Chapter

1

1 Introduction

Congratulations on choosing SpectrumSCM™ –
A Total Solution to all your configuration management needs

SpectrumSCM is the most economical full-featured, unified **S**ource **C**onfiguration **M**anagement (SCM) and Change Management application available. SpectrumSCM provides version control, issue tracking , change management, process management, release management, branching, and work flow management **all integrated in one tool - no bolt-on additions!**

- Integrated support for issue/problem tracking /change management.
- Flexible support for a development process that suits the culture of your organization, or the new culture that you wish to introduce.
- Support for teams, not just individuals.
- Support for distributed developers, including taking advantage of the Internet, Intranet and the only tool to offer full CM functionality over the web.
- Management of complex projects over multiple platforms, with multiple implementation strategies.
- A common configuration management system for Multiple Platforms.
- Simple migration from existing environment.
- Simple, quick and intuitive to install and get started.

SpectrumSCM is not just a Software Configuration, Source Control or Version Control system. **SpectrumSCM** provides all these functions in the context of comprehensive product life cycle tracking of product source, whether it is *software code, requirement docs, training docs, test plans, test scripts, build scripts, models, training docs, engineering drawings, images, web pages, documents, excel, powerpoints, contracts, proposals, curriculum material, SAS program files*, etc. All source is tracked from the day it is placed under SpectrumSCM control through any and all product builds or releases. Any product release is fully reproducible at any time in the future. Multiple releases (by customer or operating system, for example) can be developed concurrently with both shared and unique components.

1.1 Why is Software Configuration Management Required?

In today's world, software is often a company's most valuable asset. Far too often the source code and documentation are left unprotected in various directories on some multi-purpose machine. This is hardly considered safe and secure considering the time, money and effort spent producing that software.

Fixing a problem in a previous release can cause mass confusion as developers scramble to find the source of the problem and the related components. As files have been changed over time, numerous versions of each file may exist. Which files to fix? Which files to deploy to the users? Much valuable time and effort is spent looking for the right versions.

Building a new release of the product may involve developers each having a different set of files and/or making changes to the same files with no control. Much testing time is spent finding that the wrong version of a file is being tested. How does the project team reconcile multiple versions of the same file? How do they assure that all required changes have been included? Are there dependencies among the files, and how are they communicated?

Traditional configuration management systems focus on keeping track of file versions. A definite move in the right direction, but not nearly enough. Some require so much additional work that developers resist using them.

SpectrumSCM is designed to bring order from chaos, providing a complete source configuration and management system that provides major benefits without adding overhead to the development process. It is flexible enough to work within an already established process, and it can help establish a process model where one does not already exist.

1.2 The SpectrumSCM Paradigm

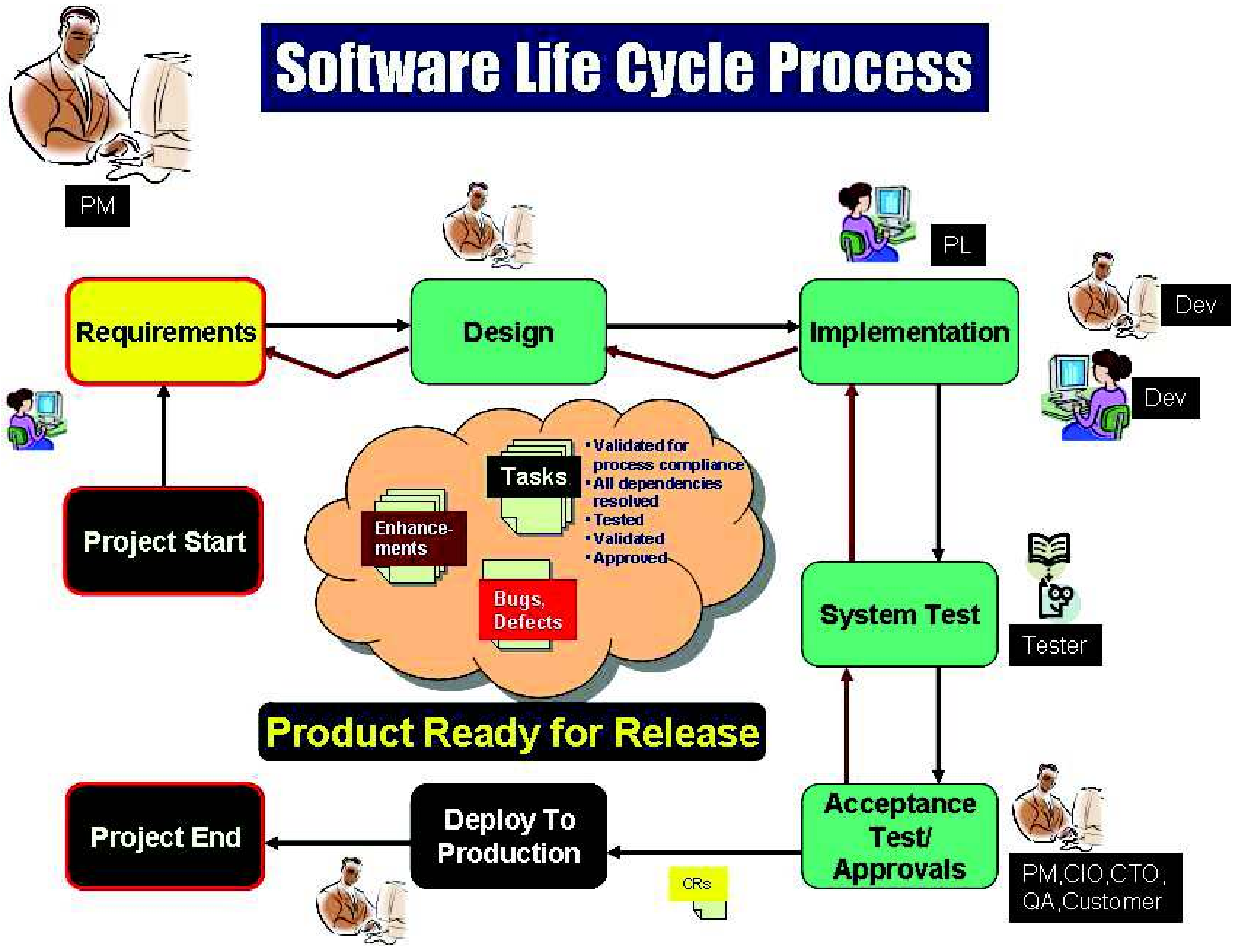
Key concepts in the SpectrumSCM Paradigm include:

- Projects
- Branches (Generics)
- Project Life Cycle Phases
- Project Team
- Roles
- Change Requests
- File Versions
- Releases

Projects can be entire applications or systems or any set of tasks (technical tasks, business tasks, bugs/defects, infrastructure related, approvals tasks, enhancements, new features etc). A group of people working on a project is the Project Team. When a project starts to use the SpectrumSCM system, it is important that the system is set up to support the team and the desired process that the team uses or wishes to use. The project team members are added to the system and assigned roles. Each person on the project team may have such specific roles as developer, tester, project leader, etc. The project's life cycle phases have to be defined to the system. They should mirror the phases of the development process or work flow used by the team. Phases can be as simple as "study, develop, test, complete" or as specific as the phases required by many government contracts.

SpectrumSCM is a **Task/Change-based configuration management system**. SpectrumSCM uses the Change Request (CR) as the basic element of work to track the changes as you had mentioned. Files of any type are logically associated with the CR and move through the life cycle with the CR. Users no longer have to worry about file labels, version labels, release labels, etc. They no longer have to worry about overlooking a file when applying labels, error in which can significantly impact builds and releases. Releases are composed of collections of CRs that have reached an approved state in the life cycle. CRs are simply dragged from the available pool into the associated release group in one easy step and all associated elements (i.e files) move into the release automatically. Only those CRs/Tasks that have been validated, approved, and have reached an acceptable state are allowed into a release.

The picture below shows an example of a simple software development life cycle using SpectrumSCM. In SpectrumSCM you can customize to a process that fits your business/project needs.



As work comes in, it is defined in the SpectrumSCM system by adding a new **Change Request** that defines the feature, fix or other reason why the work needs to be done. As the team members work, files are only added, changed and deleted in connection with a CR.

Traditional configuration management systems focus on keeping track of file versions. SpectrumSCM **Change Requests (CRs)** are the glue that assures that features, changes and fixes are connected to the file changes that implement them. CRs are the mechanism that allow management to track the history, activity, and status of particular issues or units of work. Using SpectrumSCM, files are managed by CRs that document why the work is being done. All changes, additions and deletions are traceable to a CR and user.

The project team may be working on various components of a large release or they may be working simultaneously on parallel versions of a release (designed for different customers or operating systems) or there may be fixes to a previous release while a new release is being developed. SpectrumSCM manages work within a project by defining generics. A **generic** is a branch of work that contains one or more features and therefore one or more files. A generic can be the first release of a system, a long-lived branch of continuing development work based on a previous release, or a short-lived branch created to fix a problem in an existing system. A generic contains files. The files can be source code, documentation, design, test plans, test results, or any type or file that must be managed. **Generics** can also be used to define/organize **sub projects** within a project. For example if different components are being built which are related to the same larger project but individually the desire is to have its own separate containers and change requests that makes its work items.

A project team can be working on multiple generics at any point in time. There are many ways a project team can define and divide work using various branching patterns. Some files may be the same as in another branch ("common"); some may be new or changed. ("uncommon") As files are changed over time, different **versions of a file** exist.

A release is a set of files, each at a particular version, that when extracted from the system make up a single version of the product. Managing release formation can be a tedious, time-consuming job on some CM systems. To create a release with the correct versions of individual files, a system needs to be able to properly track the file changes that make up each file version and present that information in some meaningful form. In the **SpectrumSCM** system this is easily accomplished with the built-in issue tracking system that ties each file change to a specific CR that describes what the changes were made. With SpectrumSCM a release is built from a set of CRs which automatically define the versions of each source file. Nothing is left to chance. CR dependency checking is done to assure release integrity.

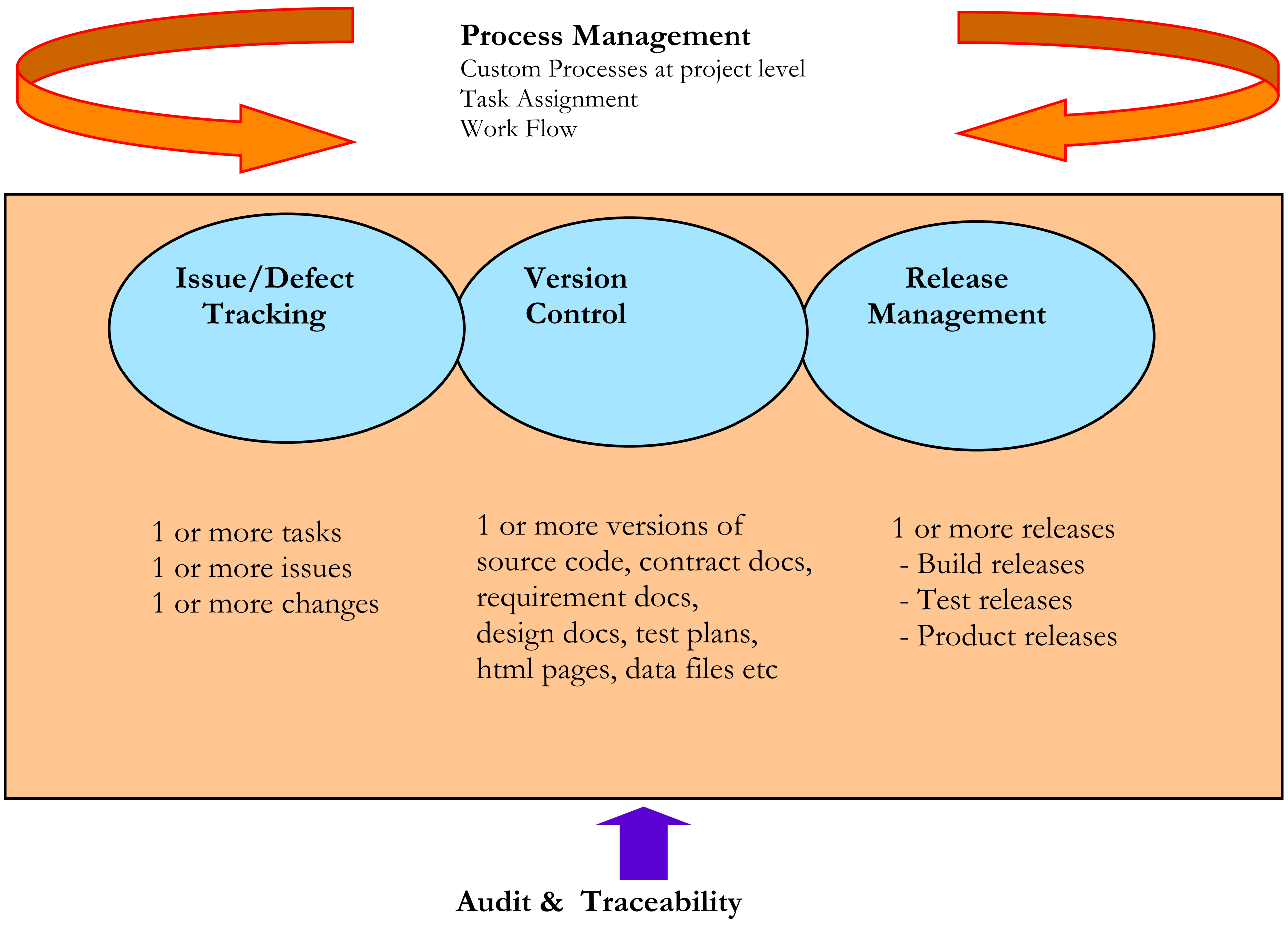
A release (a specific version of the product) can therefore be easily re-created at any time by automatically extracting the relevant versions of the files associated with the CRs in a release. SpectrumSCM was designed to make release management a simple task.

SpectrumSCM assures that the correct file versions are pulled for the release by extracting only the files associated with the CRs assigned to the release. If another CR not in the release has also changed the file, or the work on a CR has not been completed, the SpectrumSCM release management process will flag those CRs and report on the dependencies.

1.3 SpectrumSCM Process Management

SpectrumSCM supports a process that suits the culture of your organization or the new culture that you wish to introduce. Project life cycle phases and milestones are defined at the project level and can be as simple or as detailed as necessary. Government projects, for example, specify required life cycle phases that must be followed and documented. SpectrumSCM easily supports these requirements.

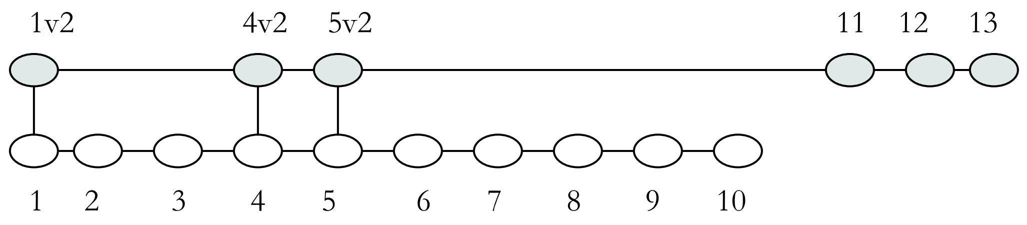
SpectrumSCM assures the quality of the product. All components are progressed through specified testing and quality assurance life-cycle phases. Some processes require a QA phase after each development step (requirements review, test plan review, test results review, etc.). Others simply require that testing be completed. SpectrumSCM allows for total customization of your process at the project level, provides the ability to verify that process was followed, and handles the storage of related process documentation associated with all phases of work All process tracking and documentation is tied to the associated change request.



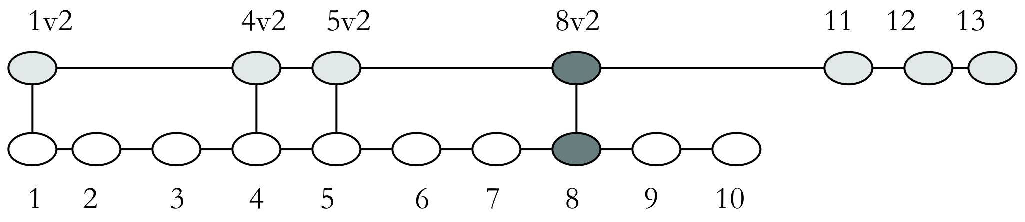
How SpectrumSCM manages Multiple Releases

For example, if a project team has developed and released version 1.0 of a system and they are currently developing version 2.0 (set up as generic 2.0), generic 2.0 will be based on the files in generic 1.0.

All files that are changed (1, 4 and 5) or added (11, 12 and 13) during the 2.0 development effort will be "uncommon" - changed only for generic 2.0. The rest of the files will be the same files as in 1.0. They are "common".



If, however, a problem is discovered in release 1.0, the fix for the problem might be made common to both generics. In this example, the problem is in file 8. The code is edited, the problem fixed and the fix is made common to both generic 1.0 and generic 2.0. A new release can be created to fix the problem (release 1.1) and deployed without disturbing the work that is going on in generic 2.0



In overview, checking a file out for edit "uncommon" will mean any file changes will only be made against that specific generic. If a check-out is performed "common" then the file changes will be made against ALL the generics that that file is currently in common with.

- **Common:** Versioned files that are physically the same across generics
- **Uncommon:** The act of physically separating versioned files from multiple generics

Checking out "common" is a powerful feature since it can be used to apply a single "fix" to multiple generics in one edit, however the developer would have to be careful of side-effects.

1.4 SpectrumSCM Technical Features

- **Powerful client-server architecture with an intuitive easy-to-use Graphical User Interface.**
    - *Works in LAN, WAN and WWW environments.*
    - *The client can even be run through your browser*
- **100% pure Java™ for complete platform independence**.
    - *Client and server can run on any platform supporting the appropriate Java Virtual Machine.*
    - *Absolutely no dependency on the underlying OS.*
- **Easy product creation and re-creation, any release, any time.**
    - *Build by Change Requests (CRs), not by file version*
    - *Build by Release (Release is a collection of CRs). It is much easier to remember a release number than it is to remember the CR number(s) that refers to a specific release.*
    - *Build by packages (a set of releases, folders or CR states) possibly from multiple projects or branches.*
    - *Reproduce, enhance or fix any release at any time.*
- **An industrial strength Object-Oriented database with full transactional integrity.**
    - *No need for database administration.*
- **Extremely scaleable and provides various levels of customizable security**
    - *Leverages the Java™ Security Model and Security Extensions with flexible Enhanced Access Control.*
    - *Projects occupy their own individual database.*
- **Source Code Extensibility**
    - *Ability to change files without disturbing previous or concurrent work.*
    - *Native file move and rename capabilities.*
    - *Workspace Analysis and Synchronization.*
- **Controlled Access to Shared Resources**
    - *Safe file sharing*
    - *All file changes traceable to user and Change Request*
    - *Versions can be compared side by side with differences high-lighted*
    - *Merge and recommon capabilities.*
    - *Access control lists (ACLs) for role based management of resource permissions*
- **Distributed Team Development**
    - *Single repository client-server model.*
    - *Remote access through Java WebStart, applet or regular client*
    - *Proxy server for bandwidth constrained environments*
    - *Work Space Analyzer and Work Space Synchronizer Utilities*

1.5 SpectrumSCM Management Features

- **Inclusion of management features as a key element of the product, not an afterthought**
    - *Full featured issue tracking and change management support*
    - *Customizable to fit the tool to your development process instead of changing your development process to fit the tool!*
    - *Pre-defined and customizable online reporting, graphs and charts*
    - *Powerful Project Performance and Metrics Dashboard*
    - *Work-flow management and tracking*
    - *Complete history of each issue and assignment*
    - *Supports meeting SEI CMM, ITIL, 21 CFR Part 1 etc. objectives.*
- **Issue Tracking / Problem Management / Change Management and Control**
    - *Ability to relate actual source changes to a living, track-able problem statement.*
    - *Parent child and peer to peer relationships between issues.*
- **Work-flow Management**
    - *Powerful Graphical workflow editor*
    - *Ability to create, assign and progress work through project life-cycle changes automatically, with real-time email notifications.*
    - *Business process rule automation*
    - *Significant cycle time reduction.*
- **Limitless versatility and total customization at the project level.**
    - *Supports the use of a process that suits the culture of your organization, or the new culture that you wish to introduce.*
    - *Enforces process without impacting development work.*

1.6 Getting Started with SpectrumSCM is Quick!

Follow the steps in the tutorial that is included on your installation CD or available for download at http://www.spectrumscm.com. The tutorial walks though the steps required to set up a project, generic, and project team and teaches the basics of using SpectrumSCM. This will enable a new user to get started in using all the basic features quickly and easily.

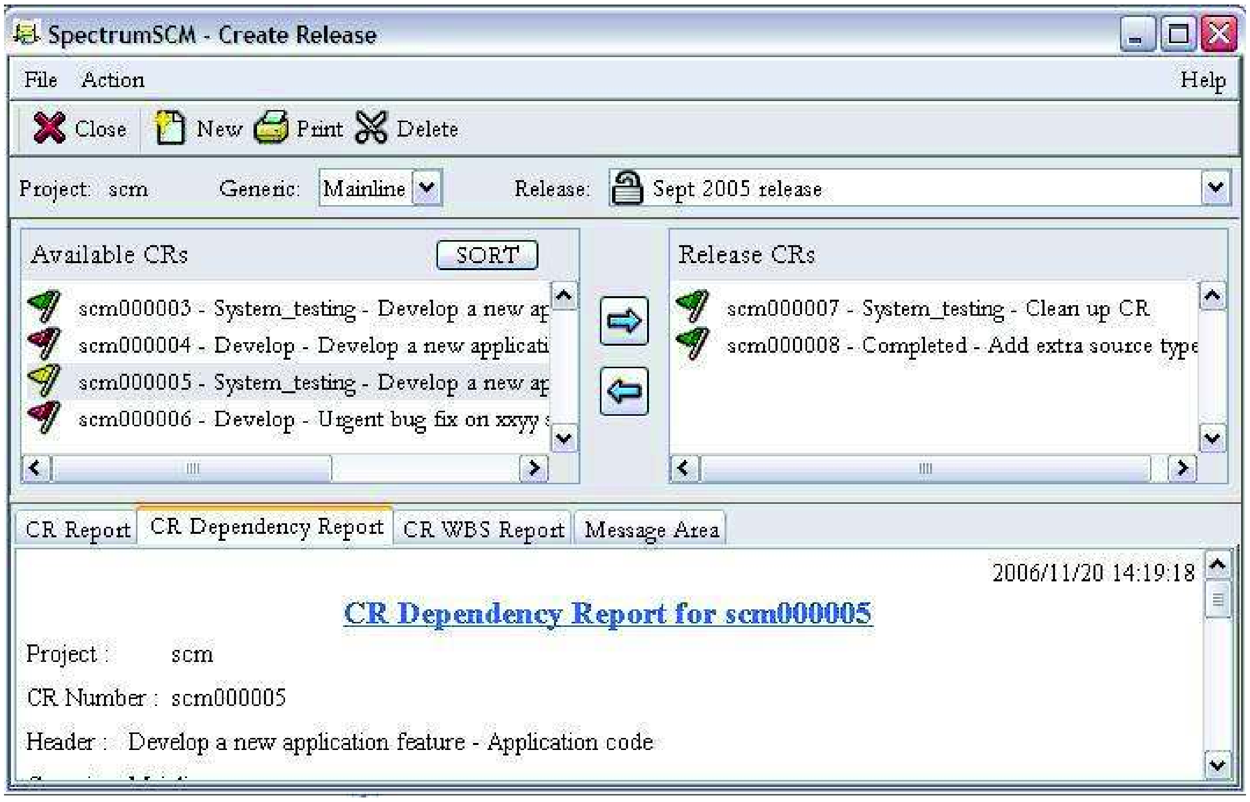
- **Installation** is automated and takes 5 - 15 minutes.
- **Create a Project**
- **Create a Generic**
- **Set up Project Life Cycle Phases**
- **Set up User roles and permissions**
- **Add project team members** as SpectrumSCM users and associated with the project.

1.7 Working with SpectrumSCM is Easy!

- **Create CR** - No work can be done without a Change Request (CR) Some projects create CRs at the feature level "develop feature 2" for example. Others create CRs at a more detailed level. A CR can also describe a problem that needs to be fixed
- **Assign CR** – A project leader will assign the CR to the first person who needs to do work on it. Depending on the size of the team and the phases in the life cycle, it might be assigned to one person to study, another to develop, and another to test. In a small team, one person might handle everything.
- **Establish local root directory** to define the location on the local hard disk where files to be checked in are found and files extracted to the hard drive are placed.
- **Load Directory Tree, Add Source, Check out files, Check in files** – the member of the project team to whom the CR is assigned might create and add files to the SpectrumSCM system or check out files, make changes, and check them back in. You can even use simple DragNDrop features to move files in and out from anywhere on your desktop.
- **Progress CR** – when a member of the project team finishes his or her work on a CR, it is "progressed", alerting the project leader that it is complete or ready to be assigned to the next person who needs to work on it.
- **Branching, Merging, and Recommoning files** as needed.
- **Create a release** – To create a release (nightly build, release for testing, or release for deployment), the Release Management feature of SpectrumSCM makes it easy to see which CRs (and their associated files) are ready.
- **Review Reports** at any point to see status of each file, CR, assignment, etc.
- **View Graphs and Charts** at any point to track, analyze, measure project, process and change request trends.

No CR goes out before its time!

SpectrumSCM assures that all components are complete and all dependencies are identified. **The Available CRs** area displays the CRs that are in this generic. Note that CRs are flagged.



scm000003 - System_testing - Develop a new ap
scm000004 - Develop - Develop a new applicati
scm000005 - System_testing - Develop a new ap
scm000006 - Develop - Urgent bug fix on xxyy

scm000007 - System_testing - Clean up CR
scm000008 - Completed - Add extra source type

2006/11/20 14:19:18

CR Dependency Report for scm000005

Project : scm

CR Number : scm000005

Header : Develop a new application feature - Application code

Only green CRs can be moved into a release.

Green – the CR's life cycle is complete. You can include this CR and its associated files in the release.

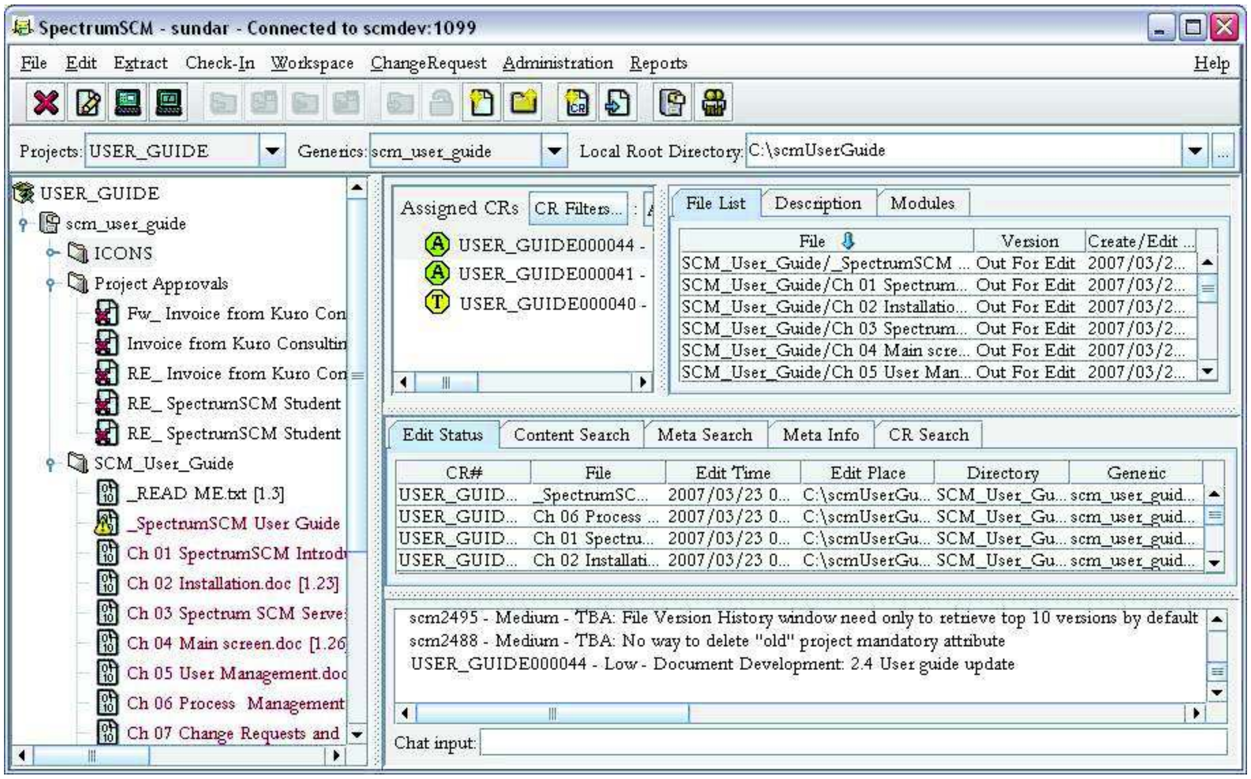
Yellow - the CR has completed development, but it is dependent on other CRs that have not yet been completed.

Red – the CR is not complete.

Dependency checking assures that all related CRs are complete before the release can be created.. **The cycle of work continues until all CRs are completed and the release is ready to go!**

SpectrumSCM can be used to manage any type of product development, not just software. For example, it can be used to manage document creation or controlled storage of and updates to documents.

The creation of this user guide was controlled with SpectrumSCM!


SpectrumSCM - sundar - Connected to scmdev:1099
File Edit Extract Check-In Workspace ChangeRequest Administration Reports Help
Projects: USER_GUIDE
Generics: scm_user_guide
Local Root Directory: C:\scmUserGuide
Assigned CRs
CR Filters...
File List
Description
Modules
Edit Status
Content Search
Meta Search
Meta Info
CR Search
Chat input:

USER_GUIDE
- scm_user_guide
  - ICONS
  - Project Approvals
    - Fw_ Invoice from Kuro Con
    - Invoice from Kuro Consultin
    - RE_ Invoice from Kuro Con
    - RE_ SpectrumSCM Student
    - RE_ SpectrumSCM Student
  - SCM_User_Guide
    - _READ ME.txt [1.3]
    - _SpectrumSCM User Guide
    - Ch 01 SpectrumSCM Introdu
    - Ch 02 Installation.doc [1.23]
    - Ch 03 Spectrum SCM Serve
    - Ch 04 Main screen.doc [1.26
    - Ch 05 User Management.doc
    - Ch 06 Process Management
    - Ch 07 Change Requests and

Assigned CRs:
- USER_GUIDE000044 -
- USER_GUIDE000041 -
- USER_GUIDE000040 -

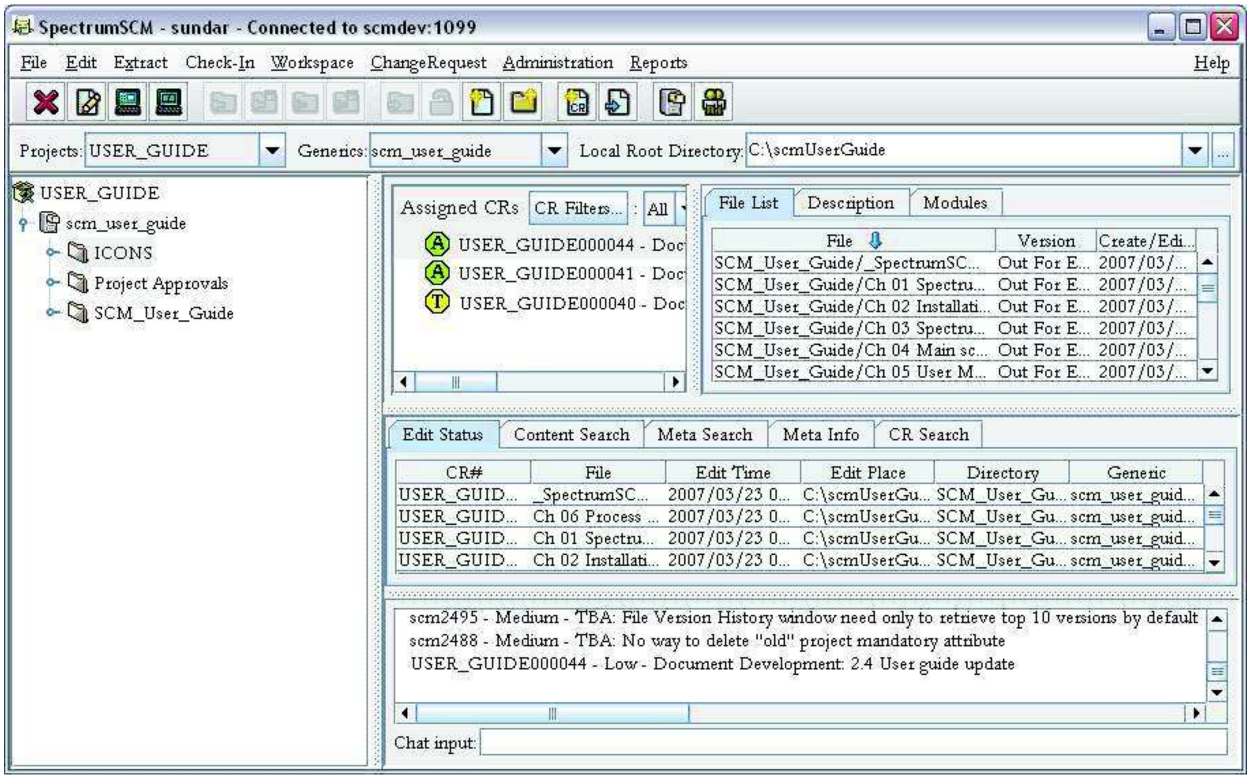
File	Version	Create/Edit ...
SCM_User_Guide/_SpectrumSCM ...	Out For Edit	2007/03/2...
SCM_User_Guide/Ch 01 Spectrum...	Out For Edit	2007/03/2...
SCM_User_Guide/Ch 02 Installatio...	Out For Edit	2007/03/2...
SCM_User_Guide/Ch 03 Spectrum...	Out For Edit	2007/03/2...
SCM_User_Guide/Ch 04 Main scre...	Out For Edit	2007/03/2...
SCM_User_Guide/Ch 05 User Man...	Out For Edit	2007/03/2...

CR#	File	Edit Time	Edit Place	Directory	Generic
USER_GUID...	_SpectrumSC...	2007/03/23 0...	C:\scmUserGu...	SCM_User_Gu...	scm_user_guid...
USER_GUID...	Ch 06 Process ...	2007/03/23 0...	C:\scmUserGu...	SCM_User_Gu...	scm_user_guid...
USER_GUID...	Ch 01 Spectru...	2007/03/23 0...	C:\scmUserGu...	SCM_User_Gu...	scm_user_guid...
USER_GUID...	Ch 02 Installati...	2007/03/23 0...	C:\scmUserGu...	SCM_User_Gu...	scm_user_guid...

```
scm2495 - Medium - TBA: File Version History window need only to retrieve top 10 versions by default
scm2488 - Medium - TBA: No way to delete "old" project mandatory attribute
USER_GUIDE000044 - Low - Document Development: 2.4 User guide update
```

SpectrumSCM's Main Screen provides access to all functionality and a clear view to each user of his or her assignments and current work status. When each user logs on, his or her assignments are clearly visible.

- If the CR is assigned to the user for work, a green A is displayed.
- If the CR is assigned to the user for work and has supporting attachments, a green A with a "gem clip" is displayed.
- If the CR is assigned to the user for review or testing where no edits are allowed, a blue A is displayed
- If a CR has been progressed and is in a TBA (to be assigned) state, it is shown to all users in the generic engineer role and those with assignment capabilities, with a yellow T.



USER_GUIDE000044 - Doc
USER_GUIDE000041 - Doc
USER_GUIDE000040 - Doc

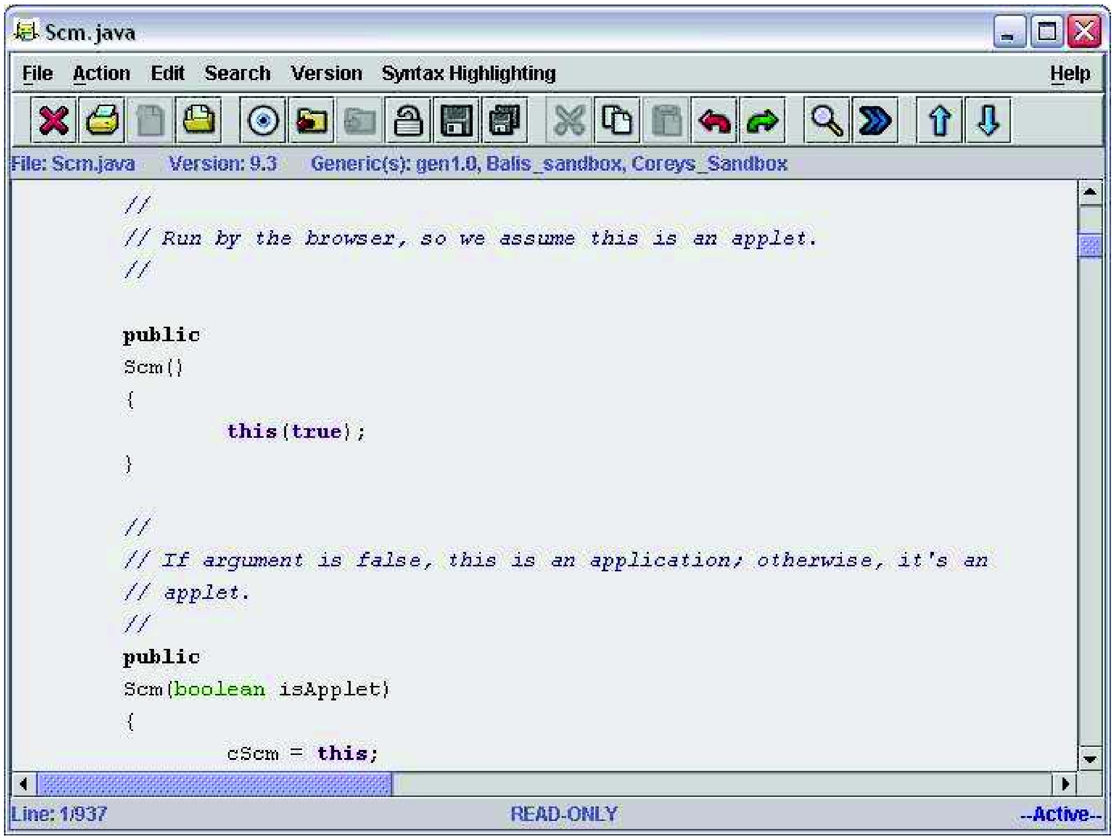
File	Version	Create/Edi...
SCM_User_Guide/_SpectrumSC...	Out For E...	2007/03/...
SCM_User_Guide/Ch 01 Spectru...	Out For E...	2007/03/...
SCM_User_Guide/Ch 02 Installati...	Out For E...	2007/03/...
SCM_User_Guide/Ch 03 Spectru...	Out For E...	2007/03/...
SCM_User_Guide/Ch 04 Main sc...	Out For E...	2007/03/...
SCM_User_Guide/Ch 05 User M...	Out For E...	2007/03/...

CR#	File	Edit Time	Edit Place	Directory	Generic
USER_GUID...	_SpectrumSC...	2007/03/23 0...	C:\scmUserGu...	SCM_User_Gu...	scm_user_guid...
USER_GUID...	Ch 06 Process ...	2007/03/23 0...	C:\scmUserGu...	SCM_User_Gu...	scm_user_guid...
USER_GUID...	Ch 01 Spectru...	2007/03/23 0...	C:\scmUserGu...	SCM_User_Gu...	scm_user_guid...
USER_GUID...	Ch 02 Installati...	2007/03/23 0...	C:\scmUserGu...	SCM_User_Gu...	scm_user_guid...

```
scm2495 - Medium - TBA: File Version History window need only to retrieve top 10 versions by default
scm2488 - Medium - TBA: No way to delete "old" project mandatory attribute
USER_GUIDE000044 - Low - Document Development: 2.4 User guide update
```

The file development process is supported and enhanced by

- A full featured graphical editor that supports syntax highlighting, the ability to walk backwards and forwards in time (file versions) and can handle enormous text documents with ease.



File: Scm.java   Version: 9.3   Generic(s): gen1.0, Balis_sandbox, Coreys_Sandbox

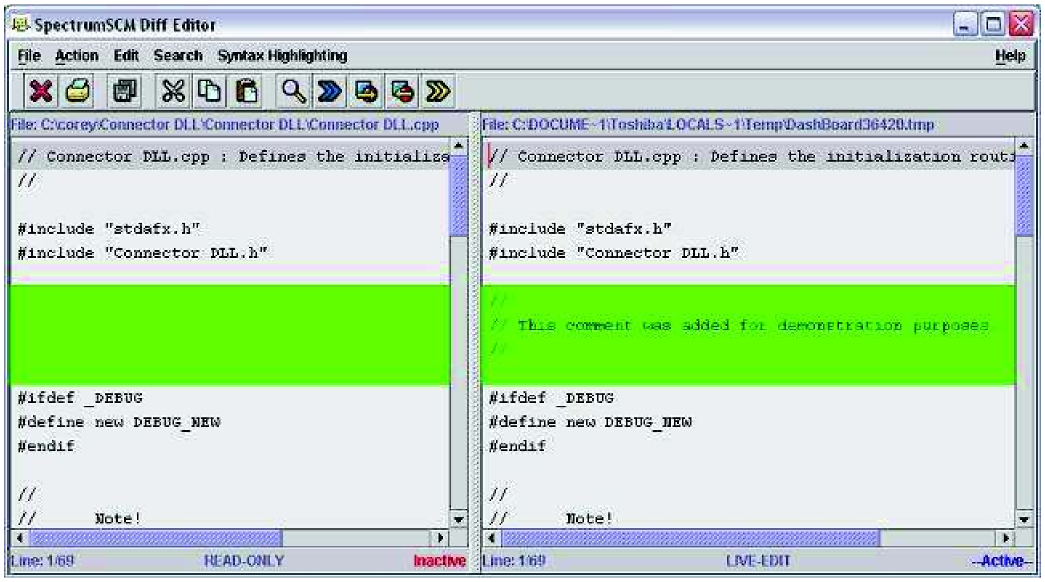
```
        //
        // Run by the browser, so we assume this is an applet.
        //

        public
        Scm()
        {
                this(true);
        }

        //
        // If argument is false, this is an application; otherwise, it's an
        // applet.
        //
        public
        Scm(boolean isApplet)
        {
                cScm = this;
```

Line: 1/937   READ-ONLY   --Active--

- A color-enhanced, side-by-side difference viewer / editor. Side-by-side difference viewer can also be used to aid merging two versions of code or text.



File: C:\corey\Connector DLL\Connector DLL\Connector DLL.cpp

```
// Connector DLL.cpp : Defines the initializa
//

#include "stdafx.h"
#include "Connector DLL.h"




#ifdef _DEBUG
#define new DEBUG_NEW
#endif

//
//      Note!
```

Line: 1/69   READ-ONLY   Inactive

File: C:\DOCUME~1\Toshiba\LOCALS~1\Temp\DashBoard36420.tmp

```
// Connector DLL.cpp : Defines the initialization routi
//

#include "stdafx.h"
#include "Connector DLL.h"

//
// This comment was added for demonstration purposes
//

#ifdef _DEBUG
#define new DEBUG_NEW
#endif

//
//      Note!
```

Line: 1/69   LIVE-EDIT   --Active--

Even more exciting are the merging and recommon capabilities of SpectrumSCM. As members of the project team, work on different branches, multiple versions of a file are created and must be synchronized.

Merging in SpectrumSCM is the act of copying the changes made in one version of a file into another file. **Recommoning** makes the files identical; changes made in one are copied into the other and vice-versa.

Both merge and recommon use the **SpectrumSCM Merge Editor.** The Merge Editor highlights the differences between two versions of a file. Inserts show up in green, changes/differences in yellow and deletes in red. Differences can be generated and changes can be applied in either direction.

To learn about these and the many other features of SpectrumSCM, download the **SpectrumSCM Tutorial** (or access the tutorial from your installation CD) and follow the instructions step-by-step to create your first project in SpectrumSCM. Then read through the User Guide and review the examples. Many of your questions and even concerns will be answered as the power and flexibility of the SpectrumSCM system becomes apparent.

Stay current with new features and ways to use the product. The SpectrumSCM web site is constantly updated with new topics. Download and read the white papers available at www.spectrumscm.com.

1.8 Glossary

Term	Definition
SCM	*Source Configuration and Management* - Configuration Management for ALL of your product source whether it be software, web pages, requirements documents, whatever ...
Configuration Management	Process control that includes version control, audit trails, release management, and issue management working together to produce solid products that are both testable (what is in this release) and maintainable (what was in a previous release).
CR (Change Request)	An issue to be recorded and tracked. SpectrumSCM uses change requests (CRs) to drive its system. Changes made to files are tied to a specific CR, which lets developers quickly find all changes resulting from any CR. CRs allow files to be managed by issues, features, or fixes.
Branch	A separate version of a project that supports different feature sets or bug fixes from other branches within the same project.
Generic	A branch of work that contains one or more features and therefore one or more files. These files may or may not be branched (see common/uncommon files). Can be used to define sub projects or components within a project as well.
Life-cycle	A connected set of phases applied to a project change request. For example a CR might be created in the *TBA* state or assigned to *specific* state (phase). The CR might then progress through the *defined life-cycle* phases for that project before being considered complete. This set of phases is a life cycle.
Issue Management	The capability to record problems or issues and track them through the development process to their final release. This includes relating the actual source that fixes the issue to the issue itself. The capability to annotate the issue with progress notes is also significant.
Problem Tracking	See *Issue Management*
Version Control	Tracking changes to every file, and providing the capability to access any version of the specific file.

Release	A release is defined as a set of files, each of a particular version, which when extracted from the system together make up a single version of the product. SpectrumSCM ties each CR to specific versions of one or more files, so generating a release is just a matter of selecting the appropriate CRs to include. When it creates a release, SpectrumSCM displays a list of all CRs from which users can select specific (completed CRs). The user creating the release selects the CRs to include in each release.
Interim Release	A phase based informal release which extracts all the file versions at the specified workflow/life-cycle phase. This is most useful to perform automatic development/testing builds before formal release management gets involved.
Release Management	The ability to easily establish and maintain control over product releases at any time in that release's lifetime. This includes the ability to recreate it at any time and also the ability to extend it. Specifically a release is made up of a set of specific versions of the product's source files.
Package/Component Management	The ability to easily establish and maintain control over a complete product even if its components are maintained within separate projects or generics within the SpectrumSCM repository. Once defined a package can be recreated at any time. Specifically a package is made up of a set of specific releases, interim releases or folders.
Audit Trail	All changes to the product must be recorded so that control can be maintained. Who changed what, when and why?
Parallel Development	Supports development of multiple separate features from the same source base.
Uncommon File	A file that has been established in a separate generic AND has been edited (made different) under that generic (i.e. branched).
Common File	A file that shares its current version and version history with other branches.
Merging a file	The action of merging the contents of two uncommon versions of a file to create an updated version of one of those files.
Recommoning a file	The action of merging the contents of two uncommon versions of a file to create a single new version.