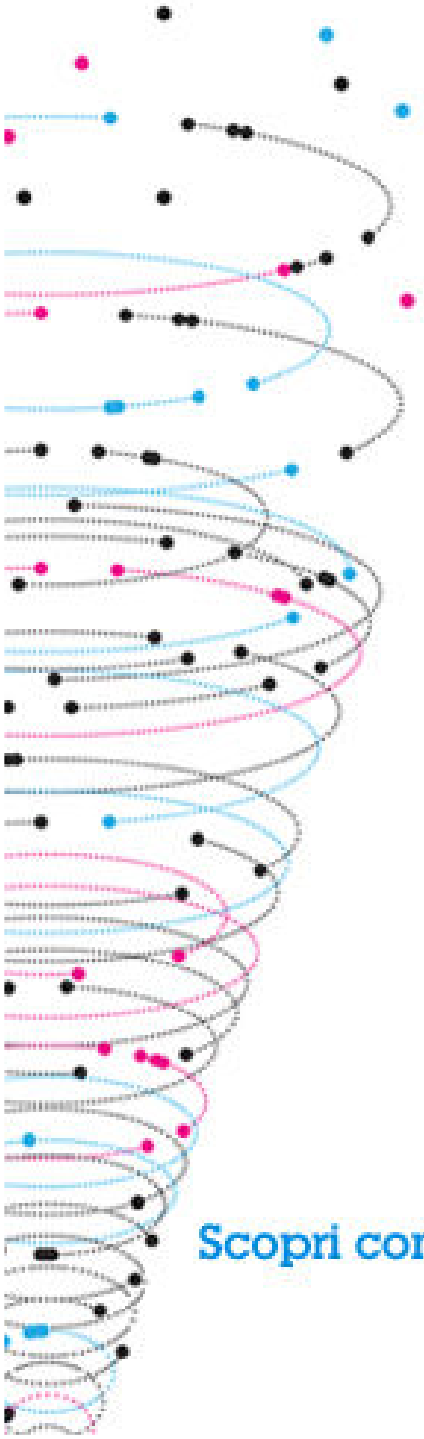


Achieving Agility at Scale

Alan W. Brown
IBM Rational CTO for Europe
alanbrown@es.ibm.com

Scopri come far lavorare in perfetta sinergia i tuoi team e i tuoi progetti.

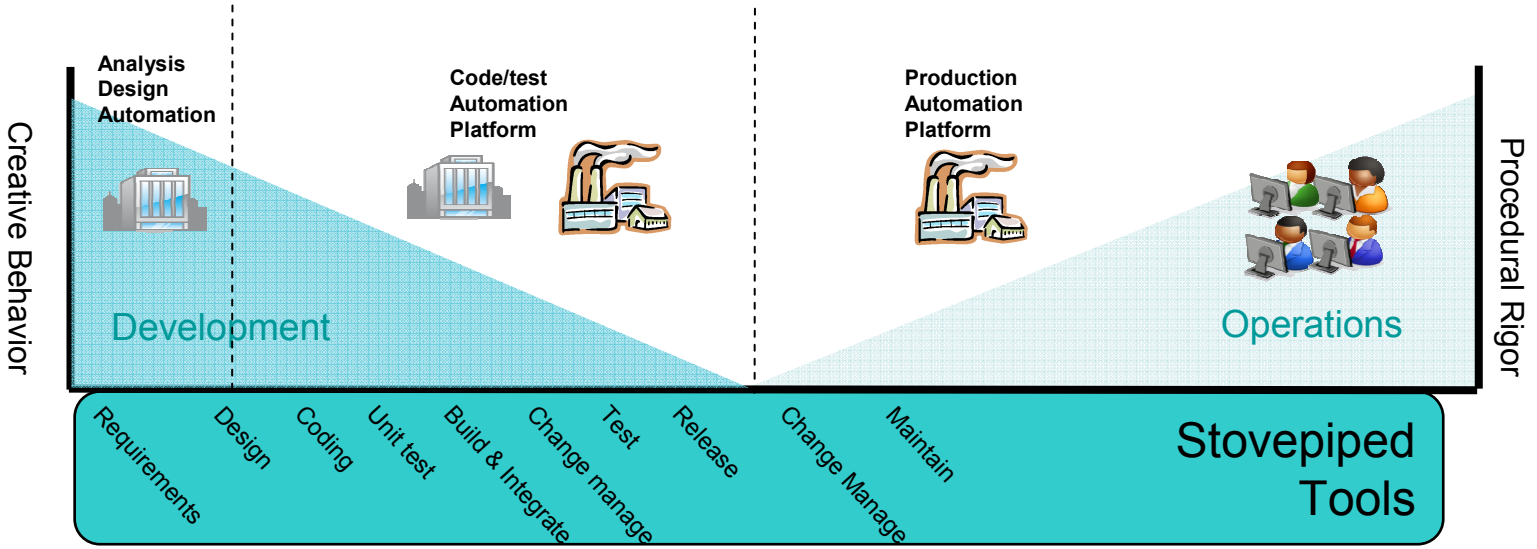
Rational. software

Topics

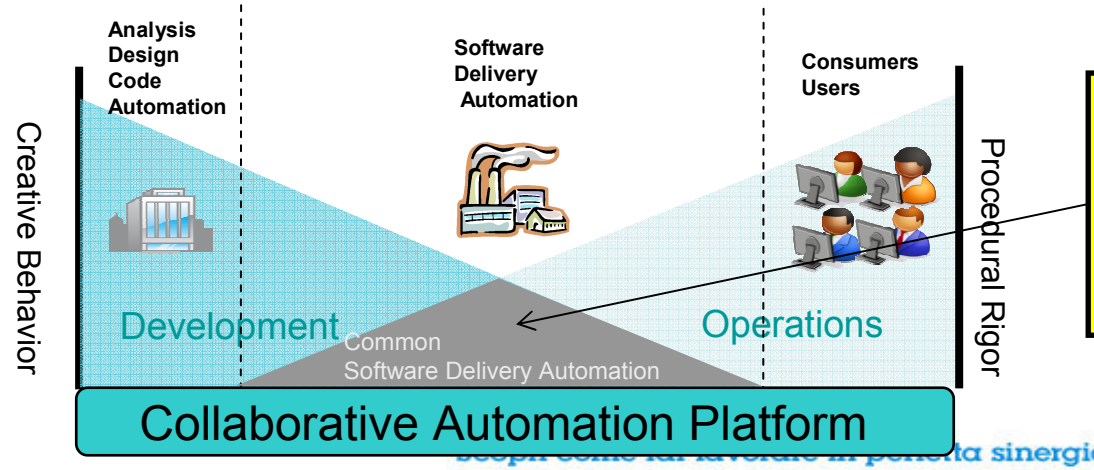
- **Summary**
- **Becoming agile**
- **Thinking agile**
- **Staying agile**
- **Where to begin…**

Rethinking Software Delivery

Waterfall Process Platform



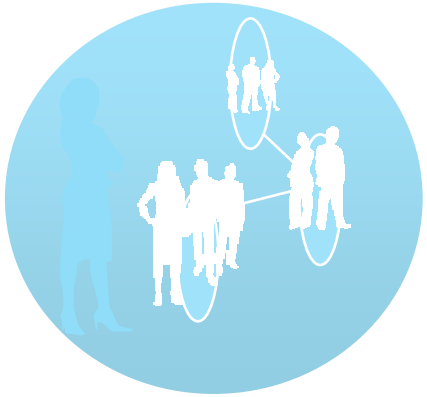
Agile Delivery Process Platform



Implications

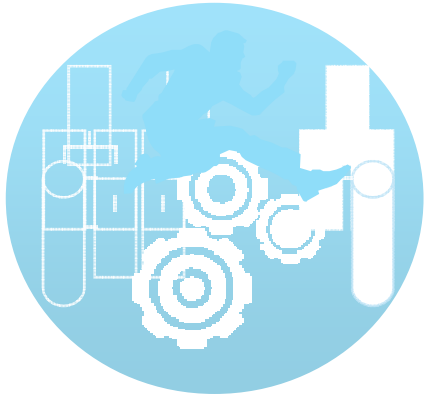
Automation
Measurement
Close Customer Relationship
Project visibility

The 3 Key Areas for Effective Agile Software Delivery



Collaboration

Drive organizational consensus on priorities and improve workforce productivity



Automation

Lower costs and improve quality by automating workflows based on real-time information

Visibility

Continuously improve by measuring progress against desired business outcomes

How Do You Scale Agile Software Delivery?

- Focus on the key Agile practices
  - Match them to your organization, people, maturity, projects, culture……etc…
- Reinforce the delivery practices that support your teams
  - Find out what works….grow the skills and practices
- Change the delivery rhythm, and make it more transparent
  - Push teams to work in shorter cycles with greater feedback and input
- Support practices with automated tooling
  - Help overcome the collaboration and integration issues for larger, distributed teams….make them part of the daily work habits
- Measure and report to get management buy-in and support
  - Clearly align technology improvements to business goals, and demonstrate the value to the business

Summary

- Collaborative life-cycle management is the key set of practices and technologies that unify your organization
- Achieving agility at scale requires new ways of thinking, acting, and sustained transformation
- Rational´s Jazz platform
  - A unified platform that includes collaboration, automation and reporting can dramatically improve the business process of software delivery
  - Embracing open integration strategies, enables IBM and its partners to leverage and develop best-of-breed solutions
  - Achieving business differentiation with agility and confidence is a reality today!

Topics

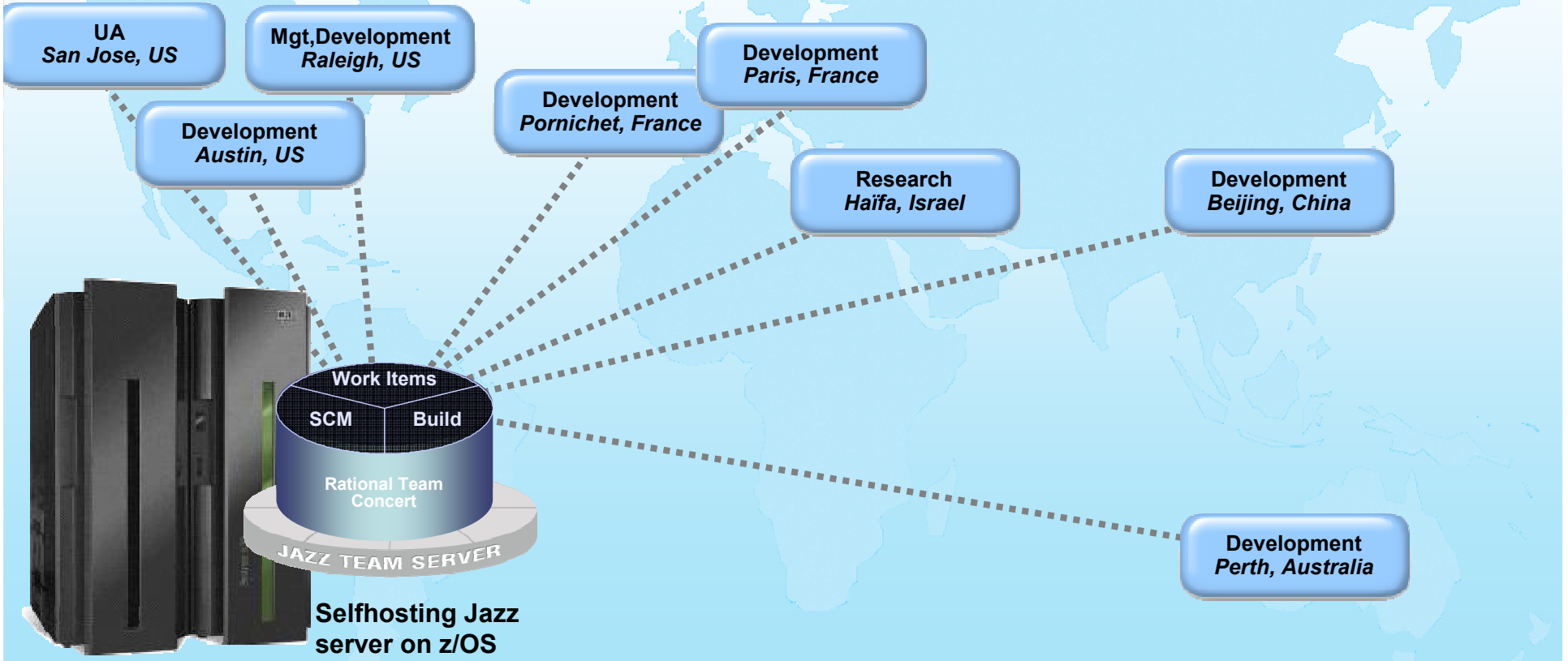
- Introduction
- **Becoming agile**
- Thinking agile
- Staying agile
- Where to begin…

Agile in Context

- There is a difference between:
  - Scaling agile practices in collaborative teams
  - Introducing agile practices to scaled collaborative teams
  - Providing agile collaboration to scaled teams
- What are the practical implications?

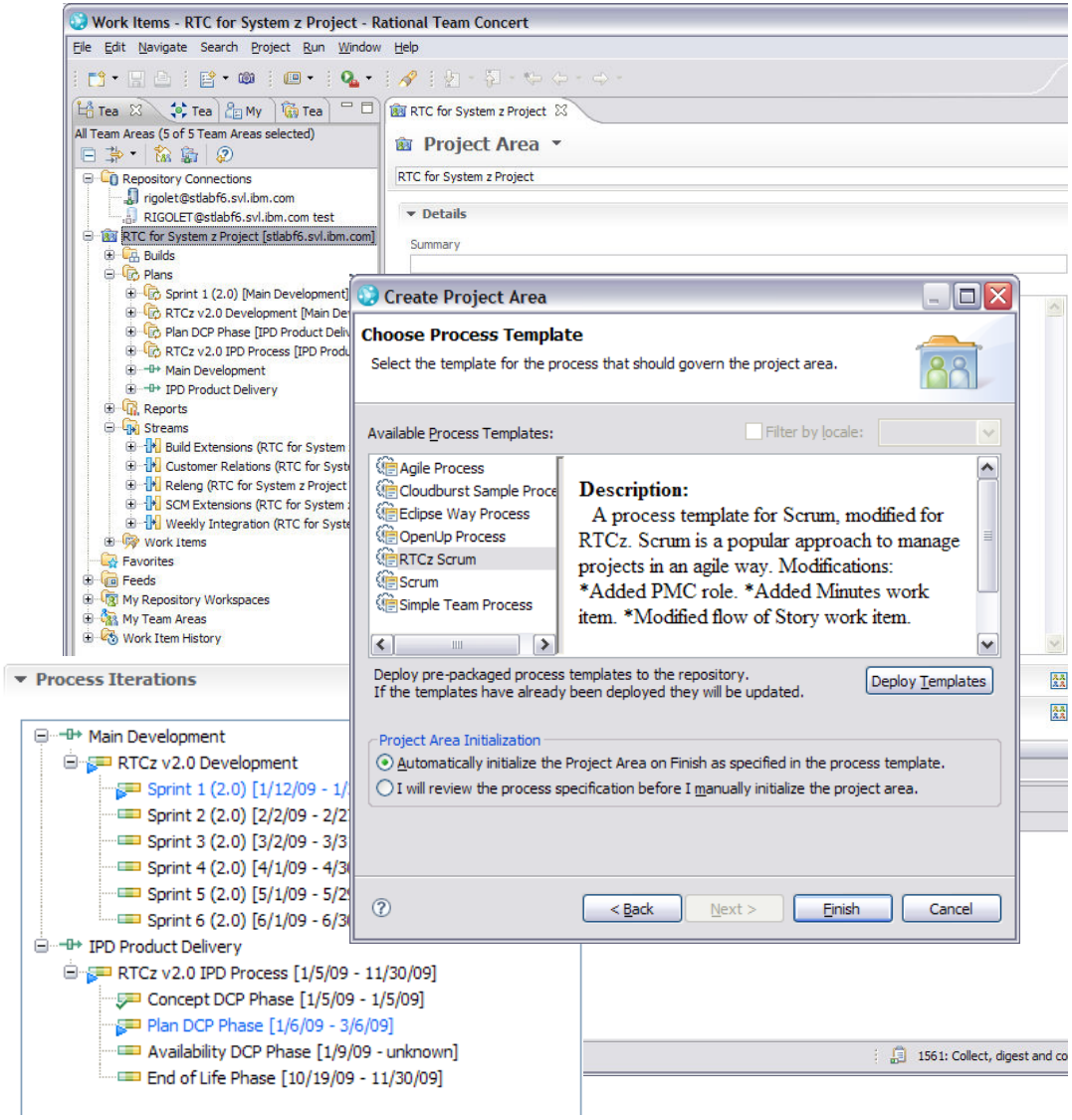
Example 1: Distributed Agile Team

Approx 25 people

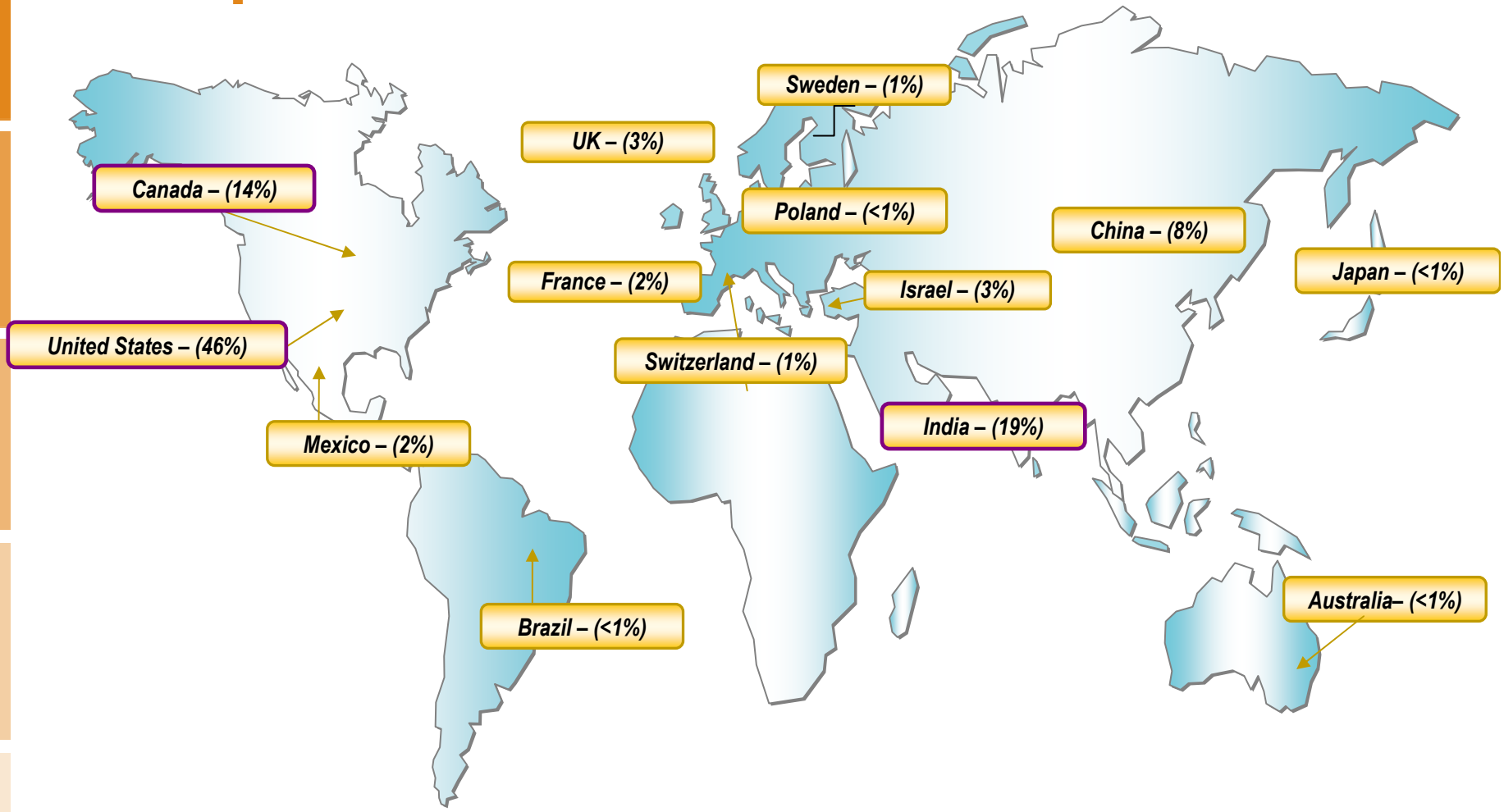


Successful Delivery of Agile Solutions

- **RTCz development project**
  - Selfhosted on *System z*
    - *Access from Jazz.net*
  - 'RTCz for System z Project'
  - Based on the Scrum template
- **Geographically Distributed Development**
  - 3 main Scrum teams
    - RTP (Raleigh, US)
    - FASL (France & Australia)
    - BF (Austin, US)
- **2 parallel development lines**
  - Main development
    - Release v2.0
    - Post v2 development
  - IPD Product Delivery



Example 2: IBM Rational Core Dev. Team



Approx. 2000 people worldwide
Data from 2010

Successful Delivery of Agile Solutions

IBM

Executive Dashboard

Development Health

- Build Health
- Project Velocity
- Staffing Variance
- Process Timeliness
- Iteration/Milestone Status
- Severity Analysis
- Security Vulnerabilities
- Static Code Analysis
- Requirements Met
- IPD Timeliness

Perceived Quality

- Transactional Survey
- PMR / Call Rates
- Critical Situations
- Cost of Support
- Installability
- RFE SLAs
- Usability
- Consumability
- Scalability
- Integrations with other products
- User Experience / Doc
- Time to Resolution
- APAR:PMR ratio
- PostGA metrics
- Transparency

Development Quality

- Defect Backlog
- Test Escapes
- Functional Test Trends
- Critical Situations
- System Test Trends
- S-Curve Progress
- Automation Percentage
- Customer Testcases
- Consumability Scorecard
- Defect Latency
- Quality Plan Commitments
- Test Coverage
- Defect Density

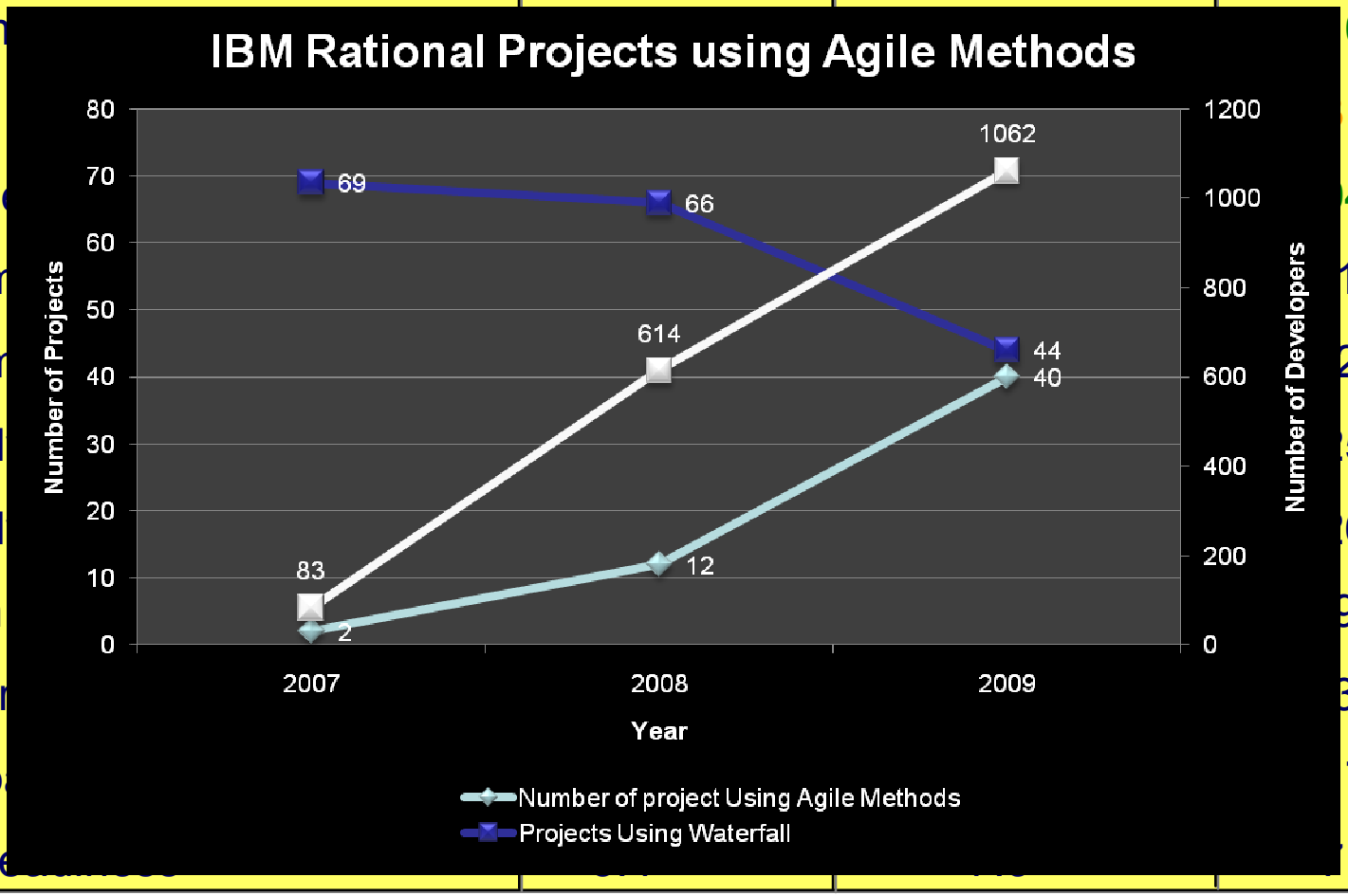
Business Health

- Sales Plays
- Partner Enablement
- Support Enablement
- Technical Enablement
- Sales Enablement
- MCIF Index
- Alt Packaging
- OEMs
- XL hits
- Tactics
- ROI
- Pipeline / Multiplier
- Revenue

Practices

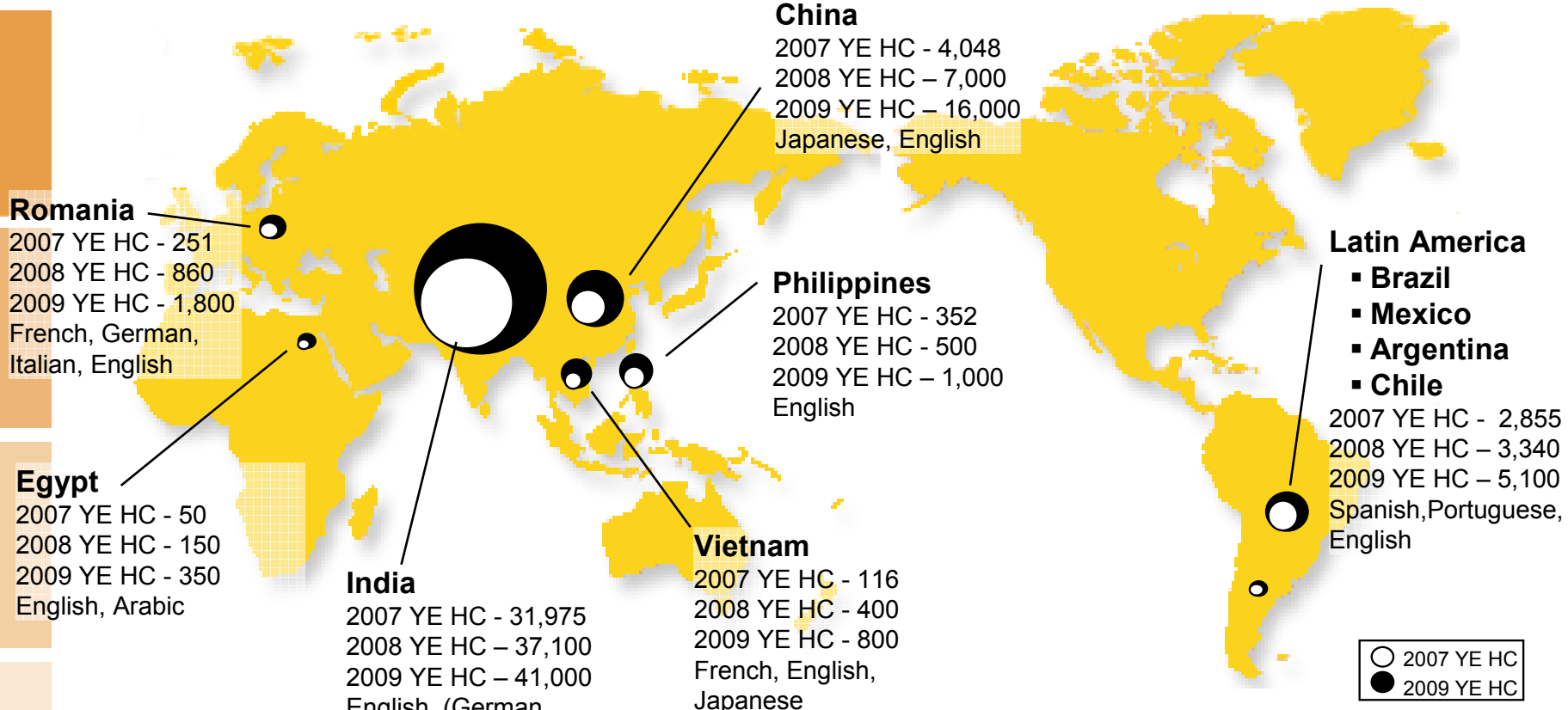
Evolutionary Architecture
Vulnerability Assessment
Concurrent Testing
Requirements Management
Whole Team
Test Driven Development
Team Change Management

Metric	2006 Measurement	2008 Measurement	2009 Measurement
On Tim			00%
Defect			.5 months
Beta D			4%
Custom			16%
Custom			20%
Lab Ad			55
Lab Ad			69
Design			91
Beta Pr			33
Transp			7
Ship Re			.6



Example 3: Global Software Integrator

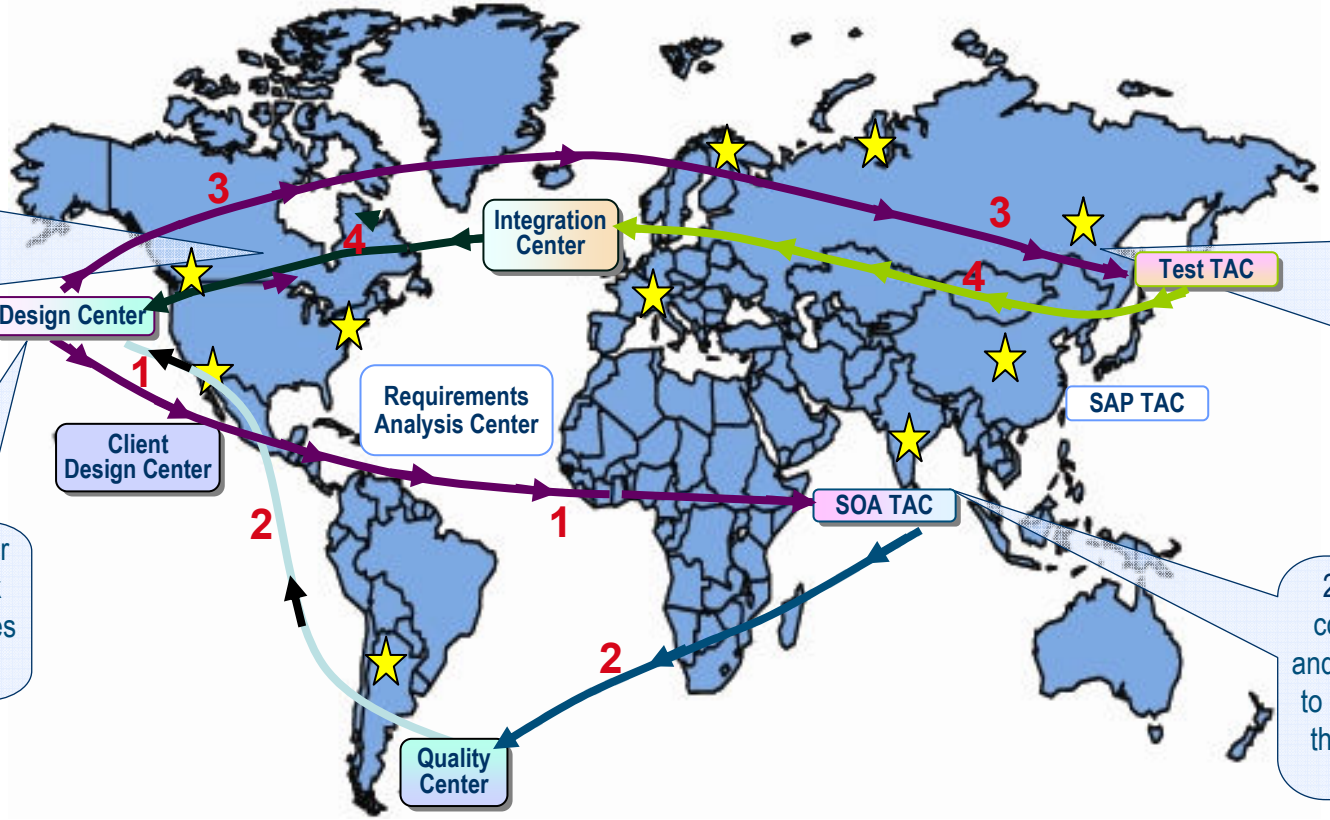
Global Delivery Supply Staffing Plan



Software Factory – Virtual Application Optimization Services Environment

- *Design and technology assembly centers are integrated through a standardized work request/response mechanism. Each center is a delivery team providing specialized services in one or more capability areas*
- *Work packets enable mobility of work while capturing complete, consistent and reusable instructions for successfully delivering high-value solutions with lower cost and risk*

1. The Design Center packages necessary artifacts into a Solution Design Work Packet and dispatches it to the SOA TAC



4. The solutions are developed, tested, quality checked and the appropriate work packet reflecting this is sent back to the Design Center

3. The Design Center creates a Test Work Packet and dispatches it to the Test TAC

2. The SOA TAC completes its work and dispatches it back to the Design Center through the Quality Center

TAC = Technology Assembly Center

Case Study: A Global Enterprise Focused on Improving Productivity and Efficiency

- Collaboration across Global Delivery Teams
  - Multiple suppliers, multiple geographies, multiple business units
- Reduce Waste and Optimize Resources and Assets
  - Aligned practices to provide a consistent and integrated development approach with standardized tooling across the organization
- Optimized Reuse of Core Assets and Practices
  - Catalog, categorize, and assess the value of current asset inventory to make it more accessible across the organization
  - Asset categories from development, delivery, and deployment
- Business Cost Management Focus
  - Greater cost transparency redefine expense ratios
  - Move toward virtualized and cloud-based infrastructure
  - Continual monitoring of project health across the portfolio of projects, and across a wide variety of tools and practices

IBM Case Study : An Evaluation of Potential ALM Savings

- Worldwide AD project chosen for evaluation
  - c. 2,800 Man Day AD project which is part of broader programme
  - IBM managed and resourced programme based on T&M
  - Programme assets developed that have and can be re-used
    - Includes : code, components, documentation, security
  - *80% of resource effort on project has been offshore*
    - Handpicked offshore team based on skill-sets required
    - Offshore PM's and technical leads landed during design phase
    - Regular 'High-touch' visits by core team and customer
    - Expectation management, specific instructions and follow-up key
    - Online collaboration tools critical (IM, Live meetings)

ALM will make this more of the norm based on industrial tools & processes vs. the exception based on the talents, effort and visibility of a single team

- Detailed ex-Post analysis of effort across project phases established a further 15% cost saving potential based on use of ALM tools and process
  - Development > 25% productivity savings (resources mostly offshore)
  - Test > 25% productivity savings (resources mostly offshore)

Representative productivity saving for rest of programme

IBM Case Study : De-risking More Aggressive Off-shoring of project

Same 2800 Man Day Project

De-risking on/offshore Resource Mix

On:Off Ratio	Avg/Cost Day
20:80	Baseline Cost
30:70	+11%
40:60	+49%

More Aggressive on/offshore Resource Mix

On:Off Ratio	Avg/Cost Day
20:80	- 33%
30:70	- 26%
40:60	Alt. Baseline Cost

ALM will de-risk more aggressive off-shoring through enhanced :

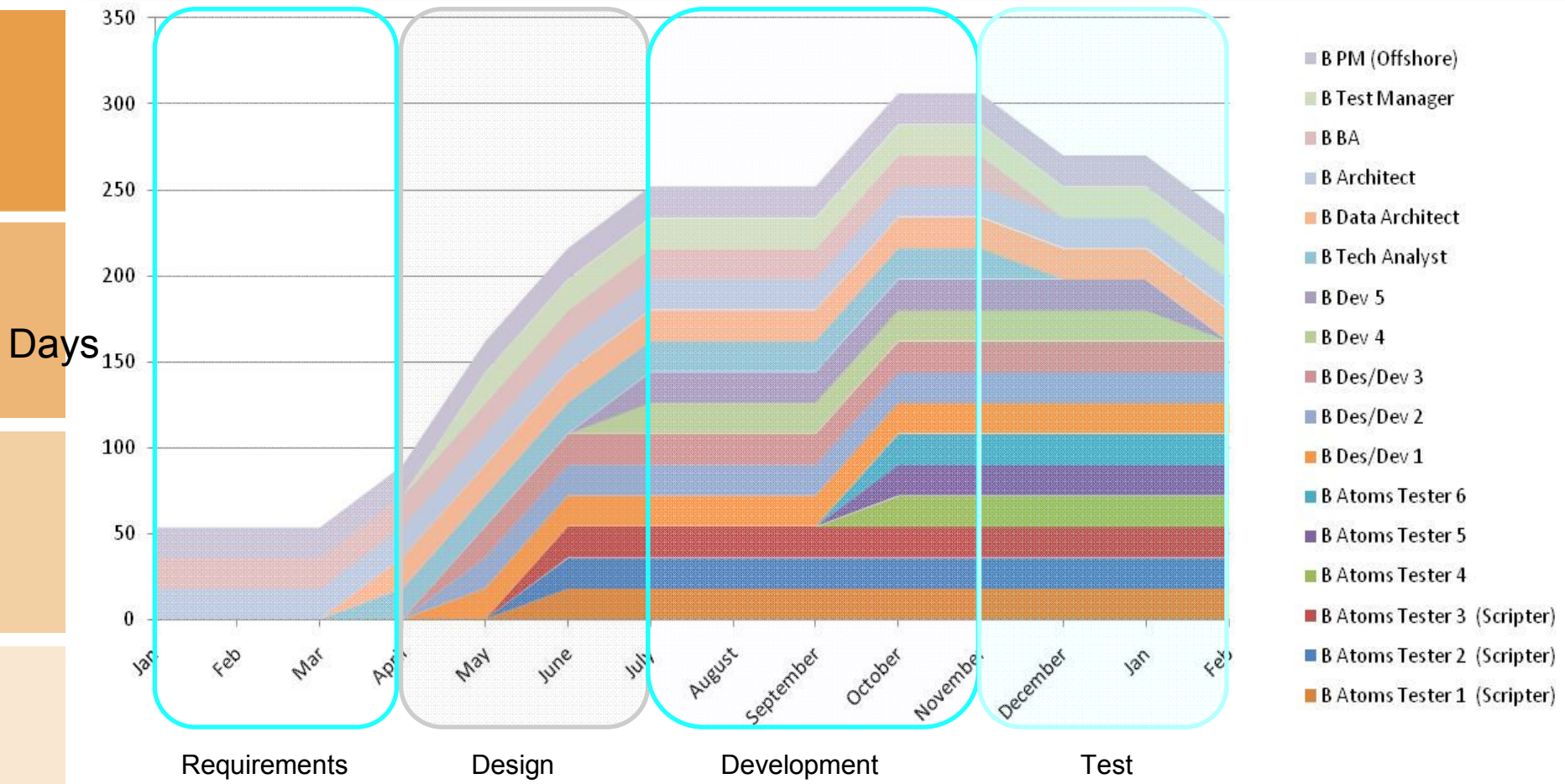
traceability,
componentisation
collaboration, and
governance

…..based on integrated workflow and performance management metrics

AGILE development

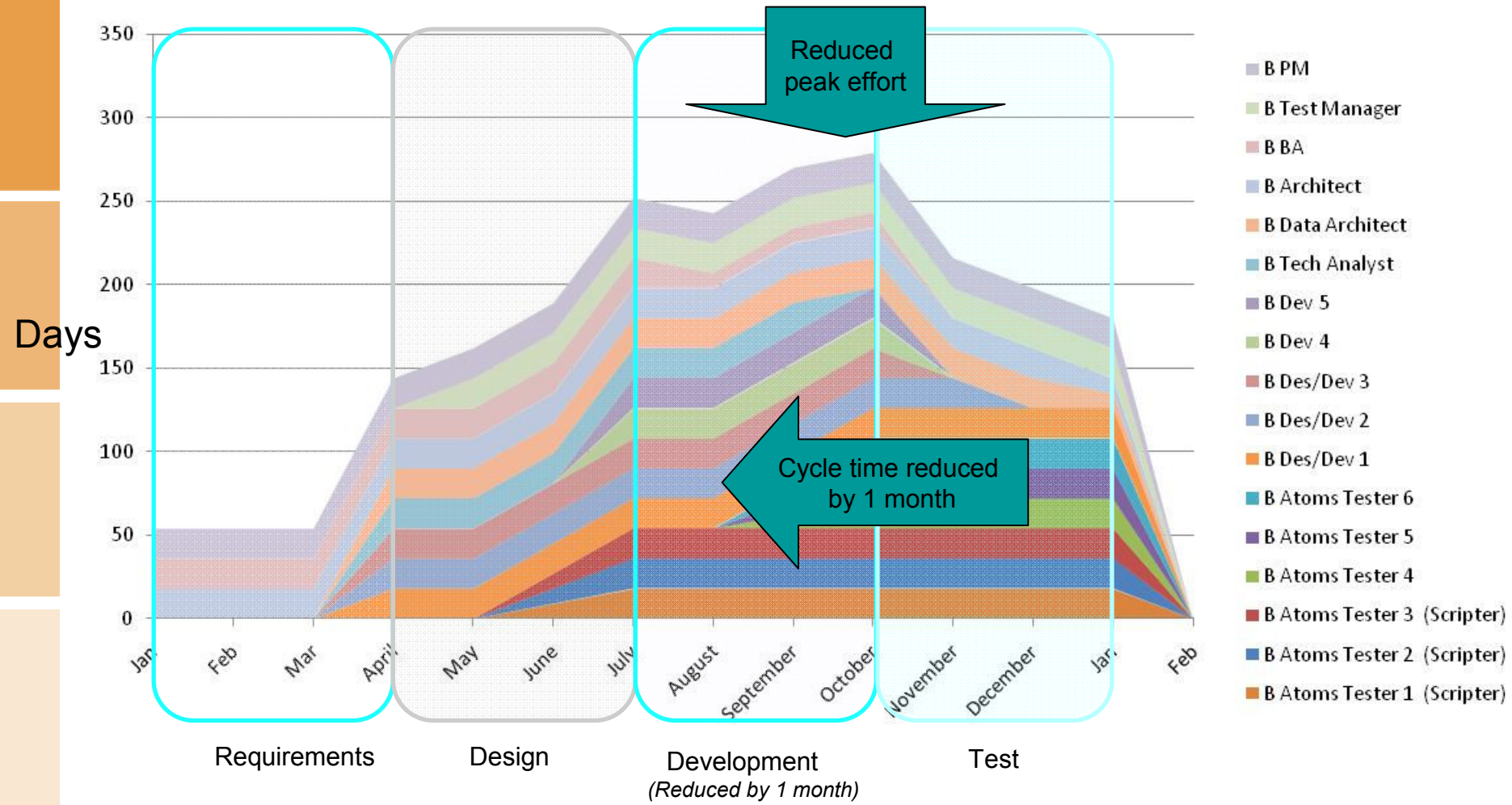
IBM Case Study : Baseline Man Days Billed for Project

Staffing of project has been an average of 20:80 onshore/offshore



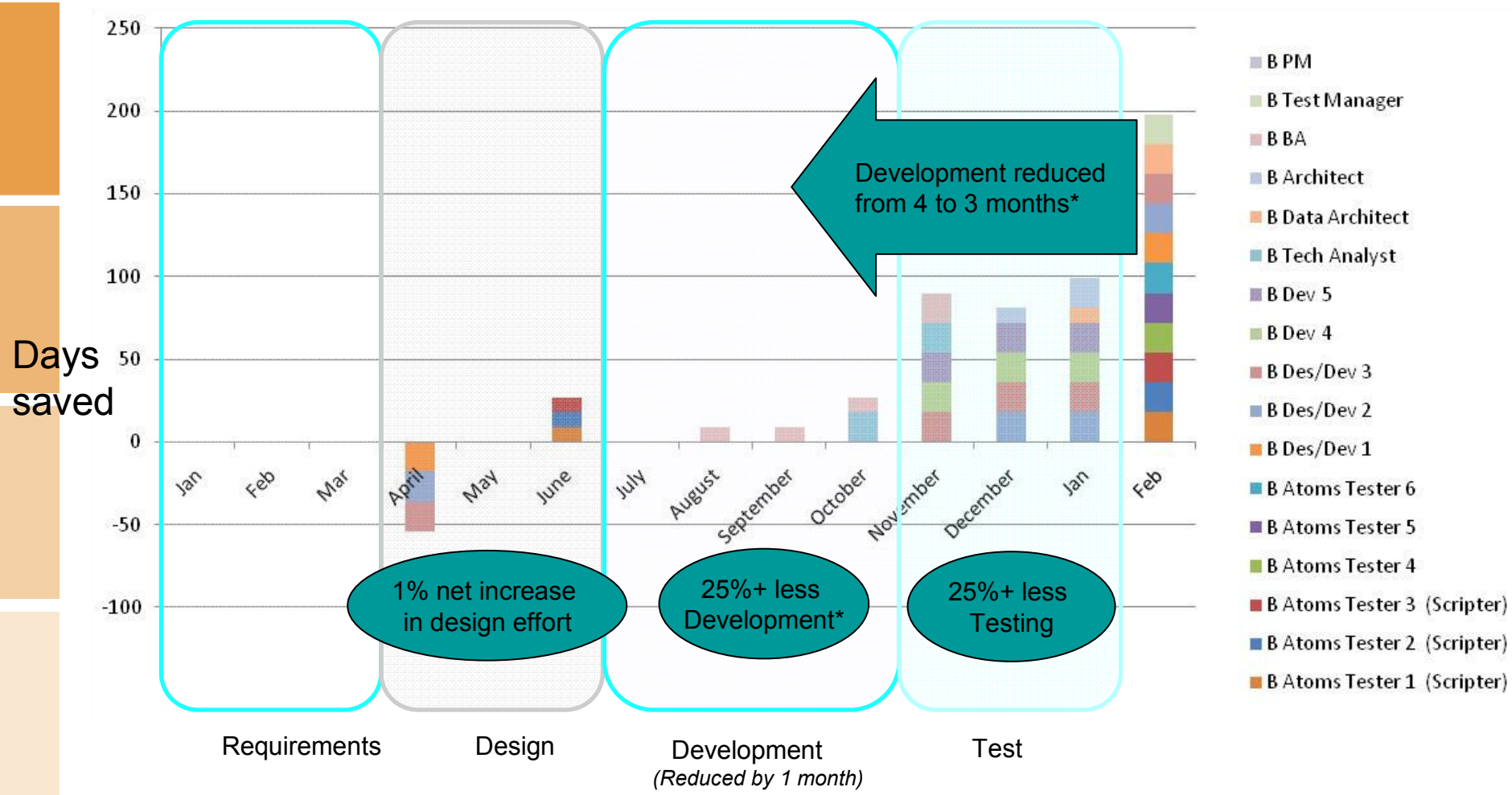
IBM Case Study : Man Days Projected with ALM Discipline (ex-Post)

FTE Savings - Major reduction in Development and Test effort expected due to enhanced definition and tracking of requirements and 'decoupled' test cycles



IBM Case Study : ALM Productivity Savings Between Project Without vs. With ALM Discipline

Estimated total Project Days saved = 18% *(*excludes any asset reuse)*



IBM Case Study: Source of Productivity Savings

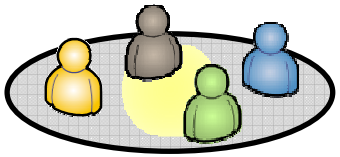
- **Quality Management**
  - Reporting / Quality assessment supported by tooling increased productivity of onshore management team.
- **Data Management**
  - ALM Tooling enables data analysis and modelling, increasing quality of data used to test, reducing development and test timelines.
- **Requirements Traceability**
  - Design and code development from requirements reduces design gaps and misunderstanding
  - Significant time saved in Development from not having to query requirements for unclear design.
  - Reduced critical and major defects in test as build is more focused at requirements and design.
  - CR's more easily scoped for estimated impact when considering impacted existing requirements, design, test scripts.
- **End to End Environment Management**
  - Faster environment procurement
  - Predefined developer profiles – resources effective immediately
- **On boarding from Dev/Test factory**
  - Faster on-boarding
  - Guaranteed skill sets

******Not considered but could equally save more in productivity***

- *Jump start design using template blue prints and other assets*
- *Jumpstart teams using blue print software components e.g. security component*
- *Reduced risk allows more aggressive offshore model*

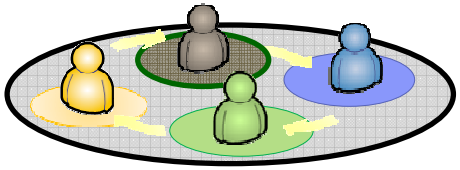
Three Common Solution Patterns

Vertically aligned
> Centralized ALMaaS



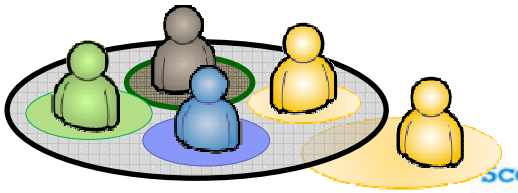
- **Integrated team with collaborative, transparent and automated workflows**

Divided by Function
> Integrated ALM Cloud



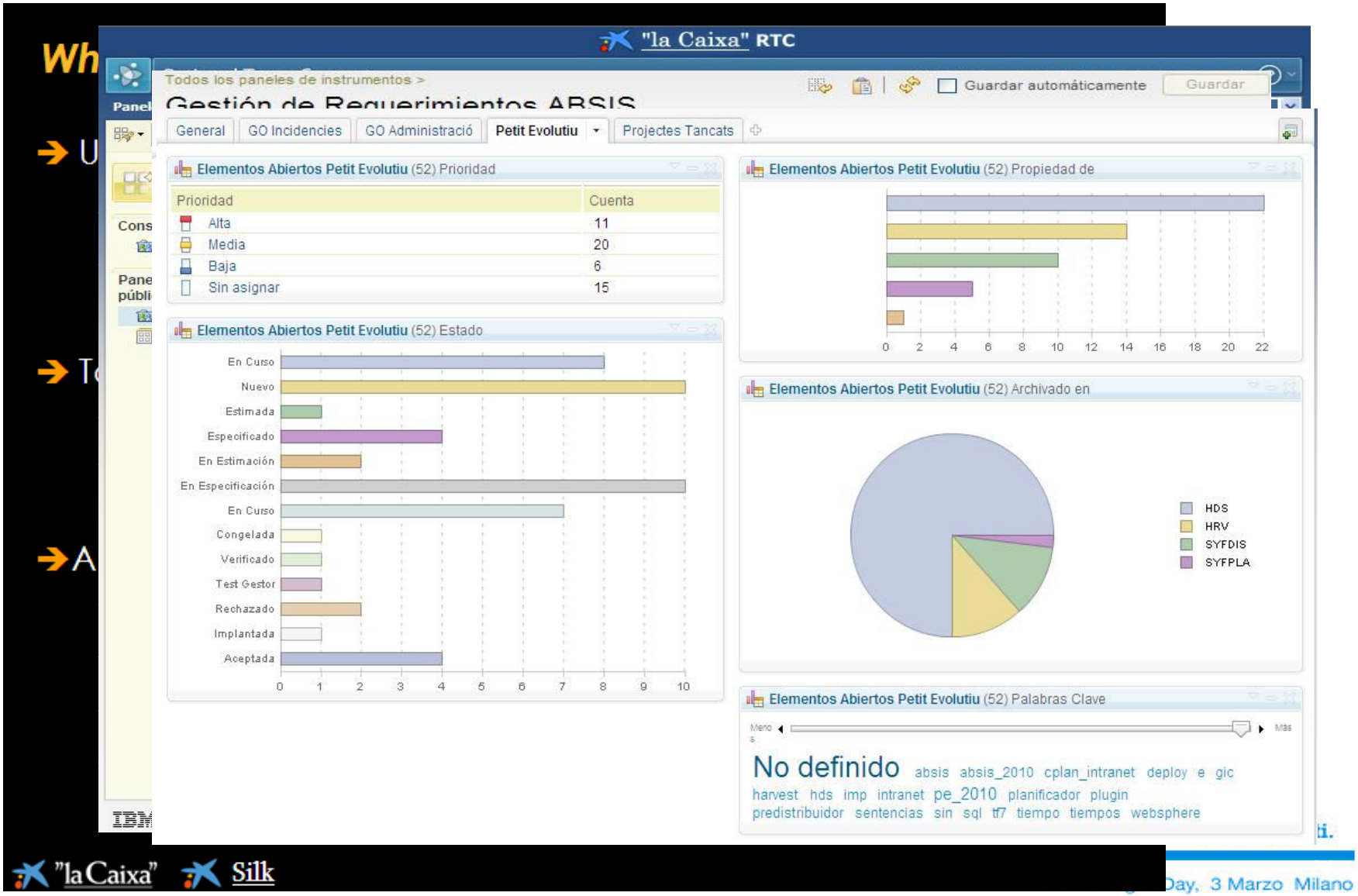
- **Functional silos, organized by discipline and line organization, form software delivery chain**

Outsourced
> Secure and Connected



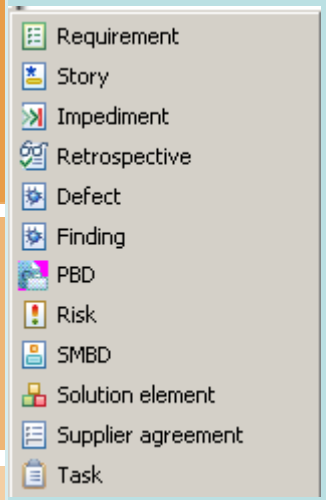
- Organizations depending on functions and **contributors outside corporate boundaries, while preserving IP security**

Example: Managing Service Providers at La Caixa

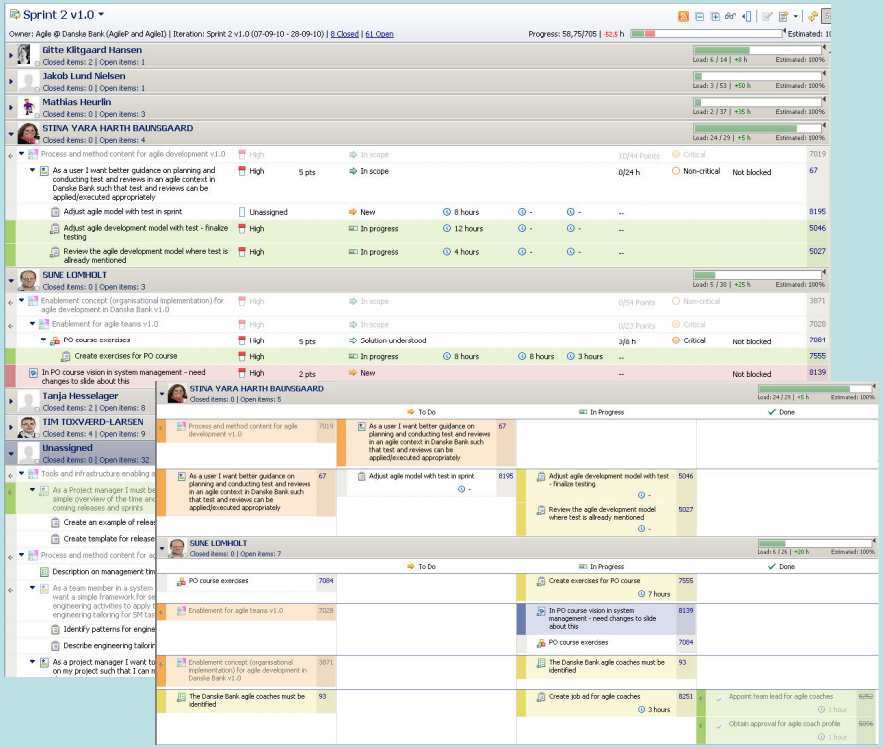


Example: Agile at scale adoption at Danske Bank

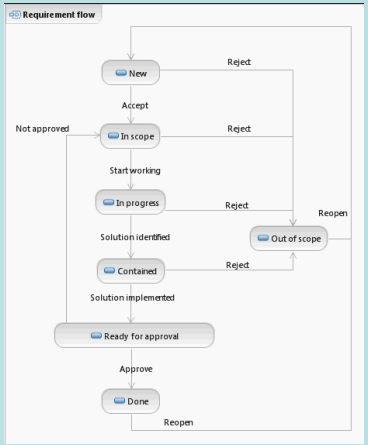
Work item types

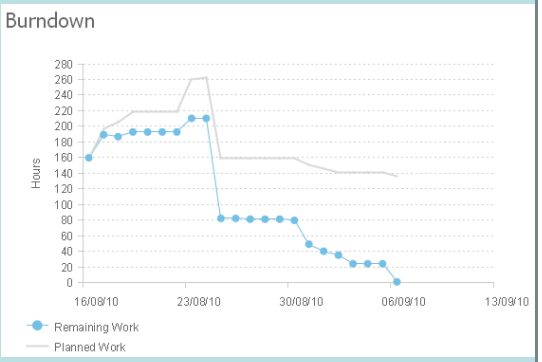


Plan layout



Timelines, workflows and permissions





- Dashboards and reports
- Integration with HP Quality Center
- Integration with existing task management system

From: Robert Baillargeon, SW Engineeing Manager, Panasonic Automotive Systems

Example: Visibility and Transparency at Panasonic Automotive

Results of using RTC

- Changing Status Meetings
  - Focus on the status to plan (rather than what is done)
  - Results are actions to stay on plan
- Changing the Management-Developer Relationship
  - Potential for Support (without meddling)
  - Focused discussions on the meaningful... specific issues and challenges
- Clear (and Transparent) Management to the Imperatives
  - On-Time
  - On-Budget
  - With Quality
- We know where every project is on a daily basis
  - And we have the evidence to back it up!
- Developer adoption is increasing over time
  - More developers are engaging beyond simply their own work
  - Not seeing the drop-off in usage that is often observed with new tools

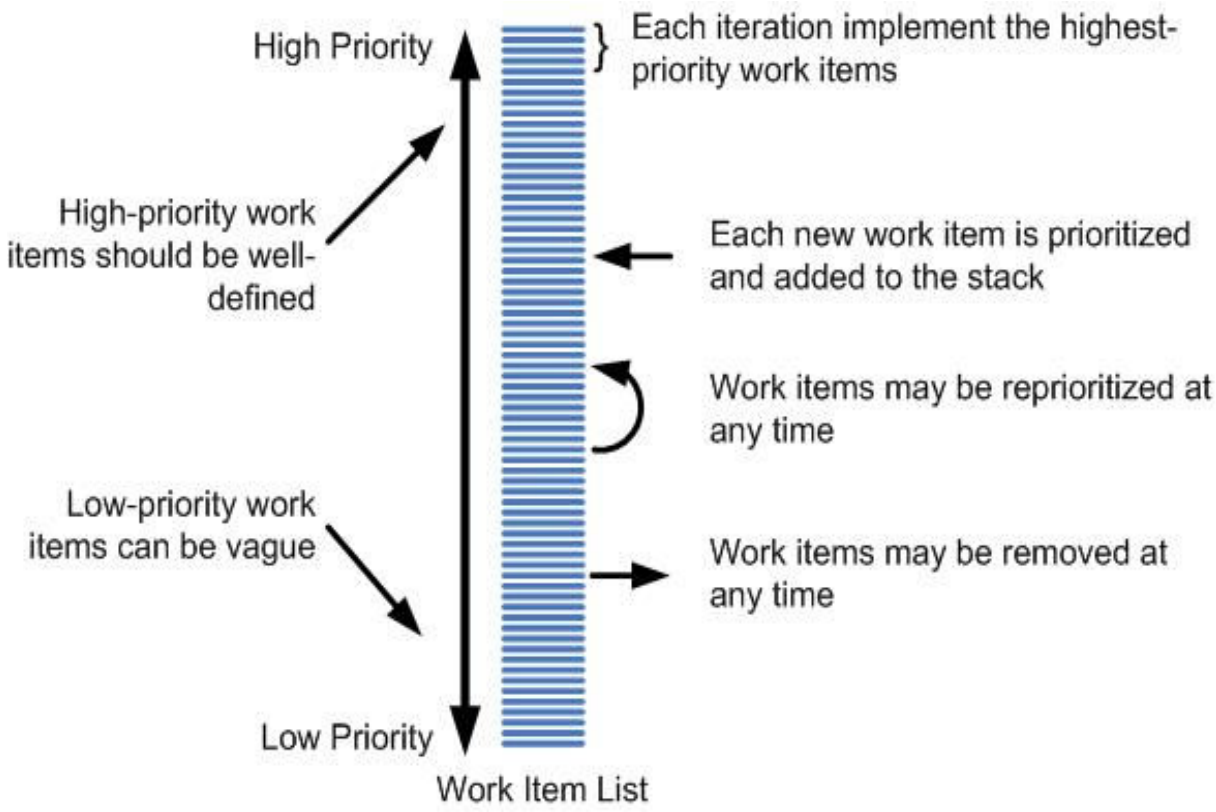
RTC doesn't Manage Projects, but it does provide the information to support managing projects better

Topics

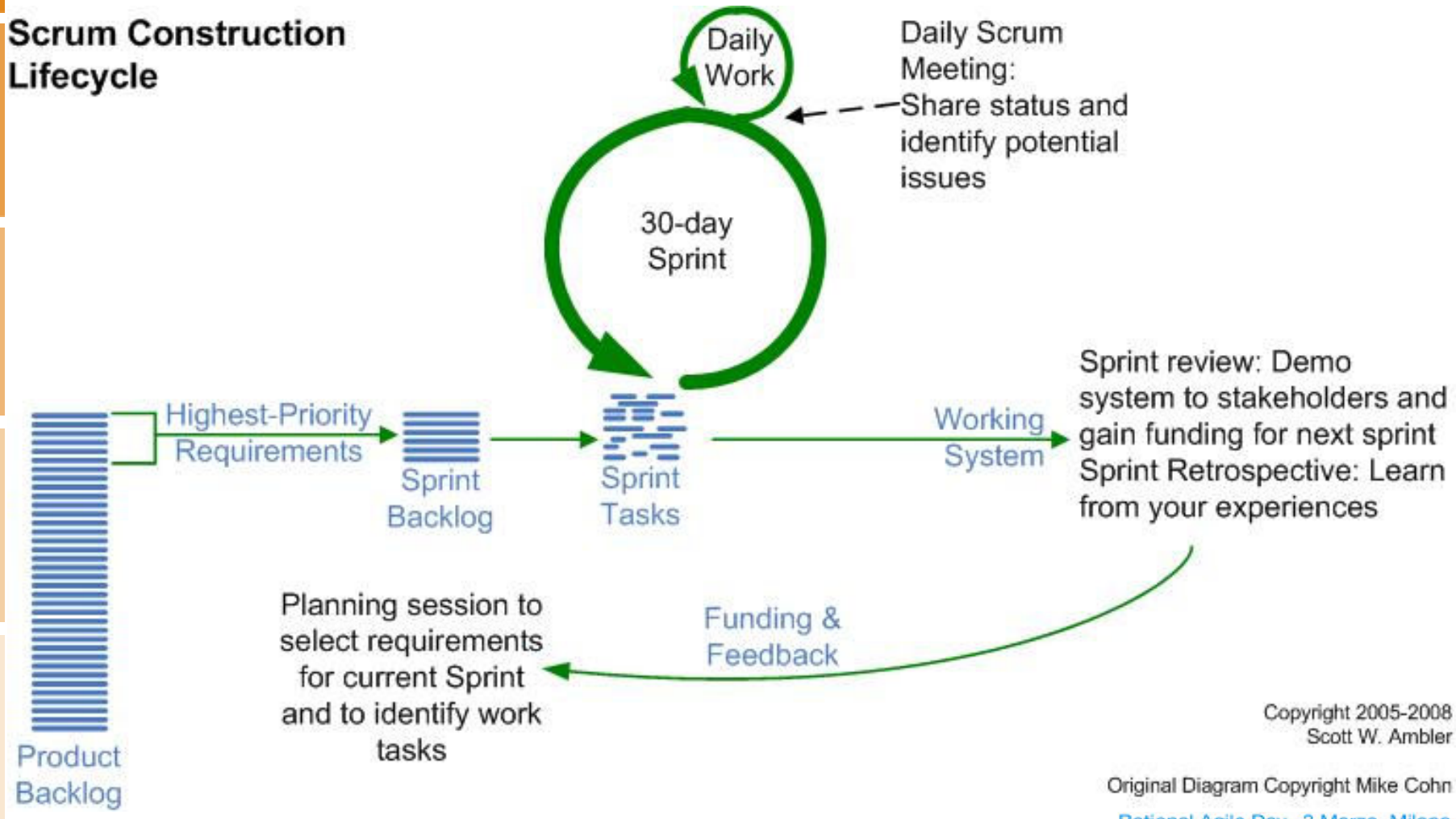
- Introduction
- Becoming agile
- **Thinking agile**
- **Staying agile**
- Where to begin…

Mainstream Agile Practices

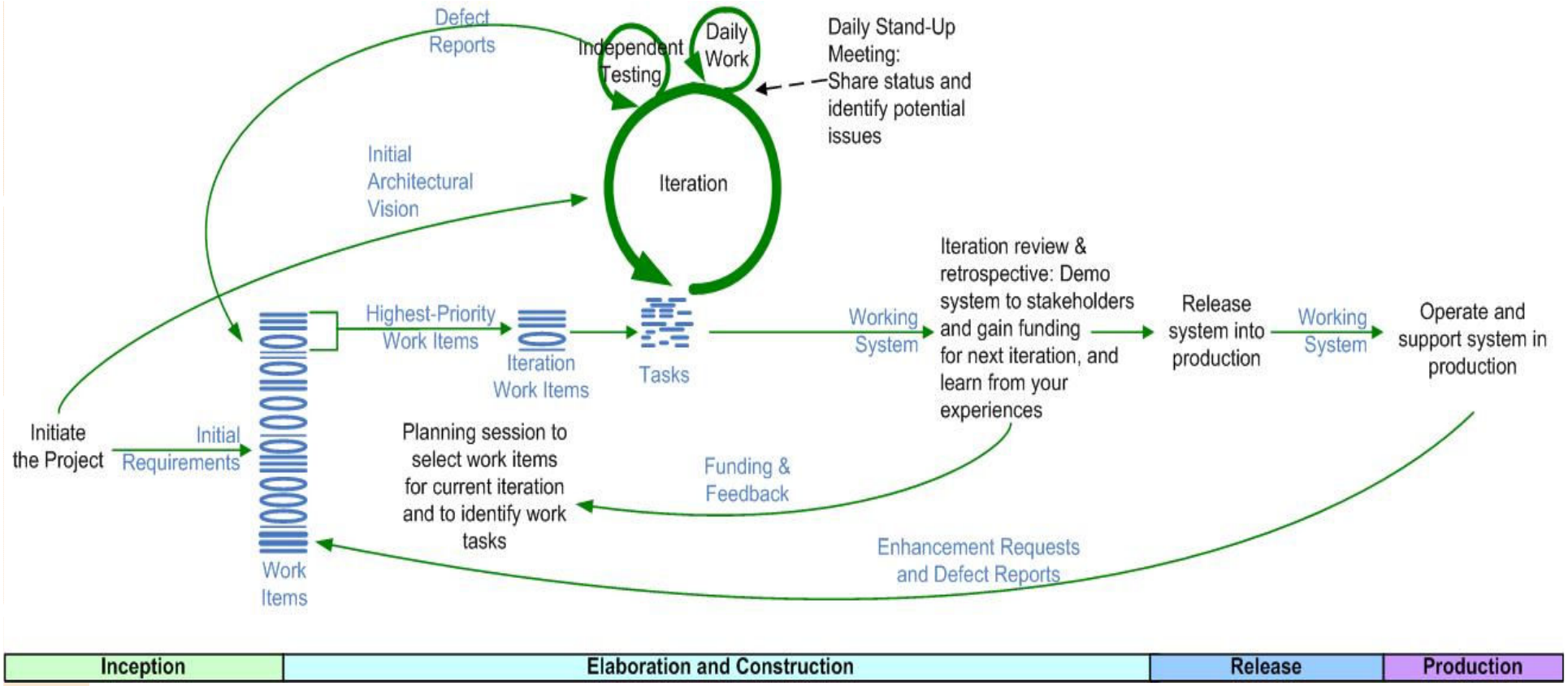
- Regular Deployment of Working Software
- Non-Solo Development
- Refactoring
- Continuous Integration
- Configuration Management
- Test Driven Development (TDD)
- Agile Testing
- Agile Documentation



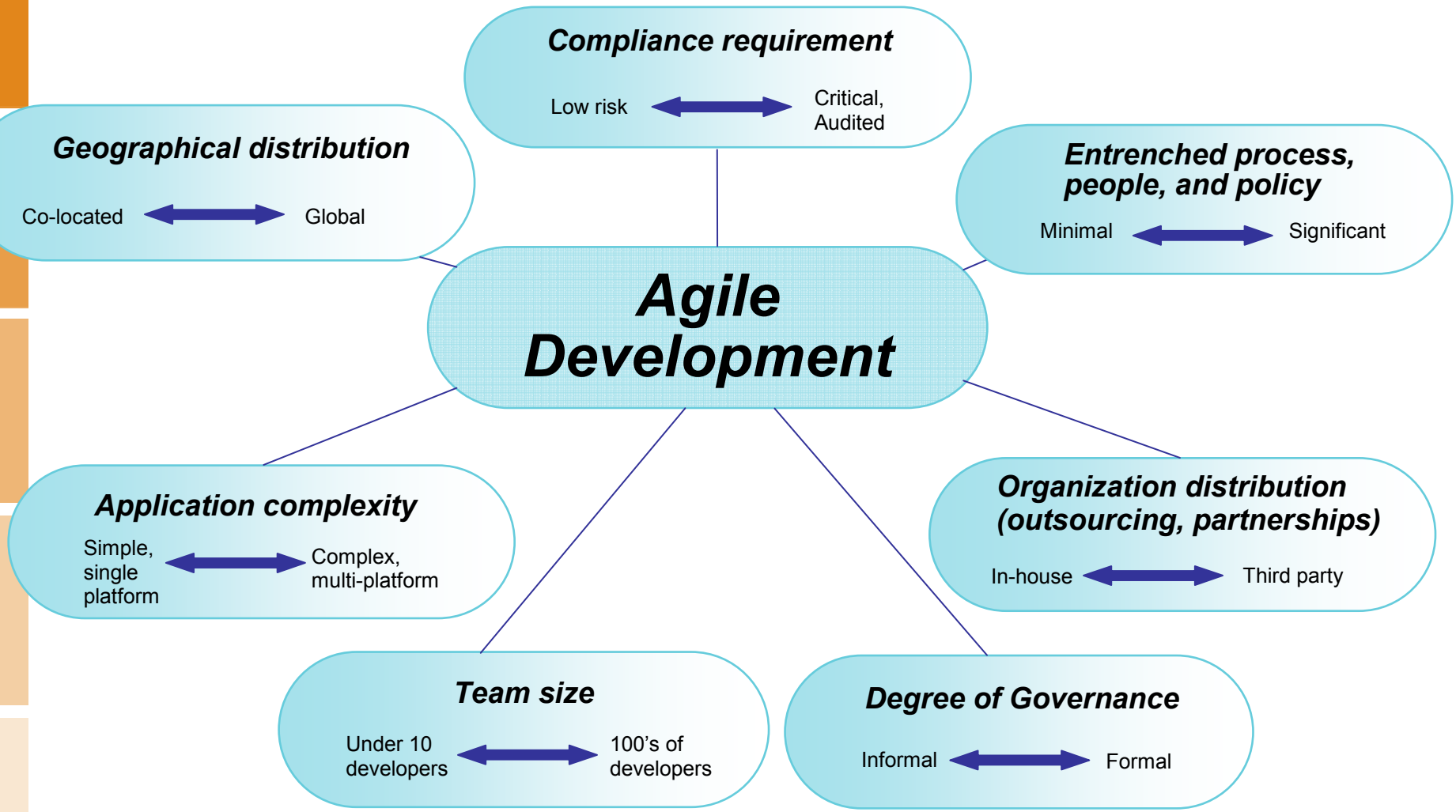
The Agile Construction Lifecycle



The Full Agile Delivery Lifecycle



Challenges with Agile in the Mainstream



Achieving Agility at Scale

Disciplined agile teams:

1. Produce working software on a **regular basis**.
2. Do **continuous** regression testing, and better yet take a Test-Driven Development (TDD) approach.
3. Work **closely** with their stakeholders, ideally on a daily basis.
4. Are self-organizing, and disciplined teams work within an **appropriate** governance framework.
5. **Regularly** reflect, and **measure**, on how they work together and then act to improve on their findings in a **timely** manner.

Critical culture shifts in improving software economics

Conventional Governance

Activity-based management
- Mature processes, PMI/PMBOK
- Plan in detail, then track variances

Adversarial relationships
- Paper exchange, speculation

Requirements first
- Assumes certainty in desired product
- Avoid change

Early false precision
- "More detail = higher quality"

Apply too much or too little process
- Process is primary, blind adherence

Agile Governance

Results-based management
- More art than engineering
- Plan/steer/plan/steer…

Honest collaborative communication
- Progressions/digressions, facts

Architecture (*risk mitigation*) first
- Admits uncertainties
- Manage change

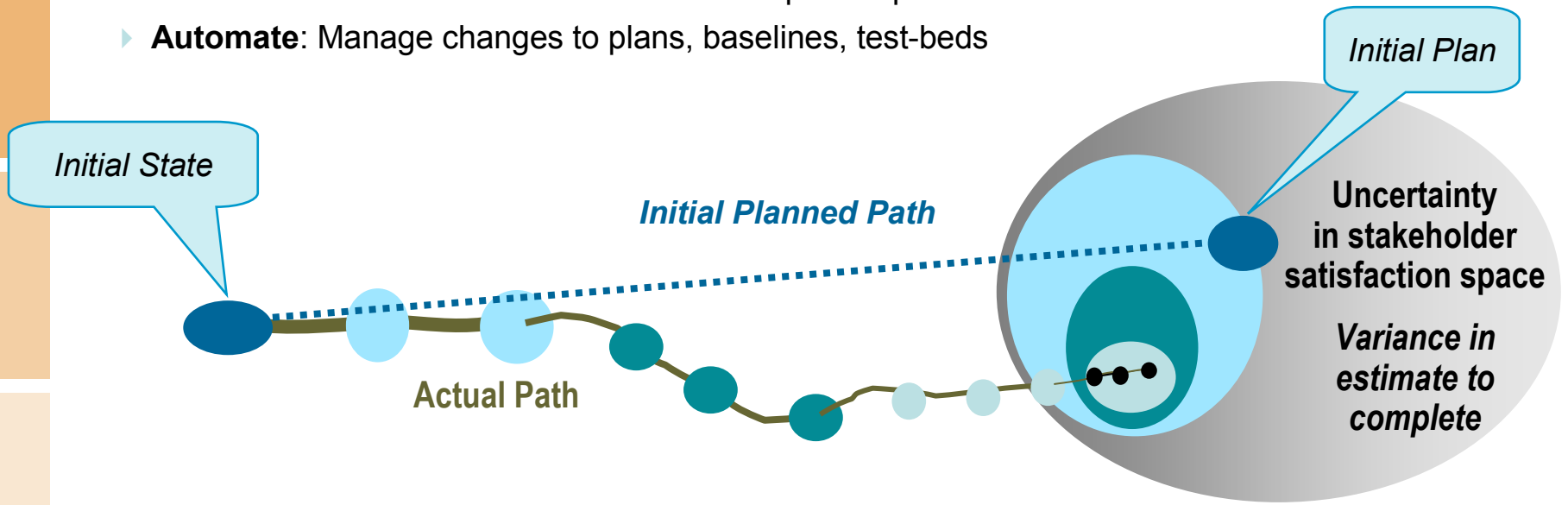
Evolving artifacts
- Scope (Problem specs)
- Design (Solution specs)
- Constraints (Planning specs)

Right-size the process
- Desired results drive process
- Manage variances

Thinking Agile means "Measure and Steer"

- At onset of program
  - **Report:** Establish estimates/variances of effort, cost, establish initial plan
  - **Collaborate**: Set initial scope and expectations with stakeholders
  - **Automate**: Establish a collaborative development environment

- At each iteration, improve estimates and report
  - **Report**: Values and variances of progress achieved, quality achieved, resources expended
  - **Collaborate**: With stakeholders to refine scope and plans
  - **Automate**: Manage changes to plans, baselines, test-beds

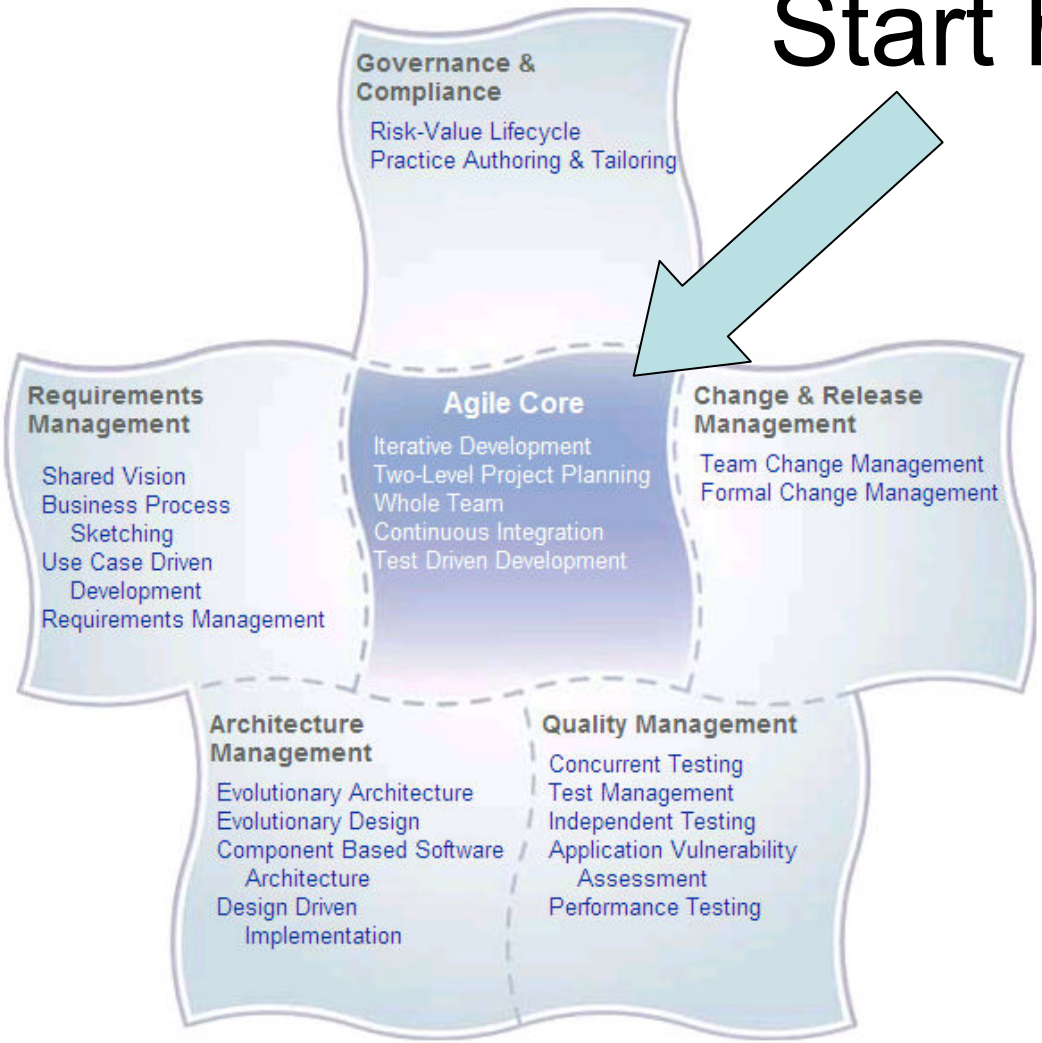


Case Study: A Large-scale Agile Improvement Effort

- A large Scandinavian bank
- 2000+ developers
- 6 business units
- Development teams are often geographically distributed

IBM Practice Library

Start here!



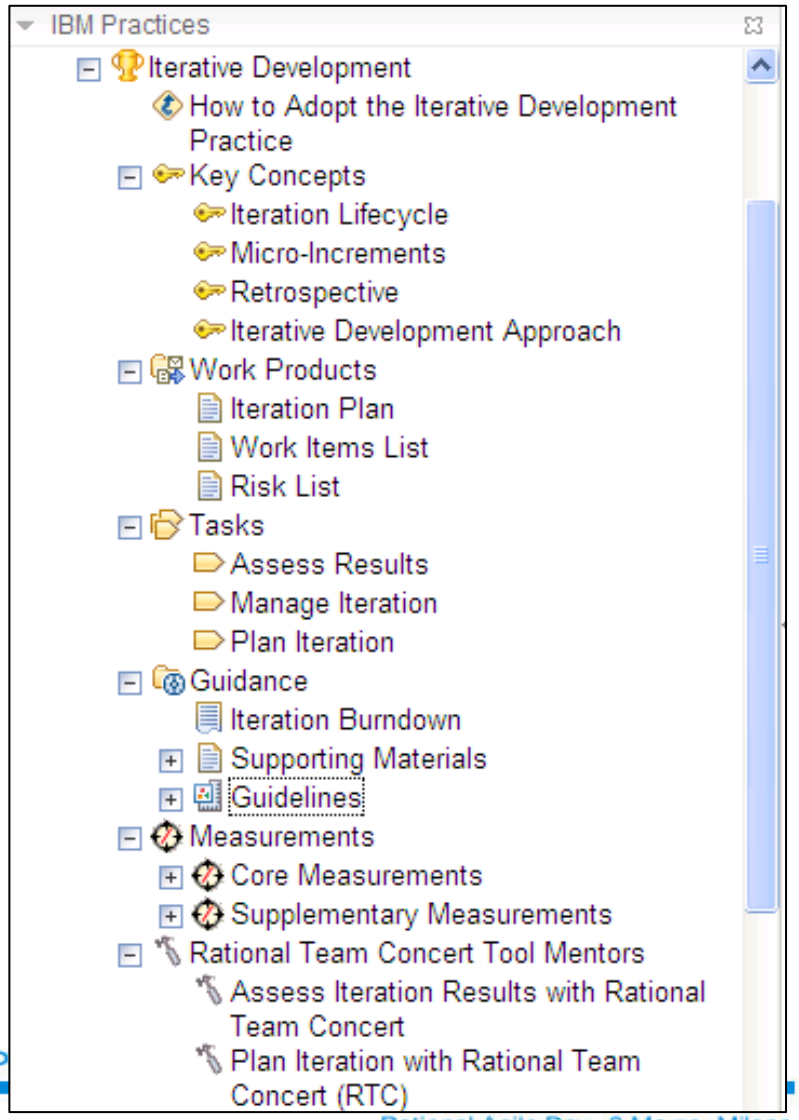
A version of these practices is available in OpenUP

Case Study – Practices by Priority

- Foundation
  - Iterative Development
  - Two-Level Planning
  - Team Change Management
  - Shared Vision
  - Continuous Integration
  - Whole Team
- High
  - Risk-Value Lifecycle
  - Test-driven development
  - Use case-driven development
- Medium
  - Evolutionary Architecture
  - Concurrent Testing
- Low
  - Business Process Sketching
  - Evolutionary Design
- Ultra Low
  - Process authoring and Tailoring
  - Requirements Management
  - Formal Change Management
  - Component Based Software Architecture
  - Design Driven Implementation
  - Test Management
  - Independent Testing
  - Application Vulnerability Assessment
  - Performance Testing

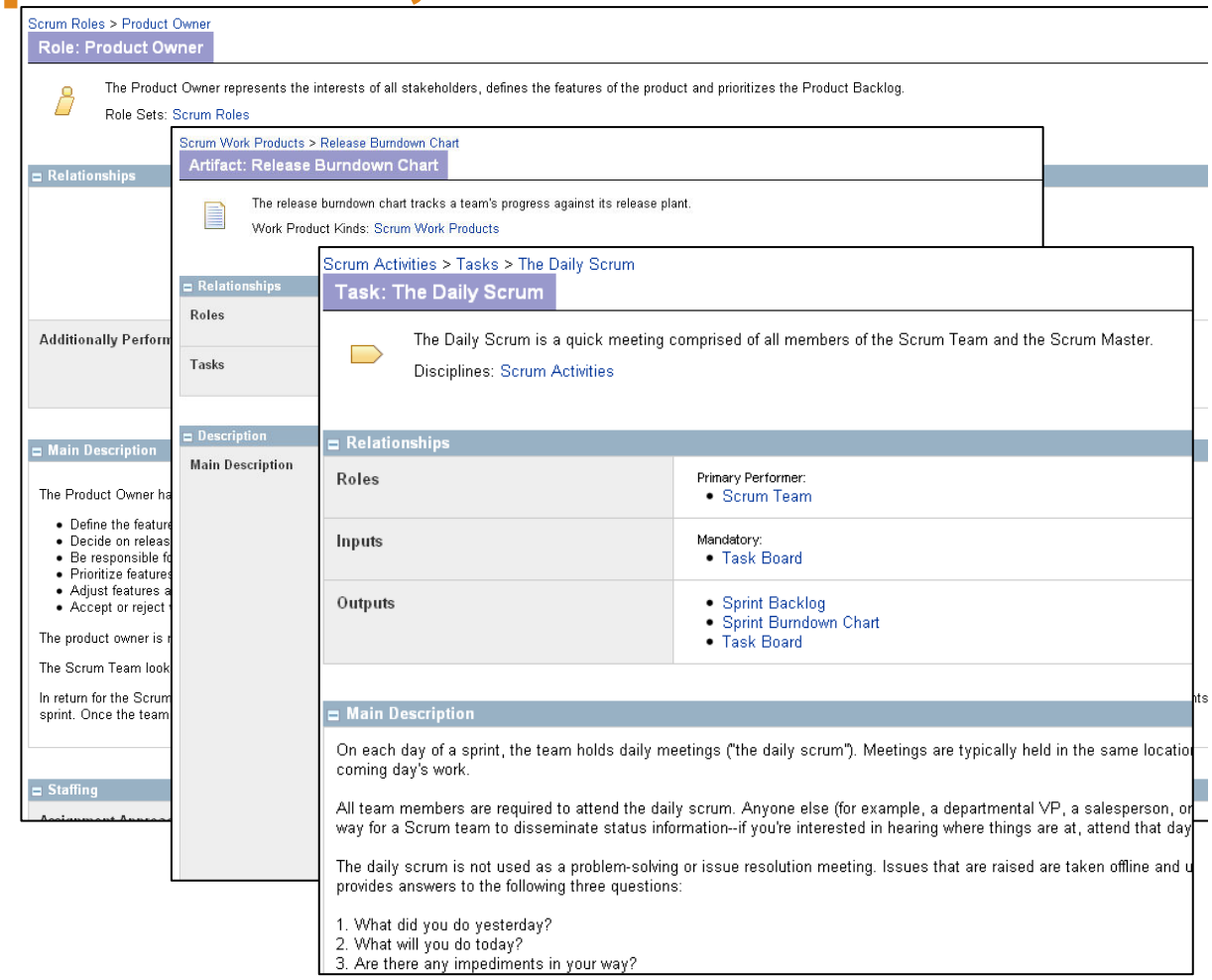
What's in a Practice?

- Key concepts
- Work products
- Tasks
- Guidance
- Measurements
- Tool mentors

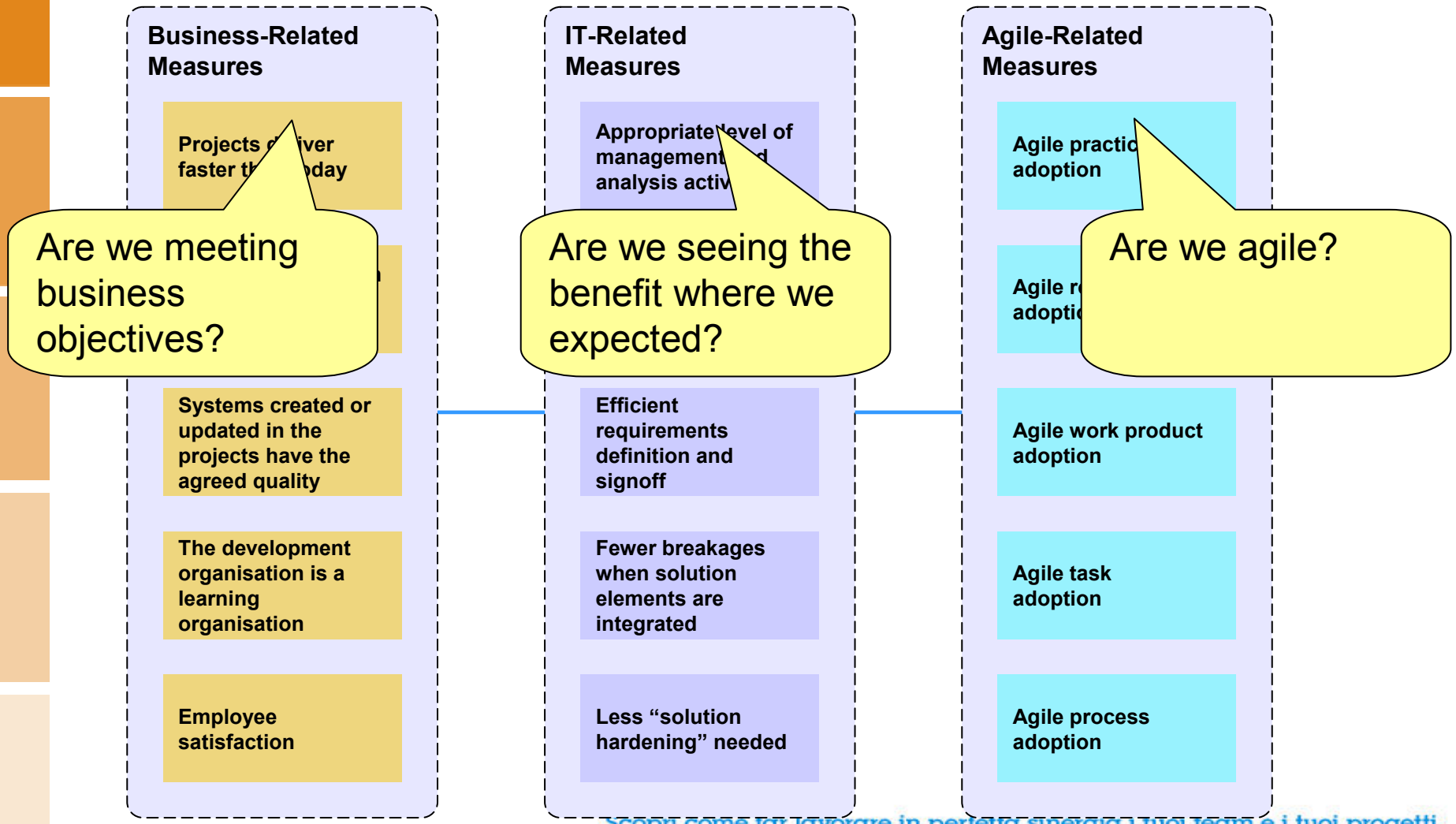


Roles, work products, tasks

- Roles
  - Product owner
  - Scrum master
- Work Products
  - Product backlog
  - Blockers list
  - Sprint Goal
  - Task Board
  - Epics
  - User stories
- Tasks
  - Various

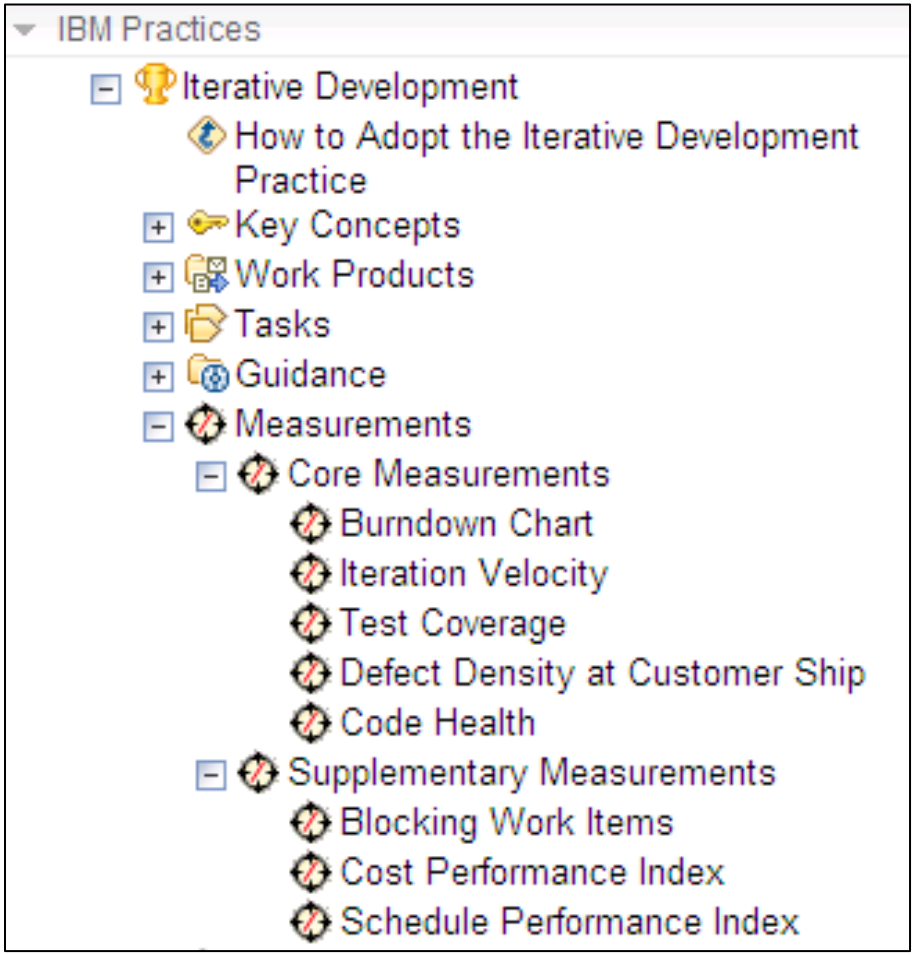


Measures help answer key questions



Selecting Measures (Metrics)

- Simple criteria
  - Who cares?
  - Will it add value?
  - Will collection be intrusive?



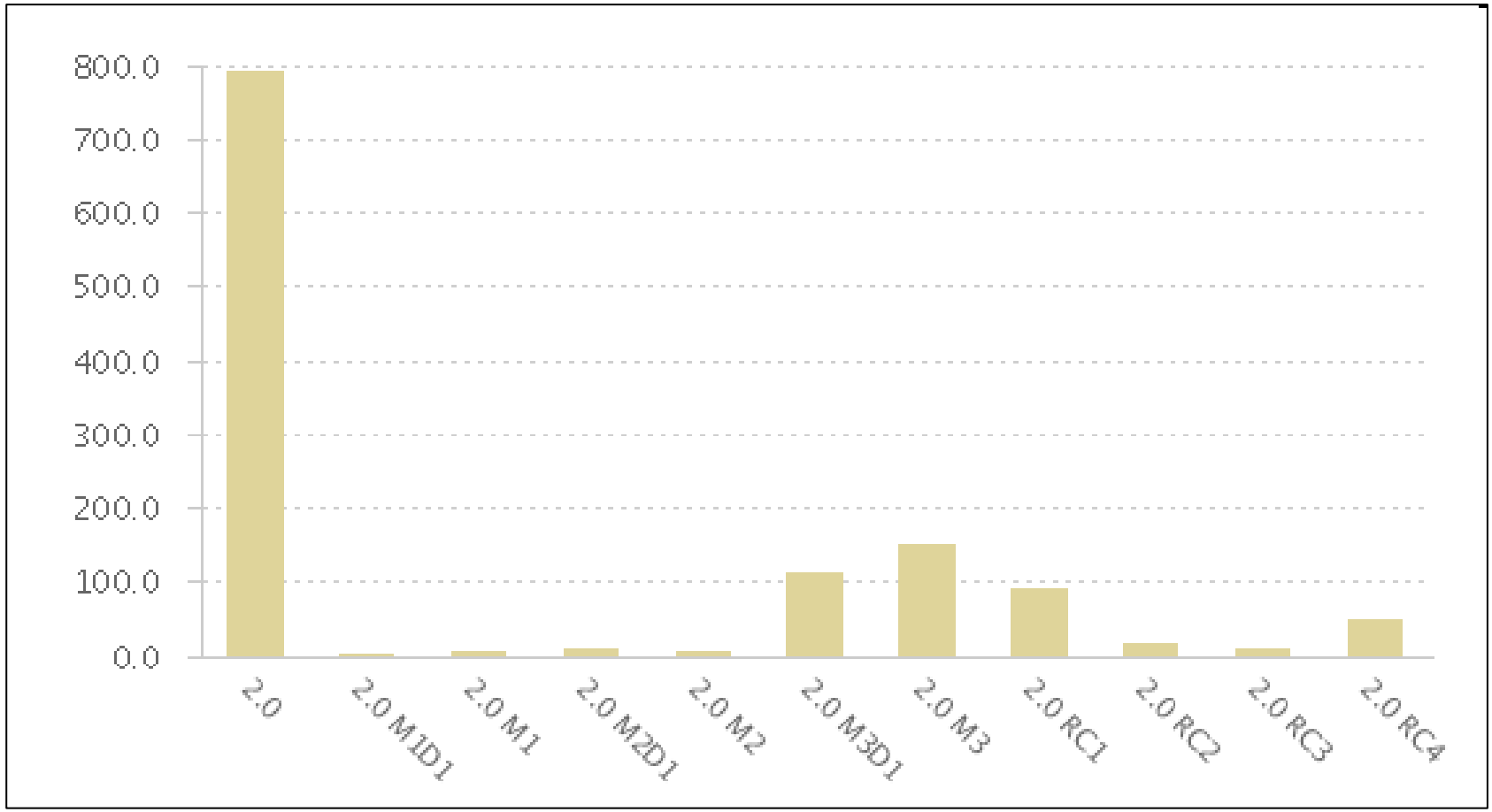
Case Study – Initial Metrics

	Business-related	Agile-related
Cycle time reduction	•Time spent from project initiation to delivery of first increment •Time spent from project initiation to project closure	•Sprint velocity •Blocking work items
Quality	•Defects (severity 1 and 2) in production per 100 FPs	•Defect trend
Continuous optimisation	•Process maturity level	•Adoption of agile practices
Productivity	•Function points per man year	•Sprint burndown chart •Release burndown chart

Category: Cycle time reduction
Metric: Sprint velocity

Objectives	Sprint velocity is used to measure the performance (and therefore capability) of the team. The velocity is useful in identifying the trend of how much work a team can complete in a sprint.
Baseline Metric	The number of points is plotted on the Y-axis and sprints on the X-axis. In initial sprints, the team velocity is typically low but subsequently increases and stabilises as the project proceeds. If the velocity rises or falls dramatically then it needs the immediate attention.
Unit	Velocity can be measured in term of points, days, hours, or any other unit the team is using for estimation.
Responsibility	Project Manager.
When to Measure	During project execution.
Manual/Automated	Automated in Rational Team Concert.
Data Repository	Available in Rational Team Concert.
Project Calculation	Velocity, calculated as the number of units of work the team has completed in a given sprint. Units can be points, days, hours or any other unit your team is using for estimation.
Example	See over for chart.
Target	A trend of a steady or increasing number of work items addressed over time.

Sprint Velocity Example

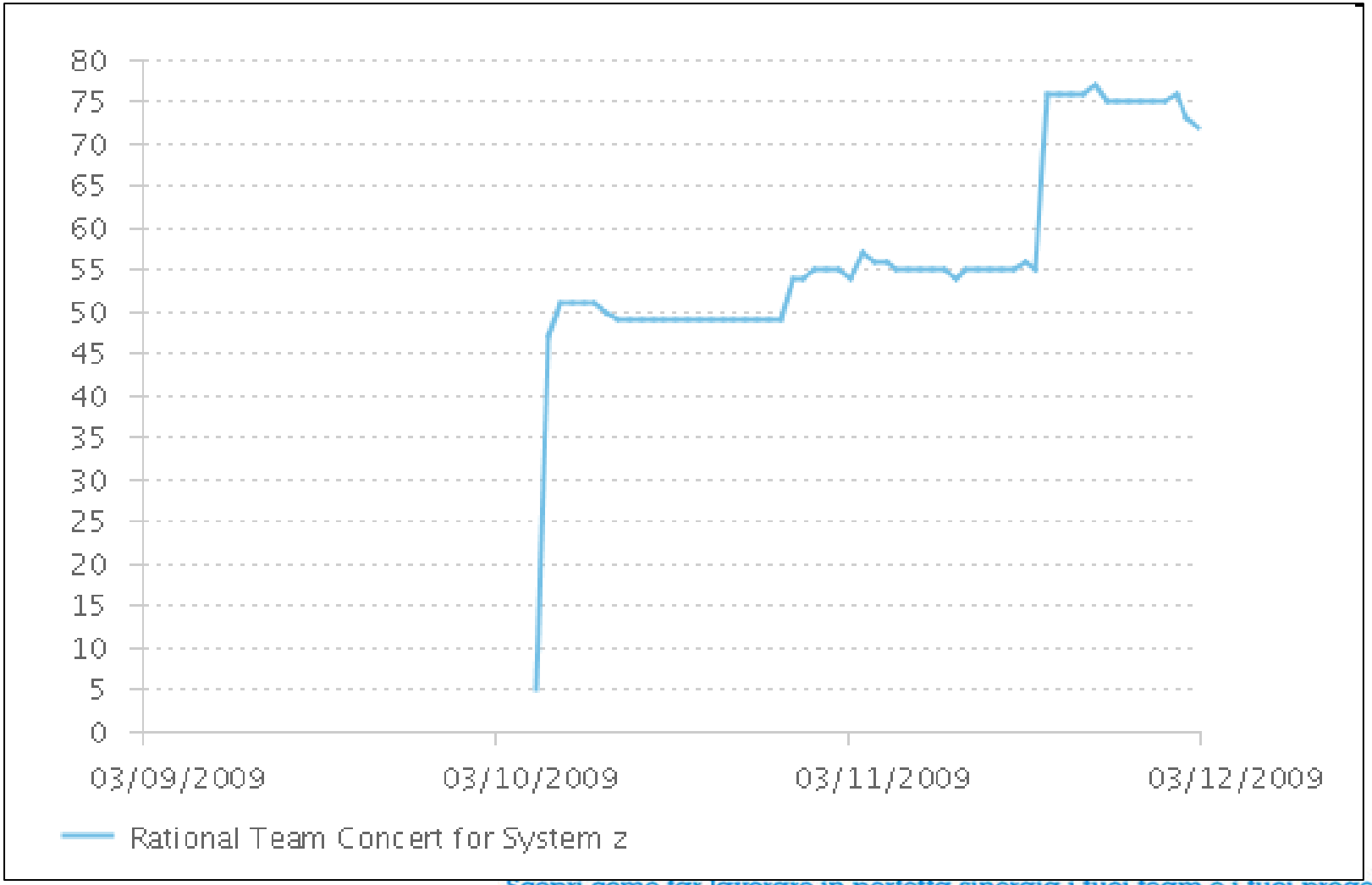


Taken from RTC 2.0 project at jazz.net on 3rd December 2009

Category: Quality
Metric: Defect trend

Objectives	The defect trend is used to ensure that arrival and closure rates have some correlation (i.e. that your arrivals don't consistently outpace your closure, resulting in a high defect backlog), to determine the remaining defect backlog, to project the future defect arrival/close rate up to (and after) customer ship.
Baseline Metric	Slope of a trend chart showing total cumulative defects (total found – total closed) over time. Ideally, the slope should be flat or decreasing.
Unit	Chart slope.
Responsibility	Project Manager.
When to Measure	During project execution.
Manual/Automated	Automated in Rational Team Concert.
Data Repository	Available in Rational Team Concert and Rational Quality Manager.
Project Calculation	•Number of defects found for each unit of time (usually a week, but could be day or month, depending on sprint length). •Number of defects closed for each unit of time. •Total cumulative defects (total found - total closed).
Example	See over for chart.
Target	A trend of a steady or decreasing number of defects over time.

Defect Trend Example

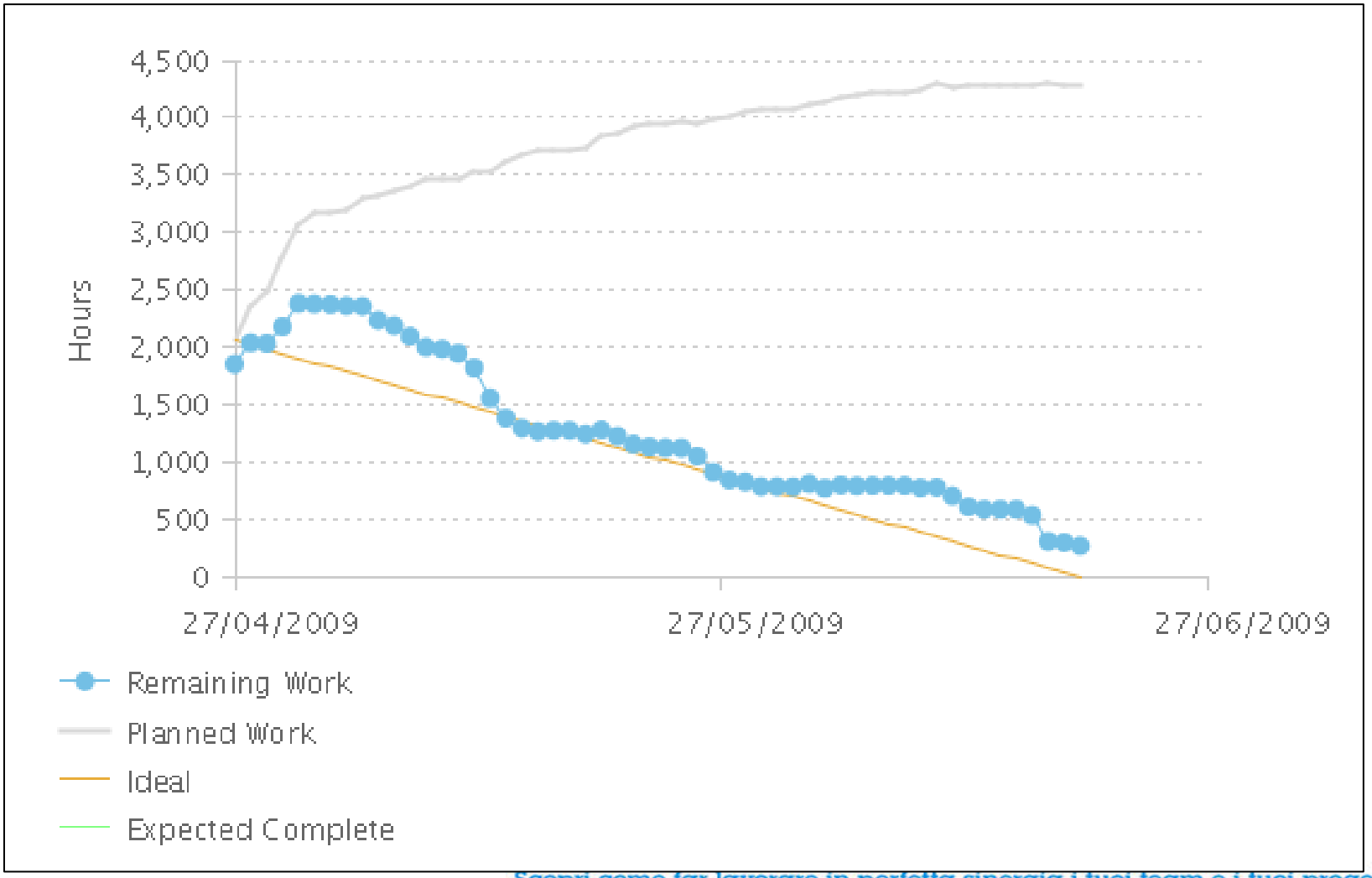


Taken from RTC 2.0 project at jazz.net on 3rd December 2009

Category: Productivity
Metric: Sprint burndown chart

Objectives	A sprint burndown chart allows the progress of the sprint to be measured.
Baseline Metric	Slope of the chart. The number of remaining units (such as work items or hours) is shown on the Y-axis, together with the number of planned units, and time is shown on the X-axis. Ideally, the trend of remaining units should go down as time progresses.
Unit	Chart slope.
Responsibility	Project Manager
When to Measure	During project execution.
Manual/Automated	Automated in Rational Team Concert.
Data Repository	Available in Rational Team Concert.
Project Calculation	•Number of planned units during time I for the sprint. •Number of actioned units during time I for the sprint.
Example	See over for chart.
Target	A trend of a decreasing number of remaining units over time.

Sprint Burndown Example



Taken from RTC 2.0 project at jazz.net on 3rd December 2009

Agile Adoption Example (detail)

Instruction:

1) Participants fill out the answers in the data section (light blue area) below. One column for the answers of each participant
2) Answer each question using a scale of 1 to 5. 1 = never, 3 = halfway job (or half the time), 5 = always
You can use fractions. You can leave blank for NA. 0 means that you hate this practice.
3) If desired, right-click and insert a comment in the cells in the data area.

ITERATIVE DEVELOPMENT
(based on sprints)

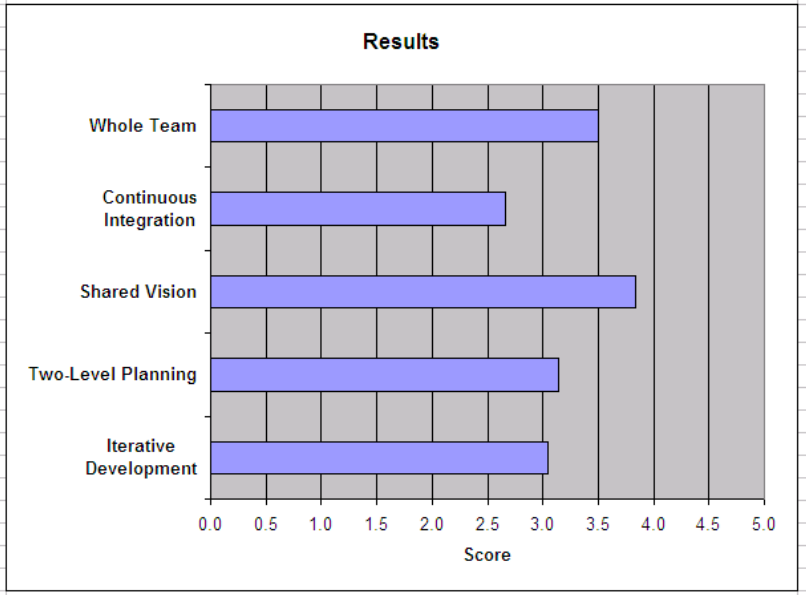
Target	Question	Results Section			Data Section								
		Average	Deviation	Talk?	a	b	c	d	e	f	g	h	i
Time-boxed Sprints	Do you hold your sprint end dates fixed, and adjust content of that sprint if needed?	2.5	2.121	Talk	4	1							
Daily Scrum	Do you hold a daily Scrum meeting?	2.5	2.121	Talk	4	1							
Scrum Master	Do you have a Scrum Master assigned to the project?	2.5	0.707		3	2							
Sprint Planning Meeting	Do you detail the plan for the next sprint at the end of the current sprint?	3.5	2.121	Talk	2	5							
Sprint Review Meeting	During a sprint review, do you calibrate progress made with project goals by discussing what worked well, what didn't work well, and how to improve? Do you improve planning for the next sprint and update the long-range plan accordingly? Do you use feedback, including test results to improve your process?	2.5	2.121	Talk	1	4							
Estimating the Product Backlog	Do you involve the entire team in estimation. Do you re-plan your work for each sprint based on your previous "Velocity" (how much work got done in previous sprints)? Do you update overall plan and stakeholder expectations based upon actual progress.	3.5	2.121	Talk	2	5							
Prioritizing the Backlog	Do you select content for your next sprint from a prioritized set of work items (including functionality and defects)?	3.5	0.707		3	4							
Working Increment	Does each sprint (except perhaps the earliest ones) result in a stable executable release (internal or external), with code that you can demonstrate?	3.0	0.000		3	3							
Feedback Used	Do you use feedback from key stakeholders such as sponsors, partners, users to adjust the content of subsequent sprints?	3.5	2.121	Talk	5	2							
Micro-Increments	For each sprint, do you define measurable tasks for sprint objectives, and are these tasks small enough to be performed by one or a few people?	3.0	0.000		3	3							

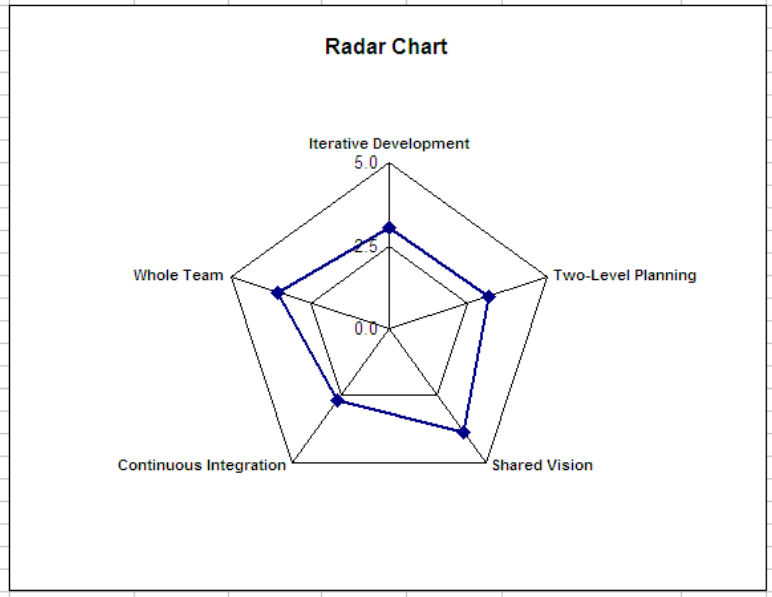
Agile Adoption Example (summary)

AGILE TEAM PULSE (summary)

Score out of 5

Practice	Score
Iterative Development	3.1
Two-Level Planning	3.1
Shared Vision	3.8
Continuous Integration	2.7
Whole Team	3.5





Case Study – Automation

	Automated	Manual
Cycle time reduction	•Time spent from project initiation to delivery of first increment •Time spent from project initiation to project closure •Sprint velocity •Blocking work items	
Quality	•Defects (severity 1 and 2) in production per 100 FPs (*FP count is manual*) •Defect trend	
Continuous optimisation		•Process maturity level •Adoption of agile practices
Productivity	•Function points per man year (*FP count is manual*) •Sprint burndown chart •Release burndown chart	

Topics

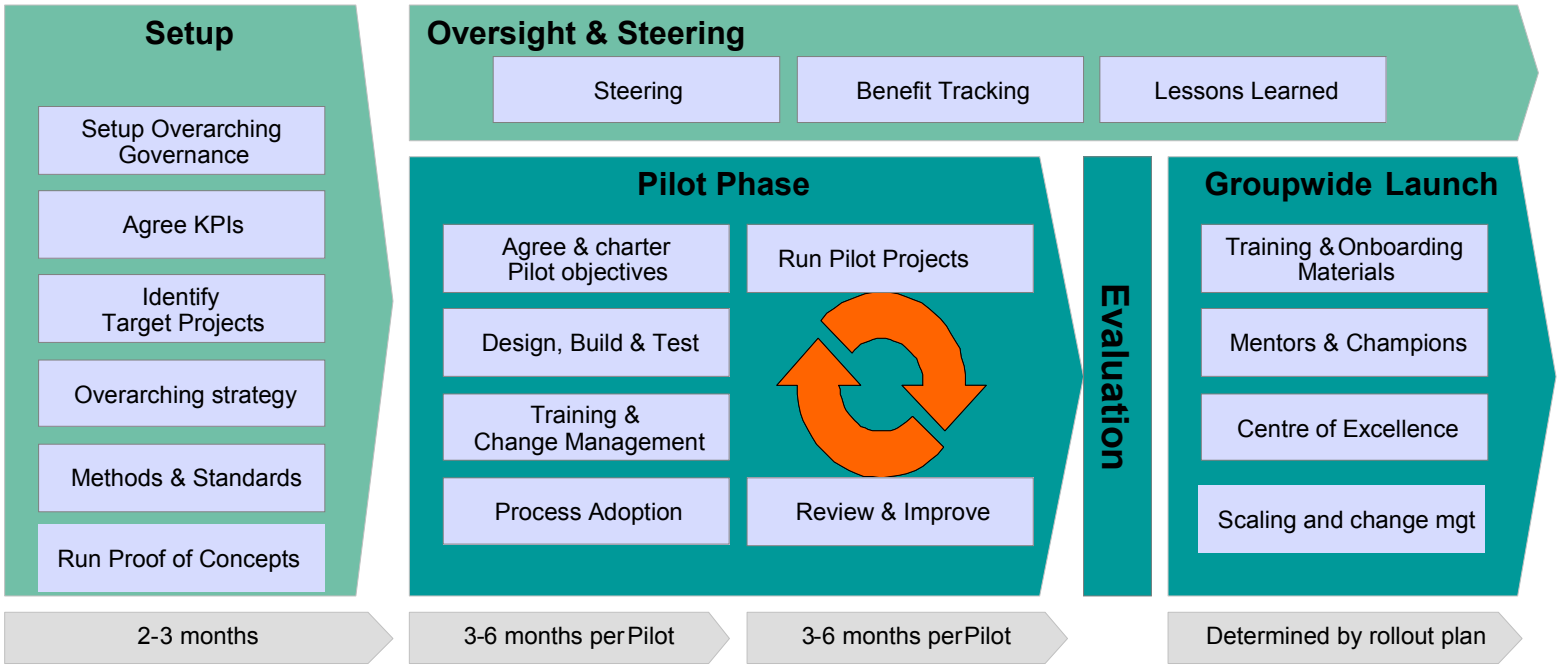
- Introduction
- Becoming agile
- Thinking agile
- Staying agile
- **Where to begin…**

Consider all the elements

- Delivering agility in your organization requires several coordinated elements

- **Process and method content** based on content from standard frameworks (SCRUM and OpenUP) augmented with content from the organization´s existing processes extended with guidance from IBM.
- **A workbench** delivering the appropriate supporting capabilities to automate, accelerate, guide and measure adoption of the new practices.
- **Decision framework** for selection of lifecycle practices that are appropriate to the project characteristics, and guide process adaptation.
- **Metrics** and dashboards for assessing projects, BU, and organizational KPIs.
- **Pilot** strategy and criteria for selecting pilots, managing candidate pilot project adoption, and adjusting practices and tool based on pilot progress.
- **Organizational improvement** through scheduled training activities, support/coaching concept and communication, and broad educational tasks.

Typical programme structure & plan for large-scale Agile at Scale rollout



The programme must be structured as:

- Initial **setup phase** to define standards, setup benefit tracking mechanisms and setup programme governance and oversight
- An ongoing **oversight and steering** stream to enforce standards, ensure continuity and track benefits across the disparate projects
- For each 'practice area':
  - A set of **Pilot projects** on a small pool of users per area (2-5 projects). These would typically take 3-6 months to setup and then require 3-6 months of 'running' to evaluate the concept and make improvements
  - Once the pilot has completed, a separate **'launch' scale out** is needed to be rolled out across the organization

A Call To Action

- **Consider an Agile Pilot Project**
  - See it work for yourself
  - Get mentoring help
- **Get some Agile training**
  - Project management training is critical
  - Training modelers, developers, … is also critical
- **Get an Agile Health Check**
  - Look at key agile practice areas
  - Use a Measured Capability Improvement Framework (MCIF) to establish target
- **Adopt appropriate Agile practices**
  - Select agile practices that optimize you project characteristics
  - Align with control mechanisms and risk-mitigation strategies
  - Support with tools that automate those practices

Go to IBM