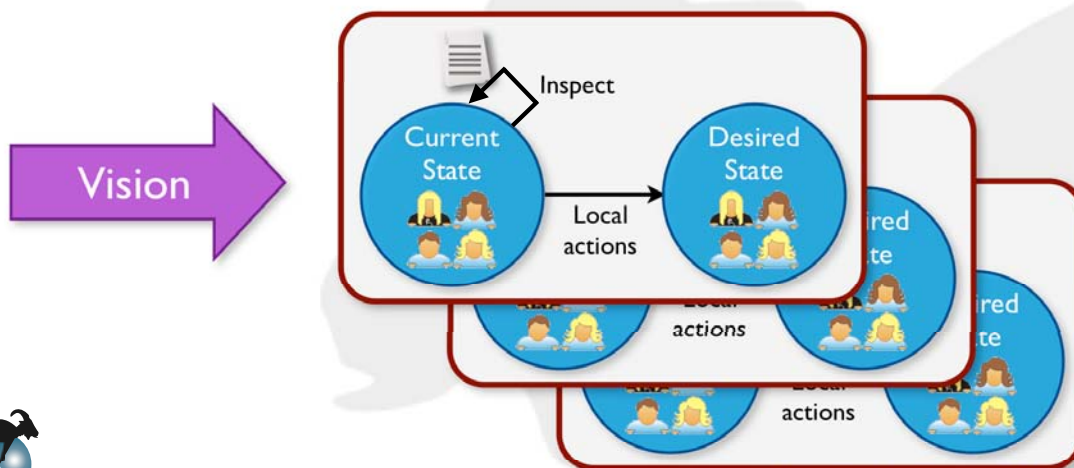


© Mountain Goat Software, LLC

13

Local goals and gaps

- Local agents (individuals, project teams, discipline coworkers) identify local gaps based on their local goals



© Mountain Goat Software, LLC

14

- Each paired statement below and on the next slide describes either the traditional or CAS view of how to change an organization
- Put an X in the appropriate column to indicate which describes the traditional view and which the CAS view



	Traditional view	CAS view
Behavior is predictable and controllable		
Behavior is unpredictable and uncontrollable		
Direction is determined through emergence and by many people		
Direction is determined by a few leaders.		
Every effect is also a cause		
Every effect has a cause		

© Mountain Goat Software, LLC

15



	Traditional view	CAS view
Relationships are directive		
Relationships are empowering		
Responsiveness to the environment is the measure of value		
Efficiency and reliability are measures of value		
Decisions are based on facts and data		
Decisions are based on patterns and tensions		
Leaders are experts and authorities		
Leaders are facilitators and supporters		

© Mountain Goat Software, LLC

16

Discuss these questions



Looking back on the previous two slides, circle the item in each pair that is most closely aligned with agile.

1

Are those views the predominant views in your organization today?

2

If not, what problems do you expect to encounter while transitioning?



An Agile
Transition
Framework

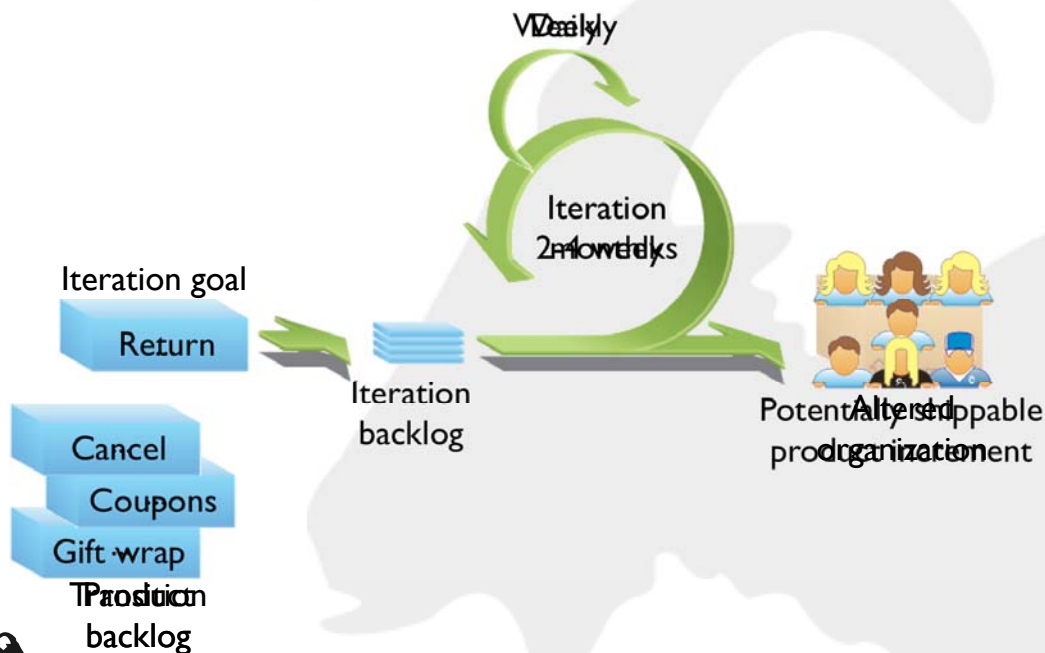


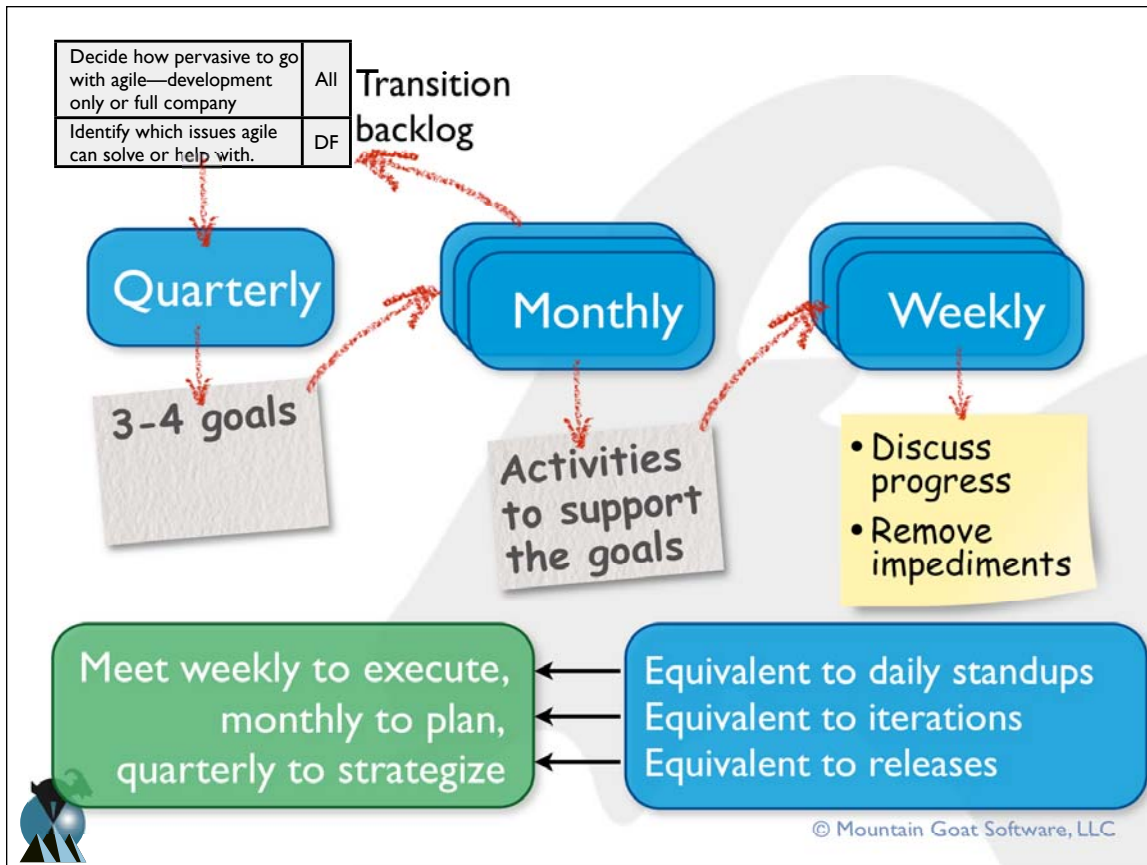
- On projects we learn we cannot precisely anticipate:
 - our users' requirements
 - how long it will take to develop a feature or entire system
 - which design will be best
 - the set of tasks necessary to develop a feature
- So we devise alternative approaches:
 - Rather than ask for upfront specs, we deliver partial solutions, solicit feedback, and repeat
 - Rather than design the whole system, we design incrementally and adjust based on what we learn

We need to do the same for the transition effort

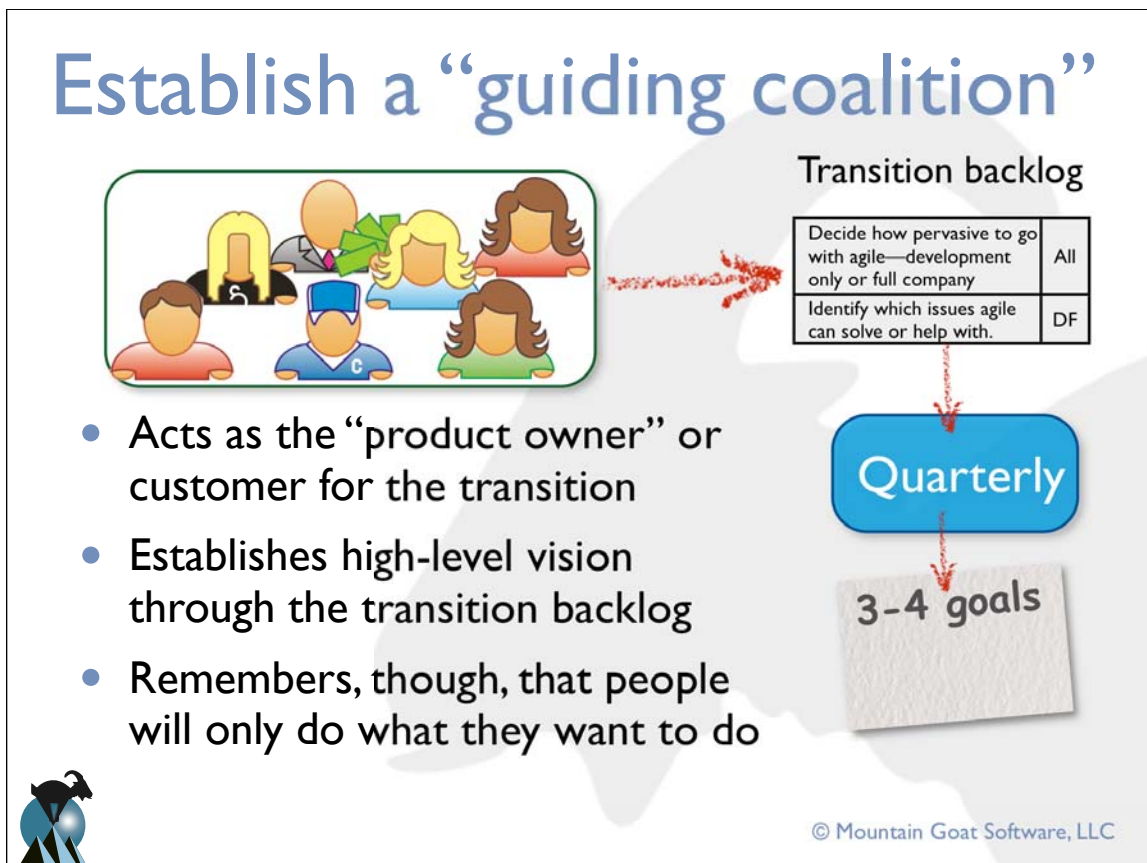


An agile transition process





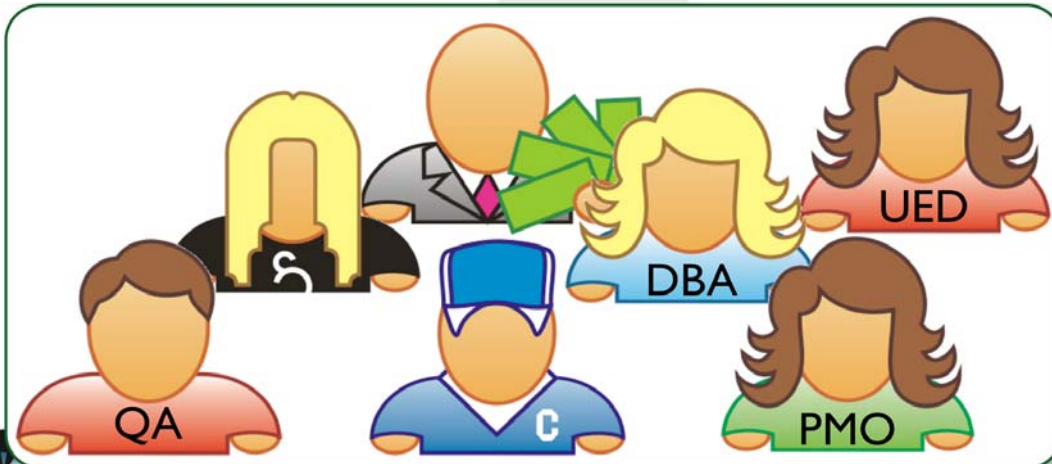
21



22

Guiding coalition members

- Sponsor—senior person responsible for success
- Area managers or leads who can make it happen



© Mountain Goat Software, LLC

23

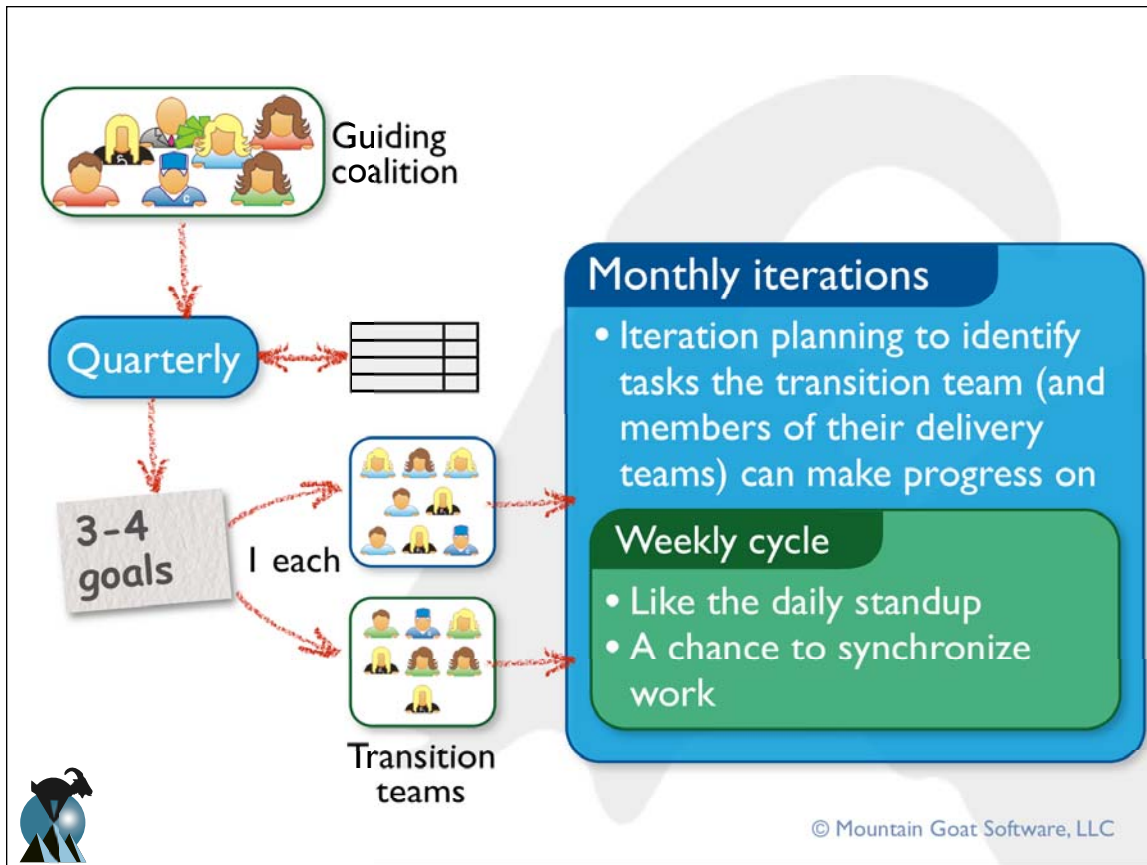
Transition teams

- Usually multiple teams pursuing different goals
- Organized around achieving specific goals in the organization
 - e.g., test automation or user experience design
- Some teams in an organization will be organic
 - Individuals notice something needs to be achieved
- Others will be formally-sponsored
 - Guiding coalition puts someone in charge of achieving a goal that hasn't been picked up
 - Usually best only if an organic team doesn't form



© Mountain Goat Software, LLC

24



25

Transition team members

- Try to form these teams organically
 - Possible with a point person to start the team
 - True product owner for the team is the guiding coalition
 - But this starting person acts as a combination day-to-day product owner and ScrumMaster
- Initial membership
 - Start with 1-3 members who “get it”
 - Ask each of those members to pick 1-2 more



26

Transition team member considerations

- Think about
 - Who has the power to make or break the transition to agile?
 - Who controls critical resources or expertise?
 - How will each be affected?
 - How will each react?



Additional considerations

- Who will gain or lose something by the transition to agile?
- Are there blocs likely to mobilize against or in support of the transition?
- Do team members have sufficient credibility that the teams' opinions and results are taken seriously?
- Can team members put their personal interests aside in favor of the organizational goal?



Who should *not* be on these teams

- People with big egos
 - Big egos fill the room; leave little space for others
 - Don't understand their own limitations
- Snakes
 - Someone who poisons relationships among team members
- Reluctant participants
 - Lack time or enthusiasm
 - But may have needed expertise or political clout



Your transition team



1. Who might desperately want the transition to fail?
 - Why?
 - What might you be able to do to prevent them from sabotaging the change?
2. Who might want to be on the transition team who shouldn't be on the team?
 - Why?
3. What hidden agendas will people bring to the transition team?
 - What can you do to counter (or make use of) those hidden agendas?
4. What can you do to handle snakes who need to be on the team?





The Role of Leadership

© Mountain Goat Software, LLC

31

Leading an agile transition

- Transition team and other formal leaders must lead the transition
 - but cannot do so in the usual ways
- Self-organizing groups still require leadership
- Lead through example, questions, and focus
 - “Nudge” the organization; Poke and prod;
 - See how the organization responds

© Mountain Goat Software, LLC

32

Pre-requisites of self-organization

Container

- A boundary within which self-organization occurs
 - Company, project, team, city, role, nationality

Differences

- There must be differences among the agents acting in our system
 - Technical knowledge, domain knowledge, education, experience, power, gender

Transforming Exchanges

- Agents in the system interact and exchange resources
 - Information, money, energy (vision)



Glenda Eoyang: *Conditions for Self-Organizing in Human Systems*

© Mountain Goat Software, LLC

33

Using the CDE model

- When stuck thinking about how to nudge the organization think of the:
 - **C**ontainers
 - formal teams, informal teams, clarify (or not) expectations
 - **D**ifferences
 - Dampen or amplify them within or between containers
 - **E**xchanges
 - Insert new exchanges, new people, new techniques or tools



© Mountain Goat Software, LLC

34

Containers

- Enlarge or shrink teams
- Enlarge or shrink the responsibility boundary of teams
- Change team membership
- Create new teams or groups



Differences

- Don't require consensus
 - Creativity comes from tension
 - Quiet disagreement is not as good as fierce debate that leads to behavior change
- Ask hard questions
 - Then expect teams to find solutions



Transforming exchanges

- Encourage communication between teams and groups
 - Who isn't talking that should?
- Add or remove people from exchanges
 - Change reporting relationships
 - Relocate people
- Encourage learning



You are the ScrumMaster or project manager...

- The next set of slides describes some teams with some trouble spots. Think about how you might help them by changing their **Containers**, amplifying or dampening **Differences**, or changing their **Exchanges**.
- For each case, identify at least one thing you'd do.
- Note whether you are tweaking their Container, Differences, or Exchanges. (You might be affecting more than one.)





1

The team consists of four developers, two testers, a database engineer and you. The developers and testers are not working well together. Developers work in isolation until two days are left in the iteration. They then throw the code “over the wall” to the testers.

2

The team is failing to deliver potentially shippable software at the end of each iteration. None of the items they start are 100% finished. Their close but work is always left to be done in the next iteration.



3

The team seems to be consistently undercommitting during iteration planning. They finish the work they commit but it doesn't seem like much. The product owner hasn't complained yet but you're worried she will soon.

4

Your organization has 20 different agile teams. Each team has its own testers who are starting to go in different directions in terms of preferred tools and approaches.





5

Jeff, a senior developer, is very domineering. During iteration planning the team defers to him on every decision even though he is a horrible estimator. You notice the glances that other team members exchange when he suggests very low estimates on some tasks.

6

You are responsible for two teams. Team members on one discuss all sides of various issues before making a decision. This has been working well. On the other team, discussions drag on endlessly because they pursue absolute consensus in all cases.



Patterns of
Agile Adoption



Two types of patterns

Adoption patterns

- Technical practices first
- Iterative first
- Requirements first
- Start small
- All in
- Stealth mode
- Public display of agility
- Impending doom

Expansion patterns

- Split and seed
- Grow and split
- Internal coaching



© Mountain Goat Software, LLC

43

Technical Practices First

Advantages

- Very rapid improvements are possible
- The transition can be quick

Disadvantages

- Technical practices support each other in subtle ways
- There is likely to be strong resistance to some practices
- Outside coaching will likely be needed

Useful when

- The most pressing issues facing the project are ones that can be solved with technical practices.
- You aren't starting a huge number of teams at once
- Team members have solid technical backgrounds
- There is a desperate need to improve



© Mountain Goat Software, LLC

44

Iterative First

Advantages

- It's easy to start
- It's hard to argue against

Disadvantages

- The team may not choose to add the technical practices

Useful when

- You want to transition more than a handful of teams concurrently
- You are starting with a stalled project
- Lots of different technologies are in use by various teams



Requirements First

Advantages

- Starting with agile requirements makes it hard to avoid being agile later
- It makes introducing other practices easier

Disadvantages

- You have to wait until the right project is ready to start
- Starting the project takes longer than it should

Useful when

- There is general agreement on what to build
- You are starting a new project or restarting a failed project
- You have the discipline and skill to do this quickly



Start Small

Advantages

- Cost of mistakes is minimized
- You can almost guarantee success

Disadvantages

- Conclusions may not be compelling
- It takes a lot of time
- Agile teams will need to work with non-agile teams

Useful when

- There is reluctance to commit fully to agile
- The risks of failing an all-at-once transition outweigh the advantages
- You can afford the time it takes



All In

Advantages

- It's over quickly
- There's no organizational dissonance from using two processes at once
- It can reduce some resistance

Disadvantages

- It's risky
- It's costly
- It will likely require a reorganization

Useful when

- You want to send a clear message
- Time is critical
- Your team isn't too small or too big



Stealth Mode

Advantages

- There's no additional pressure
- No one knows about it until you tell them
- No one can tell you not to do it

Disadvantages

- You won't have any organizational support
- Skeptics will only hear about success, they won't witness it

Useful when

- You want to experiment
- You don't have any organizational support
- You expect strong resistance



Public Display of Agility

Advantages

- Everyone knows you're doing it so you're more likely to stick with it
- It establishes a vision to work toward
- Makes a firm statement that you are committed to transitioning

Disadvantages

- Announcing something before you do it can make you look foolish
- Resistors will come out of the woodwork

Useful when

- You are confident in the approach and committed to achieving it
- You are likely to face stiff resistance and want to face it all at once



Impending Doom

Advantages

- It can shock the team out of complacency
- Admitting that a project is headed toward disaster can free the team to experiment
- It can help overcome a lot of resistance
- The transition can be quick

Disadvantages

- It isn't always an option
- A big change in a time of trouble can increase stress on the team

Useful when

- A project is on its way to failure unless dramatic action is taken
- Apathy has set in among team members



Patterns of agile adoption

Discuss these questions:

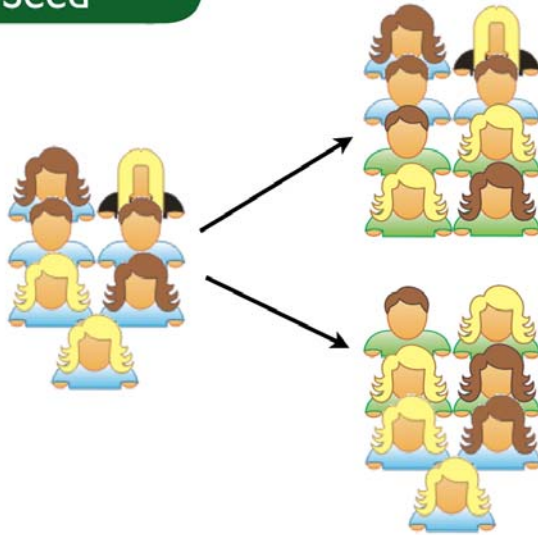
- Which of these techniques have you used in the past?
 - Was the transition successful?
 - If not, would a different pattern have helped?
- What advice would you give to someone about to use one of these patterns you've used in the past?
- What pattern would you prefer to use in the future? What conditions would you like to be true for you to use that pattern?



Expansion patterns

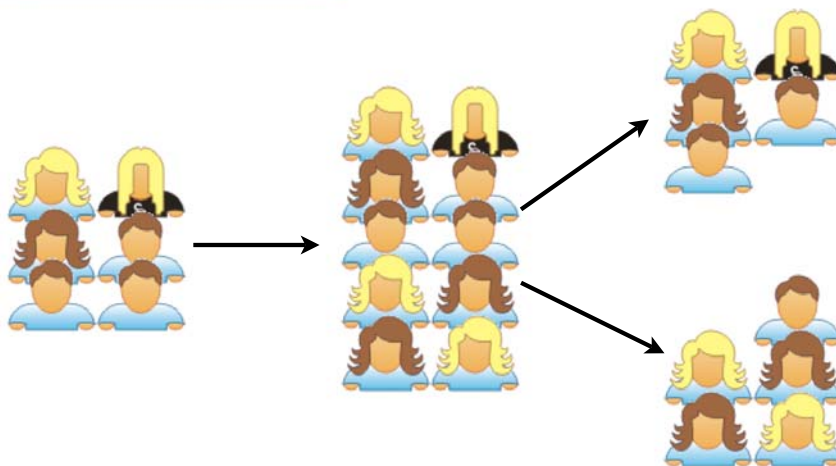
1

Split and Seed



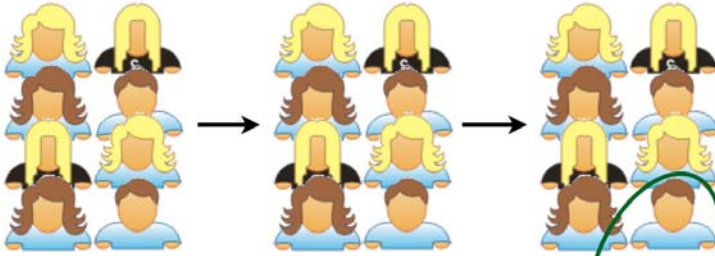
2

Grow and split



Internal coaching

3



Give coaches specific duties such as:

- Attend planning meeting
- Attend 2 daily scrums per week
- Spend 4 hours with the team per sprint

© Mountain Goat Software, LLC

55

What do you advise?

Read the following case study and recommend a course of action.



© Mountain Goat Software, LLC

56

An Ill-Timed Start?

The cold coffee on his desk did nothing to improve John's mood. As the vice president of product development, he knew that today was the day he'd have to make the call. Should his Cabo project team continue with Scrum or should they go back to their more sequential process? He'd already stalled a week since some of the Cabo team members came to him with their concerns. Maybe it had been a bad time to start the transition but he'd come back from the Certified ScrumMaster class so excited he couldn't wait. With version 7.0 "in the can" and entering testing, most of the team would be freed up to start work on 8.0. All of the testers and a handful of programmers would need to remain on 7.0 but everyone could get started on 8.0 using Scrum immediately.

SpiffyPricer version 7.0 had been in development for a little over a year, much longer than the company's traditional pace of a release per quarter. The development team seemed to be working as hard as before—harder in fact in many cases—but they just couldn't get products released as quickly as before. This was part of what drove John's decision to adopt Scrum. A dedicated and conscientious manager, John couldn't stand to see his team working more hours and getting less done.

A large enterprise-scale application used by retailers worldwide to monitor and set prices on all manner of goods, SpiffyPricer had done extremely well in the market during its short five-year existence. Over 50,000 licenses were currently in use and sales were continuing to boom. SpiffyPricer's 200-person team had been divided into nearly thirty Scrum teams shortly after John returned from CSM training.

"John," Tonya said, interrupting John's thoughts. "I got your email saying you wanted to talk this morning," she continued, her voice making it a question.

"Yeah, yeah, thanks. Come in."

Tonya, SpiffyTech's quality assurance director, had been with the company since the beginning. She wasn't a founder but she had been the sixth employee hired. John sometimes wondered why she continued to work at such a demanding job after she'd done so well in SpiffyTech's IPO a few years earlier.

"I'm worried about version 7."

"Me, too, John. But you know me—I always worry. That's why I'm in QA. Defect rates are higher than they were last release at this time. We're still finding about 35 bugs a day even after six weeks of testing."

"I know, Tonya. That's what has been hurting our initial sprints. I thought most of the team would be ready to move onto version 8 shortly after we started testing version 7 six weeks ago. But the programmers keep telling me they aren't able to make any progress on version 8 because all their time goes to bugfixing on version 7."

"We're going to try to speed things up. Many of the testers are coming in this weekend."

"I appreciate their dedication. I wish we didn't need them to do that, though."

"There's only seven weeks left in the schedule. We've all been anticipating this."

“Yeah, but, with Scrum we’re supposed to work at a ‘sustainable pace.’ I don’t think being in the office for the next seven Saturdays is all that sustainable. What I’m worried about is that we started Scrum too soon.”

“It’s only been three weeks. You’re not thinking of abandoning it, are you? You know I’m skeptical and wouldn’t mind dropping it.”

“Well, no, I don’t want to drop it exactly,” John said. “But I’m worried that maybe we adopted it at the wrong time. I had no idea we’d find so many bugs and that so many of the programmers would be needed to wrap up version 7. I thought it was mostly testing at this point.”

Tonya spent the next 20 minutes sharing graphs and trend reports on the defects her team had found. These did nothing to make John feel any better about the state of the 7.0 release, but he at least knew the release was in good hands with Tonya looking over it.

* * * * *

After a cold lunch at this desk, John walked over to Tyler’s office where he found an impromptu meeting going on. “Mind if I listen in,” he asked.

Tyler, one of the lead developers on SpiffyPricer, slid an Aeron chair from around his conference table. The four other programmers in the meeting nodded agreement and John joined the meeting.

“Rama found a bug in the point-of-sale interface. It’s nasty and we’re trying to figure out how to fix it without rewriting that whole interface,” Tyler said to John.

“I don’t see a way around, Tyler. Randy’s code is crap. We need to start over,” Kristy said, continuing the discussion that John’s appearance had interrupted. Randy had left the company a few months earlier, right before it was discovered that either much of his code was indeed crap or that he became a convenient scapegoat for many of the 7.0 delays.

“I still think the impact is isolated. The design holds water. I’m sure we don’t need to rewrite the whole thing.”

“You think that now. What if you get into, spend time trying to avoid rewriting, and then find you do need to rewrite,” Kristy continued. “That will take even longer. Let’s just do it now and not in four weeks. The system is stabilizing. The testers are finding fewer bugs than they were.”

“We might have time to rewrite if we didn’t spend time in those daily scrum meetings. That half hour a day adds up,” Shannon joked as all eyes turned cautiously toward John. The disdain for the daily scrum meetings was well-known, but everyone had also heard John’s message that stopping the meetings was not an option.

“Why are the meetings taking a half hour, Shannon? They’re only supposed to be fifteen minutes,” John asked.


“Oh they take at least a half hour. I get interrupted from what I’m doing. I go to the meeting and Tom, our ScrumMaster, has us give an update on 7.0 bugfixing. Each of us do that. Then we do an 8.0 daily meeting where we go around the room again. That part is quick because hardly any of us get time on a given day to work on 8.0. So we go around the room and each say ‘No progress.’ Then I need to walk back to my desk, get

my mind back into what I was doing before the meeting. It just adds up. If Tom wants to know where I'm at, he can look in Bugzilla. Everything is up to date—as soon as I fix a bug I mark it as fixed. No one asks any questions in those meetings. It's pretty clear they're just for Tom.”

“I think I'll watch your meeting tomorrow. Thanks for being honest.”

As the discussion of how to fix the point-of-sale interface bugs continued John found he was no longer paying attention to the meeting. He knew Scrum was the right direction for the company. In the three weeks since they'd begun he'd already noticed some encouraging changes. So while he knew that Scrum was the right thing to do, he didn't know if it was the right thing to do *now*. What he did know he had to do now was meet with Carlos, the CEO, and let him know what he'd decided.

What should John do in this situation?



Overcoming Resistance

© Mountain Goat Software, LLC

57

Overcoming resistance

- Sell the problem, not the solution
 - No one wants a solution to a problem they don't (think they) have
 - Be open to hearing better solutions than you have
- Communicate why the change and why now
- Put team members in touch with customers
 - Let them hear the problems you are hearing
- Emphasize benefits of the change
- Help resisters find new roles

© Mountain Goat Software, LLC

58

Engage the change agents

Change agents...

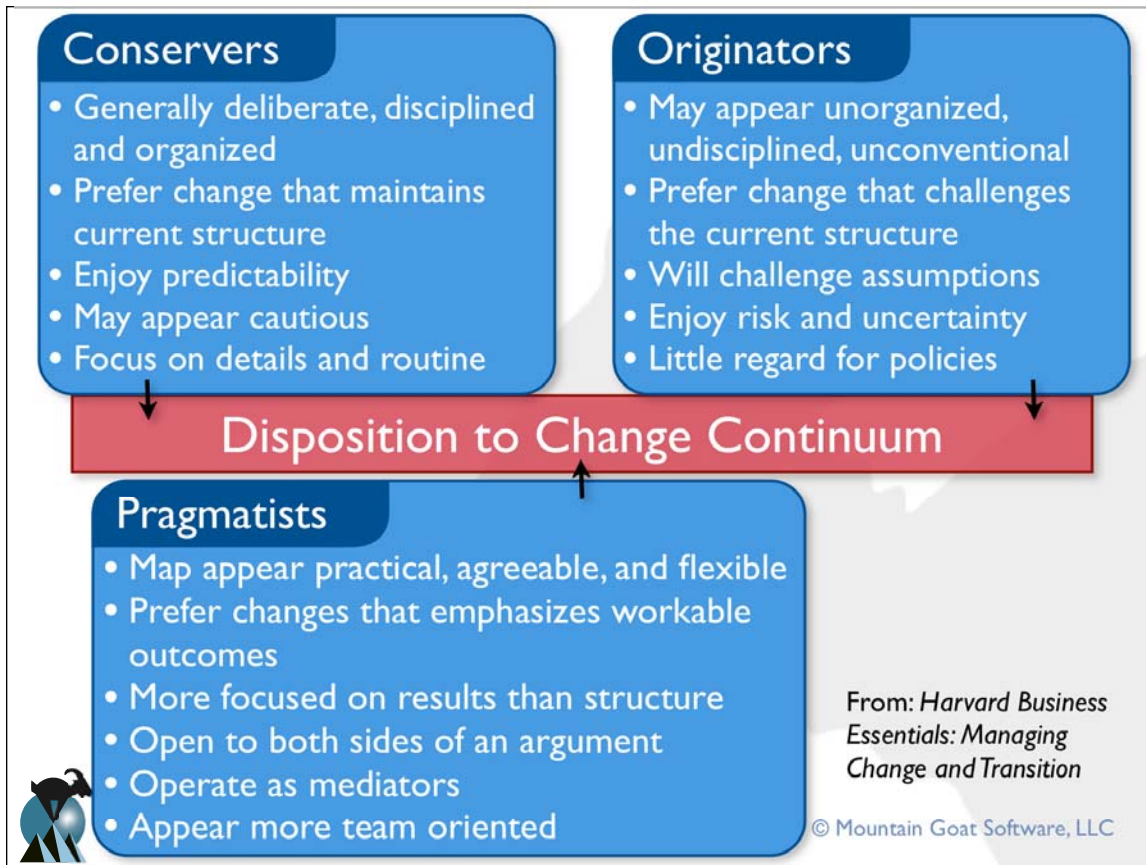
- help others see problems and address them
- articulate the need for a change
- are accepted as trustworthy and competent
- can see and diagnose problems
- motivate people to change
- work through others to translate intent into action



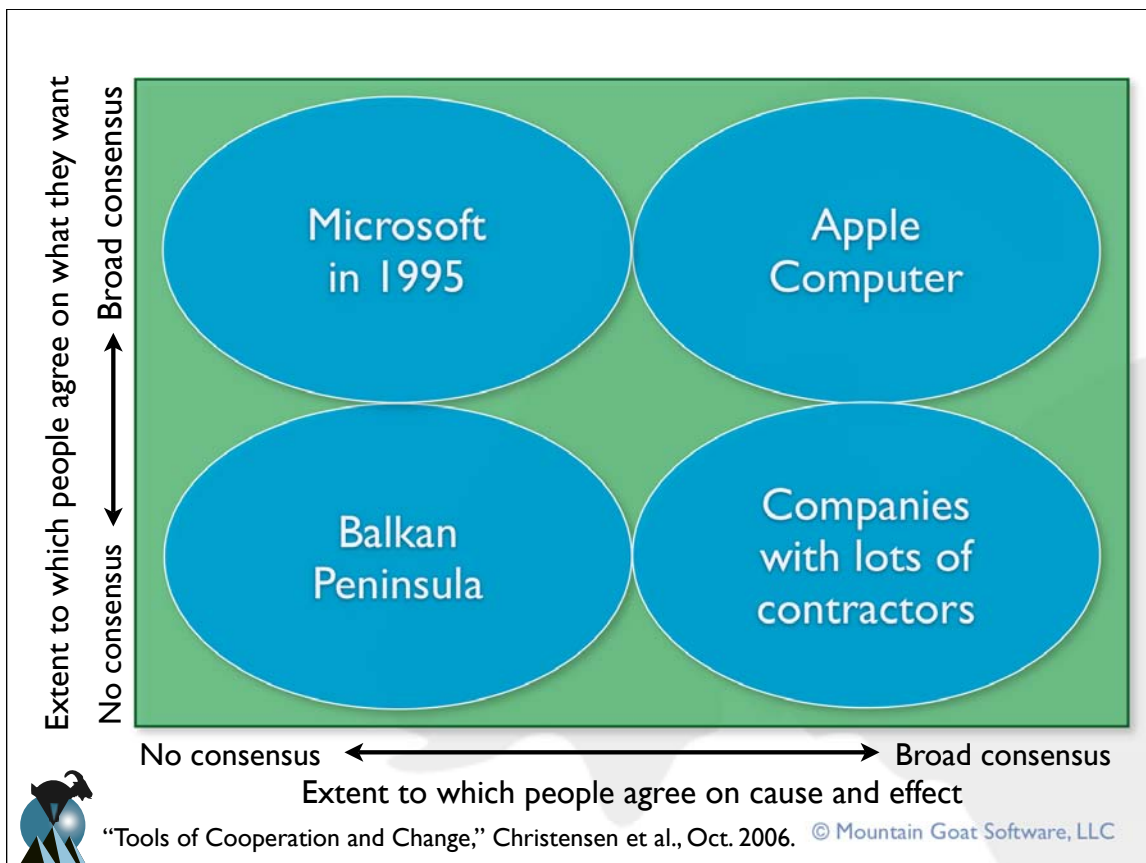
Identifying change agents

- Find out who people listen to
 - These may not be people with formal authority
- Look for people who think differently
 - Change agents aren't satisfied with the status quo
- Consider new employees or others who may not be infected with a common mindset yet
- Consider people with different backgrounds
 - The programmer with the art history degree

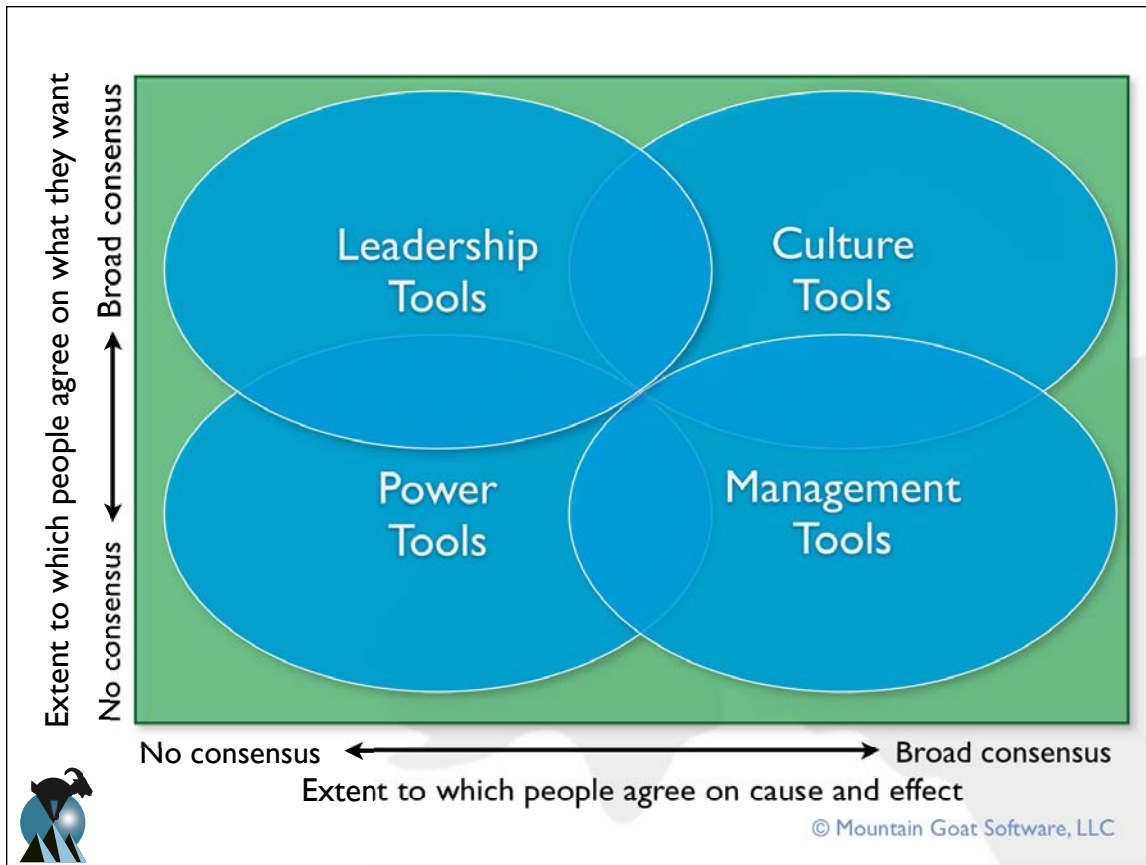




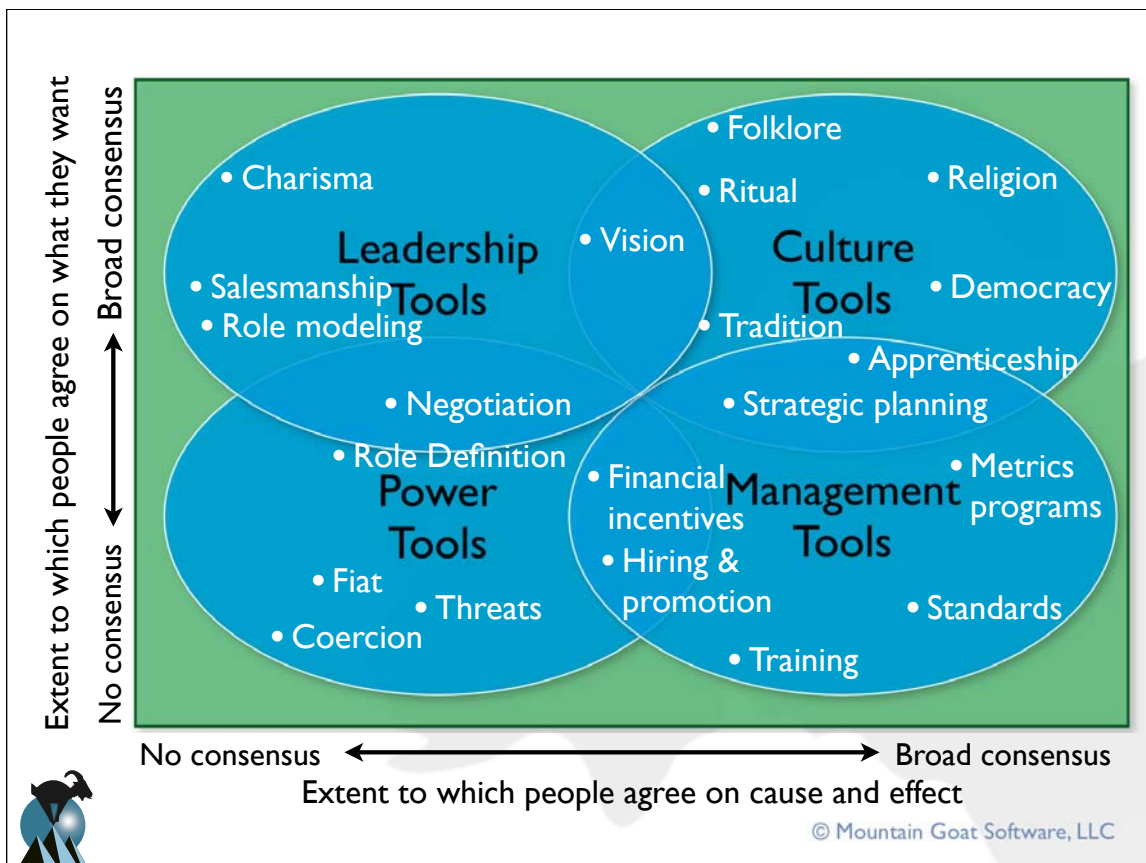
61



62



63



64

Upcoming public classes

Date	What	Where
Jan 15-16 Jan 17	Certified ScrumMaster Agile Estimating and Planning	Atlanta
Feb 24-25 Feb 26	Certified ScrumMaster Agile Estimating and Planning	Seattle
April 8-9 April 10	Certified ScrumMaster Agile Estimating and Planning	Dallas
June 3-4 June 5	Certified ScrumMaster Agile Estimating and Planning	Washington, DC (Reston)
Other classes in London, Oslo and Stockholm if you're up for a longer trip.		

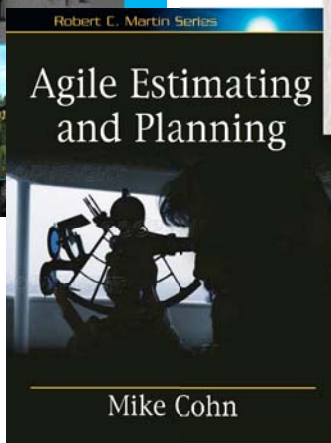
Information and registration at
www.mountaingoatsoftware.com



© Mountain Goat Software, LLC

65

Mike Cohn contact info



mike@mountaingoatsoftware.com

www.mountaingoatsoftware.com

(720) 890-6110 (office)

(303) 810-2190 (mobile)



© Mountain Goat Software, LLC



66