Branching and Merging with Team Foundation Server 2010

Pieter Gheysens
Team System MVP
Certified Scrum Master
User Group Lead VISUG

What I expect you to know for this session

• Basic understanding of version control
• Familiarity with a version control system
Agenda

- Version Control
- Branching
- Merging
- Branch Plans
- Demo "Branching & Merging"
  - Branching Visualization in TFS2010
  - Tracking Changesets
  - Basic Branch Plan Scenario
  - Versioning with Team Build
- Best Practices
- Takeaways
- Q & A

What is Version Control?

- Version control is a repository of files (mostly source code) with monitored access
- Keeps track of which changes were made
  - WHO
  - WHAT
  - WHEN
  - WHY
- Uses basic concepts like check-out, check-in, get latest, labeling, ...
- Foundation of the software development lifecycle
Branching

- Branching enables parallel software development activities
  - Implement different features on branches for same codebase
  - Maintain different releases in branches
- Branching = ISOLATION
- Most common Branching patterns
  - Branch by Release
  - Branch by Feature
  - Branch by Team
- First rule for branching: do NOT branch ... Keep it simple!
- Branching ≠ Labeling

Merging

- Merging allows you to move changes from one branch to another
  - Forward Integration (FI)
  - Reverse Integration (RI)
**Ubiquitous Language**

- *Development* branch: active development changes for next release(s)
- *Main* branch: junction branch between Development and Release
- *Release* branch: the bits that go into production
- *Hotfix*: change to fix a specific blocking bug or service disruption
- *Service Pack*: collection of hotfixes (+ features) for previous release
- *Forward Integration*: merge from parent branch to child branch
- *Reverse Integration*: merge from child branch to parent branch
- *Baseless merge*: merge between branches with no direct relationship

---

**Basic Branch Plan [1 dev branch]**

- concurrent development for next releases
- stable Main Branch for testing
- Release Branch for Bug Fixes
Basic Branch Plan [2 dev branches]

- Execute a FI Merge (Main to Dev) before a RI Merge (Dev to Main)
- Resolve Merge conflicts in the Dev branches as early as possible
- Avoid direct check-ins on the Main branch

Best Practices for (multiple) Dev branches
ALM with Visual Studio 2010

Basic Branch Plan Scenario

DEMO
Demo Summary

- Branching Visualization
- Track individual changeset across branches
- Branch properties
- Why Basic Branch Plan?
  - Supports maximum number of scenarios
  - Minimal number of branches
  - Handles complex situations
- Version Control History
  - Fine-grained permissions
  - Team Builds / branch + versioning

Extension 1: Standard Branch Plan
Extension 2: Advanced Branch Plan

When MAIN is ready to release, create the SERVICE PACK, HOT FIX, and RELEASE branches at the same time.

The two DEV branches are created as sequential tasks, but as one unit of work.

The RTM branch is a read-only copy of what was released.
Which branch plan to choose?

Basic
Small Team
Support for Major Releases
3 branches

Standard
Medium Team
Support for Service Packs
4 branches

Advanced
Large Team
Support for Hotfixes
5 branches

Visual Studio TFS Branching Guide 2010 at Codeplex
http://tfsbranchingguideiii.codeplex.com/

Best Practices

• Use meaningful branch names where applicable
• Always prefer Branching over “code-freeze”
• Integrate early and often to avoid “bing bang merge”
• Avoid cascading branches
• Preserve the physical integrity of the branch
• Isolate Change
• Isolate Work, not People
• Avoid baseless merges
Takeaways

- Branching Visualization with TFS2010 will be a big help in understanding parallel development
- TFS2010 can be installed on a client OS: ideal for experimenting with Version Control, Work Item Management and Build Automation!
- Invest time in setting up your branching strategy
- Be aware of the costs of branching! Keep it simple!
- It’s all about discipline! Stick to the right process to avoid merge hell!
- Merging requires skill and knowledge. Tooling cannot resolve everything!
- Provide different hardware environments for your branches
- Learn from your mistake(s)

Resources

- Visual Studio 2010 RC
- Visual Studio TFS Branching Guidance 2010 (ALM Rangers)
- Slot Mode operation for Version Control in TFS2010 [Blog Matt Mitrik]
- Custom Workflow Activities for Team Build 2010 [Blog Jim Lamb]
- Branching and Merging Anti-Patterns
- Into VSTS [My Blog]
  [http://www.intovsts.net](http://www.intovsts.net)
Q & A

Contact

pieter.gheysens@sparkles.be
www.sparkles.be
www.intovsts.net