

6 Process Management

This chapter describes how to set up the project environment to support your development process, including setting up a project under SpectrumSCM, establishing the project life cycle, and creating a generic. It also describes how to set up the project directory structure and load existing files and file structures into the SpectrumSCM system.

SpectrumSCM Process Management

- Ensures that components are progressed through chosen life cycle phases before being released. For example, to verify that testing and quality assurance occur before a component is cleared for release.
- Allows for total customization of your process at the project level.
- Allows the use of a process that suits the culture of your organization or a new process that you wish to introduce
- Change Management is integrated tightly into the entire product life cycle.

Work through this chapter and follow the steps to set up a sample project. SpectrumSCM provides complete flexibility to map your current process or customize the system to handle specific project and process needs.

6.1 Setting up a New Project under SpectrumSCM

To add a new project to SpectrumSCM, click on the **Create Project Wizard** menu item under the **Administration** menu on the SpectrumSCM main screen to access the **Project Creation Wizard**.

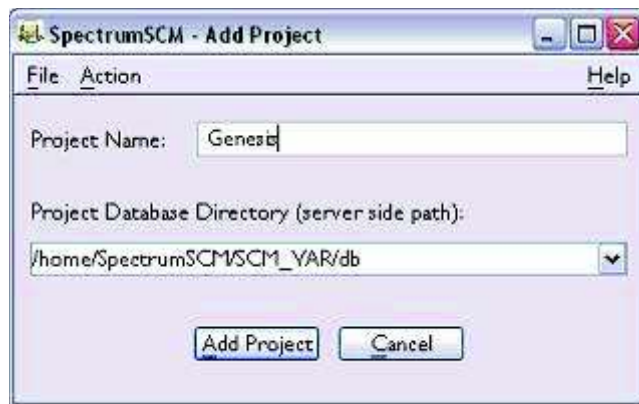


The Project Creation Wizard is an aid to creating new projects in SpectrumSCM. The wizard guides the user through all of the steps necessary to create a new project, set up a new branch, and all of the other steps required to get the project ready for first use. The Project Creation Wizard screen is broken up into two separate sections. The table area at the top contains the list of activities that the user will execute in order to setup the new project. Note that some of the activities are required and some are not. The steps that are not required can be skipped during execution. The bottom portion of the screen is a text area that describes the step that the user is about to execute. It also provides screen snapshots of each individual screen that will be presented during execution. At the bottom of the screen are three buttons, **Cancel**, **Next>>** and **Execute Task**. These buttons allow the user to navigate through the tasks and to execute them when instructed. The user can press the **Next>>** button at any time to move to the next task. This allows the user to read through all of the instructions first, and then return to actually execute the tasks.

As tasks are completed, a check mark is added to the table entry for the completed task. Tasks that have been completed cannot be executed again. Tasks that have not been completed can be revisited and executed at any time.

6.1.1 Project Creation

In this step the user will be prompted to create a new project. The text area at the bottom of the screen will contain information about the project creation screen itself and provides instructions concerning the project creation screen.



In the project creation screen the user must enter the name of the project that they want to create, and the server side path to the database directory. The database directory path will always default to the SCM_VAR/db directory, located under the server side installation directory.

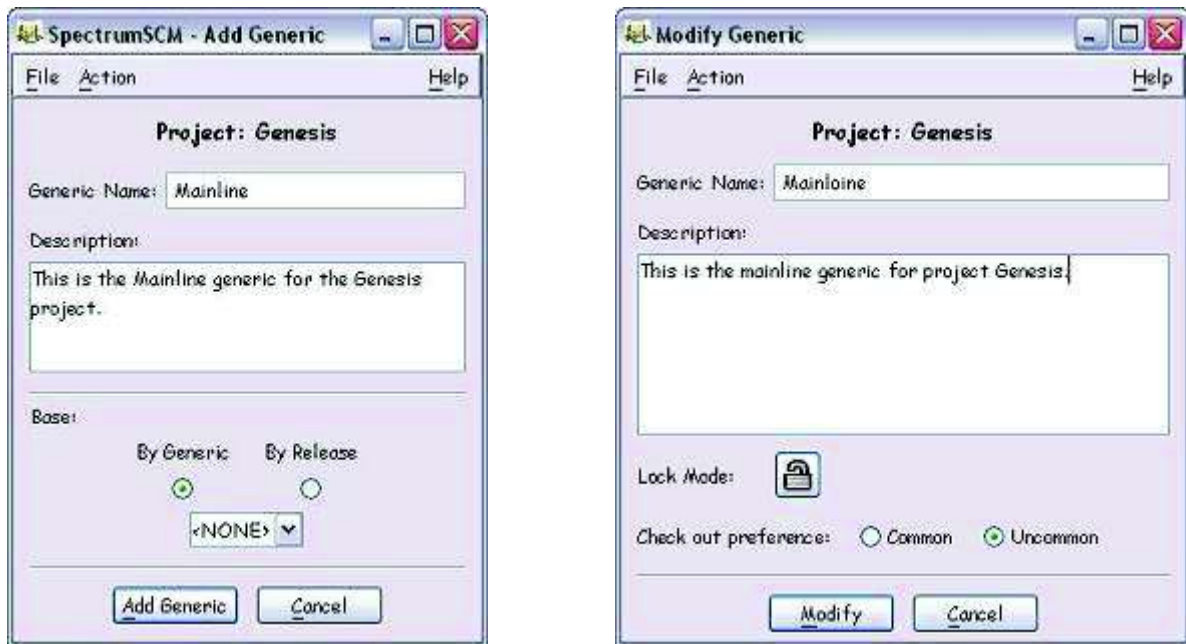
NOTE – The directory entered into the Project Database Directory path needs to be backed up on a regular basis. If a backup strategy is already in place for this server, make sure to extend it to include this new directory. If no backup strategy is effect for this server, make sure to create one right away and start backing up the files located under this directory on a nightly basis.

Once the project has been successfully created, the user will be presented with a confirmation dialog indicating that the project has been created.

6.1.2 Mainline Creation

Projects can contain many generics (branches). Upon creation, a new project does not contain a generic. Before work can begin on the project, a new generic must be created. The Generic (branch) creation screen prompts the user to enter a name for the new generic as well as a short description of what the generic should be used for. For new projects the radio buttons at the bottom “By Generic” and “By Release” have no meaning, since there are no other generics or releases for this project at this time. Enter a name for the new generic and a short description and then press the “Add Generic” button.

Once the new generic has been created, the system will respond with the Modify Generic screen.

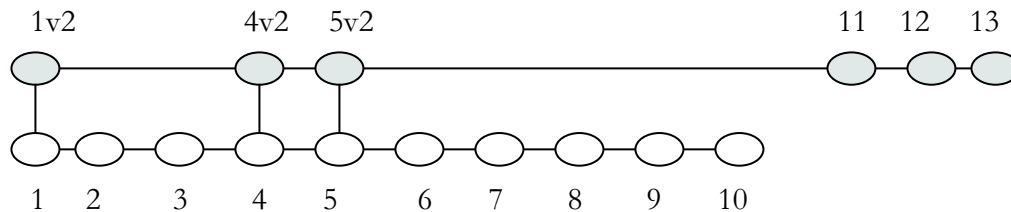


This screen allows the user to set the check out preferences for the buttons on the main screen and to mark the generic as “Locked” or not. Locking a generic fixates all of the files in the new generic to their current version number. Edits done on other generics, against files that are common with the locked generic, will not affect the files on the locked generic. The files will be automatically specialized (uncommoned) into the locked generic thus preserving the content of the files on the locked generic. The locking mechanism is most useful when a generic is created from a previous release. The files on the newly created generic will always contain the same content from the release that they were branched from, regardless of any actions that happen to common files on other branches. The exception to this rules comes when files are edited directly against the locked generic. Edits performed common to the locked generic are not automatically specialized into the generic.

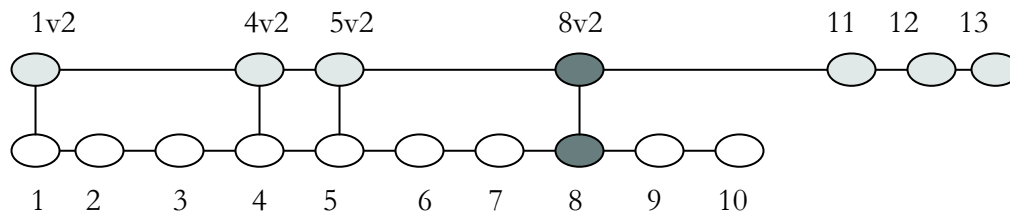
Common Checkout vs. Uncommon Checkout – The checkout preference determines how the two checkout buttons on the main screen tool bar operate. If “Common” is set, edits always default to common and if “uncommon” is set, the default behavior is to uncommon, but the user is presented with a pop-up that prompts the user to “confirm” that an uncommon check out is really desired. The generic engineer would decide how these functions should be setup.

If a check-out for edit is performed "common" then the file changes will be made against all generics with which the file is currently common (as determined during project set-ups). Sometimes there are good reasons for this to be done, for example, fixing a problem in multiple generics.

For example, if a project team has developed and released version 1.0 of a system and they are currently developing version 2.0 (generic 2.0), all modules that are changed (modules 1, 4 and 5) or added (modules 11, 12 and 13) during the 2.0 development effort will be "uncommon" - changed only for generic 2.0.



However, if 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.



When multiple generics are to be used and developed in parallel (for example to maintain 2 similar source bases for 2 different customers), the Generic Engineer must decide who is going to make the branching decisions, who will determine which of the modules will be common or uncommon with other generics. If it is to be the Developer then the generic should be left in the *unlocked* state. If the Generic Engineer will make the decisions, the generic should be set in the *locked* state and can be unlocked on a case-by-case basis.

Once the mode of the Generic has been established, if the commonality control is still with the developer (the generic is "unlocked"), then he or she can choose the appropriate option

In overview, checking out "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** of the generics with which this file is currently common.

Checking out "common" is a powerful feature since it can be used to apply a single "fix" to multiple branches in one edit, however the developer would have to be careful of side-effects. *This topic will be covered in more detail in Chapter 8.*

Create a Generic from a previous Generic. A new Generic can be created as a branch from another generic or can be created from a previous release. On the Generic Creation screen select the radio button on the bottom "By Generic". Then use the combo box to select the generic to branch from. When the new generic is created all of the files on the branched generic will become common with the new generic.

Individual specializations may then be applied by checking files out as either common or uncommon to the new generic.

Create a Generic from a previous Release. A new Generic can be created as a branch from a previous release. On the Generic Creation screen select the radio button on the bottom “By Release”. Then use the combo box to select the release to branch from. When a generic is created from a previous release, all of the files in the new generic will match the content of the files as they appear in the release itself. The version numbers of the files may or may not match depending on whether the files have been extended since the release was formed. Files that have been extended will be specialized into the new generic and those that have not been extended will remain common with the current branch. This gives users the opportunity to immediately extend release patches into the current release on the previous or mainline branch.

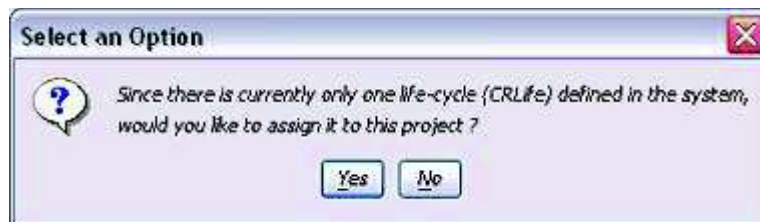
6.1.3 Life-Cycle Setup

Once the new project and its mainline generic have been created, the next step is to add a lifecycle to the project. There are two options to assigning a lifecycle to a new project. Either an existing lifecycle can be used, or a new lifecycle can be created and assigned.

If the system has any life cycles with workflow created using the SpectrumSCM LifeCycle graphical Editor, the Project Creation Wizard brings up LifeCycle/Workflow administration Editor.. Otherwise, the system brings up the linear life cycle creation screen.

6.1.3.1 Linear Life-Cycle

If only one lifecycle exists in the system at this point, the

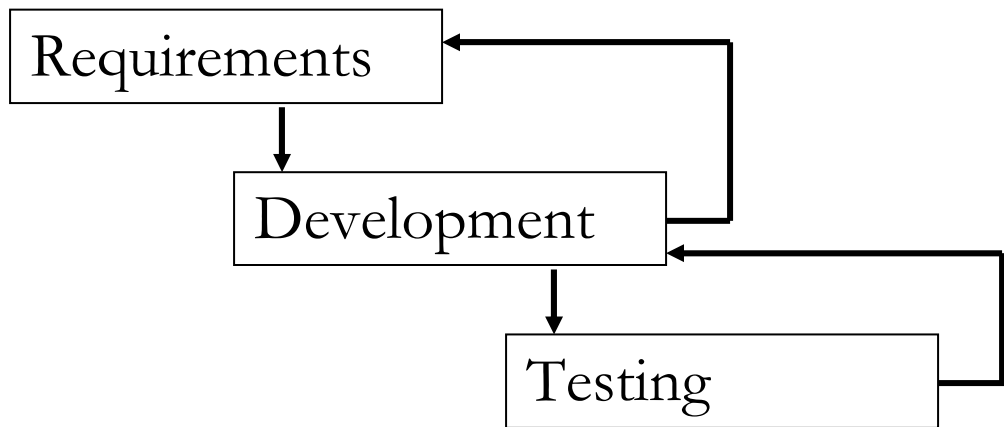


Project Creation Wizard will post the following screen to the user asking whether the system pre-defined lifecycle “CRLife” should be used or not. The user may choose to use this pre-defined lifecycle by pressing the “Yes” button, or the user may press the “No” button and the system will allow the user to create their own lifecycle and phases

A life cycle defines a set of phases, where entry and exit from each phase is well defined.

An SCM system is traditionally used to manage some form of Product Development lifecycle, for example, a software development life cycle for IT projects. Some SCM systems define and enforce a strict adherence to a pre-defined life cycle, while others can fit comfortably into an existing life cycle.

A trivial example of a software development life cycle might look like this:



In this very simple life cycle, the requirements phase is the starting phase followed by development and finally testing. The phases follow the standard waterfall model with feedback loops, which enables backtracking to an earlier phase.

More advanced life cycles are generally found in large software producing organizations where more focused phases may include several layers of testing and possibly even phases for hardware testing/profiling.

SpectrumSCM is designed to be complimentary to the process, and not an obstacle to be worked around. It is

- flexible enough to work within an already established process
- able to help establish a process model where one does not already exist

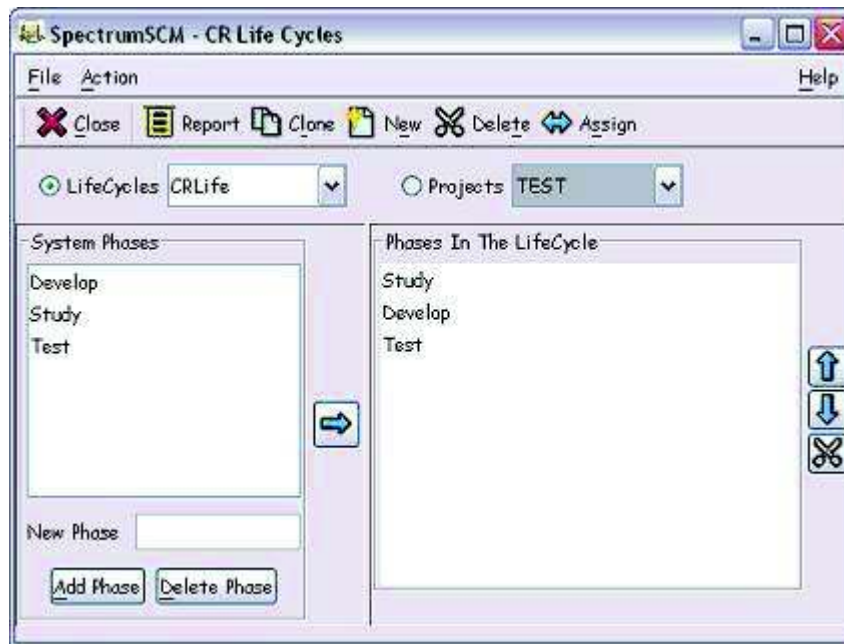
For a non-software project, life cycle phases might include development, review, revision, and approval.

6.1.3.2 Creating a new life cycle

To define the life cycle phases for a project, the **CR Life Cycles** screen is accessed via the **Main Screen, Administration / CR Life Cycle Admin**:



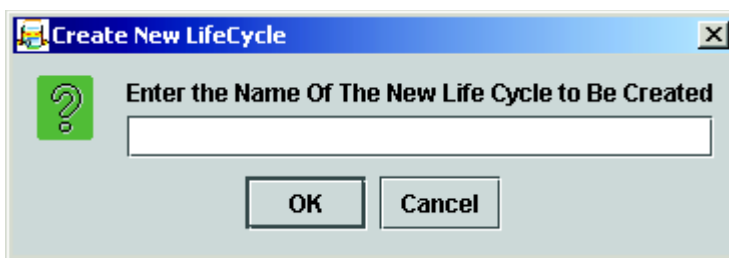
This screen can be approached two ways depending on the choice of the **LifeCycles** or **Projects** radio button just below the tool bar.



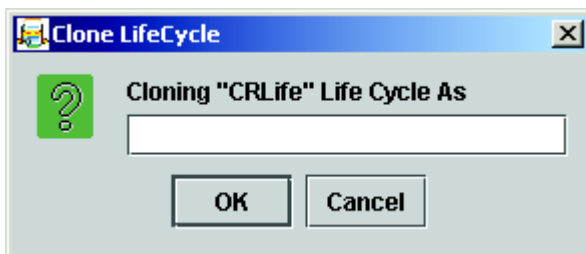
If the **LifeCycles** button is selected (the default) then the pull-down allows the user to choose from the life cycles that have been created in the system, or to create a new life cycle. The simple life cycle “CRLife” has been pre-defined. If there are any other life cycles that have been created in the system, one can be selected via the pull-down and its phases will be displayed.

Life-cycles can be administered by using the other buttons on the screen –

- **New** - to create a brand new life cycle from scratch.

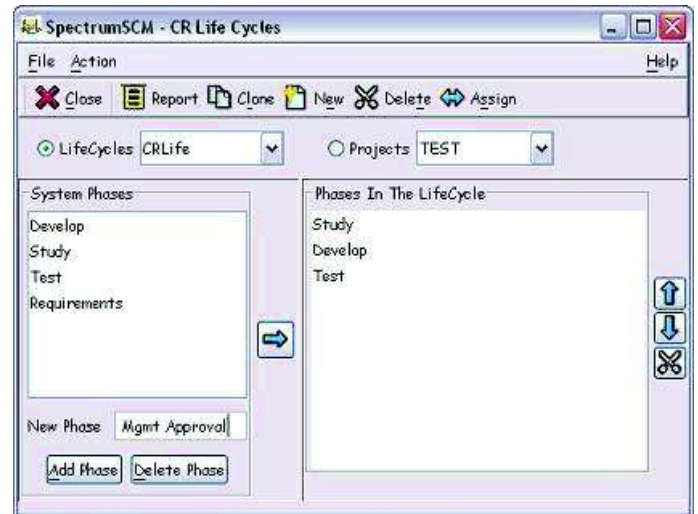
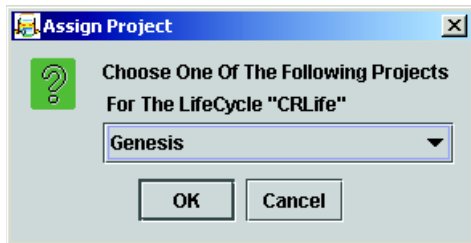


- **Clone** - to create a new life cycle from an existing life cycle.



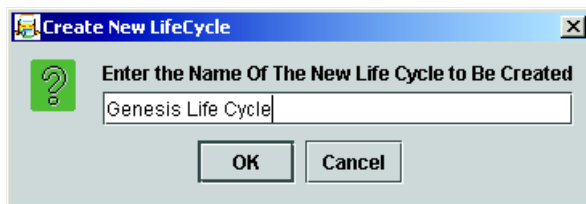
- **Delete** - to delete a life cycle. This can only be performed if no projects are using this life cycle.
- **Add phase** - to add a phase to the list of available phases.
- **Delete phase** - to delete a phase from the list of available phases.
- **Right arrow** - to assign a phase into a life cycle.
- **Up or Down arrows** - to re-sequence a phase within the life cycle.
- **Scissors** - to delete a phase from a life cycle.

The **Assign** button is used to assign a life cycle to a particular project and can be used regardless of whether the Lifecycles or Projects radio button is selected.



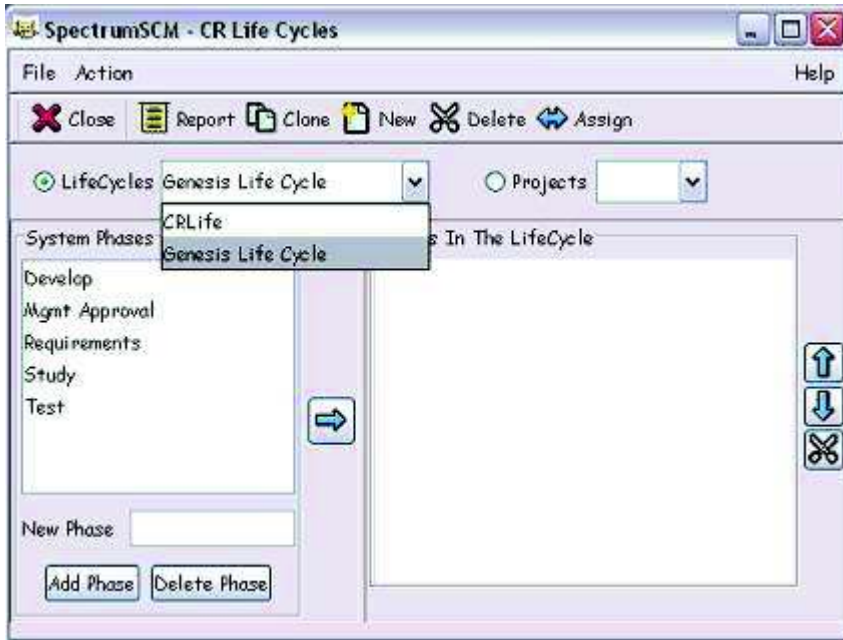
As an example, we'll set up the life cycle for the Genesis project. First, we will add two additional phases, **Requirements** and **Mgmt Approval**. To add a phase, type the name into the New Phase text field and click the **Add Phase** button, or simply hit the enter button when done typing.


Then create a new life cycle (we'll call it "Genesis Life Cycle"). Click **New** to add a new life cycle.

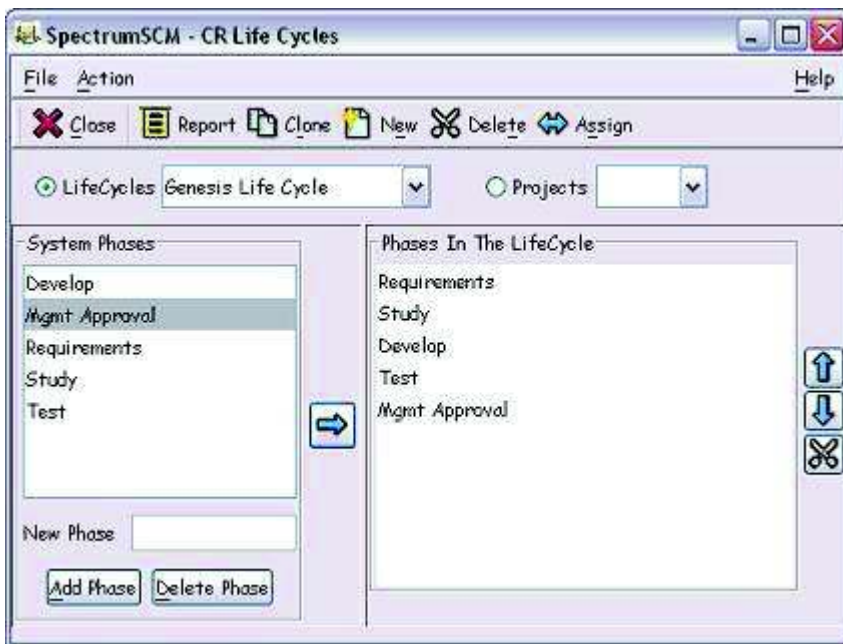


** NOTE **: Be careful modifying life cycles that already exist – they may be in use by other projects!




Select the life cycle to set up its phases.

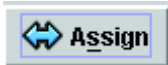


Select the phases you wish to add to this life cycle and click the  to move them into the **Phases In The LifeCycle** pane.

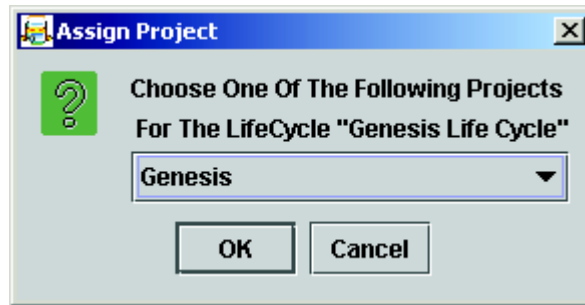




Use the   icons to order the phases and the scissors  icon to delete until you are satisfied with the phases and order.

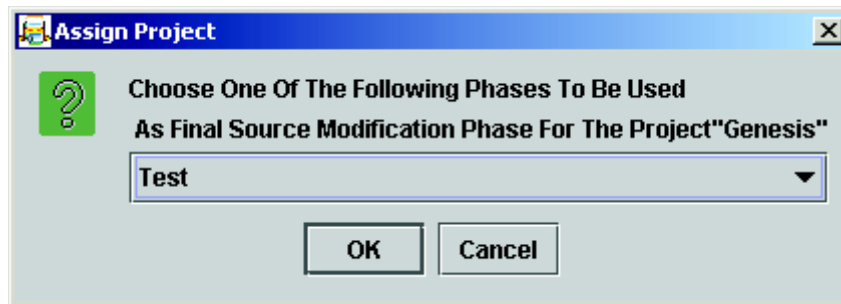


Use the  button to bring up the Assign Project screen and assign the chosen life cycle to the Genesis project.



You will then set the **Final Source Modification Phase**, the last phase during which source can be modified. It is not the last phase of the project; it may be next-to-last, but completing this phase signifies that the component is ready for inclusion in a release. **Generally, you select the phase that you want any issue, at the minimum, to have progressed past in order to allow a release to be built.**

This is an important decision. Once a component has been progressed past this phase, it is assumed to be done, tested, and ready for inclusion in a release. It will be “green-flagged.”

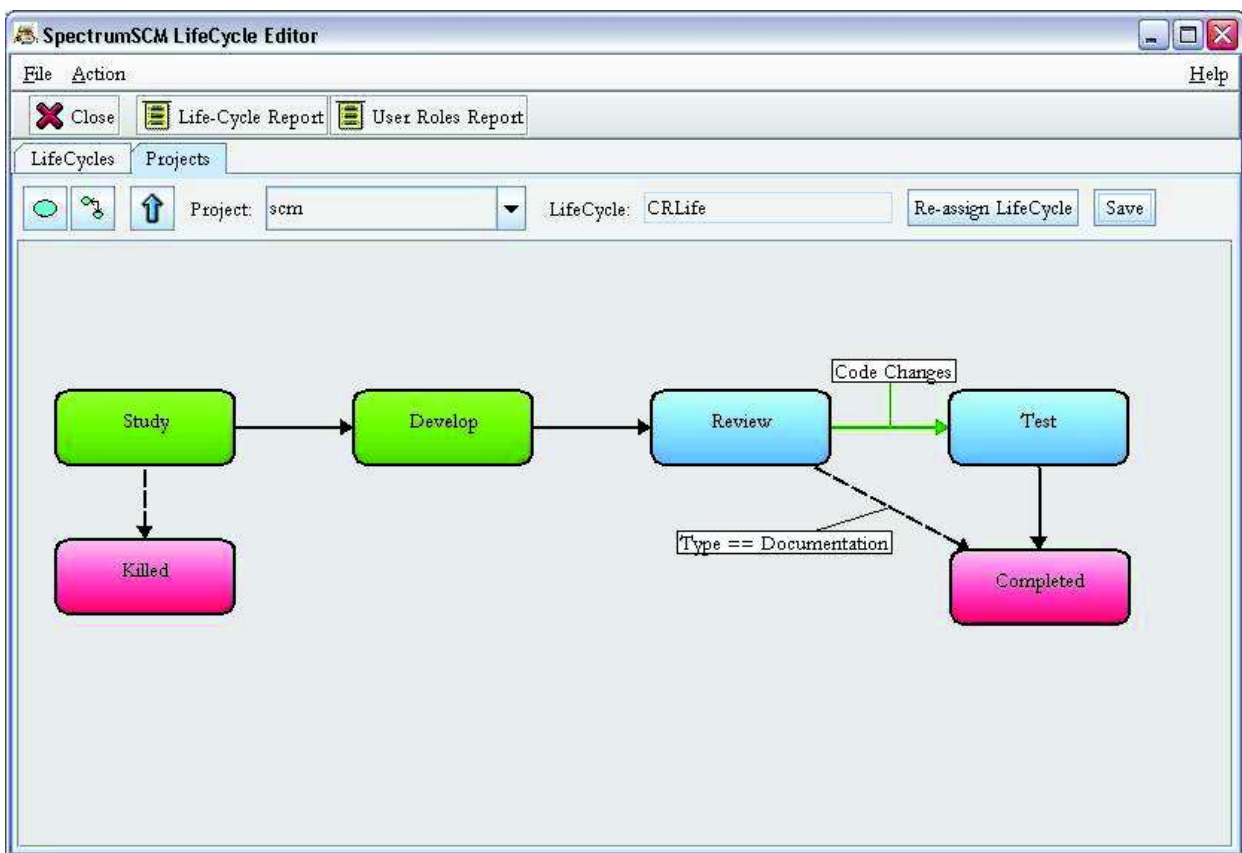


6.1.3.3 Life-Cycle/Workflow Administration

The workflow administration screen is an enhancement and replacement for the Linear Life-Cycle administration described above. All the Life-Cycle administration functionality is available under the Workflow screen. However, once the workflow functionality has been activated you cannot make changes in the Linear Life-Cycle administration screen. If you wish to revert to using the Life-Cycle administration functionality, you can, but the workflow data items will be archived.

The key functionality points for workflow administration screen are –

- The capability to specify valid transition paths, so that normal assignment tasks are focused and not complicated with unneeded options. For example, when assigning into a testing phase, only the testing users can be provided as options.
- The capability to automate transitions by specifying who is responsible for certain phases of the life-cycle.
- The capability to specify multiple transition paths, so that different types of Change Request can have different life-cycle paths.
- The capability to register callouts so that external functions/business rules can be implemented.
- Register custom e-mail triggers so that specific people or roles will receive mail about certain CR transitions but not others.
- The capability to state that particular phases are “Approval” phases, where the approver is prompted to approve (or reject) the CR.

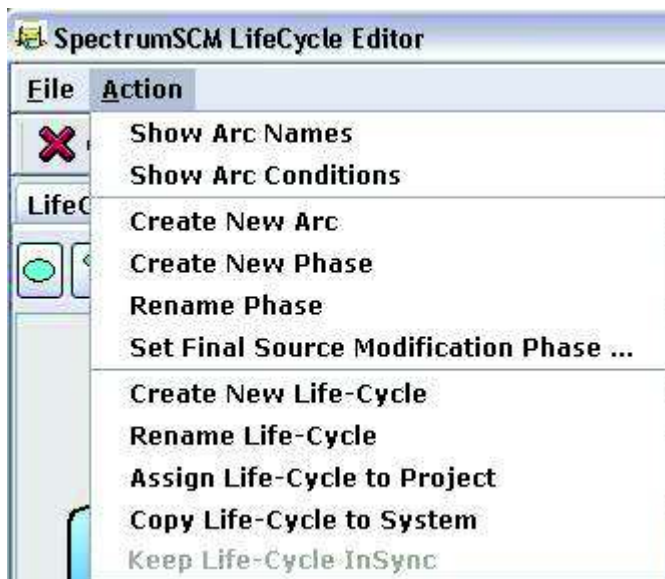


Life-Cycle/Workflow Administration Screen

Action menu item provides the following options –

- **Show Arc Names** – This is a toggle. If turned on, any arc names (like “Code Changes” above) will be shown. An arc name can be specified during arc creation or by using the right-click “Manage Arc Name” option.

- **Show Arc Conditions** – This is a toggle. If turned on, any arc conditions (like “Type == Documentation” above) will be shown. An arc’s conditional can be specified during arc creation or by using the “Manage Conditional” option on the arc right-click popup menu.
- **Create New Arc**
- **Create New Phase**
- **Rename Phase**
- **Set Final Source Modification Phase** – See below for Final Source Modification details and implications.
- **Adjust Phase Linear Order** – Since the graphical workflow can essentially define a 2 dimensional graph but yet certain application choices such as during CR creation or Assign/Modify present a linear choice-box, this option allows the ordering of the workflow to be adjusted. In general this should not be needed, but the functionality is provided in case it is needed or desired.
- **Create New Life-Cycle**
- **Rename Life-Cycle**
- **Assign Life-Cycle to Project**
- **Copy Life-Cycle to System**
- **Keep Life-Cycle InSync** – When a project change is made, a popup will ask whether this change is to be made “Common” i.e. for all the projects using this life-cycle, or “Uncommon” i.e. only for this specific project. This choice is “stored” through the “Keep Life-Cycle In-Sync” menu item, and can be changed (toggled) at a later time (if desired). **Note** – If the “uncommon” option is chosen, the life-cycle will be made specific to this project by creating a new life-cycle with the original name appended with the project name. For example: Life-cycle “CRLife” for project “abc” would become “CRLife_abc” when specialized.



There are 2 tabs on the Life-Cycle/Workflow Administration screen -

- The **“LifeCycles”** tab - This is where your system-wide life-cycle templates are defined. Projects can be assigned to life-cycles from this set. Toolbar buttons are available to create new life-cycles, delete life-cycles and clone existing life-cycles into new ones.
- The **“Projects”** tab - Where the assignment of a specific life-cycle to a specific project is made. Note that projects can use the system-level template definition as-is, or they can customize/extend it as appropriate for that specific project. If the project life-cycle is extended (made different from the template) then it will be given a new name to differentiate the project level from the template.

The two tabs together have a total of 6 toolbar buttons, 5 on the LifeCycles and 3 on the Projects tab.



Create New Life-Cycle.



Delete Life-Cycle.



Clone Life-Cycle – Create a new life-cycle from an existing one.



Create a new life-cycle phase (for both LifeCycles and Projects).



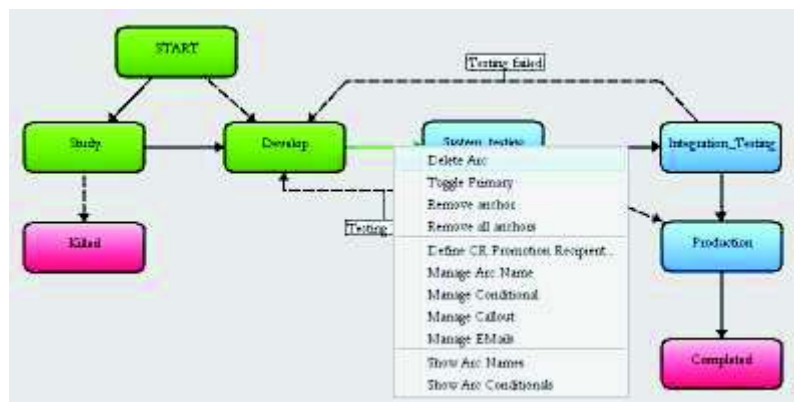
Create a new life-cycle arc/transition (for both LifeCycles and Projects).



Copy the current project specific life-cycle up to the system/template level thus making it available for other projects to use.

6.1.3.4 Phase Customization

Once a phase has been created (named) you can right-click on it to specify other attributes (or modify existing ones) if desired.



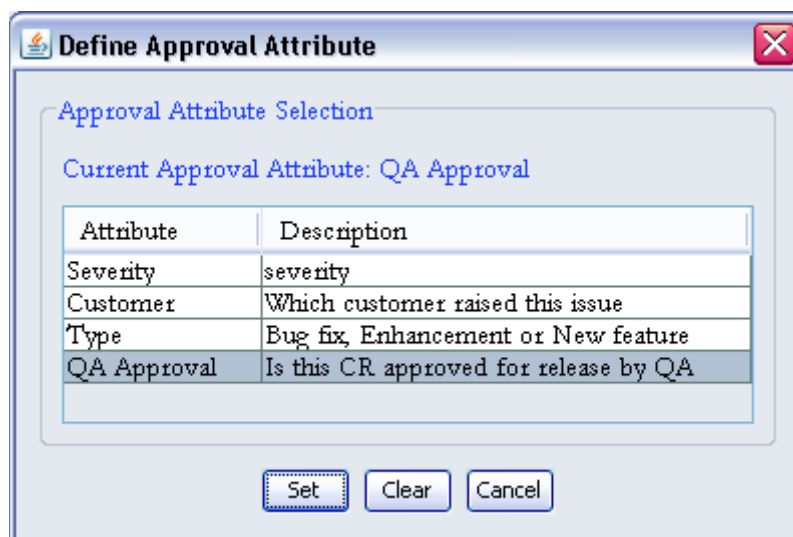
- **CR Assignment User Category** – To specify which user roles are to be available to be assigned tasks in this state.
- **CR Assignment Users** – To specify which individuals are to be available to be assigned tasks in this state. This works in addition to any assignment user categories specified above.
- **CR Promotion Recipient** – To specify who is essentially responsible for this phase i.e. if this is the “Testing” phase, then the probable promotion recipient would be the test team leader. The promotion recipient is who would be assigned tasks automatically coming into this state.
- **Approval Attribute** – If this phase is an approval phase, which CR attribute is to be used to record the approval state. See below for more details.
- **Final Source Modification Phase** - see below

Note that if there are no CR Assignment entries made then all the current project users will be presented as assignment choices.

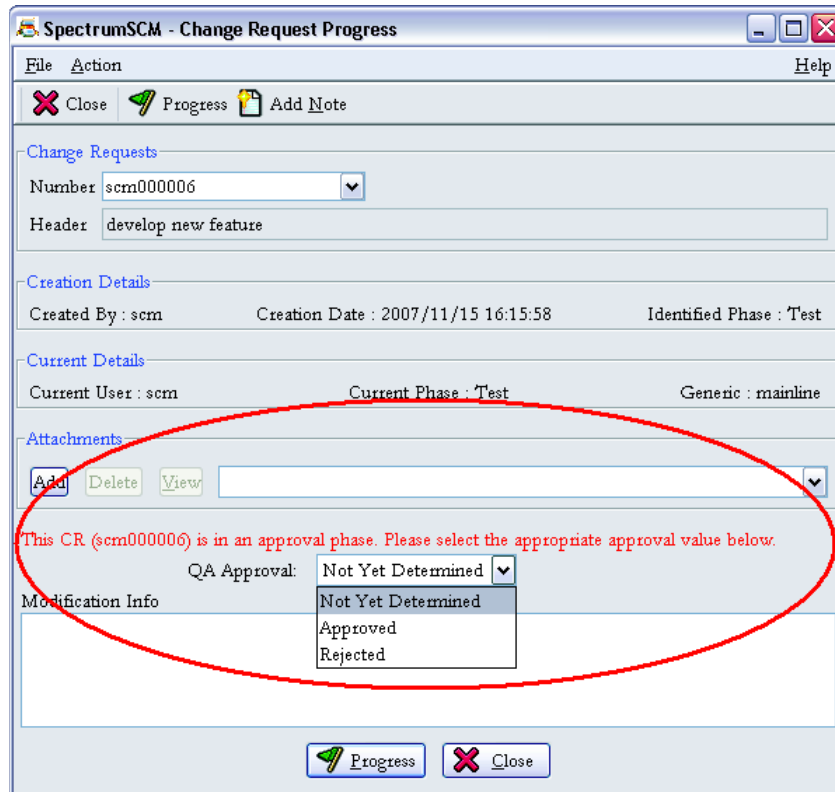
6.1.3.5 Approval Attributes

The approval mechanism is implemented by prompting the assigned user to update the approval state on the Change Request. This update, whether it be to a “passed” or “failed” state is recorded against a selected CR attribute such that it can be reported on, tracked and audited in a normal manner. Indeed, since attributes can be used to conditionally switch to different phases, such “passed” and “failed” conditions can be automatically applied to the workflow. In this way the CR can automatically transition to the appropriate success or failure phase.

As an example usage, a CR attribute called “QA Approval” could be created with values “Not Yet Needed”, “Approved”, and “Rejected”. Define the approval attribute for the “Test” phase to be “QA Approval”. Then whenever the CR is to be progressed from the “Test” phase the assignee will be prompted to set the attribute to the “Approved” or “Rejected” values. The workflow could then conditionally switch on this attribute to direct the CR forwards or backwards as appropriate.



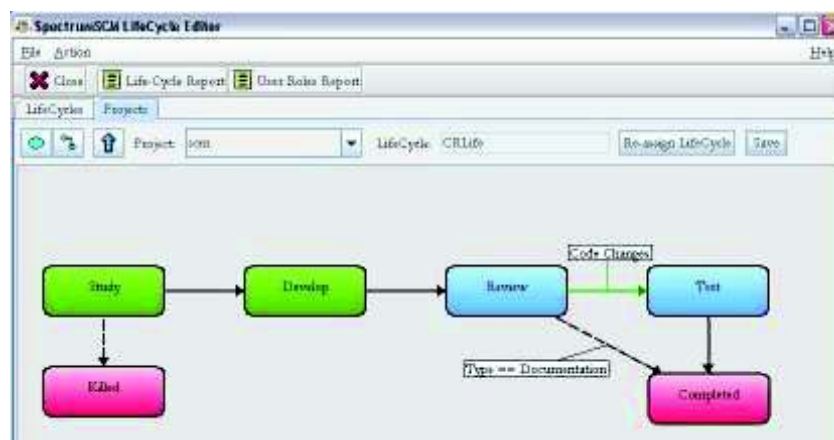
Select and “Set” the desired Approval Attribute as appropriate for the particular phase/state.



Since the approval is input through the progression screen, regular progression/ approval notes can be input at the same time before hitting the “Progress” button.

6.1.3.6 The Final Source Modification Phase

The Final Source Modification (FSM) Phase controls when Change Requests are considered eligible to be placed into a release. The final source modification phase must be on the primary path (shown with the solid line). All phases before the FSM phase will be shown in **green** to show that these CRs are OK for editing. All phases after the FSM phase will be shown in **blue**, CRs in these phases will be considered good to be placed into a release (subject to dependency checking).

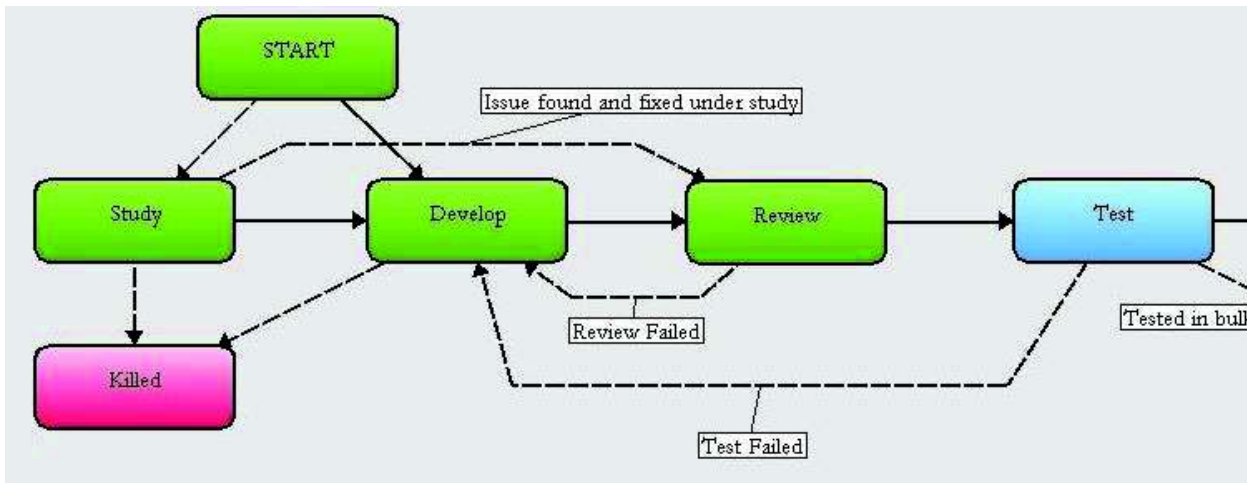


So for example in the LifeCycle/Workflow Administration screen shot above, the Final Source Modification phase can be seen to be “Develop”.

If phases on secondary paths (shown with dashed lines) need to be considered releasable then the “Set Past Final Source Modification Phase” option can be used.

6.1.3.7 The “START” Phase

If you want to constrain the CR creation process then you can define a phase with the name “START” (case sensitive). Arcs from this phase define the valid assignment/promotion paths and can even specify conditionals (if appropriate).



If such a phase is defined then the CR creation assignment will follow the assignment user and assignment roles subject to any conditionals that have been specified.

If no assignment is specified on the CR create screen, then the promotion rules will be evaluated to see if the newly created CR can be automatically assigned. If no assignment is specified and no automatic path is appropriate the CR will be placed in the “TBA” state.

6.1.3.8 Arc definition and customization

The solid arrowed lines indicate the primary transition path through the workflow. This is essentially the normal path work-items will flow. The dashed lines indicate secondary paths which can be taken if certain conditions (manual or automatic) are met.

To define a transition you can select the toolbar button. You can also select the source phase, or the source and target (use the shift key), and right-click to “Add Arc”. Any of these will present the following screen –

Create SDLC Arc

Source Phase: Destination Phase:

Conditional: ...

Callout: ...

EMails: ...

Arc Name: On Promotion Path


This is where you can specify if this transition only applies to certain types of Change Request based on the conditional constraints. By selecting the button to the right on the *Conditional* line, attribute options will be presented for specifying transition constraints. Only Change Requests whose attributes match the specified conditional expression will be considered valid for this transition.

Arc Condition Options

Attribute Name	Condition	Attribute Value
<input type="text" value="Select A Value"/>	<input type="text" value="=="/>	<input type="text" value="Select A Val..."/>

External callouts can be made by specifying the program name and any parameters. By selecting the button to the right on the *Callout* line, external callouts can be specified. Note that callouts will be executed on the SpectrumSCM server as this is the central control point, not on an individual users work-station. Also note that the callouts are executed with an additional parameter added, a file-name containing all the details of the change request and the transition being made.

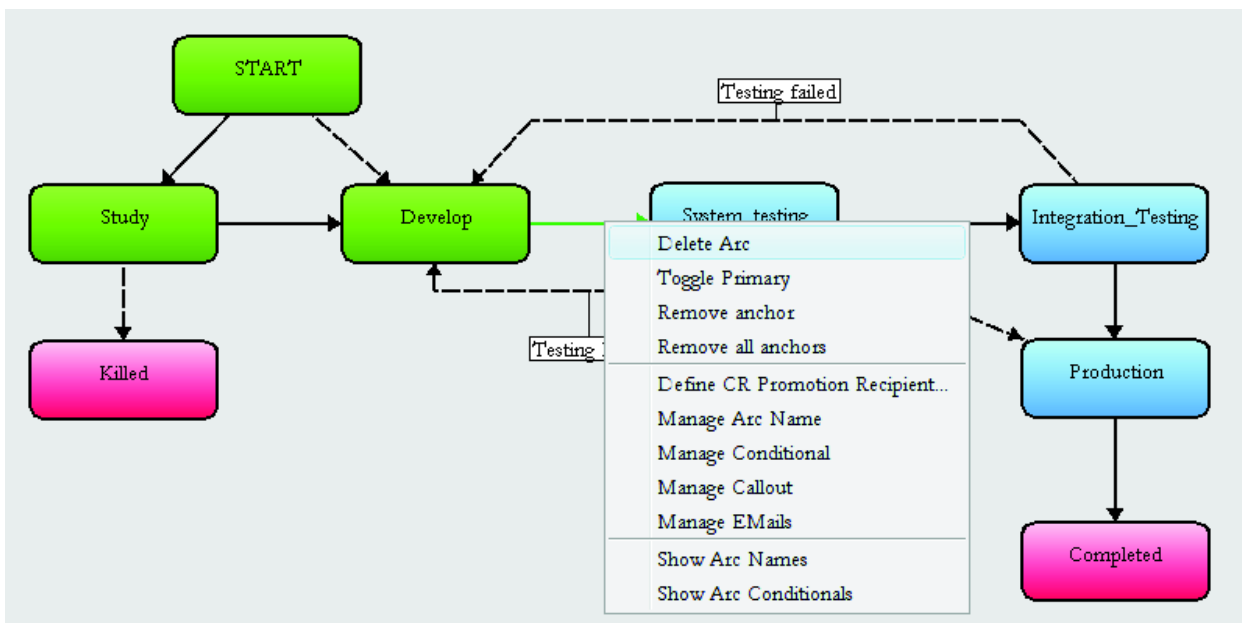


Selecting the  button to the right of on the *EMails* line allows browse and control of who should receive e-mail for this transition. Note that e-mail recipients can be selected by role or as an individual but they must have an entry under the “User Administration” screen (which gives their actual e-mail address).

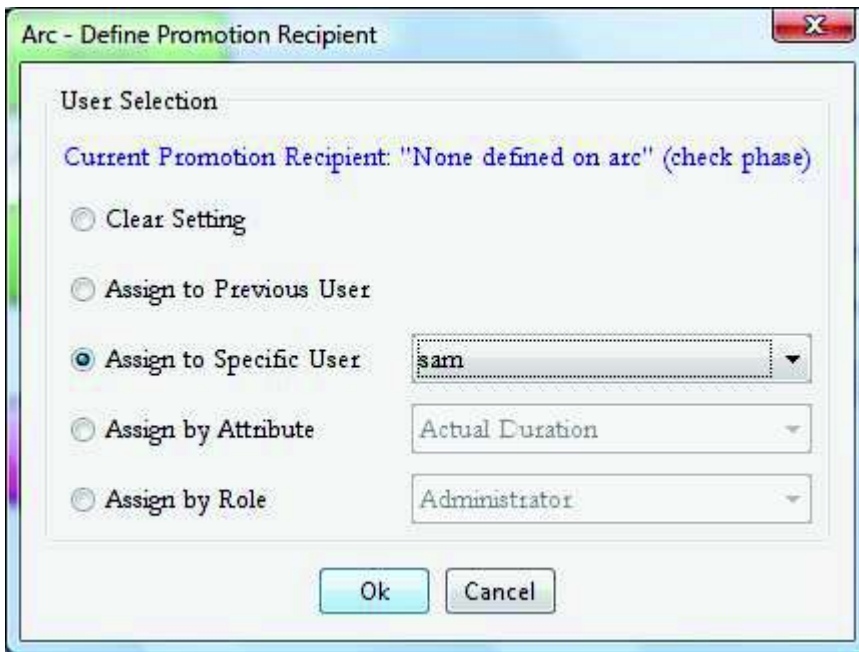
Once an arc has been created you can right-click on it to update any of these values. In addition to the functionality above, you can also -

- a. Toggle **labelling** on (or off) to show (or hide) either the arc name or the arc conditional expression.

- b. Specify an **Arc Promotion Recipient**, to further specify your automation rules.



When specifying an **Arc Promotion Recipient**, be aware that it will override the Phase based promotion recipient (if set). In this way the phase promotion recipient can be considered the default setting for the target phase, with the arc promotion recipient overriding that default, subject to the arc's conditional expressions.



In setting the Arc Promotion Recipient there are 5 options -

1. **Clear Setting** - Clears any existing Arc Promotion Recipient setting so that only the target phase promotion recipient will apply (if any).
2. **Assign to Previous User** - Determine which was the most recent user assigned this CR under the target phase and assign the CR back to that person. For example, if the CR is being rejected from a testing state and being moved back into development, the task probably wants to get assigned to the person who developed this item. Note - if no previous user is found for this target state, the CR will default to the phase promotion recipient.
3. **Assign to Specific User**
4. **Assign by Attribute** - This option is meant to allow pre-specification of an issues desired workflow routing. By setting up CR attributes such as "Responsible Tester", "Requirements Engineer", "Project Manager" etc (as appropriate in your business environment). These attribute values should either have the appropriate user ids specified or be editable (or both). The workflow will retrieve the value for the specified attribute and if it is a valid user for this project, the CR will be assigned to that person. If the attribute does not exist or its value does not specify a valid user id the CR will be defaulted to the phase promotion recipient (if any).
5. **Assign by Role** - If there is currently one person assigned to the specified role for this project then assign this CR to that person. If there is more than one person or there are no applicable user ids, then the CR will be defaulted to the phase promotion recipient.

6.1.3.9 Automation

If, when a CR is progressed, there is one and only one appropriate target phase as defined by the workflow arcs and any conditional expressions, AND the promotion recipient has been specified,

then that CR will be automatically progressed and assigned to the promotion recipient in the appropriate life-cycle phase.

Any callouts associated with the transition will be automatically executed. Any e-mails specified will also be sent. If this condition is not met, the CR will be placed in the To Be Assigned (TBA) state to await appropriate guidance/assignment.

6.1.4 User Category Setup

This topic is covered in more detail in Chapter 5.

6.1.5 Project User Setup

This topic is covered in more detail in Chapter 5.

6.1.6 Change Request Attribute Setup

This topic will be covered in more detail in Chapter 7.

6.1.7 First Change Request Creation

This topic will be covered in more detail in Chapter 7.

6.2 Setting up the local root directory or local work area

The **SpectrumSCM** system stores files internally in a directory structure that duplicates that of the native OS directory structure. A local work area is an area on the client workstation into which files can be extracted and extended without interfering with the files in the rest of the system or other workspaces. When files are extracted from the SpectrumSCM system into client workspaces, they are written into the receiving file system using that same directory. If the necessary directory structure does not exist at the time of the extraction, the directories are created as necessary.

The Local Root Directory defines the location on the client hard disk where files to be checked in are found and files extracted to the hard drive are placed. This is needed so that files stored under SCM have only relative path names. An example "root directory" would be the directory in which you perform your local product builds or compiles. Another example would be a directory that you use when you are loading source into SCM from the directory.

Subdirectory names in all local root directories need to match the SpectrumSCM file structure in the project tree. Note that you can have multiple root directories if needed. Select the one you wish to work with via the Local Root Directory pull-down menu selection box.

For example, in a Windows environment for the file path C:\temp\src\test.C, C:\temp would be the local root directory. In a Unix or Linux environment, for the file path /temp/src/test.C, the local root directory would be /temp.



For our example project, we have chosen to set the local root directory to c:\temp. Each “scm” project team member will create a folder by the same name on his or her client machine. Additional local root directories can be created by each user as needed and selected for use via the local root directory pull-down menu.

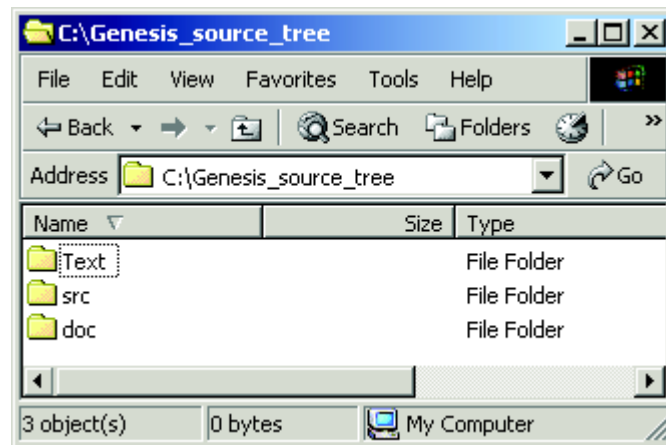
6.3 Setting up the project directory structure in SpectrumSCM

It is a good idea for the project engineer or generic engineer to set up the basic file structure for the project before development begins.

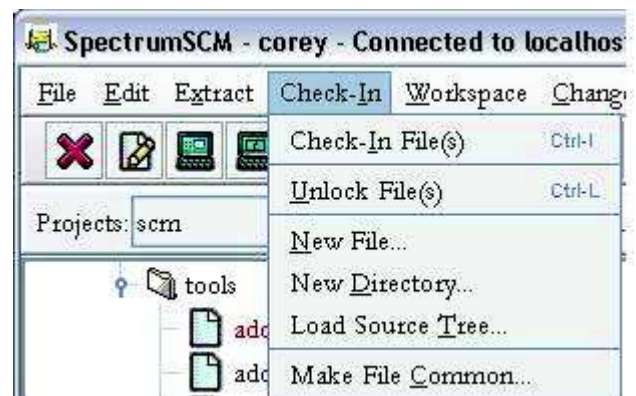
This can be done using the **Add Source Tree** screen. The Add Source Tree screen is accessed via the Main Screen **Check-In / Load Source Tree** menu option.

A CR has to be created to load the source tree (in our example, this is done under the “set up project in SCM” CR). *See Chapter 7 for details on creating, assigning, and progressing CRs*

The structure to be loaded must be in a local root directory (in this case, C:\Genesis_source_tree).



On the main screen, select the CR, the project, and the local directory containing the project structure. Access the Add Source Tree screen via the **Check-in / Load Source Tree** menu option.



The **Add Source Tree** window will be displayed.

The SpectrumSCM system will look in the specified local root directory and move its contents into the SpectrumSCM system into the corresponding project and generic directory (in this case, Mainline under the project Genesis).

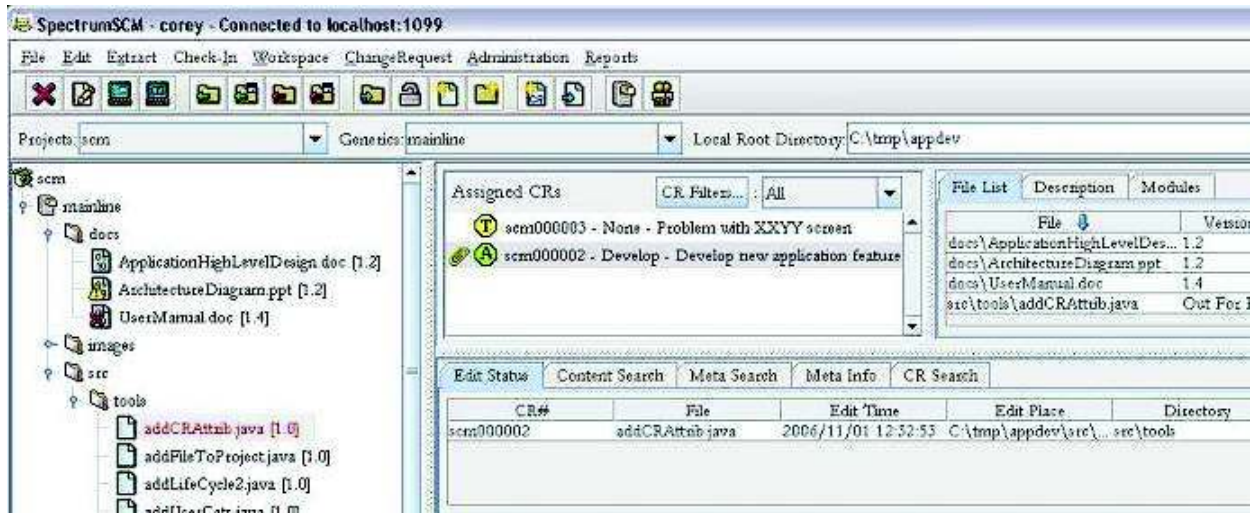


Specify **Suffix** to load only those files with a specific suffix (for example .java, .doc, .txt)

The system will automatically detect the “type” of a file (binary or text) as it is being loaded into the system. This decision can be overridden by the user by toggling the **Auto Detect** checkbox, which then enables the **Load As** (ASCII, BINARY) check boxes. By using these check boxes the user can force the system to load files of any type as another type. For instance, binary files can be checked in as text and large text files can be checked in as binary.

Specify **Recursive** if there are subfolders to be loaded. Specify **Include Binaries** if there is any non-ascii source to be loaded as part of the initial set-up. Specify **Empty Directories** if the system should add empty directories to the repository during the load.

The **Add Source Tree** function is also used to move an existing baseline source tree into SCM if you are migrating to SpectrumSCM from another CM tool or manual file versioning. To establish the structure for a new project, create a source tree of empty directories with appropriate folder names and move those into SpectrumSCM using this function. When the load is completed, a confirmation is displayed. The loaded project tree structure can be seen via the Main Screen Project tree.



Alternatively you can use the **DragNDrop feature** available in SpectrumSCM to check-in or load a single folder or an entire folder structure by simply **Dragging** the files or folders that are of interest directly from desktop or file system and **Dropping** it into appropriate folder location on the project repository window panel. This mechanism can also be used to check in individual files as well.

This feature basically automatically populates load source tree or the add source file screen with the proper source and target location without the user having to set the local root directory or screen settings.