Agile ERP: Overcoming the Challenges

Janice Aston, Agile Perspective Inc.
Gerard Meszaros, Independent Consultant
Glenn Mickelson, Independent Consultant

May 5, 2008

SAP ASUG

Agenda

- Introduction
- What is Agile?
- Why Did We Choose Agile?
- How Agile Delivers Value
- Adopting Agile Practices
- Summary

Introduction

- Janice Aston
  - Over 17 years of project mgt experience utilizing both traditional and agile methods.
- Gerard Meszaros
  - 20+ Years experience managing software development
  - Presented at Agile conferences since 2001
- Glenn Mickelson
  - 25+ Years IT Experience with 12 Years SAP Development experience

Caveat Emptor

- This a report of our experience
  - On a specific project
  - In a specific context
- We don't claim all projects are like ours
- We don't claim you'll get the same results

"Your mileage may vary!"

Agenda

- Introduction
- What is Agile?
- Why Did We Choose Agile?
- How Agile Delivers Value
- Adopting Agile Practices
- Summary

What is Agile?

Agile is an iterative and incremental (evolutionary) approach to software development
which is performed in a highly collaborative manner
by self-organizing teams
with "just enough" ceremony
that produces high quality software
in a cost-effective and timely manner
which meets the changing needs of its stakeholders.

definition by Scott W. Ambler

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:

Agile Values		*Traditional Values*
Individuals & 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.

Agenda

- Introduction
- What is Agile?
- Why Did We Choose Agile?
- How Agile Delivers Value
- Adopting Agile Practices
- Summary

Why did we choose Agile?

- Previous project was delivered using agile approach
- Hugely successful outcome
  - High quality product
  - Delivered early and under budget
  - Business ecstatic with product
  - Excellent Business-IT relationship
- Why not?

SAP Project Results

- Highly usable product
  - Workflow enhanced current business process enabling productivity gains
- Benefits seen immediately
  - End-users requesting early adoption of tool
  - Highly positive unsolicited feedback
- Business Highly Engaged & Supportive
- Recipients of President's Award

Challenges - Not all a bed of roses

- Development environment limitations
  - More easily overcome
- Constrained technical resources
  - Recruited team new to company and Agile.
- Team members' mindsets needed changing
  - Value & usability vs. product focus
- IT delivery model changed midstream
  - Best Shoring introduced

Questions to ponder:

What success have you had with SAP implementations?

Why?

Agenda

- Introduction
- What is Agile?
- Why Did We Choose Agile?
- How Agile Delivers Value
  - Highly Communicative, Collaborative Team
  - Highly Incremental Story-Based Development
  - Continuous Learning
  - Continuous Feedback
  - Modern Development Practices
- Adopting Agile Practices
- Summary

Topic Template

- Practices
  - How they work + benefits of using them
- Challenges
  - That we encountered in applying them
- How We Overcame Them
  - If applicable

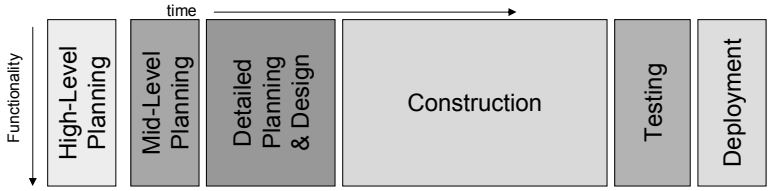
Highly Communicative, Collaborative Team

- **Highly Visible Goals**
  - **Everyone understands big picture**
    - **Business Goals**
      - Elevator Statement
      - Product Vision Box

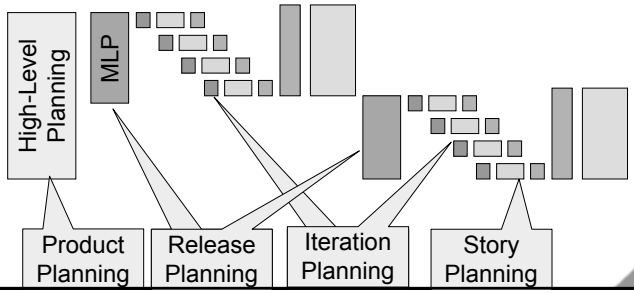
Collaboration via Radical Co-location

Continuous Planning

§ Traditional Development does all planning up front:



§ Agile Development does continuous planning:



Highly Visible Workspace

§ Everyone knows the status of the project via:
  - § UI Story Board
  - § Iteration Task Board
  - § Burn Down charts
  - § Iteration Objectives
  - § etc.

Examples of each of these on subsequent slides

UI Story Board

§ User Interface mock up annotated with story cards describing desired behavior of software.

Challenges

§ Deep but silo'ed skills

§ Document-focused tradition

§ Product vs. Value driven mindset

How We Surmounted Them

- Got the "Right" people on the bus
  - Hired For Attitude
  - Augmented the SAP skill set with highly skilled generalists
- Frequent Conversations highly encouraged
  - Avoid assumptions & surprises
- Feature breakdown (FBS), not work breakdown (WBS)
  - Incremental chunks of valuable functionality
  - worked on high value features first

Agenda

- Introduction
- What is Agile?
- Why Did We Choose Agile?
- How Agile Delivers Value
  - Highly Communicative, Collaborative Team
  - Highly Incremental Story-Based Development
  - Continuous Learning
  - Continuous Feedback
  - Modern Development Practices
- Adopting Agile Practices
- Summary

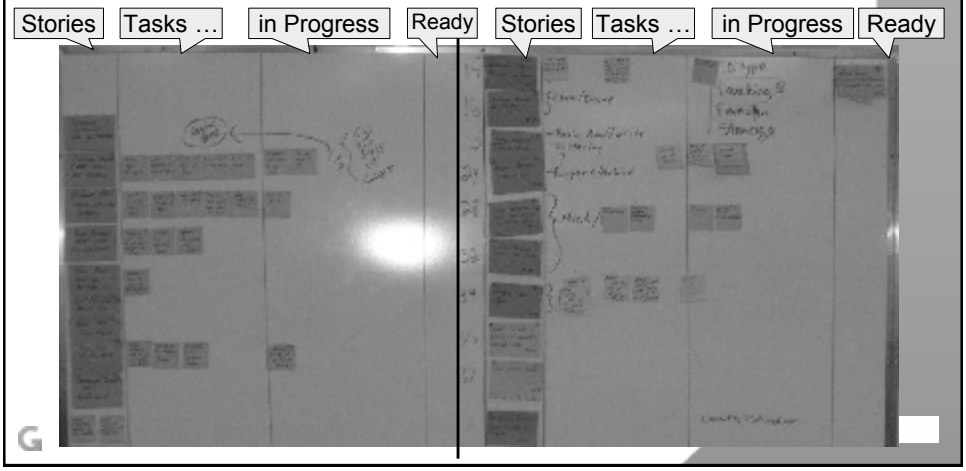
Biweekly Iteration Planning Meeting

Highly Incremental Story-Based Development

- Business Makes Decisions About What to Build Each Iteration
  - Business makes decisions about scope & priority based on ROI (value/estimate)
  - Based on Cross-Disciplinary Estimates per Story
    - BA, UI, Functional, ABAP, .Net
- Requirements are Negotiable
  - SAPers encouraged to suggest better/cheaper ways to achieve goals of business

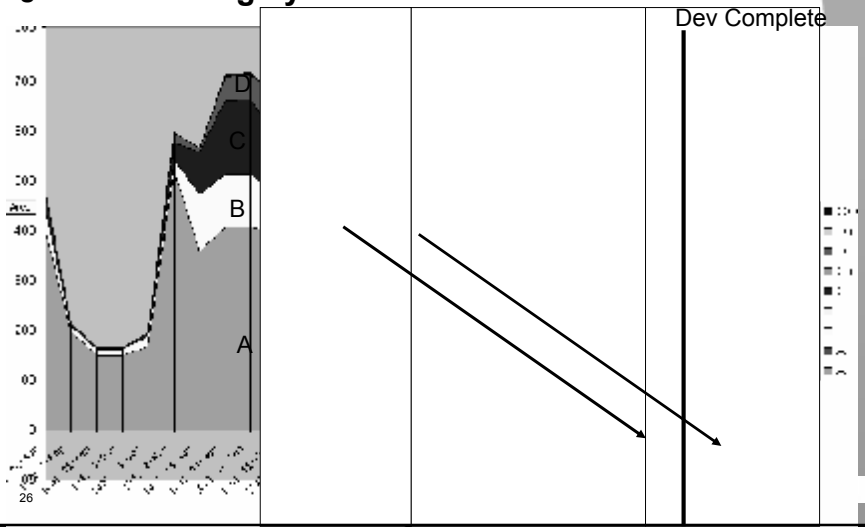
Iteration Task Board

- User Stories decomposed into Tasks to achieve them
- Tasks are moved between columns as status changes
- Board is focal point of Daily Stand-up Meeting / SCRUM



Burn Down Chart – Estimated Effort Remaining

- Used by business to track progress towards release
- Business highly motivated to remove excess functions.

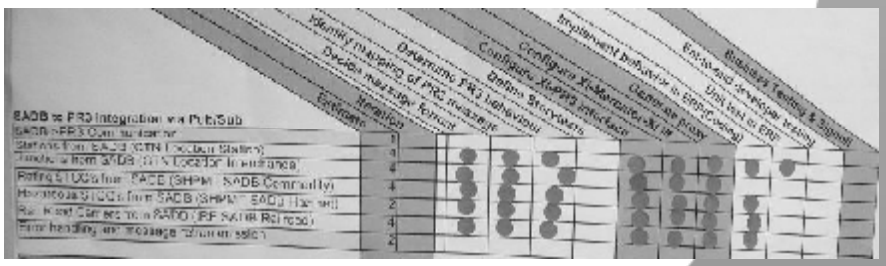


Challenges

- Similar challenges:
  - Product (Not Usability) Focus
  - Document-Driven, Waterfall Tradition
- Integrated System = All projects on same train
  - Forced fewer, less frequent releases
- Measuring Velocity with Silo'ed skills
  - Used for deriving duration from effort
- Tried different kinds of Story Points & Velocity:
  - Web Dynpro, regular ABAP, SD configuration

How We Surmounted Them

- Stressed the Business Case (Usability)
- Focused burndown on constraint (TOC):
  - Web Dynpro development velocity
- Imposed discipline
  - Deliverables (Documents or Code) Every 2 Weeks
  - Participation in Iteration Planning Meetings



TOC = Theory of Constraints

Agenda

- Introduction
- What is Agile?
- Why Did We Choose Agile?
- How Agile Delivers Value
  - Highly Communicative, Collaborative Team
  - Highly Incremental Story-Based Development
  - Continuous Learning
  - Continuous Feedback
  - Modern Development Practices
- Adopting Agile Practices
- Summary

Continuous Learning

- Bi-weekly Process Retrospectives
  - Incorporate learning's into on-going project
  - Fine-tune processes & working agreements
  - Enables self organizing teams
- Just-in-Time Decision Making
  - Continuous planning
  - Buying information via "Spikes"
  - Enables better decisions

Challenges

- Initially, team saw limited value in retrospectives
  - "Oh, not again!"
  - Team ownership of processes was foreign
- Self organizing team was a new concept
  - Equal participation sometimes a challenge
  - Not being told specifically what to do
- Feeling the need to make decisions upfront.
  - Is it really good to postpone decisions to the last responsible moment?

How We Surmounted Them

- Retrospectives each iteration
  - Ensured value was seen (ie. followup issues)
  - Utilized different collaboration techniques
- Decision Making
  - Principles became part of our working agreement
  - Documented key decisions
  - Gathered data thru a time boxed "spike"
    - When we didn't have enough info. for decision

Agenda

- Introduction
- Why Did We Choose Agile?
- What is Agile?
- How Agile Delivers Value
  - Highly Communicative, Collaborative Team
  - Highly Incremental Story-Based Development
  - Continuous Learning
  - Continuous Feedback
  - Modern Development Practices
- Adopting Agile Practices
- Summary

Continuous Feedback - Early & Often

- Wizard of Oz Usability Testing
  - of Paper Prototypes
- Acceptance Tests per User Story
  - StoryTest-Driven Development
- Automated Pricing Tests
  - Using Fit
- Manual Smoke Tests

Paper Prototyping

Wizard of Oz “Usability” Testing

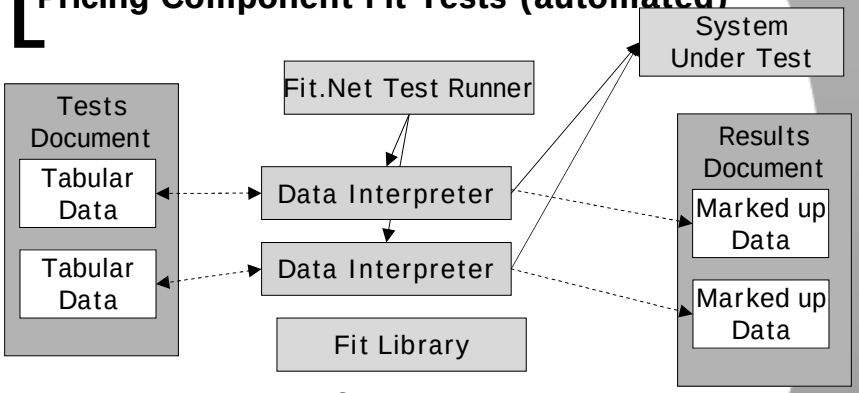
Story Tests (manual)

- Defines what "Done Looks Like"
  - Several to many tests per User Story
- Executed by developers during development
  - To make sure all cases are implemented
  - To make sure it works before showing to business
- Executed by Business as soon developer says "It's Ready"
  - Mid-iteration is better than end-of-iteration

Write StoryTest | Build Code | Test Code | Test Story

Write StoryTest | Build Code | Test Code | Test Story

Pricing Component Fit Tests (automated)



- Tabular Tests defined by the business/analysts
  - Fixtures to interpret tables built by .Net developer
- Fit Test Runner communicates with SAP via .Net Connector
  - marks up Tabular Data with test results

Sample Fit Test

PayrolFixtures.WeeklyCompensation			
Standard Hours	Holiday Hours	Hourly Wage	Pay( )
40	0	10	$400
40	0	20	$800
41	0	20	$830
40	1	20	$840
41	1	20	$870

- Our pricing tests were more complex but essentially similar:
  - A set of inputs to be injected
  - One or more outputs to be verified

Sample Fit Test Results

PayrolFixtures.WeeklyCompensation			
Standard Hours	Holiday Hours	Hourly Wage	Pay( )
40	0	10	$400
40	0	20	$800
41	0	20	$830
40	1	20	$840 *expected* $800 *actual*
41	1	20	$870 *expected* $830 *actual*

- Our pricing tests were more complex but essentially similar:
  - A set of inputs to be injected
  - One or more outputs to be verified

Usability Testing of Early Versions

- Run against early versions of application
  - Remember, we always have a working system; it just has more functionality later in the project
- Based on tasks similar to Wizard of Oz testing
  - But with real software, not just paper mockups
- Observers watch for usability deficiencies

Challenges

- Immaturity of ABAP Web Dynpro technology
- Lack of functional test automation tools
- Lack of ABAP implementation of Fit framework

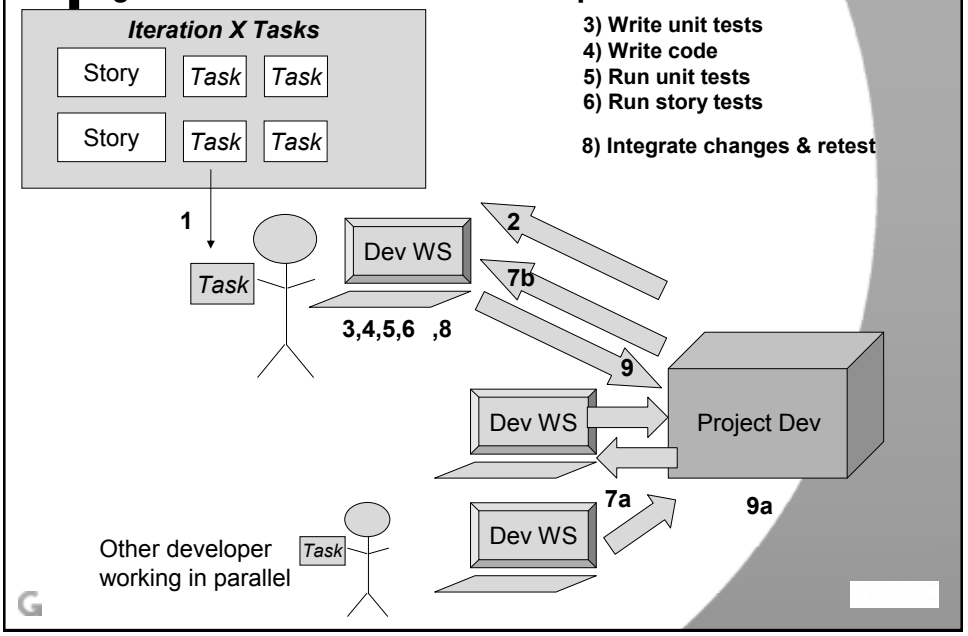
How We Surmounted Them

- Focused automation on “business logic”
  - Not the user interface
- Tested the pricing logic via Fit.Net
  - Used SAP .Net Connector to communicate
  - Could have used Fit.Java and/or web services
- Used manual smoke tests for the UI
  - Rotating responsibility through team

Agenda

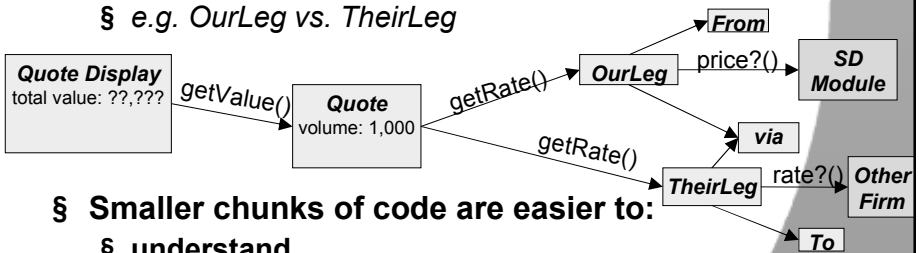
- Introduction
- Why Did We Choose Agile?
- What is Agile?
- How Agile Delivers Value
  - Highly Communicative, Collaborative Team
  - Highly Incremental Story-Based Development
  - Continuous Learning
  - Continuous Feedback
  - Modern Development Practices
- Adopting Agile Practices
- Summary

Agile Practice - Concurrent Development



Object-Oriented Design

- Code is structured around business concepts
  - Quote, Subquote, Legs, Places
- Polymorphism allows "design for variability"
  - Run-time binding picks right logic to run
  - *e.g. OurLeg vs. TheirLeg*



- Smaller chunks of code are easier to:
  - understand
  - version control (check in/out)
  - unit test
  - reuse

Automated Unit Testing

- We unit tested most logic in ABAP objects
  - Tests written first, before code
  - Using ABAPUnit (part of NetWeaver 2004s)
  - Business logic in business objects
  - UI logic via “Humble Dialog” pattern
- Software is retested many times a day
  - Units tests rerun before transporting
  - Complete ABAPUnit check done nightly
- Developers dared to change code as needed
  - Tests act as “Safety Net”
  - Much less time spent in debugger

Refactoring

- Improving code structure
  - e.g. Retrofitting older code with more recently learned techniques
  - Ensures consistency, minimal code duplication
  - Reduces cost of further development
- Software Development’s “Clean Desk Policy”
  - Pays down the “technical debt” that usually accumulates in code as it is maintained

Challenges – Server-Based Development

- Non-deterministic Automated Test environment
- Other developers can break you at any time
- Business testing is frequently interrupted

Challenges - Lack of Refactoring Support in Tools

- Invert Boolean
- Safe Delete
- Move Instance Method
- Inline Constant
- Extract Subclass
- Replace Method Code Duplicates
- Convert To Instance Method
- Renaming of packages, classes, methods, fields, method parameters and local variables with reference correction
- Moving classes and packages with reference correction
- Moving static members with reference correction
- Move Inner Class to Upper Level
- Change Method Signature
- Make Method Static
- Copy/Clone Class
- Extract Method
- Introduce Variable
- Introduce Field
- Introduce Constant
- Introduce Parameter
- Extract Interface
- Extract Superclass
- Use Interface Where Possible
- Pull Members Up
- Push Members Down
- Replace Inheritance with Delegation
- Inline Local Variable
- Inline Method
- Convert Anonymous Class to Inner
- Encapsulate Fields
- Replace Temp With Query
- Replace Constructor With Factory Method

Refactorings supported by the Java IDE IntelliJ IDEA

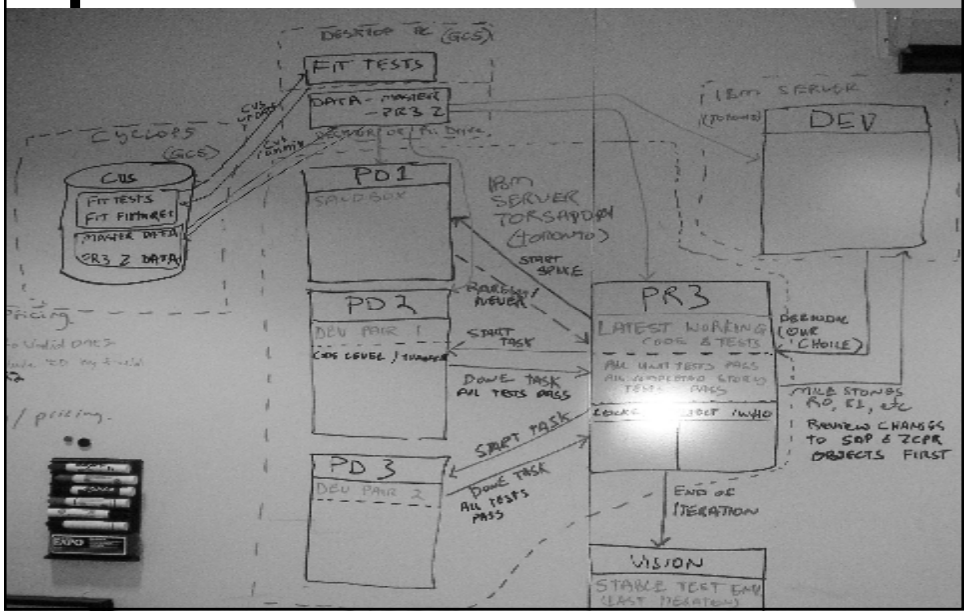
Challenges - Lack of OO Skills

- Few ABAPers have true Object-Oriented Programming (OOP) experience
  - Objects as glorified function modules
- Traditional development in older systems does not encourage OOP
  - Not available, poor performance, no objects inside
- Objects ‘scare’ many ABAP Developers
  - New concept, tools
  - Outside their comfort zone

Challenges - Lack of Unit Test Automation Skills

- Abap Developers’ First Exposure
  - Concept not fully understood initially so slow to pick up
- Little documentation on how to use AbapUnit
  - Documentation that is available is ‘clunky’.
  - Object Courses do not cover Abap Unit tests
  - AbapUnit hard to find in Development Environment

How We Surmounted Them – Concurrent Dev't



Taught/Mentored on the Job

- Leveraged experienced Java/.Net resources
  - Author of xUnit Test Patterns book was one of the team members
- Development team was open to learning
  - Mentored in Objects/Classes, Refactoring, Unit Testing, Test-Driven Development,
  - Many late night 'learning' sessions

Allowed For Several Attempts to Get it Right

- Refactoring was accepted as the price of learning.
- Manual Refactoring took much longer
- Had several “zero point iterations”

Agenda

- Introduction
- What is Agile?
- Why Did We Choose Agile?
- How Agile Delivers Value
  - Highly Communicative, Collaborative Team
  - Highly Incremental Story-Based Development
  - Continuous Learning
  - Continuous Feedback
  - Modern Development Practices
- Adopting Agile Practices
- Summary

Adopting Agile Practices (A.K.A. Teaching Old Dogs New Tricks)

Two Basic Strategies:

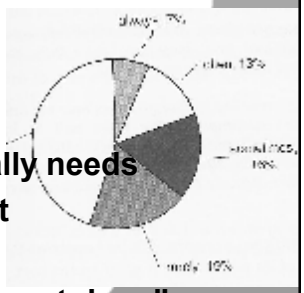
- **All at Once**
  - **What we did - because we had prior experience with agile (IT and business)**
- **Individual Practices**
  - **The addition of any agile practice can add value on a traditional project.**

Adopting Individual Practices

Practice	Individual	Sub-Team	Team	Enterprise
Test-Driven Development	Yes	Yes	Yes	Yes
Story-Based Dev't	No	Yes	Yes	Yes
Pair Programming	No	Yes	Yes	Yes
Daily Standup	No	Yes	Yes	Yes
Iteration Planning	No	Yes	Yes	Yes
Concurrent Development Environment	No	Yes	Yes	Yes

Summary (1 of 2)

- Agile practices add value by delivering applications users love to use
  - Agile and SAP are not at odds.
  - Can deliver highly usable applications that users love (in SAP); you just have to value this outcome



- Agile Practices Increase ROI by
  - Only building stuff business really needs
  - Reducing waste while building it

"Agile is the art of maximizing work not done"

Summary (2 of 2)

- Practices can be adopted as a group or individually
- Some practices are harder to adopt in SAP
  - Agile practices are technology independent
  - Some are a bit harder to implement than in Java/.Net but they are still doable

You *Can* Teach Old Dogs New Tricks!

Reference Books

- Agile Management for Software Engineering
  - Applying the Theory of Constraints for Business Results
  - David J. Anderson (Prentice Hall)
- Agile Retrospectives
  - Making Good Teams Great
  - Esther Derby & Diana Larsen (Pragmatic Programmers)
- Collaboration Explained
  - Facilitation Skills for Software Project Leaders
  - Jean Tabaka (Addison Wesley Professional)
- Fit for Developing Software
  - Framework for Integrated Tests
  - Rick Mugridge & Ward Cunningham (Prentice Hall)
- Paper Prototyping
  - The Fast and Easy Way to Design and Refine User Interfaces
  - Carolyn Snyder (Morgan Kaufmann)
- xUnit Test Patterns
  - Refactoring Test Code
  - Gerard Meszaros (Addison Wesley Professional)

How to contact us:

- Janice Aston:
  - janice@agileperspective.ca
- Gerard Meszaros:
  - asug2008@gerardm.com
- Glenn Mickelson:
  - grmickelson@shaw.ca