Collaborate, Integrate, Test, Deploy: Essential SCM Practices for Teams

Steve Berczuk
Sr. Software Engineer
Fast Search and Transfer, Inc.
Boston, MA
steve@berczuk.com

fast

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Agenda & Goals

- Agenda
  - ❑SCM and The Development Process
  - ❑SCM Concepts
  - ❑SCM Patterns for a More Agile Team
  - ❑Questions
- Goals:
  - Discuss some common problems.
  - Learn how taking a "Big Picture View" of SCM will you make your process more effective.
  - Understand how working with an Active Development Line model simplifies your process

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Opening Questions

- What is SCM?
  - Version Management
  - Configuration Identification
  - Anything Else?
- Why do We do SCM?
  - Control?
  - Adaptability?
  - Robustness?
- Who does SCM?
  - Release Engineers?
  - Developers?
  - Customers?

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

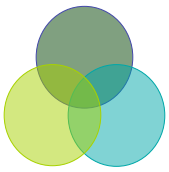
The Problem: Ineffective SCM

- Not Enough Process:
  - "Builds for me…"
  - "Works for me!"
  - "The build is broken again!"
  - "What branch do I work off of?"
- Process Gets in the Way:
  - Pre-check-in testing takes too long
  - Code Freezes
- Long integration times at end of project
  - "Fixing it" in integration
- Silos of Knowledge
  - "I don't know how this code works"

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

The Context

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



The Goal: Working software that delivers value.

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Solution

- An Agile Approach to SCM
  - Effective (not Unproductive) SCM
  - Agile Manifesto Principles applied to SCM
- The SCM Pattern Language
  - A Pattern Language to help you realize an Agile SCM Environment

Traditional View of SCM

- Configuration Identification
- Configuration Control
- Status Accounting
- Audit & Review
- Build Management
- Process Management, etc

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Agile/Effective SCM

- Who?
- What?
- When?
- Where?
- Why?
- How?

Focus on how processes add value.

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

What is *Agile SCM?*

- *Individuals and Interactions* over Processes and Tools
  - SCM Tools should 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 should facilitate communication among stakeholders and help manage expectations
- *Responding to Change* over Following a Plan
  - SCM is about facilitating change, not preventing it

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Agility and Transparency

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

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

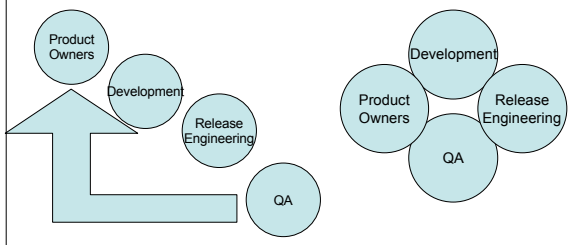
What Agile SCM is Not

- Lack of process
- Chaos
- Lack of control

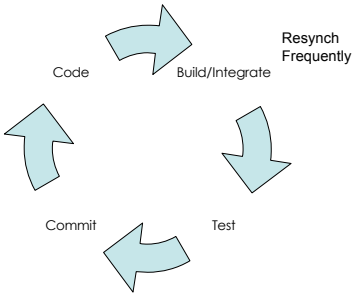
Agile SCM is about having an Effective SCM process that helps get work done.

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Feedback and The Team



The Agile SCM Cycle



SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Agenda

- Agenda
  - ✓SCM and The Development Process
  - ❑SCM Concepts
  - ❑SCM Patterns for a More Agile Team
  - ❑Questions

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

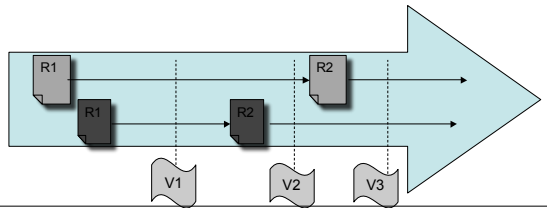
SCM Definitions

- Codeline/Branch
- Versioning Concepts
  - Configuration
  - Version
  - Revision
  - Label
- Workspace

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Codeline

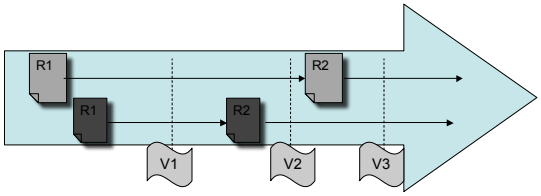
- A **codeline** contains every version of every artifact over one evolutionary path.



SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Versions, Revisions, Labels

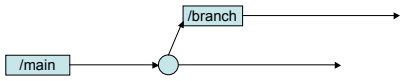
- Revision: An element at a point in time
- Configuration: A snapshot of the codeline
  Version: A *labeled* configuration



SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Branches

- **Branch**: A *codeline* that contains work that derives (and diverges) from another codeline.
- **Branch** of a file: A revision of a file that uses the trunk revision as a starting point.



Workspace

- Everything you need to build an application:
  - Code
  - Scripts
  - Database resources, etc

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Agenda & Goals

- Agenda
  - ✓SCM and The Development Process
  - ✓SCM Concepts
  - ❑SCM Patterns for a More Agile Team
  - ❑Questions

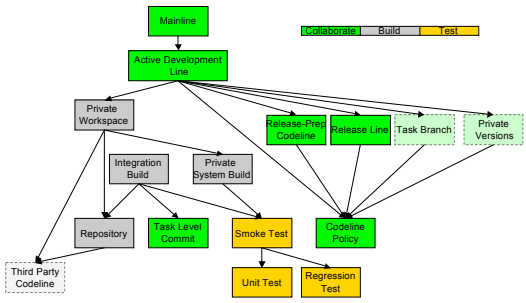
SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Creating an Agile SCM Environment

- Decide on a goal.
- Choose an appropriate Codeline Structure and 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.

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

The SCM Pattern Language



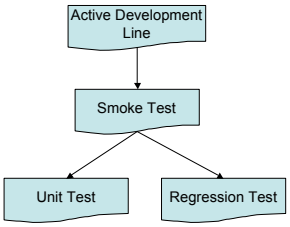
SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Patterns and *Pattern Languages*

- A *pattern* is a solution to a problem in a context
- Patterns capture common knowledge
- *Pattern languages* guide you in the process of building something using patterns
  - Each pattern is applied in the correct way at the correct time

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Context



- *Smoke Test* "completes" *Active Development Line*
- *Smoke Test* applies in the context of *Active Development Line*
- Arrows point from context to the "next" pattern

Mainline

- You want to simplify your codeline structure.
- **How do you keep the number of codelines manageable (and minimize merging)?**

SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

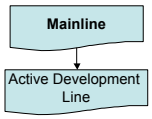
Mainline (Forces & Tradeoffs)

- A Branch: tool for isolating yourself from change.
- Branching can require merging.
  - Merging can be difficult.
- Separate codelines: a way to organize work.
- Integration with everyone's work is required.
- You want to:
  - maximize concurrency
  - minimize problems cause by deferred integration.

SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Mainline (Solution)

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



SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

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)?**

SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

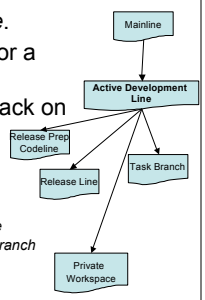
Active Development Line (Forces)

- A Mainline is a synchronization point.
- More frequent check-ins are good.
- A bad check-in affects everyone.
- If testing takes too long: Fewer check-ins:
  - Human Nature
  - Time
- Fewer check-ins slow a project's rhythm.

SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

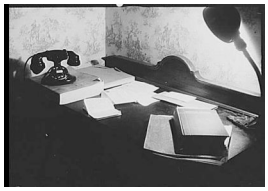
Active Development Line

- 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?**



SD BEST PRACTICES INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Private Workspace (Forces)

- Frequent integration helps avoid working with old code.
- People work in discrete steps: Integration can never be "continuous."
- Sometimes you need different code.
- Too much isolation makes life difficult.

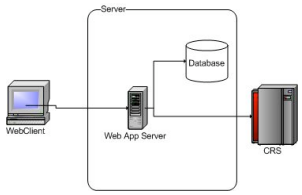
SD BEST PRACTICES INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Private Workspace (Solution)

- Create a *Private Workspace*
  - It contains everything needed to build a working system.
  - You control when you get updates.
- Before integrating your changes:
  - Update your workspace.
  - Build your workspace.
  - Test your code and the system.

SD BEST PRACTICES INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Private Workspace Example

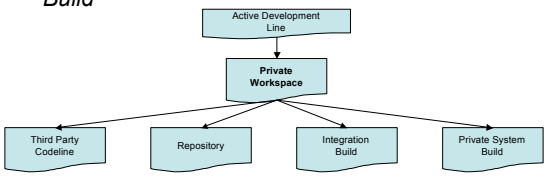


- Workspace
  - App Server
  - Database Schema
  - Code for Web App
  - Test CRS Login
  - (Build/Deploy and Configuration Tools & Scripts)

SD BEST PRACTICES INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Private Workspace (Unresolved)

- Populate the workspace: *Repository*
- Manage external code: *Third Party Codeline*
- Build and test your code: *Private System Build*
- Integrate your changes with others: *Integration Build*



SD BEST PRACTICES INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Repository

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

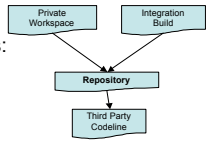
Repository (Forces & Tradeoffs)

- Many things make up a workspace:
  - Code
  - Libraries
  - Scripts.
- You want to be able to easily build a workspace from nothing.
- Components could come from a variety of sources (3rd Parties, other groups, etc).

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Repository (Solution)

- Have a single point of access for everything.
- 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.
- Unresolved:
  - Manage external components: *Third Party Codeline*



SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Types of Tests

Common Name	Author	Created	Isolation	Purpose
Unit/Programmer	Developer	During Unit Dev	High	Testing functional components
Smoke (Integration)	Developer QA	"Integration"	Low	Verify minimal operation.
Regression	Support QA Developer	Post Release	Low	Verify that problems do not resurface

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Smoke Test

- You need to verify an *Integration Build* or a *Private System Build* so that you can maintain an *Active Development Line.*
- **How do you verify that the system still works after a change?**

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

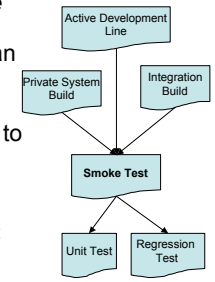
Smoke Test (Forces)

- Exhaustive testing is best for ensuring quality.
- Longer tests imply longer check-ins
  - Less frequent check-ins.
  - Baseline more likely to have moved forward.
- People have a need to move forward.
- Stakeholders have a need for quality and progress.
- Test Execution Time is often idle time.

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

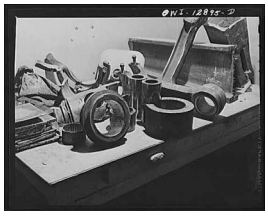
Smoke Test (Solution)

- Subject each build to a *Smoke Test* that verifies that the application has not broken in an obvious way.
- A *Smoke Test* is not comprehensive. You will need to find:
  - Problems you think are fixed: *Regression Test*
  - Low level accuracy of interfaces: *Unit Test*



Unit Test

- A *Smoke Test* is not enough to verify that a module works at a low level.
- **How do you test whether a module still works after you make a change?**



SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

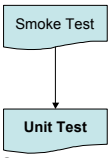
Unit Test (Forces)

- Integration identifies problems
  - But makes it harder to isolate problems.
- Low level testing is time consuming.
- When you make a change to a module you want to check to see if the module still works before integration
  - You want to isolate problems.

SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Unit Test (Solution)

- Develop and run *Unit Tests*
- *Unit Tests* should be:
  - Automatic/Self-evaluating
  - Fine-grained
  - Isolated
  - Simple to run
- Also known as *Programmer Tests*
  - J.B. Rainsberger



SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Regression Test

- A *Smoke Test* is good but not comprehensive.
- **How do you ensure that existing code does not get worse after you make changes?**

SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

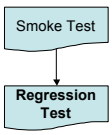
Regression Test (Forces)

- Comprehensive testing takes time.
- It is good practice to add a test whenever you find a problem.
  - You can't anticipate everything.
- When an old problem recurs you want to be able to identify when this happened.

SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Regression Test (Solution)

- Develop *Regression Tests* based on test cases that the system has failed in the past.
- Run *Regression Tests* whenever you want to validate the system.
- You can run these tests as part of an automated Integration build (nightly or more frequently).



Release Line

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

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

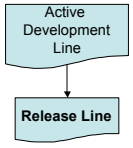
Release Line (Forces)

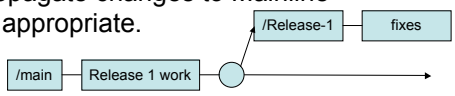
- A codeline for a released version needs a *Codeline Policy* that enforces stability.
- Day-to-day development will move too slowly if you are trying to *always* be ready to ship.

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Release Line (Solution)

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

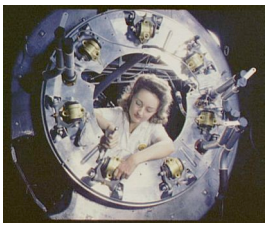




SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

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*?**



SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

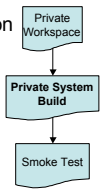
Private System Build (Forces)

- Developer Workspaces have different requirements than the system integration workspace.
- The system build can be
  - Complicated.
  - Time Consuming.
- Checking things in that break the *Integration Build* is bad.

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Private System Build (Solution)

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



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?**

SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

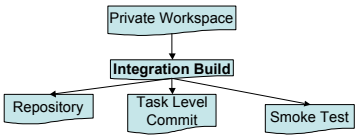
Integration Build (Forces)

- People do work independently.
- *Private System Build*s are a way to check the build.
- Building everything may take a long time.
- You want to ensure that what is checked-in works.

SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Integration Build (Solution)

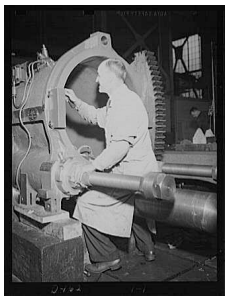
- Do a centralized build for the entire code base.
  - Use automated tools: Cruise Control, SCM tool Triggers, etc
- Still Unresolved:
  - Testing that the product of the build still works: *Smoke Test*
  - Build products may need to be available for clients to check out
  - Figure out what broke a build: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



SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Task Level Commit

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



SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

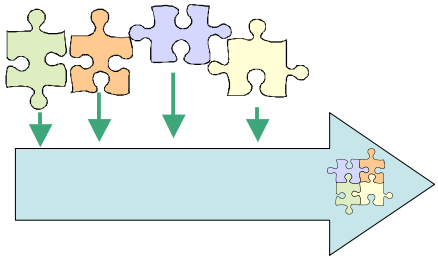
Task Level Commit (Forces)

- The smaller the task: easier roll back.
- A check-in requires some work.
- It is tempting to make many small changes per check-in.
- You may have an issue tracking system that identifies units of work.

SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Task Level Commit (Solution)

- Do one commit per small-grained task.



Codeline Policy

- *Active Development Line* and *Release Line* (etc) need to have different rules.
- **How do developers know how and when to use each codeline?**

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

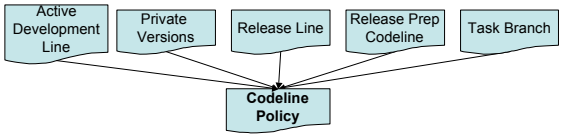
Codeline Policy (Forces)

- Different codelines have different needs, and different rules.
- You need documentation.
  - But how much?
- How do you explain a policy?

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Codeline Policy (Solution)

- Define the rules for each codeline as a *Codeline Policy*.
  - The policy should be concise and auditable.
- Consider tools to enforce the policy.



SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

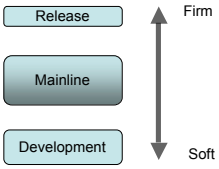
Sample Codeline Policies

- Active Development Line
- Release Line
- Other

SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Policies: The Tofu Scale

- Consider:
  - How close software is to being released.
  - How thoroughly must changes be reviewed and tested.
  - How much impact a change has on schedules.
  - How much a codeline is changing.
- See *Practical Perforce* Laura Wingerd Perforce Software) for more info



SD BEST PRACTICES CONFERENCE & EXPO 2006 INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Release Prep Codeline

- You want to maintain an *Active Development Line*.
- **How do you stabilize a codeline for an imminent release while allowing new work to continue on an active codeline?**

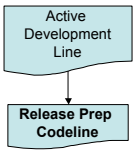
Release-Prep Codeline (Forces)

- You want to stabilize a codeline so you can ship it.
- A code freeze is the traditional approach
  - Slows rhythm too much.
- Branches have overhead.

SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Release Prep Codeline (Solution)

- Branch instead of freeze.
  - Create a *Release Prep Codeline* (a branch) when code is approaching release quality.
- Leave the *Mainline* for active work.
- The *Release Prep Codeline* becomes the *Release Line*
  - Release line has a stricter policy.



SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Essential SCM Practices

- Frequent feedback on build quality and product suitability through:
  - Version Management
  - Release Management
  - Build Management
  - Unit & Regression Testing
- These steps enable agility.

SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Resources/Places to Go

- www.scmpatterns.com
- www.berczuk.com
- www.cmcrossroads.com
- steve@berczuk.com
- *Software Configuration Management Patterns:Effective Teamwork, Practical Integration*

SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Other Books of Interest

Pragmatic Version Control Using Subversion 2ed

by Mike Mason

Pragmatic Version Control Using CVS

by Andy Hunt & Dave Thomas

JUnit Recipies

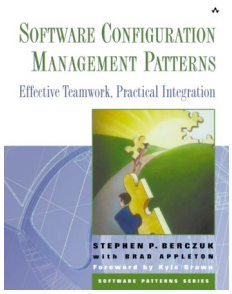
by J. B. Rainsberger

Pragmatic Project Automation

by Mike Clark

SD BEST PRACTICES 2006 CONFERENCE & EXPO
INNOVATE • ADVANCE • SUCCEED
SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

The SCM Patterns Book



- Pub Nov 2002 By Addison-Wesley Professional.
- ISBN: 0201741172

SD 2006 BEST PRACTICES CONFERENCE & EXPO

INNOVATE • ADVANCE • SUCCEED

SEPTEMBER 11-14, 2006 • HYNES CONVENTION CENTER, BOSTON, MA

Questions?