



## Agile ERP

You Don't Know What You've Got  
'Till It's Gone!



Janice Aston  
Project Manager  
Canadian Pacific  
info@agileperspective.ca

Gerard Meszaros  
Agile Coach  
clearStream Consulting  
gerard.meszaros@acm.org

Aug 13, 2007

## Company Background

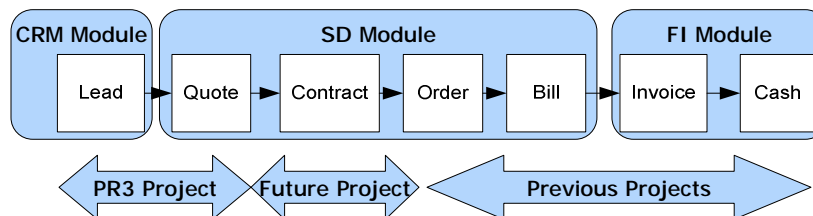
- Canadian Pacific is a very traditional railway company
  - 125 years since last spike driven on transcontinental rail line
- Embraces traditional waterfall or document-driven development practices
  - It works for laying track and building bridges!
- Agile is not widely used within CPR I.T.
- Until recently SAP confined to Finance & HR and managed under separate organizational unit

## Project Background

- Phase 2 of multi-year Pricing program
- Phase 1 (highly successful) built using .Net and eXtreme Programming
  - Storytest-driven Development using FIT
  - Paper Prototyping & Wizard of Oz testing
- Change in technology direction during planning phase

## Project Background – Why SAP?

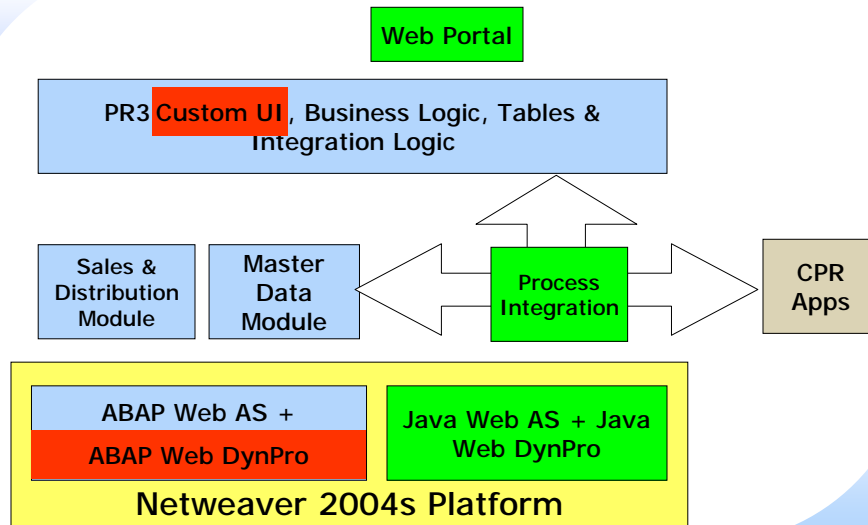
- CP adopted “SAP First” strategy between phases
  - Project revectorred to use SAP's SD module for Phase 2
- Project reported to non-SAP part of I.T.



## Why Custom Development in SAP?

- Business insisted on high usability
  - Business case was built on this!
- Significant gap between
  - "Transportation on Quote for Widgets"
    - A single "shipping" item
  - "Quote for Transportation of Widgets"
    - A number of alternate shipping scenarios each with multiple price components
- Considered building front end in .Net accessing SAP pricing engine
  - Decided it would result in too much duplication of effort

## Project Background – Architecture



## Agile Suitability Filter

### In Favour:

- Business was already in favour of Agile
- Core team preferred Agile
- SAP "Works Out of the Box"
  - SD Module implements quoting & Pricing
  - Can use "standard SAP" screens until custom UI is built
- ABAP Unit already existed

### Contra-indications (at launch):

- SAP community not familiar with Agile
  - Waterfall document-driven approach is considered Best Practice
- Agilists on project not familiar with SAP

### Other Challenges Encountered

- The subject of this paper!

## Our Agile Process

- Used fairly standard eXtreme Programming
  - 2 week iterations with IPM and Retrospective
  - Accepted Responsibility
  - Emergent design / refactoring (a challenge!)
  - Continuous Integration (a challenge!)
  - Unit Test-driven Development - mostly
  - Pairing - about 30-50% of the time
  - Onsite Customer plus several part-time customers
  - Storytest-Driven Development
    - Tests prepared by customer and executed manually
- Augmented by Usability Practices
  - As described in last year's paper

## Challenge Template

For each Challenge encountered, we describe

- **Agile assumptions:**
  - What, as .Net-based Agilists, we implicitly assumed (often without even realizing it!)
- **SAP Reality:**
  - What we encountered in the SAP world, how that differed from our assumptions and how it impacted our project.
- **Our Solution:**
  - How we addressed the newly discovered reality so that we could continue using agile development.

## Challenges: Product Centric View

Agile Assumptions:

- **We build what the customer wants**
  - The customer knows what they want, or
  - we help them figure out what they want
- **The customer makes business decisions**
  - Technical team provides data (e.g. estimates) required
- **Development Team makes technical decisions**
  - Based on business requirements

## Challenges: Product Centric View

### SAP Reality:

- SAP product has embedded Business Process
  - Considered "Best Practice" by SAP & some others
  - Expectation is everyone should adopt it
- SAP Product is Configurable by Customer
  - Data-driven extensions are preferred solution
  - Only allow predefined range of possibilities
  - Can be very difficult to extend outside this range
    - Even with custom development
- SAP not historically known for its usability

## Challenges: Product Centric View

### Our Solution:

- Recruited open-minded SAP resources
- Encouraged Negotiation ("N" in "INVEST")
  - Paper Prototypes were starting point for negotiation
  - SAP specialists could suggest alternate ways to achieve business goals
- Periodic Reality Checks in Retrospectives
  - Ensure we weren't straying too far from "standard SAP"
  - "Are we customizing too much?"

Usability

Cost



## Challenges: Specialization of Roles

### Agile Assumptions:

- **Generalists (Not very much specialization)**
  - Business vs technical only
  - Everyone signs up for work as needed
- **Develop Story by Story**
  - Requirements, Design, Code, Test within a few days
- **Single "velocity" for entire team**
  - Developers do whatever is needed by current stories
- **Stories provide Complete Functionality**
  - Includes any data, logic, UI etc.
  - Not much dependency management needed

## Challenges: Specialization of Roles

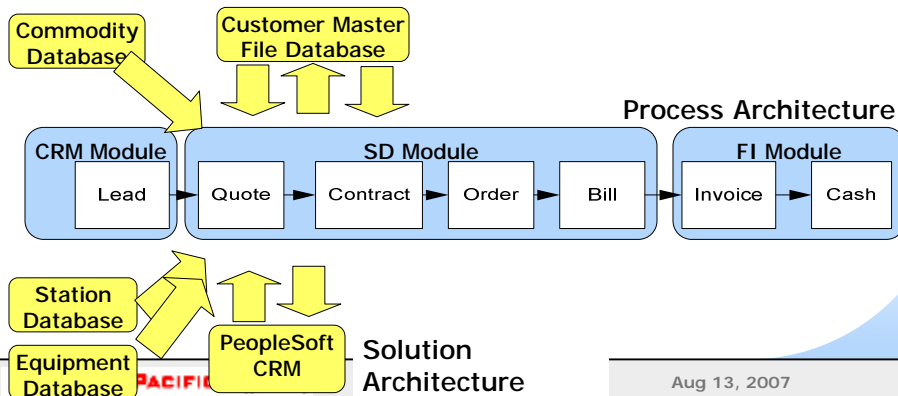
### SAP Reality 1-Specialists:

- **Functional Analysts are Product Specialists**
  - Document business process and configure modules
  - Tell Developers what tables to create and code to write
  - Not very technical
- **Developers are Technology Specialists**
  - They write code; don't typically talk to business
  - Specialize in 1 or 2 modules' API (e.g. SD and PI)

## Challenges: Specialization of Roles

### SAP Reality 2-Missing Skills:

- No UI designer or usability specialist
  - SAP products not known for their usability
- No Solution Architect or DBA
  - Architecture is provided by SAP as part of product



## Challenges: Specialization of Roles

### Our Solutions 1-Specialists:

- Separate Points & Velocity per speciality
  - WebApp Development Points
  - Configuration Points
  - Data Load & Integration Development Points
  - Focus on Critical Path backlog/burndown
- Separate Configuration vs Dev Stories
  - Typically did Config stories first
    - Tested via standard SAP Screens or FIT tests
  - Web UI development done afterwards
    - Business-friendly UI improved usability
    - For high-impact stuff only; infrequently used UI's left as SAP standard.

## Challenges: Specialization of Roles

### Our Solutions 2-Missing Skills:

- Staffed Solution Architect role
  - Experienced non-SAP architect
  - Dealt with overall system architecture
  - Delegated intra-SAP design
    - Configuration to functional analysts
    - Technical to senior developers
- Business & BA (non-SAP) did UI design and Story Definition
  - Built Paper Prototypes of all functionality
  - Wizard of Oz testing with real users
  - Defined User Stories from Paper Prototypes

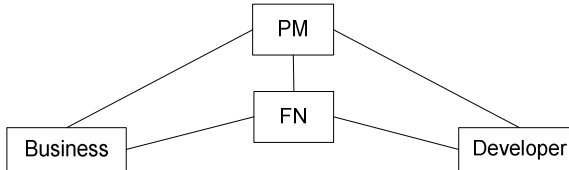
## Challenges: Communication & Collaboration

### Agile Assumptions:

- Everyone is in the know
  - Big picture
  - Shared vision
- Communication is high bandwidth
  - verbal, face to face
  - Business domain based vocabulary
- Minimize Documentation overhead
  - Just Enough
  - Just in Time
  - Minimize "Just Because"

## Challenges: Communication & Collaboration

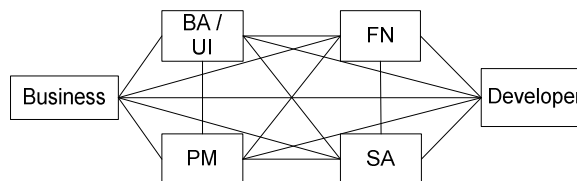
### SAP Reality:



- Hierarchical “Command & Control”
- Document-Driven process
  - Tendency to write docs/e-mails and “throw over the wall”
- Technical, product-specific vocabulary
  - Functional Analysts go-between business & developers
  - Developers not used to talking to business

## Challenges: Communication & Collaboration

### Our Solution:



- Insisted on strong face-to-face communication between all roles
  - Non-SAPers used “Secret Decoder Rings” to learn SAP
  - Added Business Analyst / UI Designer / Test Automater role
- Insisted on Deliverables each iteration
  - BDUF activities had to deliver something
  - Preferred real, testable configuration over documents
- Self-Managed Team
  - Everyone involved in planning work
  - Iteration Plan on Information Radiator

## Challenges: Server-Based Development

### Agile Assumptions:

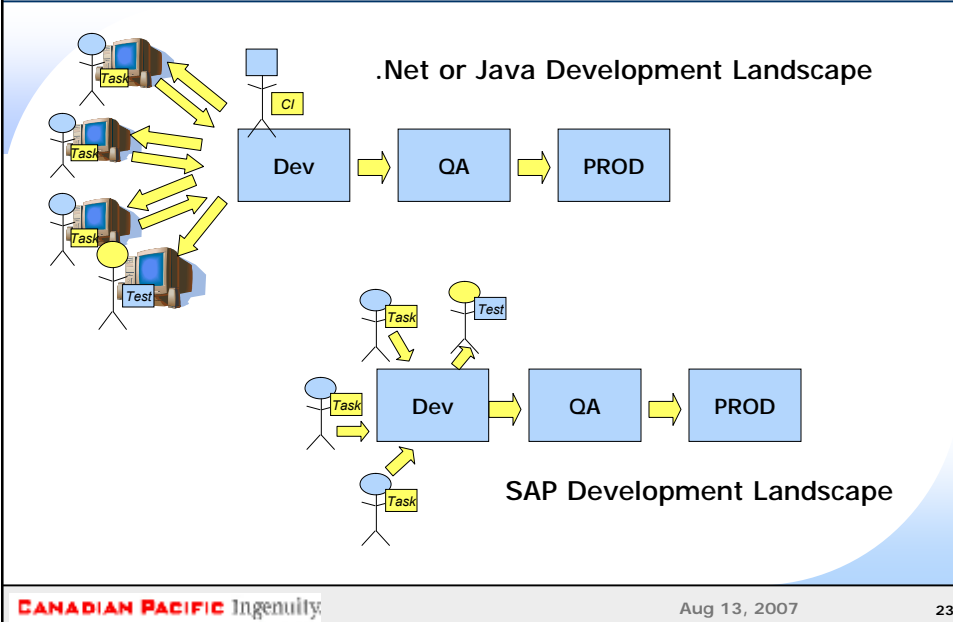
- Each Developer can work independently:
  - Sandbox per developer
  - Update from Source Code Repository
  - Make and test changes privately
  - Commit to Repository on Green Bar
- Optimistic Locking of Objects:
  - Make whatever changes are required
  - Resolve conflicts before check-in
  - Allows for the flexibility of working story by story

## Challenges: Server-Based Development

### SAP Reality 1-"Much Too Continuous Integ.":

- All development done in single DEV Server
  - Big Iron considered too expensive to replicate per team
  - High setup & management overhead
  - All code is stored in database on Server
  - "Transported" to QA server & on to PROD
- Developers use SAP GUI to change code on DEV Server
  - Code changes are visible to everyone as soon as they are made
  - ABAP Unit tests are constantly breaking
  - Development continually broken making incremental business validation impossible

## Challenges: Server-Based Development



## Challenges: Server-Based Development

### SAP Reality 2-Waterfall Mindset:

- Waterfall process & big bang integration
  - Requirements decomposed into functional & development work
  - Done by different people at different times
  - No single-point ownership of deliverable
- Pessimistic Locking to Avoid Code Conflicts
  - Developers each "own" modules / objects
  - Only they can change them
  - Forces "waterfall" style design/code/integration

## Challenges: Server-Based Development

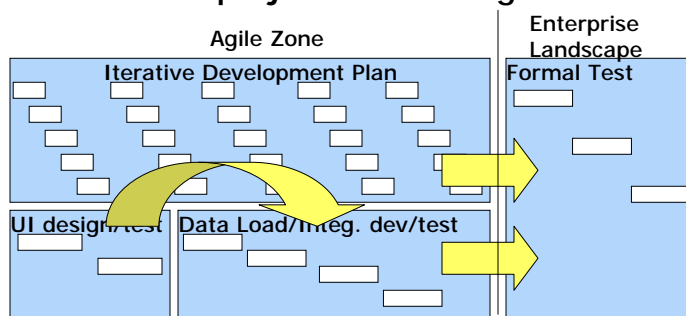
### Our Solution 1-"Much Too Continuous Integration": The "Agile Zone"

- Worked with SAP Canada to design non-standard dev landscape and process
  - Educated SAP on agile update-code/test-commit cycle
  - Based on Java/.Net-style environments per developer pair
  - Needed a lot of "selling" inside CP IT
- Procured dedicated (Non-virtual) server for team
  - 12 Week lead time!
- Implemented circular "Transport Paths" to allow Updates & Commits
  - Required extensive (& very progressive) support from SAP "Basis" personal
  - Developers had to learn to use "transport" mechanisms

## Challenges: Server-Based Development

### Our Solution 2-Waterfall Mindset:

- Plan intra-project-team work using agile methods;
  - Agile Zone allows Story-based work breakdown structure
- Plan cross-project work using traditional plans



## Challenges: SAP is an Integrated System

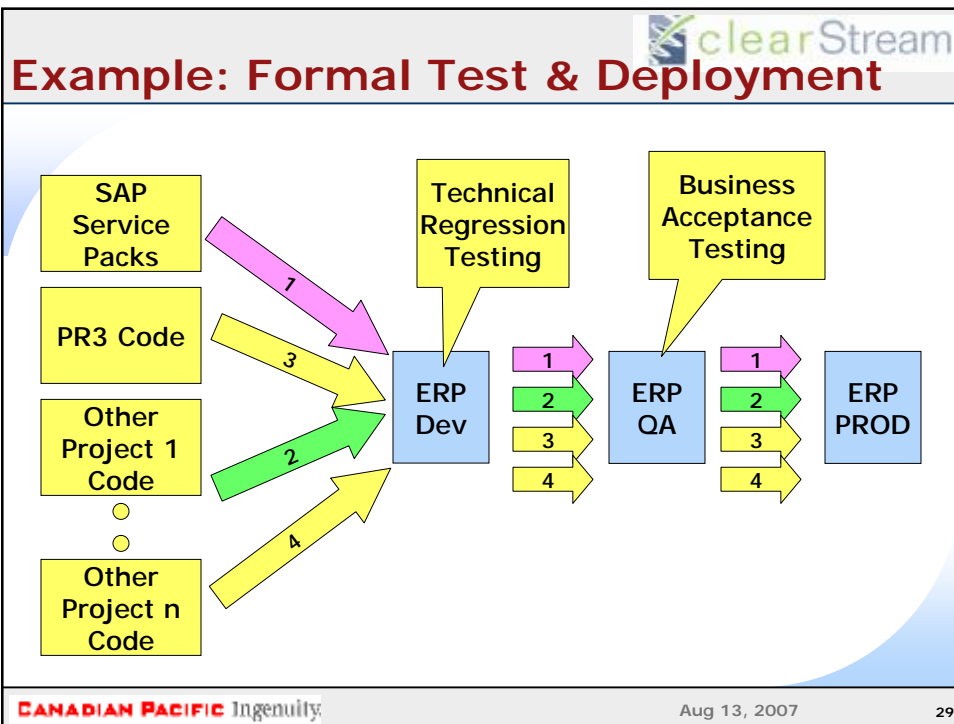
### Agile Assumptions:


- Project team is largely independent of other projects
  - Agile can “fly below the radar” if necessary
- Integration with other applications is via well-defined interfaces
- Inter-application Integration occurs when project team decides to integrate
  - e.g. when ready to Integration Test

## Challenges: SAP is an Integrated System

### SAP Reality:

- All occupants of SAP Module must co-exist
  - Everyone cares how you are changing shared code & data
  - Code/data is often shared for technical reasons
    - e.g. Quotes and Orders are stored in same table therefore changes to Quotes affect Orders!
  - Lots of FUD, requests for waterfall document deliverables
- Moving into QA & Production must be co-ordinated between projects
  - Transport dependencies constrain order of deployment
  - Regression testing req'd by all co-occupants





## Challenges: SAP is an Integrated System

**Our Solution:**

- Communicate what we're doing
  - multiple Architecture & Design Reviews
- Deploy less often
  - 5 releases over 1 year reduced to 3 releases over 1.5 years
  - Plan on long deployment interval
    - ~3 months Dev Complete to Deployed
- Deploy in Separate SAP client
  - to defer some data integration issues that the company wasn't ready to address
- Stay in Agile Zone as long as possible
  - Maintain control over our own destiny

CANADIAN PACIFIC Ingenuity Aug 13, 2007 30

## Challenges: Proprietary Tools

### Agile Assumptions:

- Object-Oriented Code is amenable to incremental development
- xUnit for automating unit tests
- FIT or Recorded Tests tools for functional tests
- Refactoring Tools keep cost of change low
- Continuous Integration enables periodic & rapid integration of work done in parallel

## Challenges: Proprietary Tools

### SAP Reality:

- Object-Oriented is new paradigm
  - Older parts of ERP inhibit incremental development
    - e.g. Each pricing conditions is "all or nothing"
  - Many SAP components can only be configured, not extended
- Clunky dev tools inhibit Refactoring
  - Pull Up/Push Down are only refactorings supported in ABAP dev tools (no Extract or Move Method)
  - Lost several iterations due to refactoring churn

## Challenges: Proprietary Tools

### SAP Reality:

- **Server-based Devt is “Much Too Continuous Integration”**
  - Had to build our own environment to separate
  - Had to build own CI tools to notify team of results
- **xUnit is available but rarely used and poorly doc'ted**
  - Took us several weeks of digging to figure out how to access it
- **Recorded Test tools didn't work with ABAP WebDynpro**
  - Neither SAP's eCATT nor Mercury/HP's QuickTest Pro
- **No text based access to code**

## Challenges: Proprietary Tools

### Our Solution:

- **On-the-job teaching/coaching by agilists**
  - Teach SAPers how to use objects
  - Teach SAPers how to refactor manually
- **Adjusted Story Granularity**
  - Some stories are larger than normal
  - More “Look Ahead” required to avoid unrefactorable tarpits
- **Adjusted Test Strategy**
  - Used FIT.Net and the SAP .Net Connector to automate pricing Story Tests
  - Scripted some Story Tests in ABAP Unit
  - Did manual Smoke/Acceptance testing of UI

## Final Thoughts

- **Would we use SAP again?**
  - Only if closer fit to business process
- **Would we custom build in SAP again?**
  - Yes, as long as the business value was there
- **Would we use Agile in SAP again?**
  - Yes, if there is support within the organization
  - Everyone is really impressed with the usability of the application
- **Would we change anything about how we did it?**
  - Absolutely!!

## Conclusions

- **ERP is a classic Agile-Hostile environment**
  - And has it's own, extra challenges
- **Agile development assumes a bunch of stuff**
  - Many assumptions are implicit
    - we don't realize we make them!
  - Many assumptions violated in SAP reality
- **Proprietary Toolset is Very Limiting**
  - Can work around technology limitations
  - but impacts productivity ...
- **PEOPLE matter most**
  - You can teach an old dog new tricks; it just takes longer!
- **Can deliver a superior product using Agile**
  - Even in SAP!

## Contact Information



Janice Aston  
Project Manager  
Canadian Pacific  
info@agileperspective.ca

www.cpr.ca

Gerard Meszaros  
Agile Coach  
clearStream Consulting  
gerard.meszaros@acm.org

www.clrstream.com  
www.gerardmeszaros.com

## References



- **User Stories Explained**
  - By Mike Cohn
  - Published by Addison Wesley
  - Ref: "INVESTing in good User Stories"
- **Adding Usability Testing to an Agile Project**
  - By Gerard Meszaros & Janice Aston
  - Agile 2006 Experience Report
- ***xUnit Test Patterns – Refactoring Test Code***
  - By Gerard Meszaros
  - Published by Addison Wesley 2007
- ***Fit for Development,***
  - By Rick Mugridge, Ward Cunningham,
  - Published by Addison Wesley 2005