Spectrum Software, Inc.
11445 Johns Creek Pkwy.
Suite 300
Duluth, GA – 30097
www.spectrumscm.com

Subject: Do **I**t **R**ight **T**he **F**irst **T**ime **- Building and Deploying Releases using SpectrumSCM®**

Issue Date: **October 1st, 2005**

From: **S. Srinivasan**
srini@spectrumsoftware.net
(770)448-8662

1.0 Introduction:

Wherever there is a desire for quality, everyone is aware of the Quality Doctrine "**D**o **I**t **R**ight **T**he **F**irst **T**ime". In software development environments, more so, it has become a challenging task to implement a disciplined process, along with use of effective tools and technologies to achieve this goal.

So why not have the goal **'B**uild **I**t **R**ight **T**he **F**irst **T**ime**' ?.** Building and Deploying a software product from requirements, design and initial coding all the way into the production environment involves a complex network of processes, people and technologies. Often in today's heterogeneous and geographically distributed software development environment, the teams involved in the various phases of an enterprise product development life cycle find it very difficult to keep up with what is being delivered to the end customer, and in many instances even to their own internal build and test environments.

There is a tremendous pressure and impact to your customer satisfaction and overall business objectives if your customer facing software applications or solutions fail. Is there a way to guarantee the integrity of your build, of your releases, of your deployment at any time all the time? Can you always reproduce what was delivered to the customer at any time? Can you provide traceability and auditability for what was delivered at any time both from internal QA objectives and regulatory compliance perspectives?

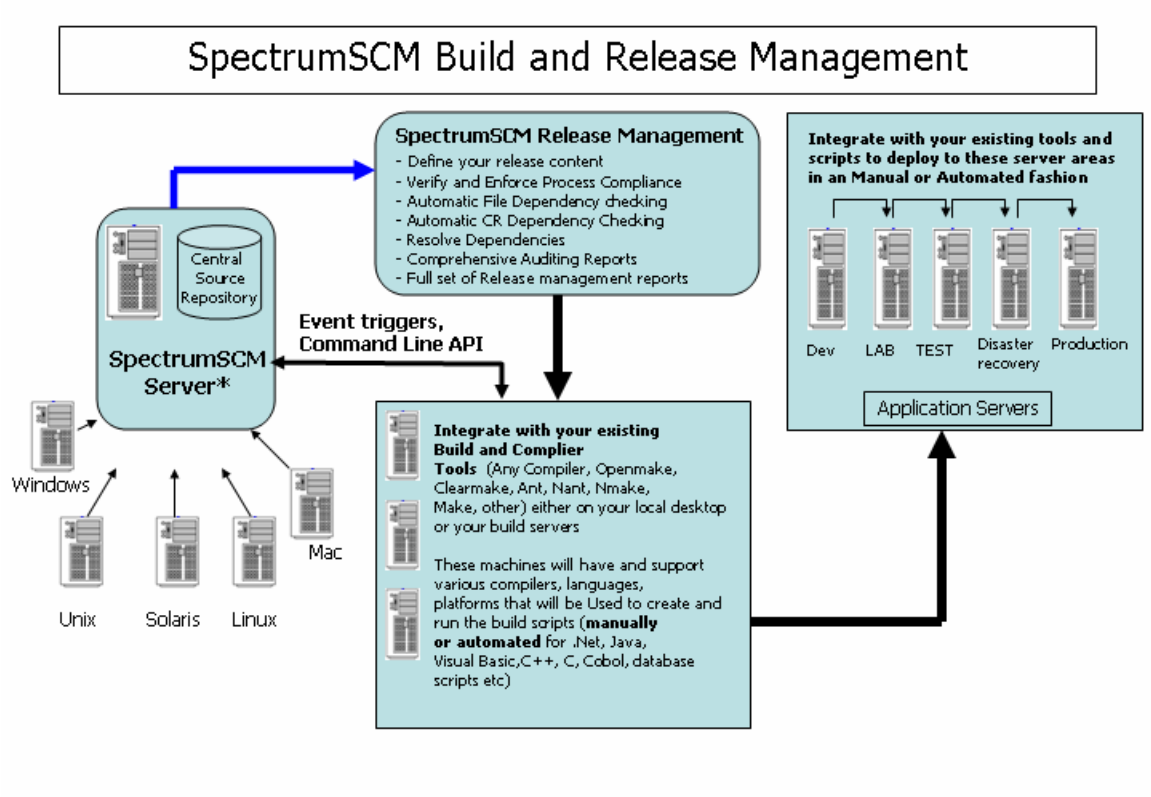
Using SpectrumSCM, a full featured, platform independent SCM tool which has process management, work flow and release management tightly integrated to its issue management and version control elements, you can now achieve these above goals efficiently. SpectrumSCM enables **end-to-end traceability** for any **'e-Asset'** (***source code, images, drawings, process documents, binary files, web pages*** etc.) from origination through delivery, maintenance, and support.

With SpectrumSCM's powerful process based release management, the activities that were traditionally the burden of build engineers and build tools are now managed through SpectrumSCM's release management features well before the sources reach your build and deployment area. SpectrumSCM will guarantee the integrity of the contents of your software product release all the time. The process of doing your build becomes much more intuitive and at the same time your build process becomes **repeatable, reproducible and fully auditable**.

So **"B**uild **I**t **R**ight **T**he **F**irst **T**ime**"** is definitely worth striving for..... and with SpectrumSCM it is no longer out of reach !!!

2.0 The Release Management Process in SpectrumSCM

The release management feature in SpectrumSCM is the most important step before you get ready to run your build scripts. 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. In order to create a release with separate versions of individual files, a CM system needs to be able to properly track the file changes that make up the individual file versions and present that information to the user in some meaningful form. In the SpectrumSCM system, this is easily accomplished with the built-in issue tracking system which is tightly integrated to its source control system.



***SpectrumSCM repository can be used as the central source for storing source and binaries including complete deployment packages.**

SpectrumSCM uses the Change Request (CR) as the basic element of work. All files associated with the change are logically attached to the CR and move through the CR life cycle with it. 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, errors in which can significantly impact builds and releases. Releases are composed of collections of CRs (defects, bug fixes, enhancement requests, etc.) that have reached an approved state in the life cycle. Release management is an intuitive, graphical process. 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. File level dependency and change request level dependency checking between CRs is an **automatic** and **continuous** process. No CR can be moved to a release if an outstanding dependency exists. CRs are color coded in the release function to give instant feedback on status – green for "ready with no dependencies", yellow for "ready but dependencies exist" and red for "not ready".

Specialized file changes, like small branches in the code, cannot be lost in the system because each change is officially recorded in the issue tracking system. Each release is a collection of active CRs in the system. Releases are built on top of previously formed releases to simplify the entire release management job. As each CR is completed and the releases are created, the number of outstanding CRs that must be accounted for is reduced to the number of open CRs since the last release was created. This keeps the number of CRs for any given release to a manageable number.

A release (a specific version of the product) can therefore be easily re-created at any time simply by extracting the relevant versions of the necessary files using the appropriate CRs or simply extracting by the release name. Outside of tracking changes to files, creating releases is the single most important job that a CM system must perform. The SpectrumSCM system is designed from the ground up with the necessary features that make release management a simple task. **SpectrumSCM thus provides alerts and problem resolution much earlier in a software lifecycle well before invoking your build scripts.** For more detailed reading on this topic please refer to Chapter 9 Release Management in the SpectrumSCM User Guide, available on our website at www.spectrumscm.com

3.0 Process Centric Approach to Building and Deploying Software in SpectrumSCM

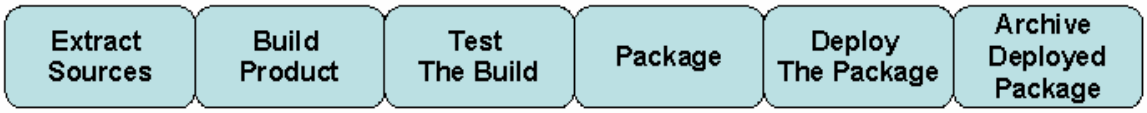
SpectrumSCM is a highly process-centric tool that uses the strength of a solid Task/Change Request based paradigm to eliminate traditional build-deployment issues. The tight integration of **Issue management** and **Version Control** combined with a customizable process workflow definition guarantees the accuracy of the source file versions that form the builds. SpectrumSCM provides the ability to manage all the tasks that will be required to accurately build and deploy the right contents of any software release, eliminating errors that are common when employing manual processes.

You leverage your investment in existing build and deployment utilities to provide complete end-to-end process for managing, controlling and auditing all of your software deliverables.

You can automate, track, audit and provide reports on the status of the tasks required to build and deploy software accurately with processes that are customizable, repeatable, enforceable and most importantly auditable.

3.1 An example Build-Deployment process flow in SpectrumSCM

With SpectrumSCM you can customize and configure your build and deployment processes according to the application needs and to fit your project goals. These process steps can be standardized, enforced and become repeatable and auditable. Below is an example of a Build_Deploy Lifecycle that can be defined and enforced in SpectrumSCM. Every task in each of these phases will be recorded, tracked and monitored using SpectrumSCM.



Extract Sources: In this phase you will extract validated and approved source contents through the use of SpectrumSCM extract features. In SpectrumSCM you can extract by release name, by life cycle phases (i.e. all files in a particular phase in the project life cycle), by generics/branches, change requests, by directories/folders/module and by individual files. You can use the SpectrumSCM API to automate this extraction.

Build Product: There are so many build systems out in the market that the decision to work with all of them rather than embedding a build function in the application. SpectrumSCM guarantees the integrity of the sources or contents of the builds as per **Extract_Sources** phase above. SpectrumSCM can also interface to external build tools through its API triggers to invoke actions in an automated fashion.

In this phase, users are then free to work with their favorite build tool (Openmake, ANT, Nmake, make, etc). You can also integrate with existing build scripts by embedding SpectrumSCM's Command-Line Interface in these scripts to access the SpectrumSCM project artifacts.

Test the Build: In this phase you will move the built software to a test area and perform testing as per your testing lifecycle activities. SpectrumSCM can be used to record and track the status of this activity.

Package: If the build is approved and ready to be deployed, in this phase you will create the complete package containing all the 'e-assets' that make up your formal product release (i.e. executables, source code, scripts, libraries, drivers, documents, release notes, web content, user manuals, etc). SpectrumSCM can be used to record and track the status of this activity.

Deploy Package: In this phase using SpectrumSCM's API's you can define event triggers to automatically launch your existing deployment scripts to install the product package. SpectrumSCM can be used to record and track the status of this activity.

Archive Deployment Package and Release Reports: In the final step you can use SpectrumSCM to be the database of record to record and track the deployment content. In the event of a failure, the end-to-end audit trail becomes instantaneously available. SpectrumSCM helps quickly pinpoint the cause of the problem and determine the status of the production or test servers. Quickly support accurate and controlled rollbacks by guaranteeing the reproducibility of previous releases, while the problem is being resolved.

4.0 Key Benefits of using SpectrumSCM as the gatekeeper and enforcer of your build and deployment process.

1. One fundamental and important distinguishing feature in SpectrumSCM, is that, SpectrumSCM is one of a handful of SCM systems that are considered truly integrated, meaning that our **version control, issue tracking, change management, process management and release management** capabilities are built right into the product itself and is not a bolt-on like in other tools. Hence it provides you the complete end to end traceability in managing change to any source documents, build components, product releases etc., that are derived from the SpectrumSCM repository.

2. SpectrumSCM allows flexible and customizable framework that allows teams to integrate their preferred build tools and compilers to create a seamless, unified view of development and to manage their build and release processes.

3. SpectrumSCM facilitates and can interface with your build and deployment tools through its API and command line features. SpectrumSCM ensures the integrity of the build and deployment activity by guaranteeing the integrity of the project artifacts that make up that build or deployment. It is also provides process centric approach to manage, track and audit your builds and production artifacts.

4. SpectrumSCM supports diverse sets of build tools, hardware platforms, programming languages, scripting languages, operating systems (including Windows, .NET, Macintosh, Linux, Mainframe

and variations of UNIX) and IDE platforms (.Net, Visual Studio, Eclipse, WSAD, any Microsoft SCCI compliant IDEs).

5. SpectrumSCM provides the necessary process framework to create and manage an integrated development environment irrespective of its complexity using your existing build tools. The strong process framework combined with its workflow engine enables the artifacts and the tasks that makes up your software products to move seamlessly through each development phase and keeps cross-functional teams in sync.

6. SpectrumSCM provides a comprehensive set of reliable bill-of-materials reports for each release (i.e. list all issues in a release, files touched in a release, full set of reports that show why a change was made, who made the change, what was changed, when the change was made, where was the change made) and most importantly provides a complete and precise audit trail to track the evolution of any project asset that is part of any release at any time. This enables all teams to become more productive. Build teams know what changed for rapid troubleshooting; developers and project lead knows the why, who, what and when of every change; QA teams know exactly what to test; and management and product support have an accurate view of what was shipped to customers.

7. SpectrumSCM eliminates most of the common error prone steps and offers many features to the build/release engineer to make better decisions.

8. SpectrumSCM supports and can integrate with Complex, Multi-Platform Development Environments through the use of both the SpectrumSCM API and the command line APIs.

9. SpectrumSCM easily integrates with existing development tools and other build tools through the use of both the SpectrumSCM API and the command line APIs.

10. SpectrumSCM enables users to detect and trouble shoot build errors quickly and efficiently.

11. SpectrumSCM facilitates mechanisms to streamline the build process by triggering automated builds and deployment to designated areas on your server.

12. SpectrumSCM enables defining and building actions (i.e. triggers) and dependencies.

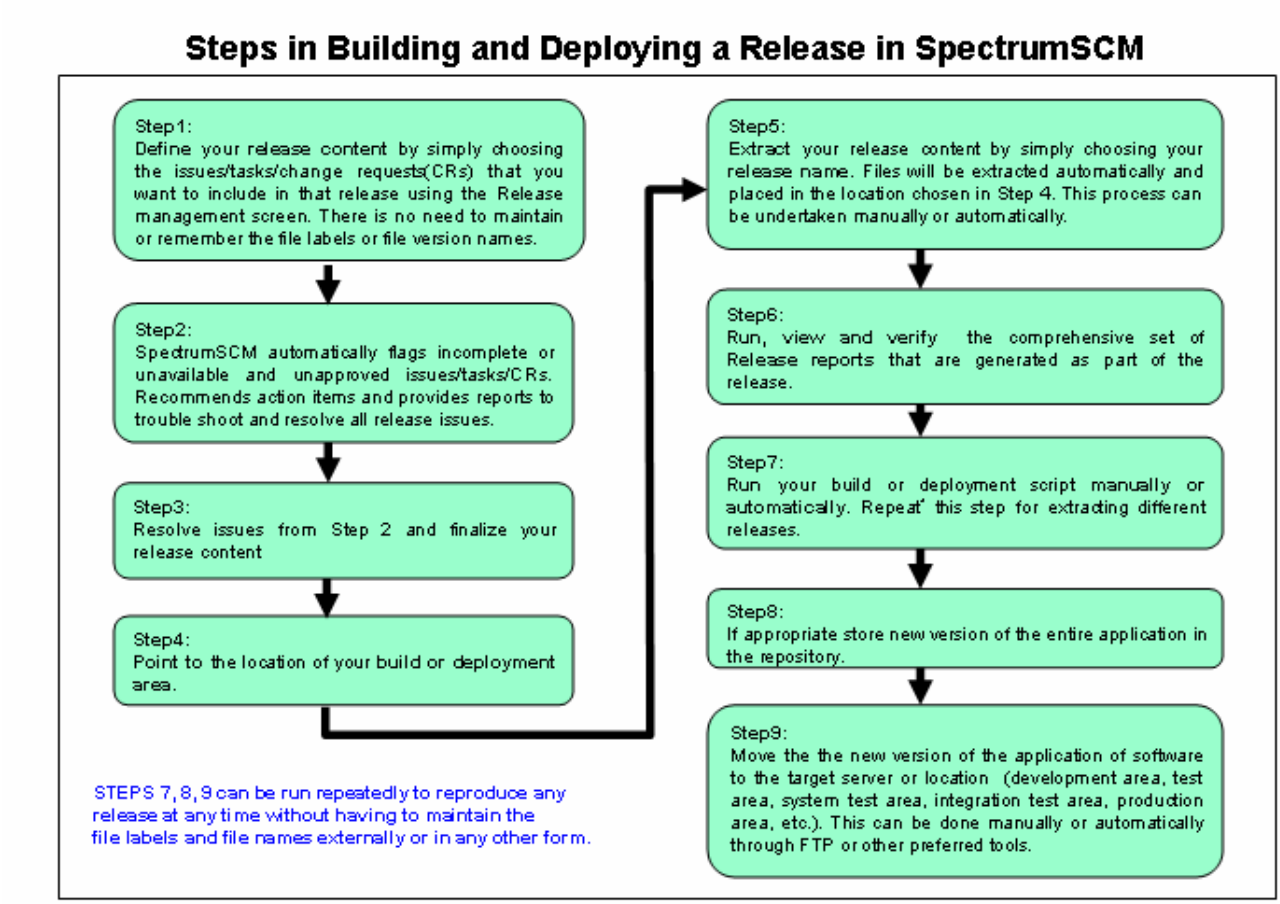
13. SpectrumSCM provides features to automate and manage dependencies using both its automated file-dependency and CR work break down structure features.

14. SpectrumSCM enables any authorized user to build or maintain any of the releases and provides very powerful features to manage individual workspaces, sandboxes, build areas, etc.

15. SpectrumSCM provides a flexible framework to maintain the same build software infrastructure for private system build, development and QA integration builds, and final release build, etc.

16. SpectrumSCM provides several powerful mechanisms to manage builds for different product versions/codeline branches.

5.0 Example of defining, building and deploying a software release in SpectrumSCM



6.0 Build and Release Reports (Tracking, Auditing and Compliance) :

SpectrumSCM supports a wide range of pre-defined configuration management reports that are available instantaneously within the system. These reports cater for the different types of users of SpectrumSCM , namely developers, managers, QA, build engineers, administrators, auditors etc. These reports provide a full audit trail of the complete product evolution from origination to delivery including the approvals and process compliance history. Reports have multiple output formats to enhance delivery process. In addition, the report data from any of the current reports can be exported to other external report writers. This data can then be used to write your own custom reports.

For example if you want to troubleshoot the source of a build or deploy problem because of the source file **SyncCheckInFiles.java,** follow the gray shaded row in each of these reports**.** The following combination of reports provides you the ability to answer **what, where, when,** and **why** easily and instantaneously. Many such audit reports are available in SpectrumSCM as part of its comprehensive set of reporting capability thus bringing in overall efficiencies and improving productivity and quality of the deliverables.

Sample Set of Reports (Tracking, Auditing and Compliance)

2005/09/06 10:53:39

Change Requests Assigned to Release: scm2.2.3.0

Project : scm
Generic : gen1.0
Release : scm2.2.3.0
Creator : ALL

CR#	Header	Creator
scm2161	XML operations not handling special chars properly	adrian
scm2163	Start looking at Java 1.5 - Continued	adrian
scm2164	Worksync improvements - take initial selection from main screen	adrian
scm2165	Worksync improvements 2 - enhance outgoing filters specifically to scan makefiles etc	bali
scm2166	Extend LoadSourceTree to handle multiples	adrian

End Of Report

2005/09/06 10:55:01

Files Touched in a Release: scm2.2.3.0

Project : scm
Generic : gen1.0
Release : scm2.2.3.0

All Files per Change Requests Associated with Release

scm2161	**XML operations not handling special chars properly**	**Version**
	src/package/do_package.sh	13.9
	src/scm/evolve/XMLUtility.java	2.2
	src/scm/transport/EditCommand_d.java	1.5
	src/scm/presentation/worksync / SyncCheckInFiles.java	1.2
scm2163	**Start looking at Java 1.5 - Continued**	**Version**
	src/applet/SpectrumSCM.jnlp	1.10
	src/scm/utilities/ScmJavaVersionChecker.java	1.4
scm2164	**Worksync improvements - take initial selection from main screen**	**Version**
	src/scm/presentation/ScmTreeComponent.java	16.8
	src/scm/presentation/worksync/SyncTreeComponent.java	1.6
	src/scm/presentation/worksync/ SyncCheckInFiles.java	1.2
scm2165	**Worksync improvements 2 - enhance outgoing filters specifically to scan makefiles etc**	**Version**
	src/help/WorkspaceSync.html	1.4
	src/scm/presentation/worksync/WorkSync.java	1.38

scm2166	Extend LoadSourceTree to handle multiples	Version
	src/scm/presentation/addSource/ScmLoadSourceTree.java	19.10

End Of Report

2005/09/06 11:04:05

File History Summary

Project : scm
Generic : gen1.0
Filename : SyncCheckInFiles.java
Filepath : src\scm\presentation\worksync

File **SyncCheckInFiles.java** is not currently out for edit.
Current version number is: 1.3, created on 2004/11/02 21:54:58 by bali, CR scm1853
File type is: TEXT

Previous Versions

Version Number	Name	Edited By	Edit Date	Change Request
1.2	SyncCheckInFiles.java	adrian	2004/10/04 17:14:13	scm2164
1.1	SyncCheckInFiles.java	corey	2004/06/30 19:11:22	scm1699
1.0	SyncCheckInFiles.java	corey	2004/01/08 13:26:44	scm1325

End Of Report

2005/09/06 11:03:15

File History Detail

Project : scm
Generic : gen1.0
Filename : SyncCheckInFiles.java
Filepath : src\scm\presentation\worksync
High Version :
Low Version :

File **SyncCheckInFiles.java** is not currently out for edit.
Current version number is: 1.3, created on 2004/11/02 21:54:58
File type is: TEXT

Source Changes

Version Number	Name	Edited By	Edit Date	Change Request	Differences
1.3	SyncCheckInFiles .java	bali	2004/11/02 21:54:58	scm1853	10I import scm.proxy.*; 85I Vector proxyUpdate = null;

					86I Pref_d prefs = (Pref_d)ScmCompManager.locate(ScmCompManager.PREFS_OBJ);
					87I if(prefs.cUseProxy) {
					88I proxyUpdate = new Vector();
					89I }
					90I
					124I if(proxyUpdate != null)
					125I proxyUpdate.add(sfi.cFileD);
					126I
					144I if(proxyUpdate != null && proxyUpdate.size() > 0) {
					145I int port;

1.2	SyncCheckInFiles.java	adrian	2004/10/04 17:14:13	scm2164	68-98D 103I ScmMainDisplay smd = (ScmMainDisplay)ScmCompManager.locate(ScmCompManager.MAINDISP_CMP); 104I reservation = smd.getReservation(fileName, sfi.cFileI.getFileType()); 105I 133D
1.1	SyncCheckInFiles.java	corey	2004/06/30 19:11:22	scm1699	119I 120I if(localRoot.endsWith(File.separator)) { 121I localRoot = localRoot.substring(0, localRoot.length() - 1); 122I } 123I

End Of Report

2005/09/06 10:59:03

Change Request Report for scm2164

Project :	scm
CR Number :	scm2164
File Detail :	Summary
Header :	Worksync improvements - take initial selection from main screen
State :	Completed
Assignee :	
Generic :	gen1.0
Creator :	adrian
Create Date :	2005/05/17 14:43:24
Create State :	Tested
Associated Release(s) :	scm2.2.3.0 (gen1.0)
Description :	2 improvements - a) Take initial tree selection from the main screen tree. b) Provide user feedback as to scan completion, currently particularly if there are no differences detected the screen just goes blank.

CR Attribute values

Attribute	Value
Release	2.2
Severity	Medium
Customer	Internal
Type	Enhancement

Files edited (summary)

File	Latest Version Edited	Commonality
src/scm/presentation/ScmTreeComponent.java	16.8	Is Common (5)
src/scm/presentation/worksync/SyncTreeComponent.java	1.6	Is Common (6)
src/scm/presentation/worksync/ SyncCheckInFiles.java	1.2	Is Common (5)

Attachments

Number	Attachment Name

CR State History

State	User	Begin Date	End Date	Note
Develop	adrian	2005/05/17 14:43:24	2005/05/17 16:09:49	State Changed from Develop to Test. User Changed from adrian to ready4release. This change was performed by adrian.
Test	ready4release	2005/05/17 16:09:49	2005/07/07 17:48:31	State Changed from Test to Develop. User Changed from ready4release to

				adrian. This change was performed by adrian. Not working - needs fix
Develop	adrian	2005/07/07 17:48:31	2005/07/08 13:25:37	State Changed from Develop to Test. User Changed from adrian to ready4release. This change was performed by adrian.
Note	adrian	2005/07/08 13:55:05	2005/07/08 13:55:05	Problem was with the generics - on the main screen the generic has the root/base information, whereas on the work-sync it does not. This was causing the "find" routine to fail. Stripping the base off of the main screen generic information before doing the compare fixes the prob.
Test	ready4release	2005/07/08 13:25:37	2005/07/26 16:10:39	State Changed from Test to Completed. This change was performed by adrian. Release scm2.2.3

End Of Report

For additional information on SpectrumSCM please visit our website at www.spectrumscm.com. Or contact Spectrum Software at 770.448.8662 or email to info@spectrumsoftware.net.