

- ▶ WSI is organized as business units
  - Products and Services to a customer base
  - Broad breath of offerings
- ▶ Heterogeneous development environment
- ▶ Different source control systems
  - CVS
  - VSS
  - ClearCase
  - SCCS
  - Perforce
- ▶ Most repositories have over 5 years of history

## ▶ Practical Operational Limits

- VSS - branching, database size, and corruptions
- ClearCase - end-of-life version, significant new hardware investment

## ▶ Business units

- Practices strongly influenced by SCM tool in use
- Limited commonality to process vocabulary
- Gains in efficiency and productivity difficult to share
- Limits to sharing infrastructure solutions and source code across business units

- ▶ More agile business units
  - Robust remote access capabilities
  - Company-wide build automation tools
  - Share standard software functionality
  - Share process and procedure
  - Reduced overall cost of ownership

- ▶ Corporate initiative
- ▶ Review several SCM and bug tracking systems
  - Support for Development Processes
  - Integrated Solution
  - Cost of ownership (license and maintenance)
- ▶ Outside Consultant
  - SCM migration
  - End user training
  - Tool integration
  - Build automation

## ▶ Reality Challenges

- Limited Perforce familiarity
- Maintaining ongoing development
- Different Business Unit migration windows
- Repository migration times
- Hand-entered defect information

## ▶ Corporate Growth Challenges

- Changes to internal network traffic
- Changes to external network access
- Changes to corporate DMZ

- ▶ Business unit Advocate
  - Volunteers
  - Represent needs of their Business Unit
- ▶ Project Champion
  - Resolve issues
  - Arbitrate disagreements
  - Remove road blocks
- ▶ Expert Guidance
  - Legacy to Perforce representation
  - Translation of vision
  - Identify conflicting organizational requirements

- ▶ Group and individual meetings
  - Artifact migration requirements
  - Migration validation
- ▶ Advocate time investment
  - Minimum was a few meetings
  - Directly related to ease of migration
  - Up to date written plans
- ▶ Avoiding typical cross-organizational problems
  - Advocates have ownership
  - Clear understanding of end goals
  - Awareness of evolving commonality

- ▶ No advocate had significant Perforce experience
- ▶ Focused user and administrator training
  - Perforce mechanics
  - Perforce concepts
- ▶ Specific training emphasis
  - Workspace management
  - Branching
  - Changelists

- ▶ Assess value of migration artifacts
- ▶ Assess legacy support needs
- ▶ Assess impact on tools
- ▶ Assess impact on delivery mechanisms
- ▶ Establish a migration validation procedure
- ▶ Establish a “go live with Perforce” strategy

- ▶ Most common adjustment
  - Fewer “critical” artifacts
- ▶ “Critical” artifacts specific to tool in use
  - Intermediate labels
  - Labels as artificial branches
  - Shared files
- ▶ Integrity of historic information
- ▶ Do not migrate potentially erroneous historic information

- ▶ Historic artifact value diminishes with relation to
  - Age
  - Frequency of use
  - Frequency of change
- ▶ WSI's core artifact histories go back 12 or more years
  - Long elapsed migration time
  - Significant amounts of information with limited usefulness
  - Supporting customers with long acceptance cycles

- ▶ Migration approaches considered
  - Tip-only
  - “Snapshots” in time
  - Full history
- ▶ Incremental migration benefits
  - Eliminate most time and dependency issues
  - Baseline migrations completed and validated as background activities in advance of “go live” events
  - High probability of a successful, short, final migration
  - “go live” scheduling flexibility
  - Resource scheduling flexibility

- ▶ Realities of conversion
  - Cross-system equivalence
  - Deterministic metadata extraction
  - Historic validation
  - Source corruption
- ▶ Managing user uncertainty
  - Read-only access to the old systems
  - Legacy has “disconnected-tips”
  - Current development has guaranteed tip equivalence

- ▶ Perform server on a Linux platform
- ▶ CVS and ClearCase already case sensitive
  - Unintentional sensitivity
- ▶ Windows based client challenges
  - Workspace directory elements
  - VSS project directory elements
  - Visual Studio solution files
- ▶ Addressing the Windows client challenge
  - Case normalization script – Visual Studio files
  - Workspace character consistency validation script
  - Case consistency validation trigger

- ▶ Web project challenges
  - Character case consistency
  - VSS project mappings
  - VSS project mapping to workspace script
- ▶ Standard practices
  - Web project specific master template workspace
  - “Validate workspace view against master”
  - “Validate workstation against workspace view”
- ▶ Perforce workspace upside
  - Deterministic client placement
- ▶ Technical Note NOTE064

- ▶ Historic validation can be difficult and time consuming
  - Source refactoring
  - SCM representation differences
  - Migration inconsistencies
  - Comparison baseline validity
- ▶ Why not use original SCM systems for legacy support?
  - Cross-system change propagation
  - Long term system access
  - Long term user training

- ▶ Manage branch as if working disconnected
  - Procedural details: Technical Notes NOTE002, NOTE015
- ▶ Technique details
  - Baseline version created by integration
  - Source selection impacts usability of historic information
  - Source selection does not impact usability of branch
  - First workspace overwrite is legacy SCM system
  - Alternative overwrite from other sources
  - Remove legacy SCM support files
- ▶ Results
  - Best case - new tips are identical to the branched sources
  - More realistic – minor source differences

- ▶ Significant factors
  - File versions converted
  - SCM command overhead
- ▶ Small test conversions to deal with
  - Processor overhead
  - Network overhead
  - Script overhead
- ▶ General Wildcard: large text files with many versions
- ▶ ClearCase wildcard: VOB labels
- ▶ Observed
  - VSS – 35 hours (3 databases, 150K file versions)
  - CVS – 25 hours (15 repositories, 176K file versions)
  - ClearCase – 240 hours (29 VOBs, 200K file versions)
  - “go live” – 15 minutes per repository

- ▶ Optional Refactoring
  - In legacy prior to migration
  - As integral part of migration
  - Refactor post-migration
- ▶ Different sources, different refactoring
  - No refactoring – retain existing branching structure
  - Forced mainline – sources don't have branching
  - Optional refactoring – refactor post-migration from a migration branch

- ▶ File types
  - Trusted source types
  - Perforce equivalent for source type
  - Typemap entries
  - CVS brought over without keyword expansion
  - Post-migration “type check”
- ▶ Checked out files
  - People “forgot” there was a migration
  - Migration forces decisions
- ▶ Labels
  - Post-migration value
  - Validation
  - “Disconnected-tips” strategy

- ▶ Source version translation
  - Created during migration
  - Critical to label creation
  - Post-migration value
- ▶ Synthetic changelists (combining source checkins)
  - CVS and VSS used synthetic
  - ClearCase migrated each file version as a changelist adding version information to submit description
- ▶ Large (500K+) text files with 100s of versions
- ▶ Identifying migrated sources
  - Migration-specific workspaces
  - Submit comments have “migration” identification

- ▶ Repository files with additional random bytes
  - Histories report correctly
  - CVS update commands accessing first version terminate
- ▶ Labels referencing non-existent file versions
- ▶ Origin of problems?
  - Single developer
  - CVS interfaces not based on common distribution

- ▶ Dealing with corruptions
  - Isolation was not an option
  - Corruptions during metadata extraction
  - Corruptions during file version access
- ▶ Restoring a structure using different character case
- ▶ Dealing with changes to history
  - Migrate to tip equivalence

- ▶ Biggest issues
  - Directory versioning
  - Workspace management
  - Branching
  - Evil twins
- ▶ Dealing with dynamic branching
  - Attempt to recreate
  - Branch unique files
  - Disconnected-tips
- ▶ Dealing with evil twins
  - No obvious selection rules
  - Tip equivalence

- ▶ Over 20 years of SCCS sources
  - Recreating a sequence of file versions
  - Selecting the appropriate version sequence
  - “Best choice” histories
  - Disconnected-tips
- ▶ Perforce to Perforce
  - Windows server to Linux server
  - Migration script eliminates character case issues
  - File tampering made validation difficult
  - Supports SCCS migration

- ▶ Product area specific depots
  - Facilitates management activities
  - Reduces general developer “clutter”
  - Pre-refactoring specific depot
- ▶ PLAYTIME depot
  - User experiments
  - Access to primary development sources
  - Transient and subject to frequent obliterate
- ▶ ADMIN depot
  - Common distribution

- ▶ Branching use cases
  - mainline model
  - baseline protocol
- ▶ Scenarios to deal with bug fixes
  - Product delivery mechanism differences
  - Propagation between general and special delivery products
- ▶ Scenarios to deal with features
  - Final product configuration
  - Propagation between general and special delivery products
  - Propagation between projects with different release cycle frequencies

- ▶ Ongoing user environment management scripts
  - Workspace view consistency
  - Workstation character case consistency
- ▶ Triggers
  - Character case validation
  - Structure creation validation
  - External LDAP authorization
- ▶ Future Triggers
  - Workspace view conformance
  - Naming convention validation

- ▶ Pre-migration
  - VSS shadow folders
  - Prototype web sites are difficult
  - Pushing “bad” versions
- ▶ Post-migration
  - Branching structures
  - Automated processes
  - Validation environments
  - Deterministic rollback

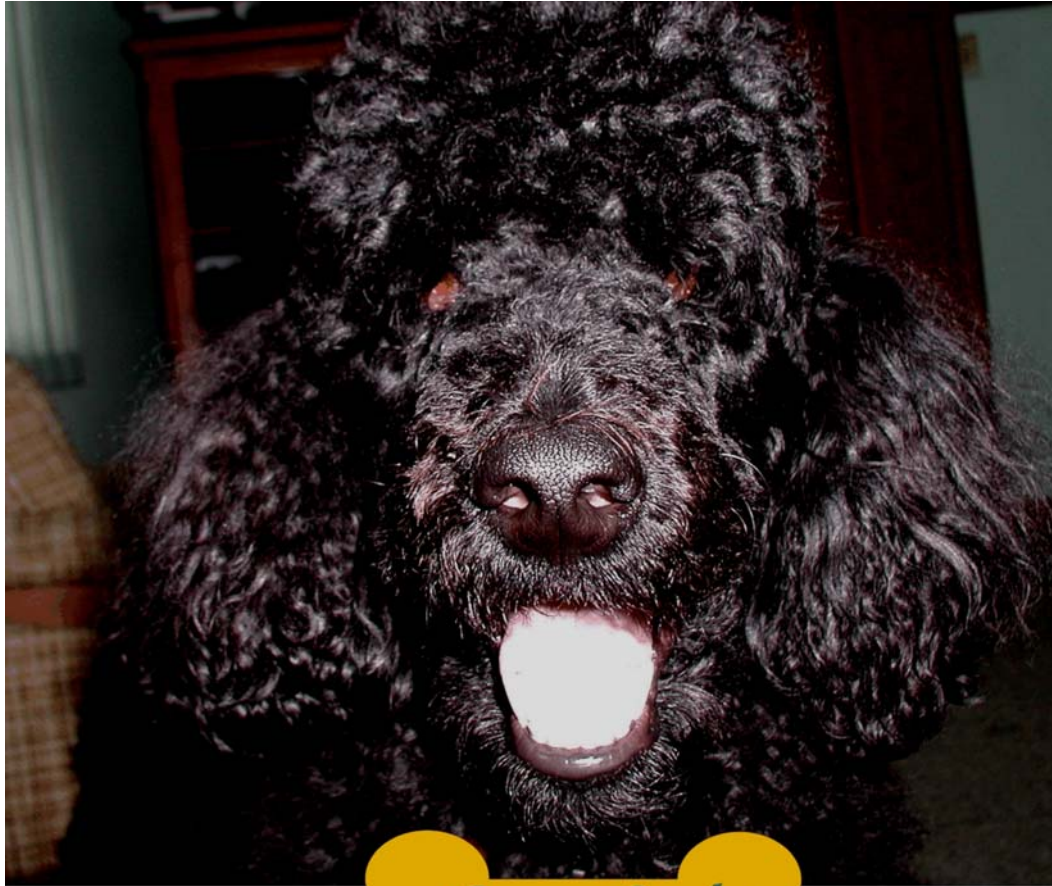
- ▶ Integration using P4DTG planned
- ▶ Software Planner features
  - WEB based interface
  - Project specific workflows
  - Project specific field configurations
  - Field value transfers
  - Support for team specific requirements, test cases, and project management
  - Graphical dashboards for quick identification of trends and details
- ▶ Post-integration plans to couple build automation with Perforce jobs
  - Single interface connection to build automation
  - Reduction of human entered information
  - Leverage defect tracking and team collaboration features of Software Planner

- ▶ Important Success Factors
  - Senior-level management sponsorship
  - Experienced leadership
  - Willing advocate participation
  - Clear understanding of the goals
  - Being flexible
  - Being adaptable
- ▶ Bottom line: things will never go as planned

A thought from Vegas

WSI

Sage Right



Sage Right

- ▶ For more information about WSI  
Weather Services International  
400 Minuteman Road  
Andover, MA 01810  
978-983-6300  
[www.WSI.com](http://www.WSI.com)
- ▶ For more information about SageRight  
508-393-6917  
[www.SageRight.com](http://www.SageRight.com)
- ▶ Post-migration Scripts  
Perforce Public Depot: `//guest/Neal_firth/SageRight`