Software Configuration Management to Enable Agility

Steve Berczuk
Cyrus Innovation

Boston SPIN
March 18, 2008

CyrusInnovation

About Me

- Software Developer
- Certified Scrum Master
- Author (SCM Patterns Book, CM Crossroads)
- Technical Lead, Cyrus Innovation (Boston)
- More Info
  - www.berczuk.com
  - www.scmpatterns.com
  - www.cyrusinnovation.com
  - steve@berczuk.com

CyrusInnovation

About Cyrus Innovation

- Offices in Boston and New York City
- Agile Software Development
- Coaching and Training

CyrusInnovation

Agenda

- **Agile: What, Why, and Why it's Hard**
- SCM: What, Why, and Challenges
- SCM Patterns
- Using SCM to Enable an Agile Approach

CyrusInnovation

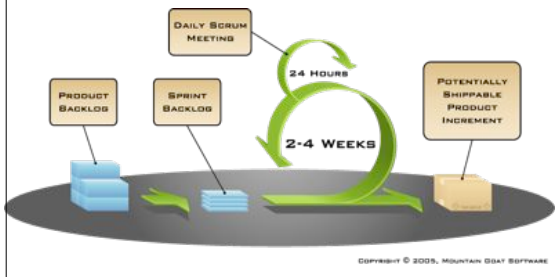
Agile Manifesto www.agilemanifesto.org

- *Individuals and Interactions* **over** Processes and Tools
- *Working Software* **over** Comprehensive Documentation
- *Customer Collaboration* **over** Contract Negotiation
- *Responding to Change* **over** Following a Plan

- People build software!
- Use the right tools and processes.
- Focus on things that add direct value.
- Adapt to change; acknowledge that change happens.
- (Common Sense Applied)

CyrusInnovation

Agile Process: Scrum



CyrusInnovation

Benefits of Agile Methods

- Easier to manage scope
- Build the right thing
- Deliver value more predictably

- Agile methods
  - Emphasize feedback and communication.
  - Avoid process steps that don't add value.
  - Address issues, don't just add processes for comfort.

Barriers to Agility

- Lack of Trust
- Fear of Change
- Process Gaps
  - Feedback mechanisms

Agenda

- Agile: What, Why, and Why it's Hard
- **SCM: What, Why, and Challenges**
- SCM Patterns
- Using SCM to Enable an Agile Approach

SCM

- SCM enables agility
  - Reproducible Workspaces
  - Feedback through builds*
- Concepts in context:
  - Branches, labels, tags
  - Builds
  - Workspaces
- Different levels of scale

Agile Context

- SCM is Part of the Puzzle:
  - Architecture
  - Software Configuration Management
  - QA/Testing
  - Culture/Organization

The Goal: Working software that delivers value.

Common (SCM) Problems

- Not Enough Process:
  - "Builds for me…"
  - "Works for me!"
  - "The build is broken again!"
  - "What branch do I use?"
- Process Gets in the Way:
  - Long Commit Times
  - Serialized Commits
  - Code Freezes
- Long integration times at end of project
  - "Fixing it" in integration
- Silos of Knowledge
  - "How does this code work?"

SCM as an Enabling Tool

- SCM gives you:
  - Reproducibility
  - Integrity
  - Consistency
  - Coordination
- SCM enables:
  - Increased productivity
  - Enhanced responsiveness to customers
  - Increased quality
- SCM done poorly can:
  - Slow down development
  - Frustrate developers
  - Limit customer options

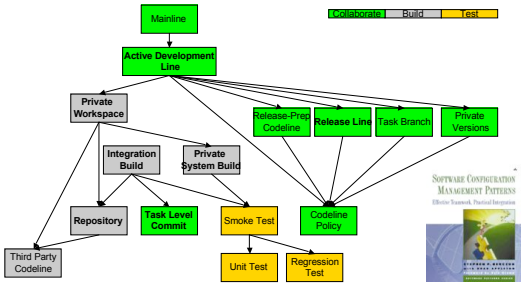
Feedback

- Agility requires feedback
- Mechanisms:
  - Build (Locally, Integration)
  - Unit Test/Integration Test
- Enablers:
  - Frequent Commits
  - Enough Testing

Agenda

- Agile: What, Why, and Why it's Hard
- SCM: What, Why, and Challenges
- **SCM Patterns**
- Using SCM to Enable an Agile Approach

The SCM Pattern Language



Mainline (Solution)

- When in doubt, do all of your work off of a single *Mainline*. Understand why you want to branch, and consider the costs.
- Unresolved:
  - Simplicity with speed and *enough* stability: *Active Development Line*

Active Development Line

- You are developing on a *Mainline*.
- **How do you keep a rapidly evolving codeline stable enough to be useful (but not impede progress)?**

Active Development Line

- Use an *Active Development Line*.
- Have check-in policies suitable for a "good enough" codeline.
- Establish practices to give feedback on the state of the codeline.
- Unresolved:
  - Doing development: *Private Workspace*
  - Keeping the codeline stable: *Smoke Test*
  - Managing maintenance versions: *Release Line*
  - Dealing with potentially tricky changes: *Task Branch*
  - Avoiding code freeze: *Release Prep Codeline*

Private Workspace

- You want to support an *Active Development Line*.
- **How do you keep current with a dynamic codeline and also make progress without being distracted by your environment changing from beneath you?**

Private Workspace

- Create a *Private Workspace* that contains everything you need to build a working system.
  - You control when you get updates.
- Before integrating your changes:
  - Update your workspace.
  - Build your workspace and Test your code and the system. (Private System Build)
- (Defer additional validations to the Integration Build)
- Have an automated way to create workspaces from a repository. (Repository)

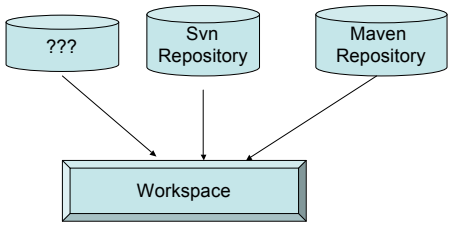
Repository

- *Private Workspace* and *Integration Build* need components.
- **How do you get the right versions of the right components into a new workspace?**

Repository (Solution)

- Have a single point of access for everything.
  - Use this mechanism at all levels (dev, integration build, etc)
  - No hard coded information.
- Have a mechanism to support easily getting things from the *Repository*.
  - Install Version Manager Client
  - Get Project from Version Management
  - Build, Deploy, Configure (Ant target, Maven goal)
  - Simple, repeatable process.
- Still to do:
  - Manage external components: *Third Party Codeline*

Repository



Active Development Line + Private Workspace + Repository

- Enable frequent demos of new features
- Add a new developer quickly.
- Create test environments.
- Create build environments.
- Reproduce problems quickly.
- Have an implicit check for inflexible configurations.

CyrusInnovation

Private System Build

- You need to build to test what is in your *Private Workspace*.
- **How do you verify that your changes do not break the system before you commit them to the *Repository*?**

CyrusInnovation

Private System Build (Solution)

- Build the system using the same mechanisms as the central integration build: a *Private Build*.
  - This mechanism should match the integration build.
  - Update to the codeline head and Build before checking in changes!
- Unresolved:
  - Testing what you built: *Smoke Test*

CyrusInnovation

Integration Build

- What is done in a *Private Workspace* must be shared with the world.
- **How do you make sure that the code base always builds reliably?**

CyrusInnovation

Integration Build (Solution)

- Do a centralized build for the entire code base.
  - Use automated tools: Cruise Control, SCM tool Triggers, etc
- Still Unresolved:
  - Testing that the product of the build still works: *Smoke Test*
  - Build products may need to be available for clients to check out
  - Figure out what broke a build: *Task Level Commit*

CyrusInnovation

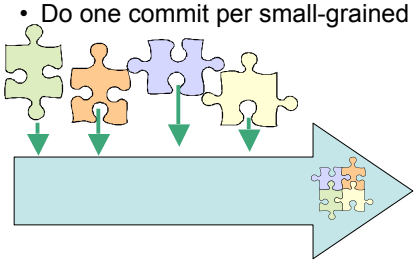
Task Level Commit

- You need to associate changes with an *Integration Build*.
- **How much work should you do before checking in files?**

CyrusInnovation

Task Level Commit (Solution)

- Do one commit per small-grained task.



CyrusInnovation

Release Line

- You want to maintain an *Active Development Line*.
- **How do you do maintenance on a released version without interfering with current work?**

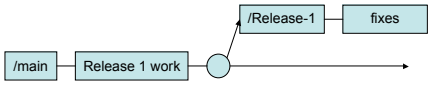
CyrusInnovation

Release Line (Solution)

- Split maintenance/release activity from the *Active Development Line* and into a *Release Line*.
- Allow the line to progress on its own for fixes.
- Propagate changes to Mainline as appropriate.

CyrusInnovation

Release Codeline



CyrusInnovation

…+ Build Patterns + Release Line + Task Level Commit

- Ability to reduce failed builds
- Ability to run more tests without delaying commits.
- Be cautious with released code but be more agile with new dev.
- Make it easier to back out when a mistake slips by.

CyrusInnovation

Agenda

- Agile: What, Why, and Why it's Hard
- SCM: What, Why, and Challenges
- SCM Patterns
- **Using SCM to Enable an Agile Approach**

CyrusInnovation

SCM for Agile Teams?

- *Individuals and Interactions* over Processes and Tools
  - SCM Tools that support the way that you work, not the other way around
- *Working Software* over Comprehensive Documentation
  - SCM can automate development policies & processes: Executable Knowledge over Documented Knowledge
- *Customer Collaboration* over Contract Negotiation
  - SCM facilitates communication among stakeholders and help manage expectations
- *Responding to Change* over Following a Plan
  - SCM mechanisms used for facilitating change, not preventing it

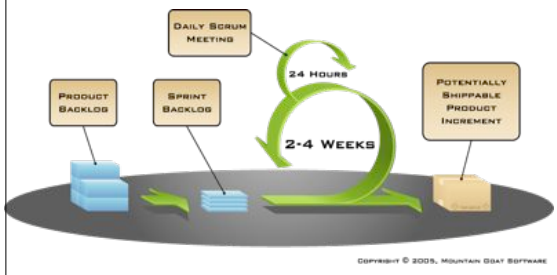
CyrusInnovation

Creating an Agile SCM Environment

- Decide on a goal.
- Choose an appropriate Codeline Structure
  - set up the related policy.
- Create a process to set up workspaces
  - Private
  - Integration
- Build & Deploy is an Iteration 0 Story.
- Integrate frequently at all levels
  - Developer Workspace
  - Integration Build
- Deploy frequently.
- Test.

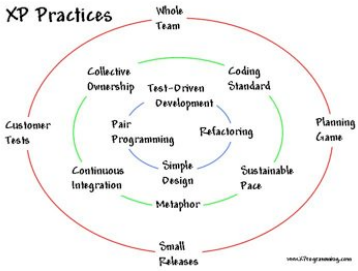
CyrusInnovation

SCM and Scrum



CyrusInnovation

SCM and XP



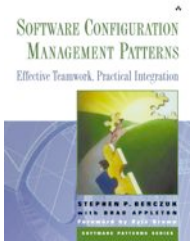
CyrusInnovation

Agile Results

- More frequent Deliveries
- Fewer Surprises
- Happier Clients

CyrusInnovation

The SCM Patterns Book



- Pub Nov 2002 By Addison-Wesley Professional.
- ISBN: 0201741172
- www.scmpatterns.com
- www.berczuk.com

CyrusInnovation

Resources

- www.cmcrossroads.com
- www.scmpatterns.com

CyrusInnovation

Questions?

CyrusInnovation