Roundtable Task Management

Randall Harp
Roundtable Product Architect

TUGBOAT
SOFTWARE

Table of Contents

1 Introduction

Managing a set of changes through the life cycle often poses a challenge to all members of a software development team. Software modifications to fulfill a change request, for example, must be identified, coded, tested, and then delivered to the end user(s). With most version control tools, promotion of changes through the development life cycle depends upon workflow systems superimposed upon the basic structures through policies practiced by the development staff, rather than constructed within the repository data itself.

Without the benefit of change identification and life cycle promotion flows within the code repository data, development shops are burdened with the extra effort of managing the identity and movement of related changes using external methods. To the degree that a change management tool can identify a set of changes throughout the development life cycle, and assist with the movement of those changes, a development team's productivity can be increased.

This white paper explains the benefits of task management, and how these features, as implemented in Roundtable TSMS, can be utilized to effectively manage the development, testing and promotion of software changes through the development life cycle.

2 Task Management Explained

2.1 Task-less Development

Before understanding the characteristics and benefits of task management, it is important to see the need for such a feature in a change management tool.

In the typical version control environment, a programmer makes changes in his or her own copy of the application. In order to make these changes available to a quality assurance (QA) team for testing, the programmer must commit the changes to the version control repository – perhaps in a branch created for the purpose of isolating these changes. With tools such as CVS or Subversion, each commitment of changes increments the revision of the code repository. In order to test the changes, each QA team member must then update their own copy of the application from the branch, based upon the particular version of the repository. The QA team members must depend upon commit notes entered by the programmer, or direct communication from the programmer in order to identify the revision level from which to populate their individual work environments.

Should a change fail testing, the programmer will have to commit corrections under another revision number, and the QA team members, again, will need to update their environments with the latest changes. This process is repeated as many times as necessary for all the changes to pass the testing phase.

2.2 Task-based Development

To prevent the cycle of developers making changes, committing those changes to the code repository, and then QA team members pulling down updates, and having to repeat that cycle if any one of the changes fails testing, a software development management system should provide the following features:

- **Change promotion without Commitment** – Changes can be promoted to a test environment without first being committed to the repository. This allows re-work on a set of changes until the changes pass appropriate testing. Changes can be both promoted and retracted without changing the revision level of the repository until the changes are accepted.
- **Persistent Label and Comments** – A set of changes made to satisfy a particular requirement should have a permanent label and comments, so that development staff can understand what changes were made, and why they were made, throughout the development lifecycle. Regardless of which branch or release the changes are incorporated in, developers, testers, and mangers can quickly identify the requirement prompting the change – even if only one resource from the set of related changes is incorporated into a particular configuration. With proper comments attached to the change set, QA team members know what to test, and can more easily troubleshoot defects in software releases.

3 Tasks Management in Roundtable TSMS

Roundtable TSMS (Total Software Management System) provides all of the features described in the previous section. A Roundtable Task defines a unit of work to be done in a Configuration (of the application). Application resources are created and modified under an active Task.

The benefits of Tasks in Roundtable are that they:

- Help developers and testers audit work done in logical units.
- Facilitate check-in of many changes at once.
- Give managers access to important information about the work being performed in the Configuration.
- Allow programmers and managers to quickly see what others are doing in the Configuration.

The following attributes make up a Task in Roundtable:

- **Configuration** - Configuration in which the Task resides
- **Summary** - High-level description of the Task
- **Manager** - The designated Task manager
- **Programmer** – One or more programmers assigned to the Task
- **User Ref** – Optional reference to an external entity, such a change request identifier.

- **Status** - The Task status. Either Work in Process(W) or Complete(C)
- **Completed** - The date the Task was completed
- **Share Status** - Default Share Status of Objects checked-out to this Task. More on this below
- **Directory** - Full path to a Task folder. This must be filled in when a Share Status other than 'Central' is chosen for the Task. More on this below
- **Description** - Detailed description of the work to be performed in the Task.

3.1 Share Status: Leveraging the PROPATH

A beneficial feature of both the OpenEdge compile and runtime environments is the session PROPATH. Since the search order for referenced resources can be programmatically manipulated at runtime, Roundtable can take advantage of this feature to facilitate Task management.

The utilization of the OpenEdge PROPATH is at the heart of Roundtable's Share Status. The Share Status of a resource determines where the changes to that resource are physically located and, consequently, to which team members they are available.

There are four Share Statuses:

- **Central** – Changes to the resource are saved in the Configuration folder. When a configuration is selected in Roundtable, the Configuration path is first in the PROPATH – the default PROPATH for Roundtable users.
- **Task** – Changes to the resource are saved in the folder identified for the Task. This is often a folder on the programmer's local PC. When the Task is selected, the Task's path is first in the PROPATH.
- **Group** – Changes to the resource are saved in the folder identified for the Task, but the programmer can copy the changes to a "Group" folder using an 'Update Group/Public Source' option. A Task may belong to a Task Group, which is an association of one or more Tasks with a particular folder. The Group folder allows testers to test changes contributed from one or more Tasks, by placing the Group folder first in the PROPATH.
- **Public** - Changes to the resource are saved in the folder identified for the Task, but the programmer can copy the changes to the Configuration folder using an 'Update Group/Public Source' option.

4 Maximizing Task Management

The following paragraphs illustrate one method of utilizing Roundtable's Task Management features. This method is neither specifically prescribed nor required, but demonstrates advantages of the features.

4.1 Task Group Setup

In order to facilitate the setup of testing environments, project managers can create a permanent Task Group in the Configuration to be tested. The folder for the Task Group should be a network shared folder, accessible to the Configuration programmers, and all members of the appropriate QA team.

With the Task Group in place, testers can have a shortcut to launch their application that has the Task Group path at the front of the PROPATH, followed by the Configuration path. This will allow the same shortcut to be used for all testing, as Tasks are added and removed from the Task Group.

4.2 Task Creation

Roundtable requires that an active Task be selected for any changes to be made to a Configuration. The manager or programmer creating Tasks in the Configuration should add the Task to the Task Group described above. The Task should be created with a Share Status of "Task", which means that as resources are checked-out by the programmer(s), the editable source will be placed in the Task folder defined for the Task. The Task folder can be a folder on the programmer's local PC (more typical) or a network share that is more likely to be backed-up on a daily basis.

4.3 Testing Changes

When a programmer is ready for his or her changes to be tested, the changes do not have to be committed to the repository. By changing the Share Status of the Task resources to "Group", the changes will be copied to the Task Group folder. Once there, testers can simply launch their shortcuts with the appropriate PROPATH to test the changes against the Configuration.

Furthermore, if the testers have access to the Task data (which they do through Roundtable) they can quickly access the programmer's notes for both the Task and the individual resource versions, so that they will know what tests need to be done. The Task manager may even create a test plan under the Task (which can be for internal use only by marking it a non-deployed object in Roundtable).

If any of the changes do not pass testing, the programmer can simply change the Share Status of those resources back to "Task". They will be automatically removed from the Task Group directory, and the programmer will again be making changes isolated from other team members, until they are ready for another round of testing. This cycle can repeat as often as necessary. During this phase, the programmer never has to commit changes to the repository, and the testers need to do nothing to update their test environments.

4.4 Completing Changes

When the changes have been approved by the QA team, the programmer or Task Manager changes the Share Status of all resources to "Central" causing them to be moved to the Configuration folder. The Task manager then removes the Task assignment from the Task Group, reviews and completes the Task (causing changes to be committed to the repository).

No matter how many changes were necessary to the resources under the Task, the versions created under the Task are forever associated with the Task – allowing the changes to be easily identified, and even promoted through the life cycle (using Roundtable's Import feature, filtered by Task).

5 Conclusion

Roundtable's Task Management can boost the productivity of development teams by significantly reducing the effort needed to identify a set of changes, and eliminating the repetitive cycle of committing changes to a repository and updating of individual testing environments that is common with simple version control tools.