Synergy Advanced Topics
Component Development

Martin Heinrich
Principal Consultant, IBM Rational
Martin.Heinrich@au1.ibm.com

Rational. software

Go to IBM

Agenda

- Module 1 - Elements of Telelogic Synergy Process Design (7.1)
- Module 2 - Patterns for Component-based Development

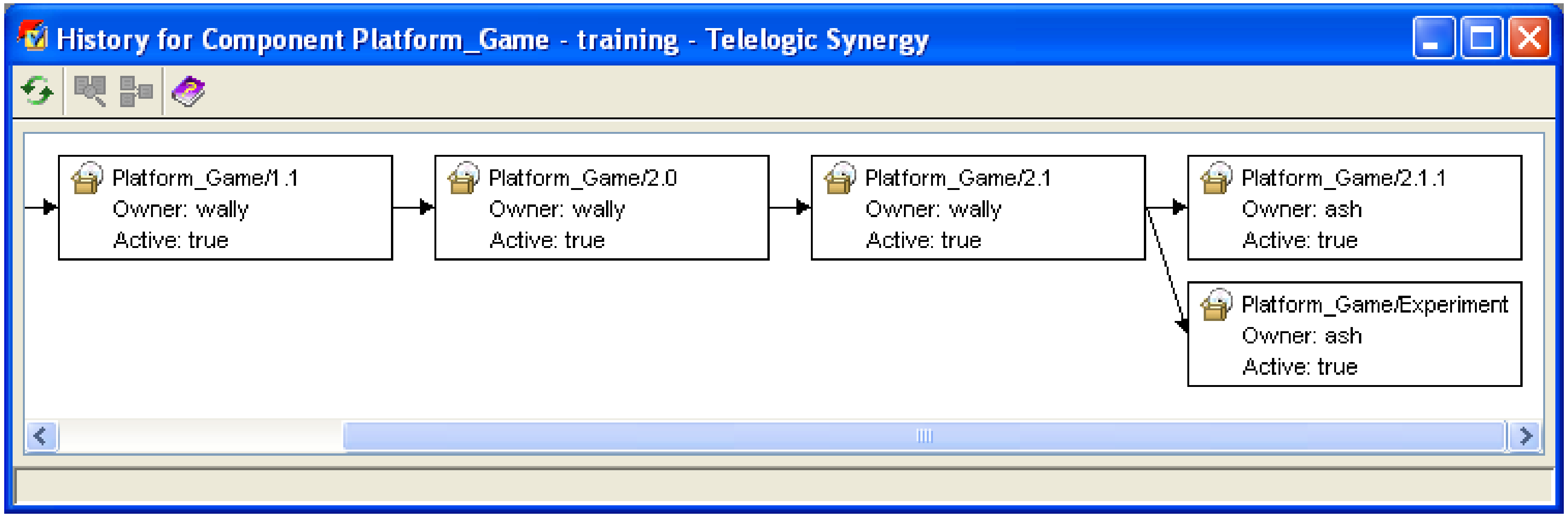
Elements of Process Design

- Releases
- Processes
- Purposes
- Process rules
- Project groupings
- Folder templates
- Update
- Use Operation and Current Task
- Component Tasks
- Mixed Release Baselines

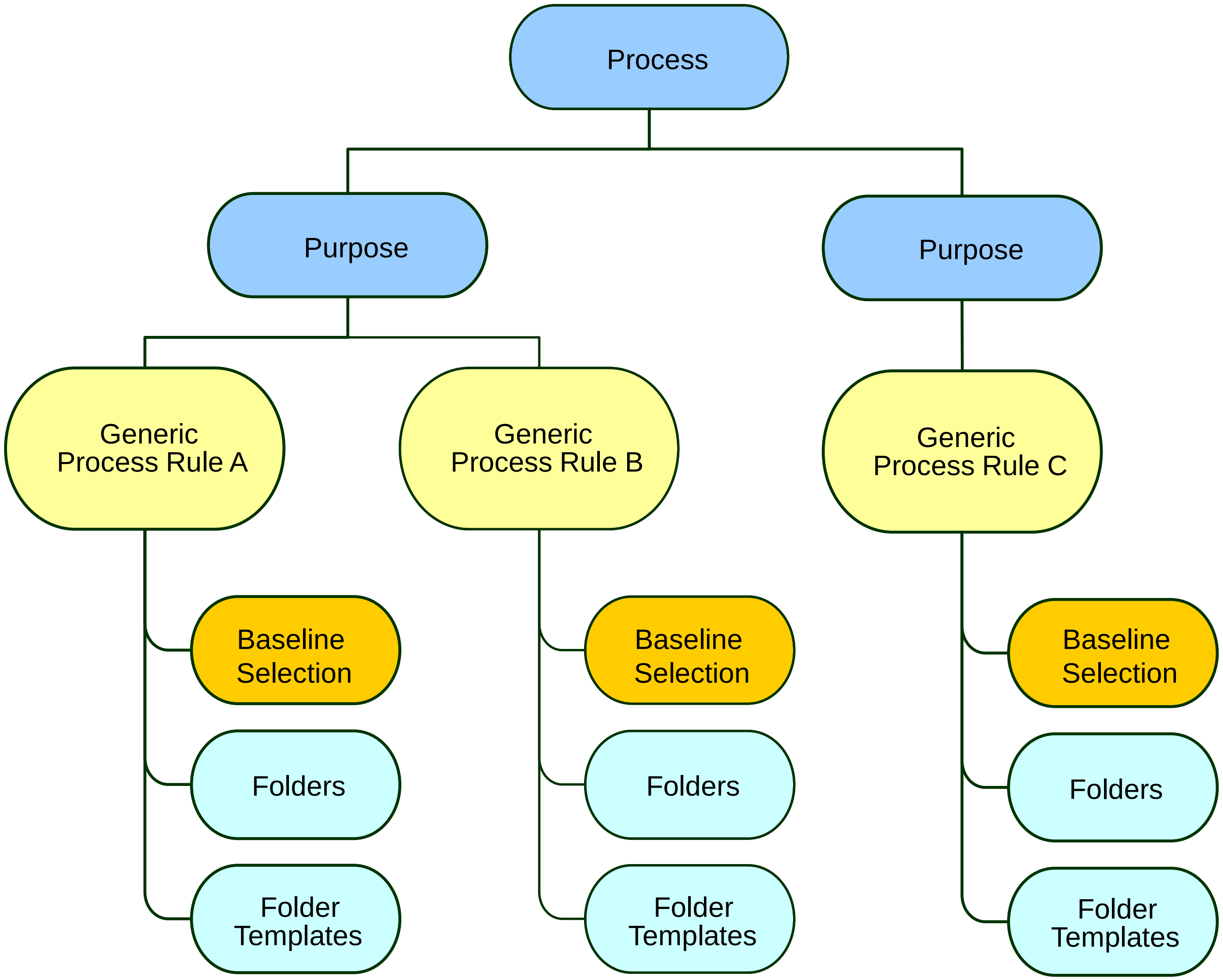
Releases

Recall that a ***release*** represents a stream of development for a particular program or set of related programs being developed for a deliverable (e.g., being shipped to customers).

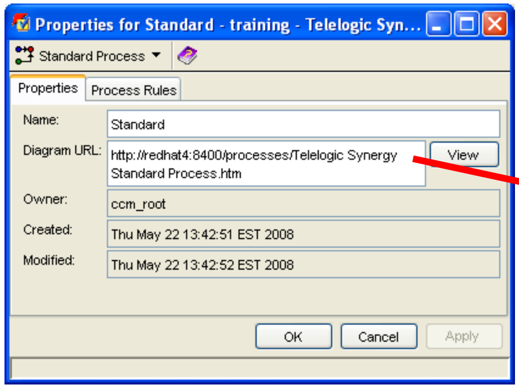
Release streams may represent products that were already shipped, or may be actively being developed. Some releases may even be developed in parallel, as shown in the release history for Platform_Game.



Processes and Process Rules



Processes: Diagram



Properties for Standard - training - Telelogic Syn...

Standard Process

Properties | Process Rules

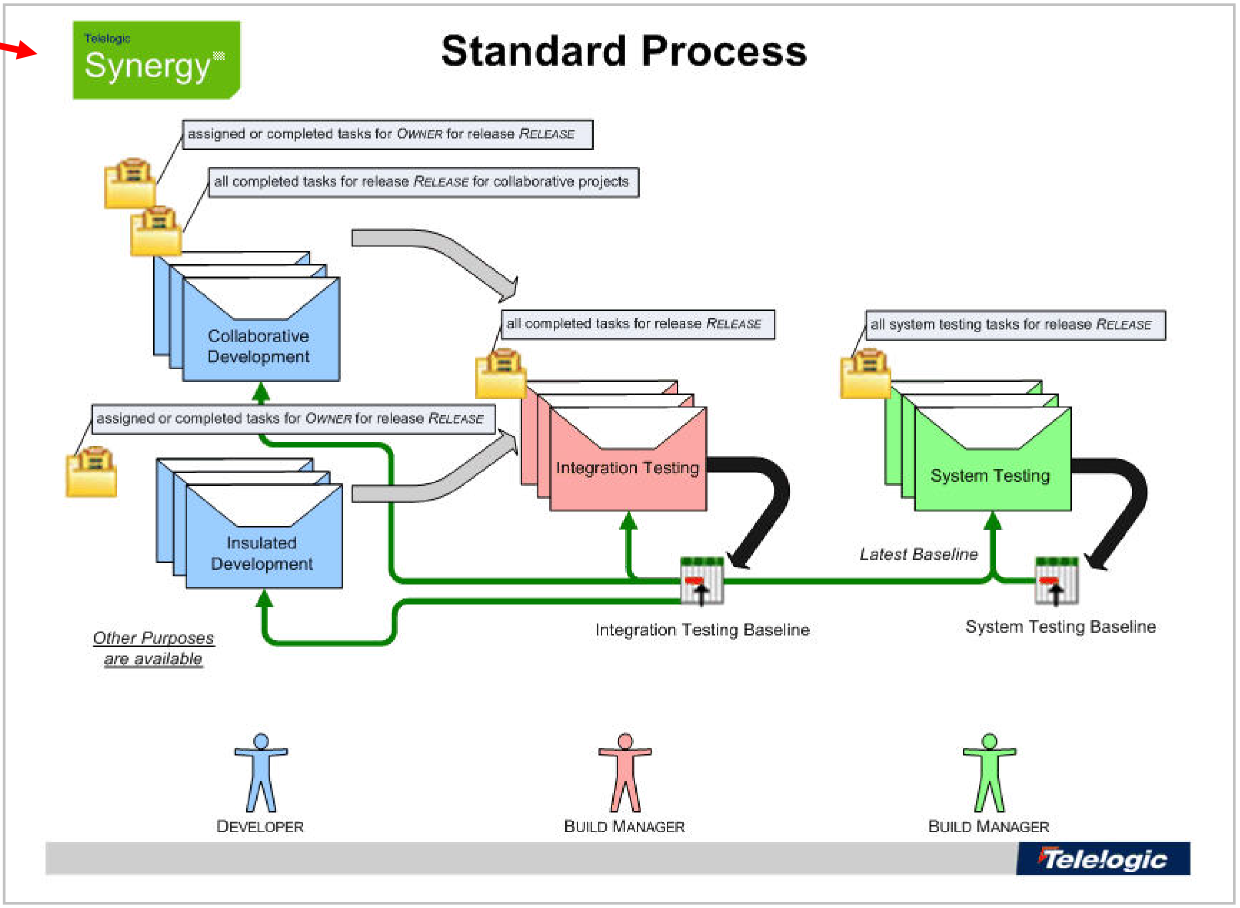
Name: Standard

Diagram URL: http://redhat4:8400/processes/Telelogic Synergy Standard Process.htm

Owner: ccm_root

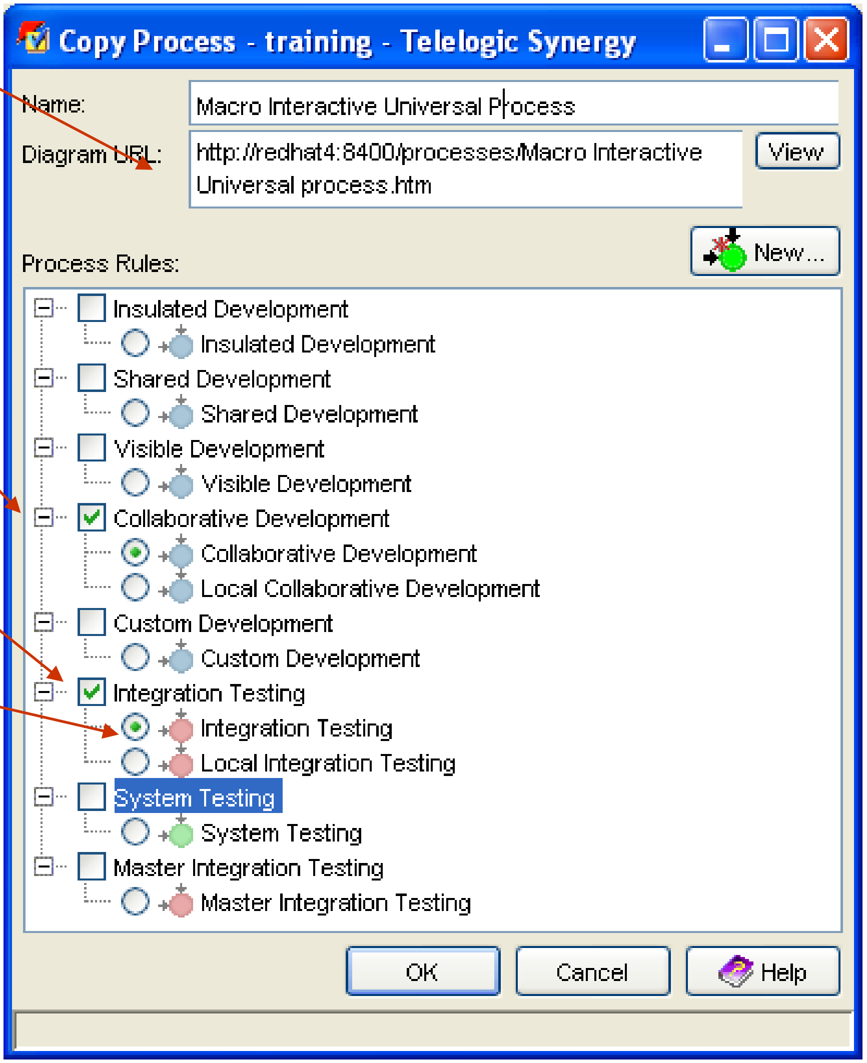
Created: Thu May 22 13:42:51 EST 2008

Modified: Thu May 22 13:42:52 EST 2008


Telelogic Synergy
Standard Process
assigned or completed tasks for OWNER for release RELEASE
all completed tasks for release RELEASE for collaborative projects
Collaborative Development
all completed tasks for release RELEASE
all system testing tasks for release RELEASE
assigned or completed tasks for OWNER for release RELEASE
Integration Testing
System Testing
Insulated Development
Latest Baseline
Integration Testing Baseline
System Testing Baseline
Other Purposes are available
DEVELOPER
BUILD MANAGER
BUILD MANAGER
Telelogic

Processes and Process Rules Built-in Processes

- See the URL for the manually created diagram.
- Select purposes for the release.
- Select which process rule should be used for the available purposes.
- Note: You can have multiple process rules for each purpose.



Copy Process - training - Telelogic Synergy

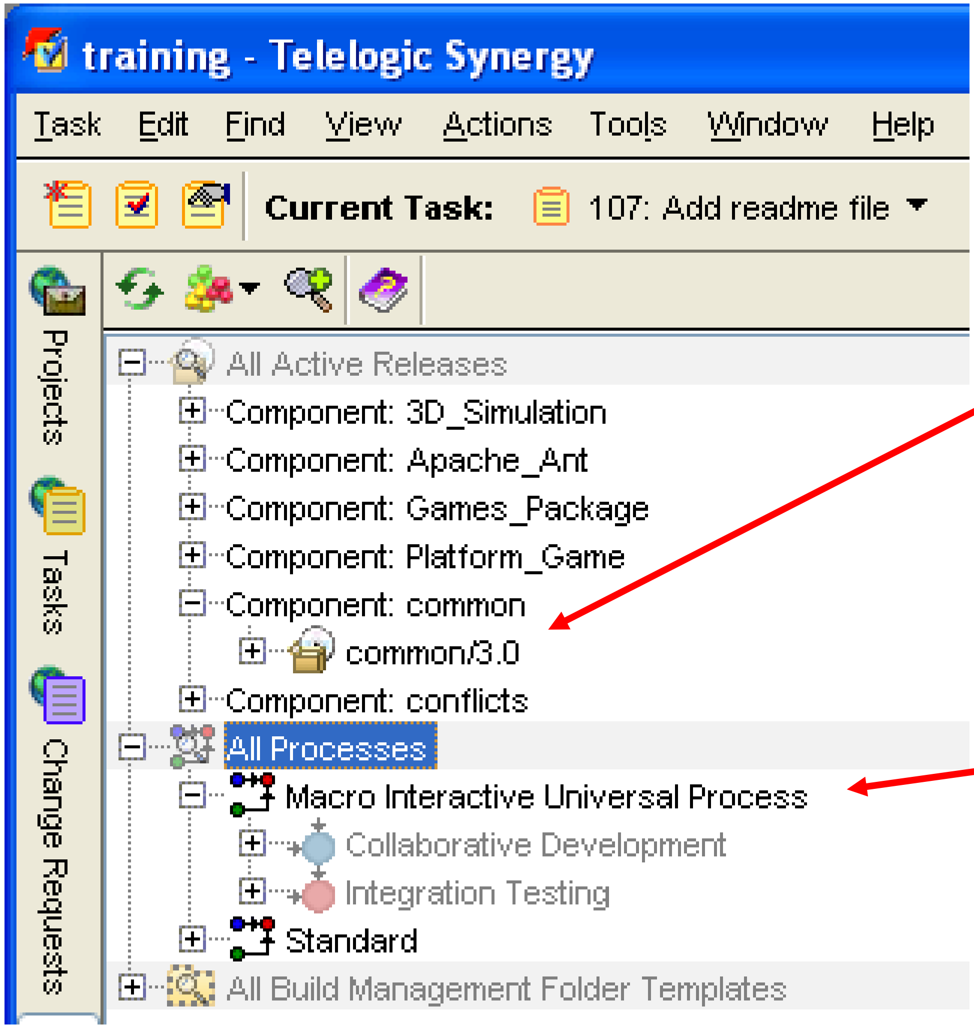
Name: Macro Interactive Universal Process

Diagram URL: http://redhat4:8400/processes/Macro Interactive Universal process.htm

Process Rules:

- ☐ Insulated Development
  - ○ Insulated Development
- ☐ Shared Development
  - ○ Shared Development
- ☐ Visible Development
  - ○ Visible Development
- ☑ Collaborative Development
  - ● Collaborative Development
  - ○ Local Collaborative Development
- ☐ Custom Development
  - ○ Custom Development
- ☑ Integration Testing
  - ● Integration Testing
  - ○ Local Integration Testing
- ☐ System Testing
  - ○ System Testing
- ☐ Master Integration Testing
  - ○ Master Integration Testing

Processes: Generic and Release Specific



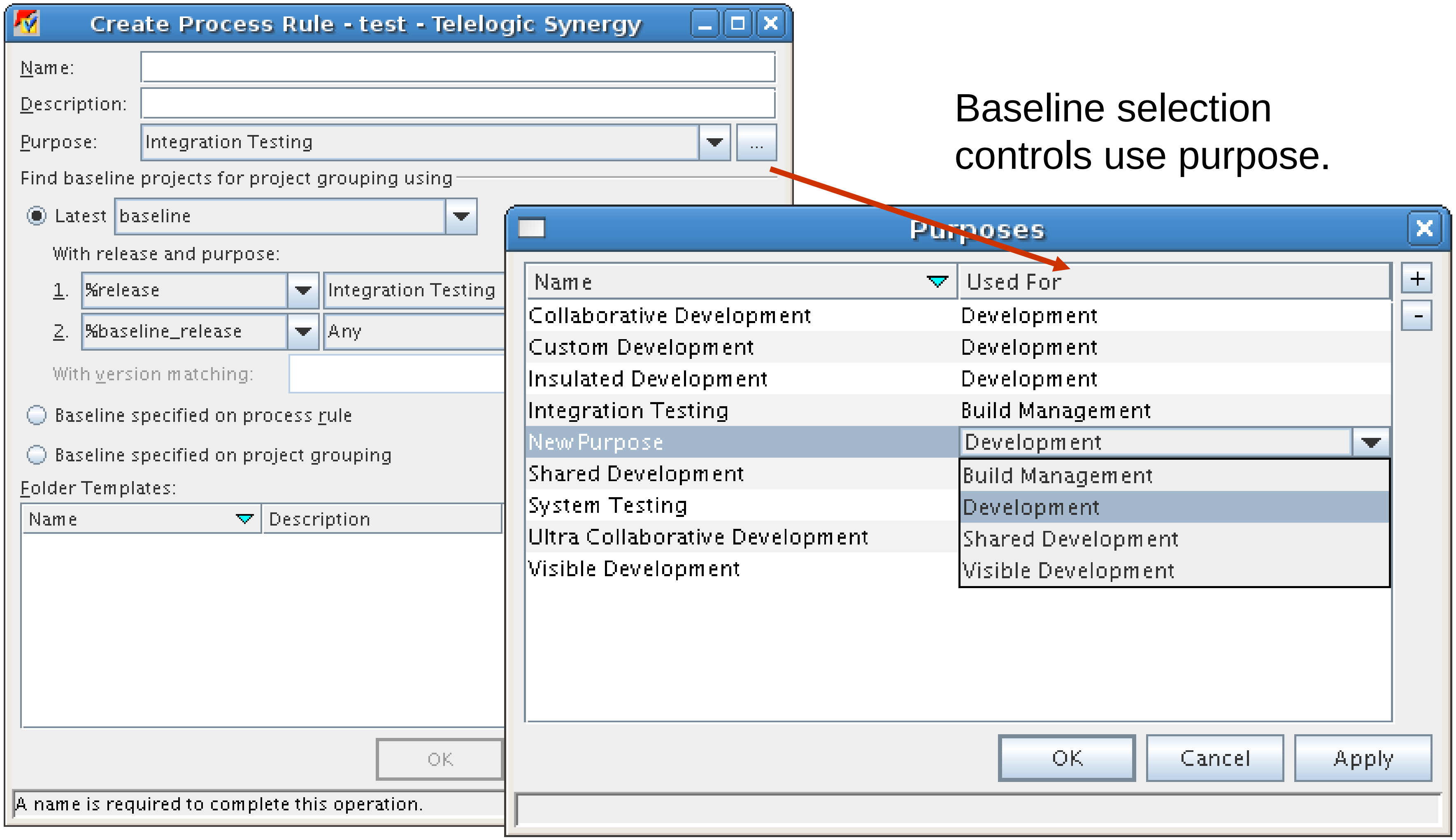
Release-specific process rules:
Change the process for a particular release.

Generic process rules:
Change the default process to be used in future releases.

Purposes

- The **purpose** describes what the project will be used for.
- Every project version has exactly one **purpose**.
- Purpose does the following:
  - Defines the state of the project.
  - Maps the process rule to the project.

Purposes


Create Process Rule - test - Telelogic Synergy
Name:
Description:
Purpose: Integration Testing
Find baseline projects for project grouping using
● Latest baseline
With release and purpose:
1. %release Integration Testing
2. %baseline_release Any
With version matching:
○ Baseline specified on process rule
○ Baseline specified on project grouping
Folder Templates:
Name Description
OK
A name is required to complete this operation.
Purposes
OK Cancel Apply

Name	Used For
Collaborative Development	Development
Custom Development	Development
Insulated Development	Development
Integration Testing	Build Management
New Purpose	Development
Shared Development	
System Testing	
Ultra Collaborative Development	
Visible Development	

Dropdown: Build Management, Development, Shared Development, Visible Development

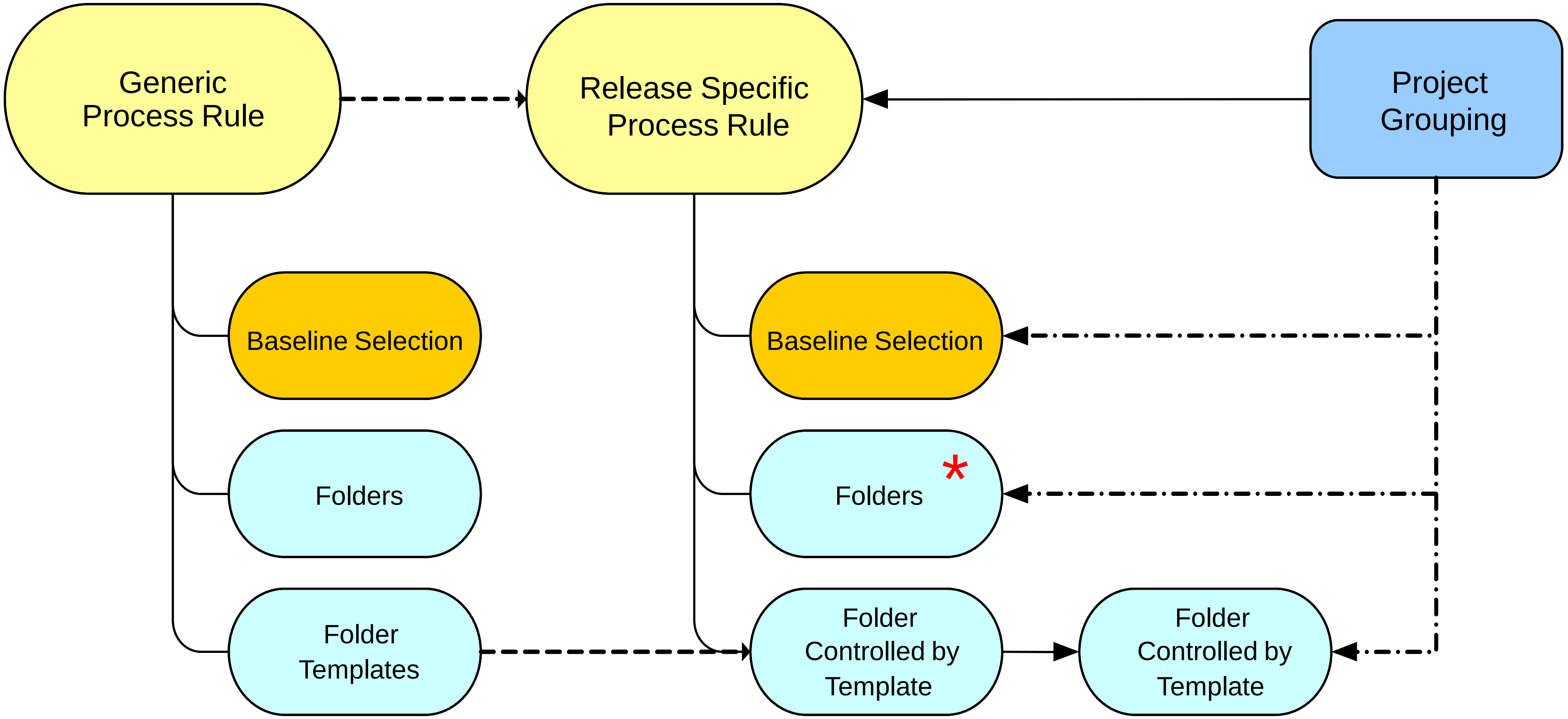
Baseline selection controls use purpose.

In the Create Process Rule you can click on the '…' to control your purposes
Keep the number of process rules to purpose reasonable.

Purposes: What they are used for

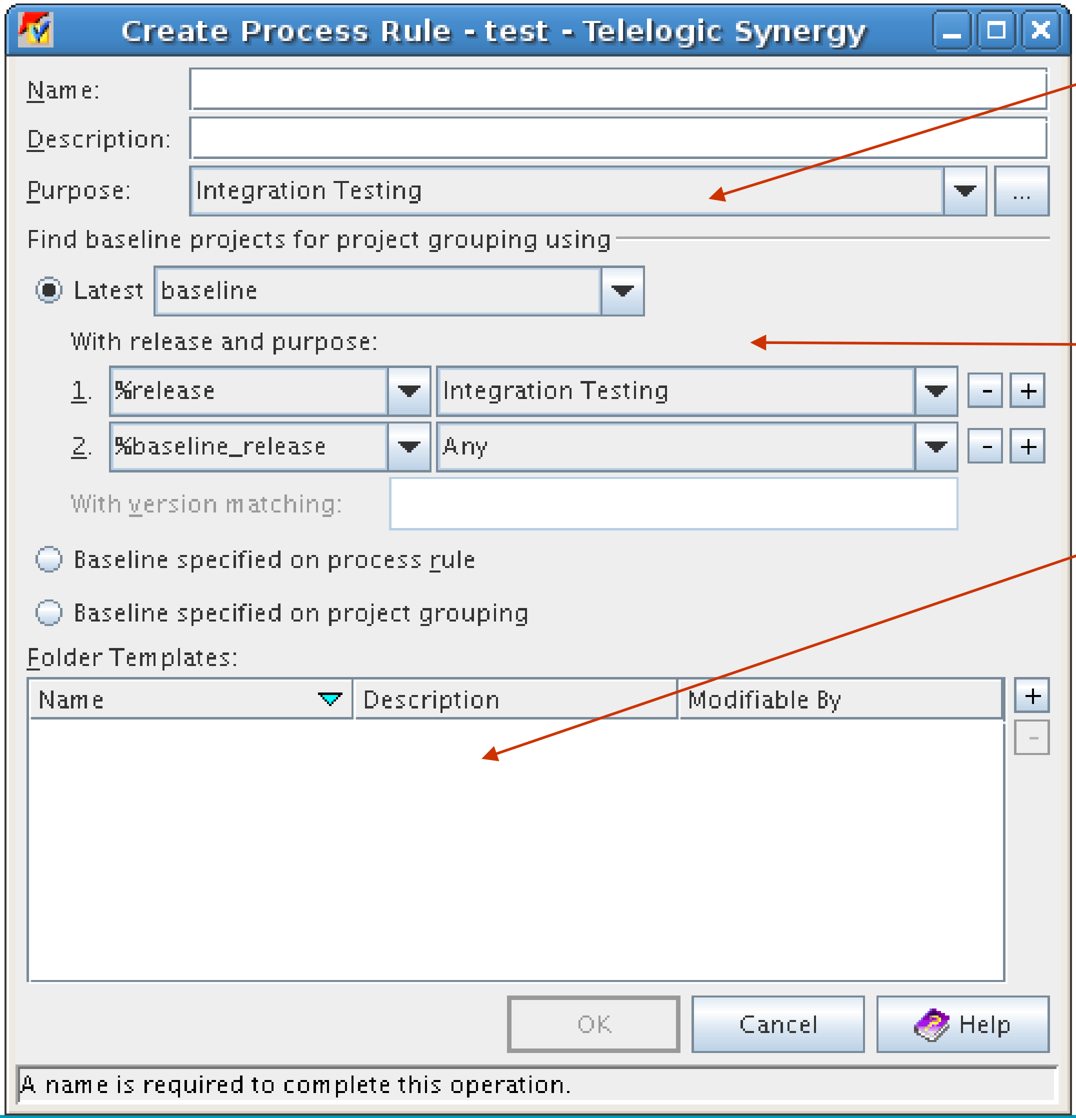
- A project’s **release** and **purpose** determine the process rule.
- Together with the release, purposes organize projects into **project groupings**.
- Purposes ensure that all projects in a hierarchy remain together.

Process Rules



Keywords expanded in the folder template description and query

Process Rules



Name:

Description:

Purpose: Integration Testing

Find baseline projects for project grouping using

(●) Latest baseline

With release and purpose:

1. %release | Integration Testing

2. %baseline_release | Any

With version matching:

( ) Baseline specified on process rule

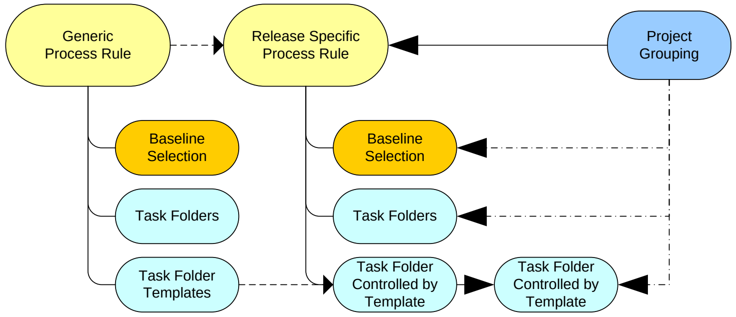
( ) Baseline specified on project grouping

Folder Templates:

Name	Description	Modifiable By

A name is required to complete this operation.

- Purpose process rule applies to
- Baseline selection
- Task selection



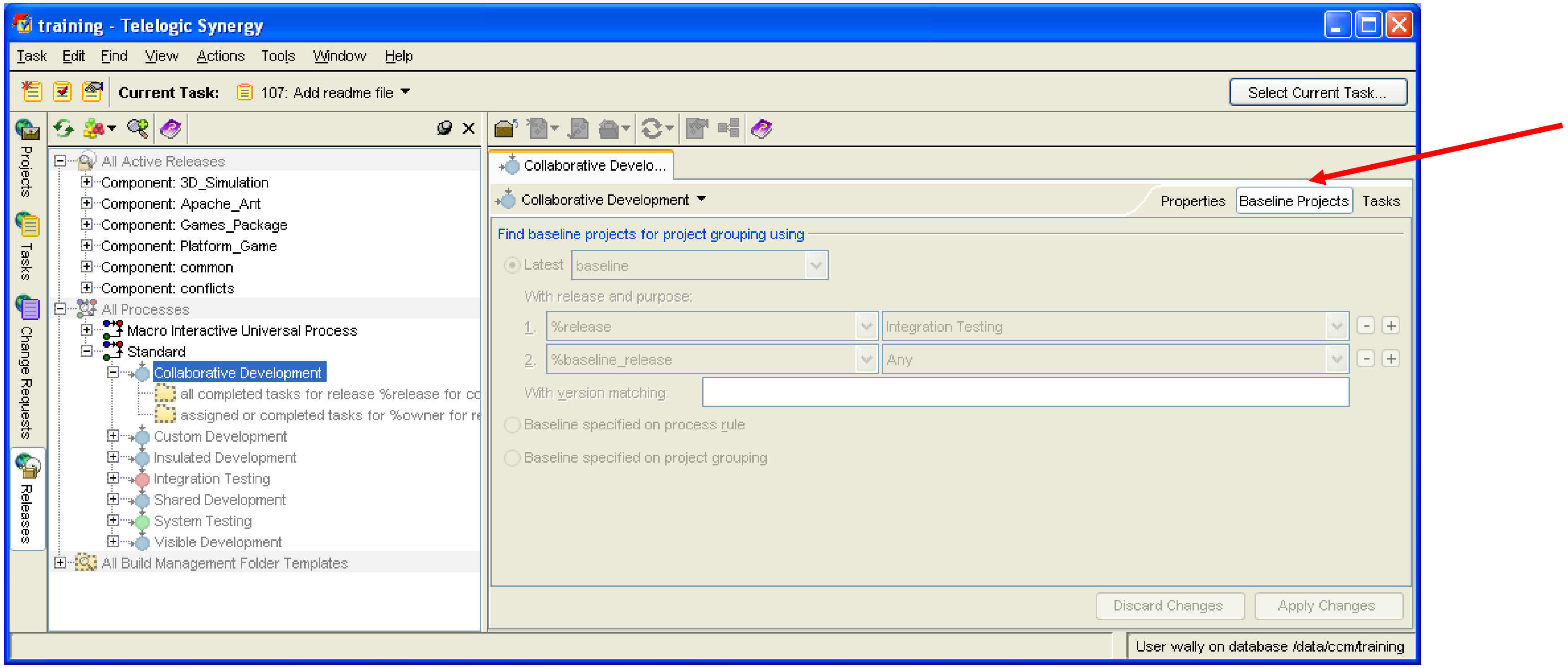
Processes Rules: How process rules work

- Identify which baseline will be used.
- Identify the list of tasks to be used.

NOTE: If you show a process rule, the dialog has three tabs: Properties, Baseline Projects, and Tasks. The Baseline Projects tab shows the rules for identifying the baseline, and the Tasks tab shows the rules for identifying the tasks.

Processes Rules: Selecting a Baseline

- Latest baseline or project
- Baseline specified on process rule – Set by administrator
- Baseline specified on project grouping – Set by Build Manager


training - Telelogic Synergy

Task Edit Find View Actions Tools Window Help
Current Task: 107: Add readme file
Select Current Task...
Projects
Tasks
Change Requests
Releases

All Active Releases
- Component: 3D_Simulation
- Component: Apache_Ant
- Component: Games_Package
- Component: Platform_Game
- Component: common
- Component: conflicts

All Processes
- Macro Interactive Universal Process
- Standard
  - Collaborative Development
    - all completed tasks for release %release for co
    - assigned or completed tasks for %owner for re
  - Custom Development
  - Insulated Development
  - Integration Testing
  - Shared Development
  - System Testing
  - Visible Development

All Build Management Folder Templates

Find baseline projects for project grouping using

Latest: baseline

With release and purpose:

1. %release | Integration Testing

2. %baseline_release | Any

With version matching:

Baseline specified on process rule

Baseline specified on project grouping

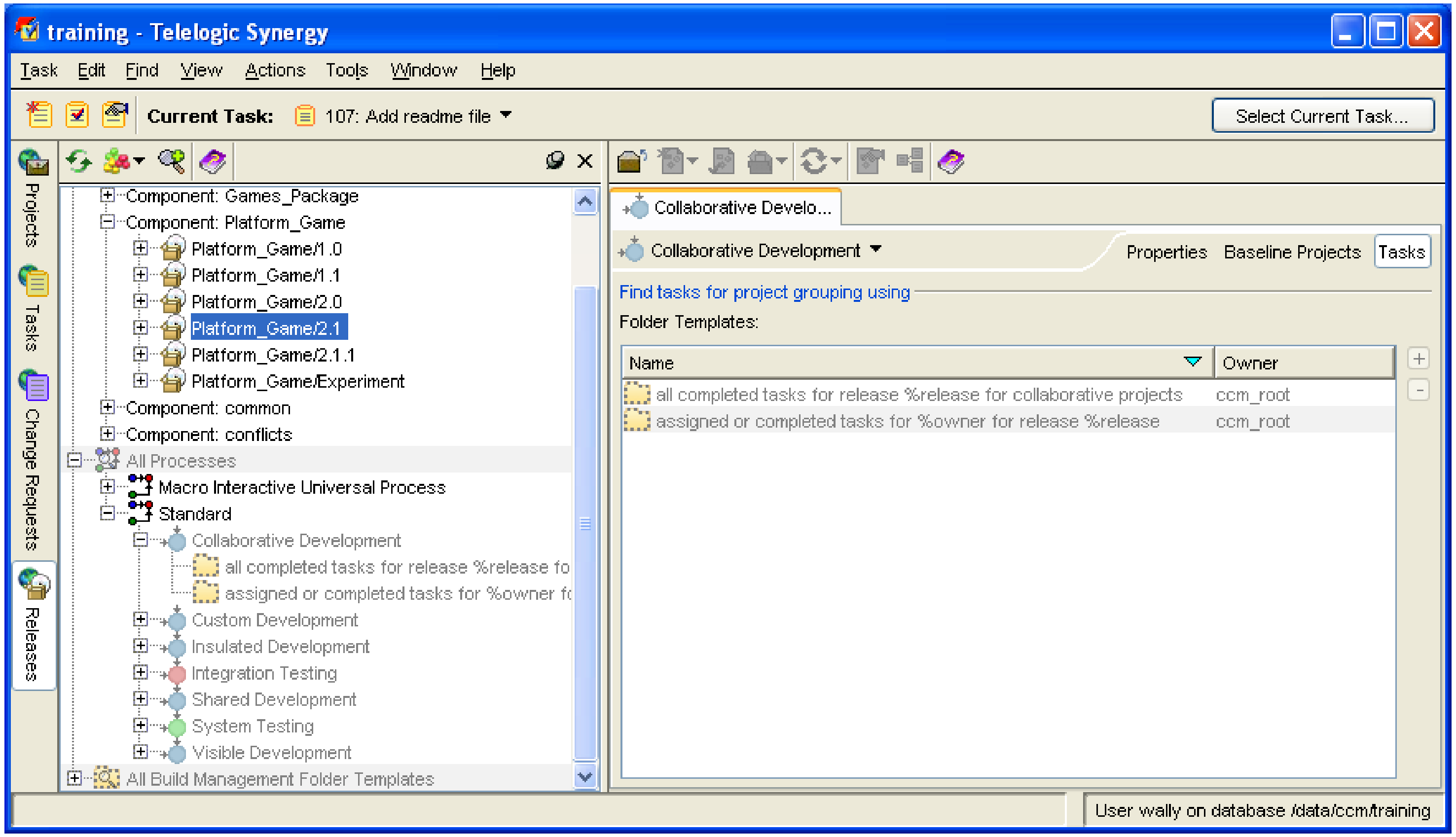
User wally on database /data/ccm/training

Processes Rules: Baseline selection guidelines

- Use %release and %baseline_release to find baselines automatically.
- Look for baselines by purpose – exploit by using multiple purposes to make selection clear.
- Official builds should be based off of known good baselines (%baseline_release) or set on project grouping (manual).

Processes Rules for Selecting tasks

- Tasks are selected by folders.
- Folders can be **manual** or **query-driven**.



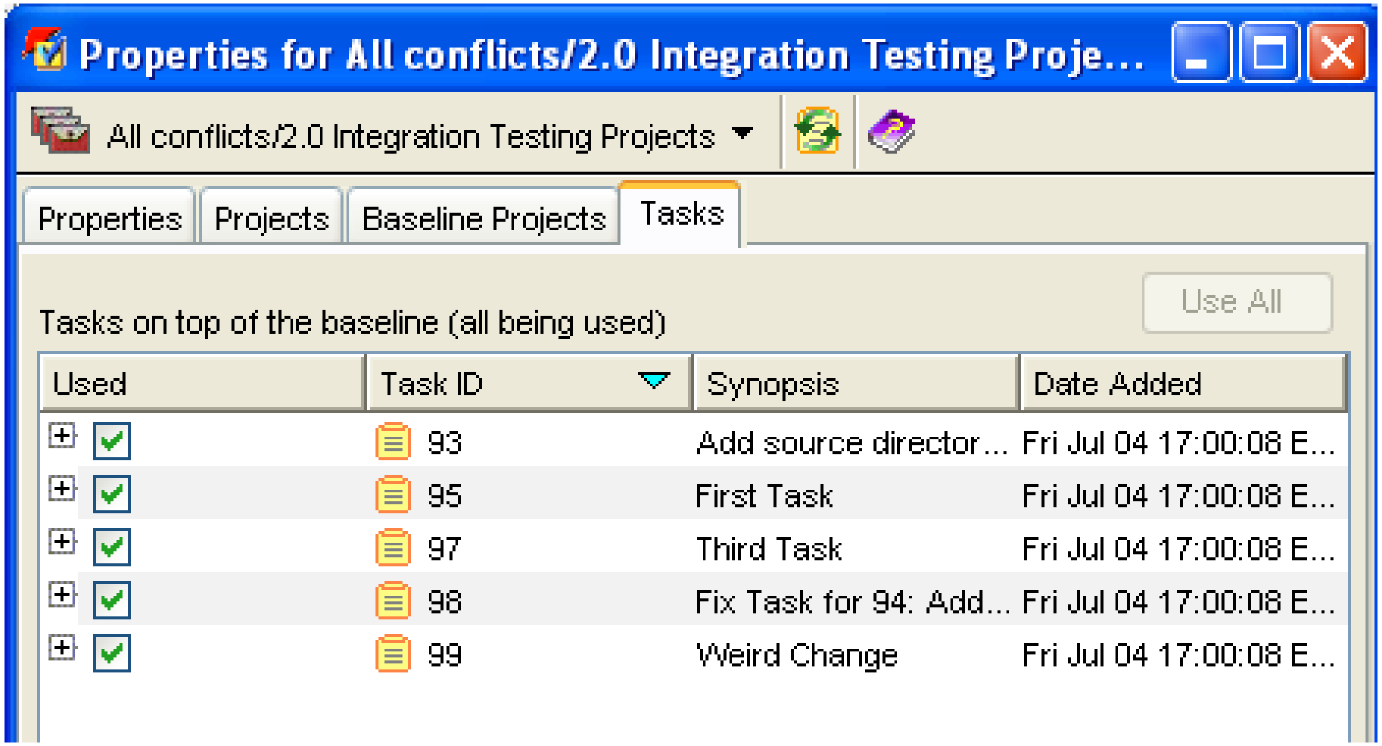
Name	Owner
all completed tasks for release %release for collaborative projects	ccm_root
assigned or completed tasks for %owner for release %release	ccm_root

Processes Rules for Selecting Tasks Selection Options

- Easy
  - All Tasks for Release X
  - All Tasks for Release X in State Y
- Complex
  - All Tasks related to CRs in State Y for Release Y
  - All Tasks included in the build manager’s project grouping
- Interesting ..
  - All Tasks for Release X not in Folder Z
  - All Tasks not in a folder managed by template X
  - All Tasks in Baseline A
- Cool (7.1)
  - Tasks associated to baseline projects and build products.

Project Grouping and tasks

- When you update a project grouping, Telelogic Synergy uses the process rules to select the baseline and tasks, but the actual baseline and tasks used are stored on the project grouping.


Properties for All conflicts/2.0 Integration Testing Proje...

All conflicts/2.0 Integration Testing Projects

Properties | Projects | Baseline Projects | Tasks

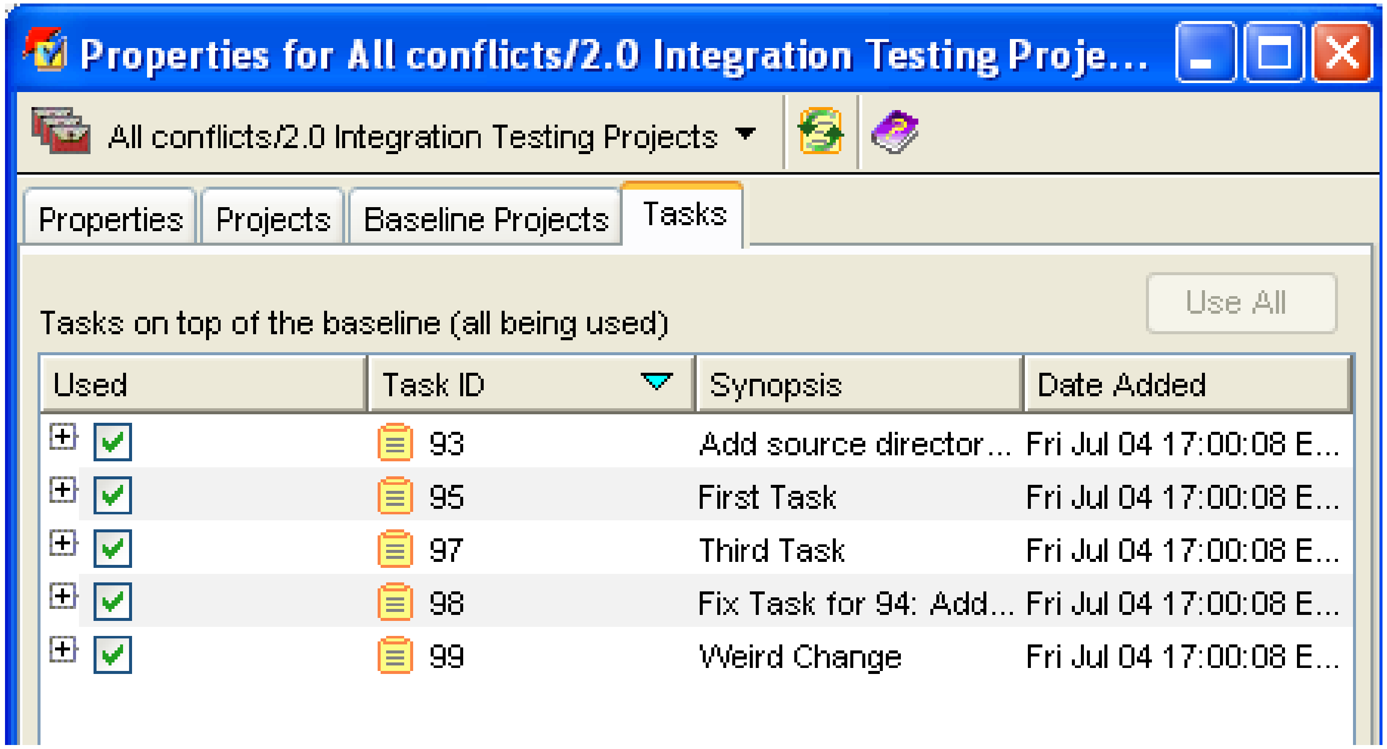
Use All

Tasks on top of the baseline (all being used)

Used	Task ID	Synopsis	Date Added
✓	93	Add source director...	Fri Jul 04 17:00:08 E...
✓	95	First Task	Fri Jul 04 17:00:08 E...
✓	97	Third Task	Fri Jul 04 17:00:08 E...
✓	98	Fix Task for 94: Add...	Fri Jul 04 17:00:08 E...
✓	99	Weird Change	Fri Jul 04 17:00:08 E...

Project Grouping: Saved Tasks

- Task included in project grouping but not included in the baseline
  - These are often referred as tasks on top of the baseline



Properties for All conflicts/2.0 Integration Testing Proje...

All conflicts/2.0 Integration Testing Projects

Properties | Projects | Baseline Projects | Tasks

Tasks on top of the baseline (all being used)

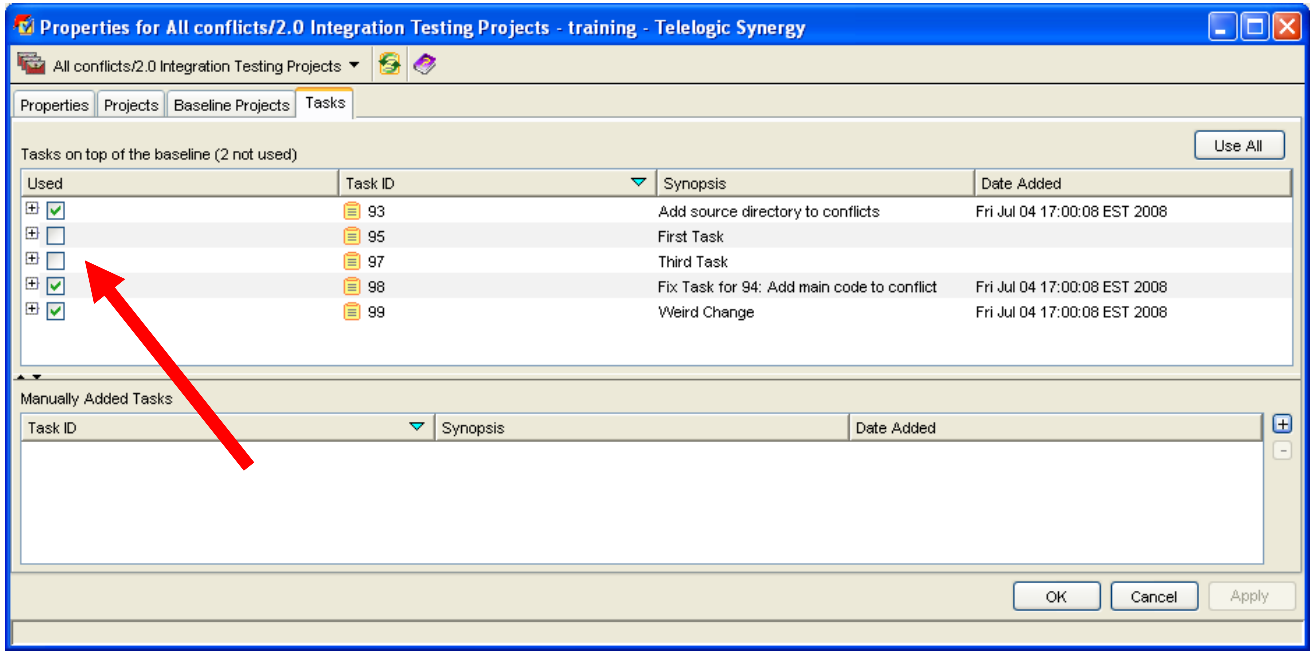
Use All

Used	Task ID	Synopsis	Date Added
✓	93	Add source director...	Fri Jul 04 17:00:08 E...
✓	95	First Task	Fri Jul 04 17:00:08 E...
✓	97	Third Task	Fri Jul 04 17:00:08 E...
✓	98	Fix Task for 94: Add...	Fri Jul 04 17:00:08 E...
✓	99	Weird Change	Fri Jul 04 17:00:08 E...

NEW 7.1 No longer shows untrusted (dirty) tasks.

Project Grouping: Removed tasks

- Tasks that you, the build manager, have removed from the project grouping


Properties for All conflicts/2.0 Integration Testing Projects - training - Telelogic Synergy
All conflicts/2.0 Integration Testing Projects
Properties
Projects
Baseline Projects
Tasks
Use All
Manually Added Tasks
OK
Cancel
Apply

Tasks on top of the baseline (2 not used)

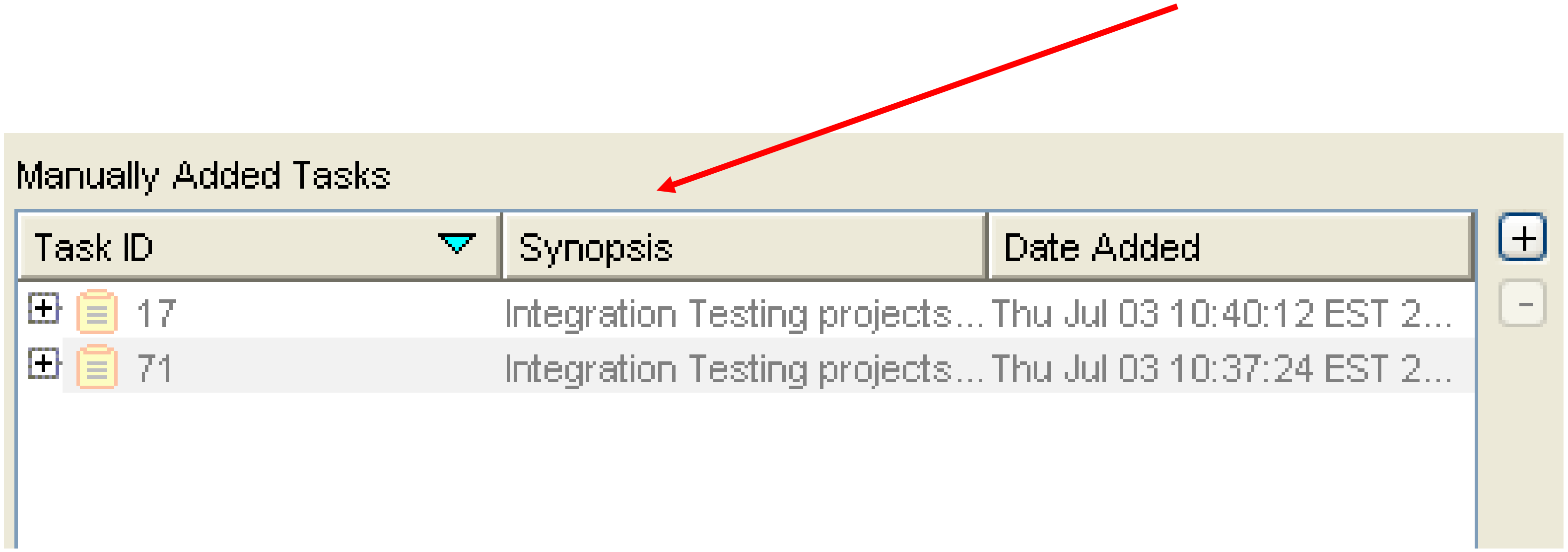
Used	Task ID	Synopsis	Date Added
✓	93	Add source directory to conflicts	Fri Jul 04 17:00:08 EST 2008
☐	95	First Task	
☐	97	Third Task	
✓	98	Fix Task for 94: Add main code to conflict	Fri Jul 04 17:00:08 EST 2008
✓	99	Weird Change	Fri Jul 04 17:00:08 EST 2008

Task ID	Synopsis	Date Added

Removed Task (Unchecked Ones)

Project Grouping: Additional tasks

- Tasks the build manager has added to the project grouping manually
- Not added by the process rule so “Manually added”



Manually Added Tasks

Task ID	Synopsis	Date Added
17	Integration Testing projects...	Thu Jul 03 10:40:12 EST 2...
71	Integration Testing projects...	Thu Jul 03 10:37:24 EST 2...

Project Grouping: Automatic Tasks

- One for each unique release, purpose, state, and owner (if in a *working* or *visible* state) for project and products.
- Helps keep structures of project and products together during updates.

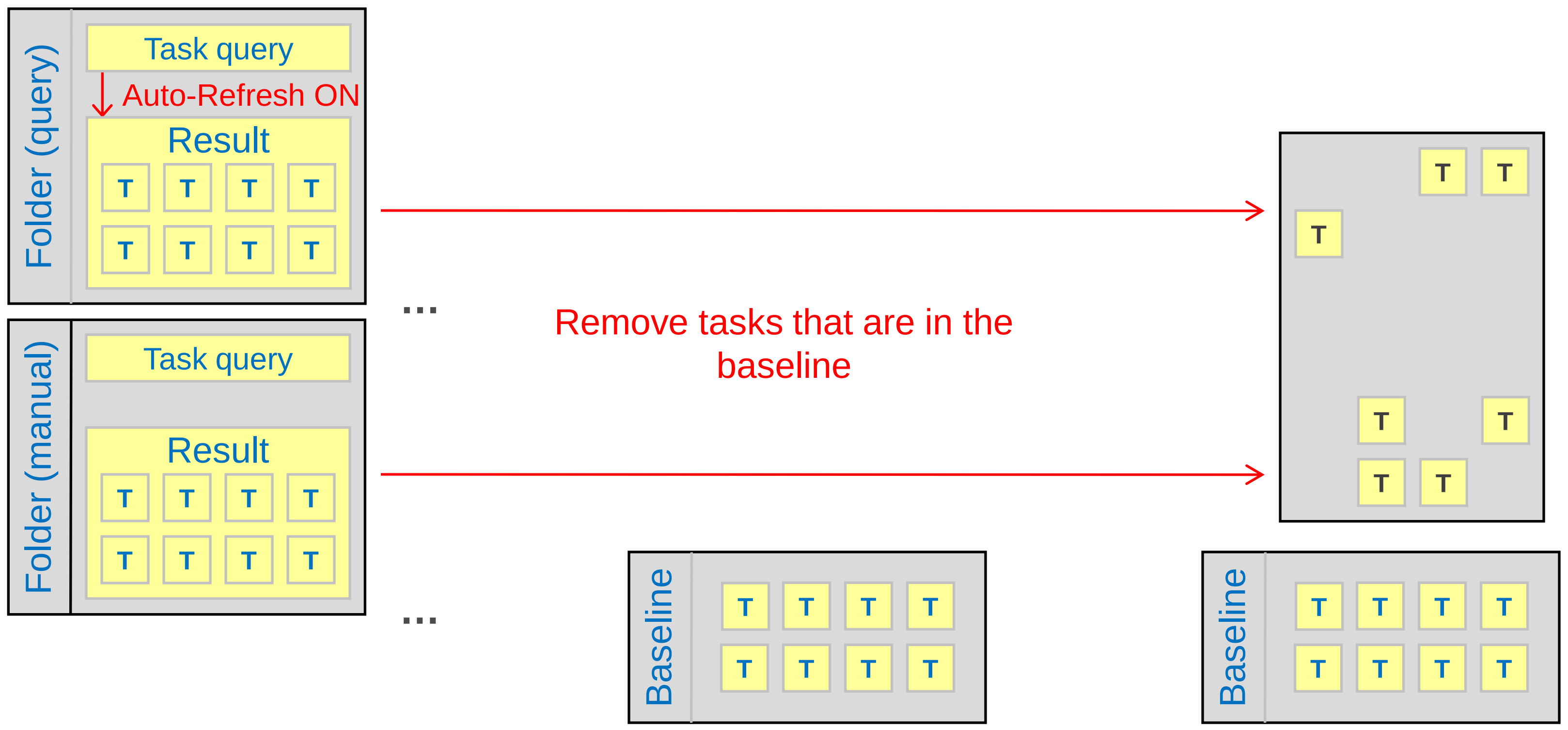
Project Attribute Values				
State	**Purpose**	**Owner**	**Release**	**Automatic Task Synopsis**
working	Insulated Development	wally	1.0	wally's Personal Projects for Release 1.0
prep	System Testing	n/a	2.0	System Testing Project for Release 2.0
prep	Performance Testing	n/a	2.1	Performance Testing Projects for Release 2.1

Note: Not visible in the project grouping.

Update: How Update Chooses Members

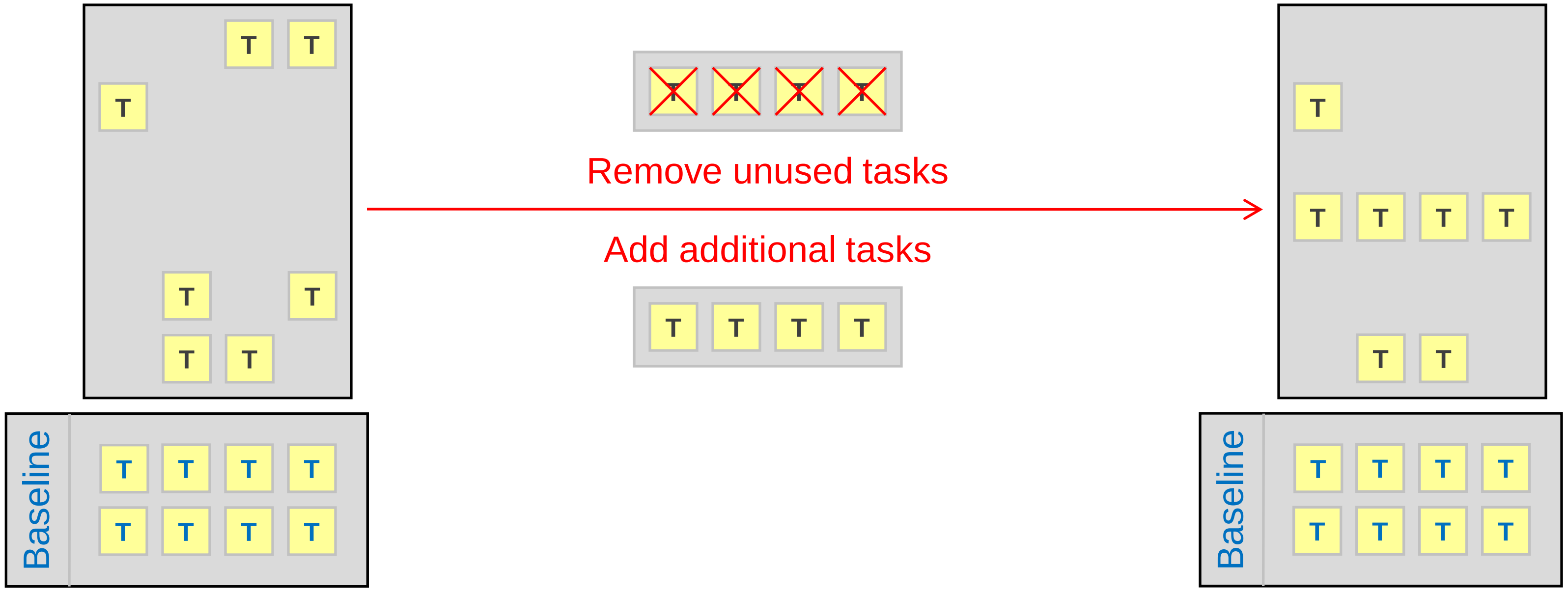
- Update Project Grouping for the project being updated if Auto Update is On
  - Identify baseline.
  - Set Tasks Since the Baseline from all process rule folders.
- Identify the baseline project copy for this project.
- Get all versions from baseline and tasks.
- For each subproject, directory, and file…
  - Find each candidate version.
  - Evaluate each candidate using
    - Apply Selection Rules.
    - Apply Exclusion Rules.
  - Use highest scored candidate in project.
    - If tied, use newest version.

Update: How Update chooses members



Folders + Baseline => Baseline with additional tasks

Update: How update chooses members



Baseline
with additional tasks

+

Project Grouping

=>

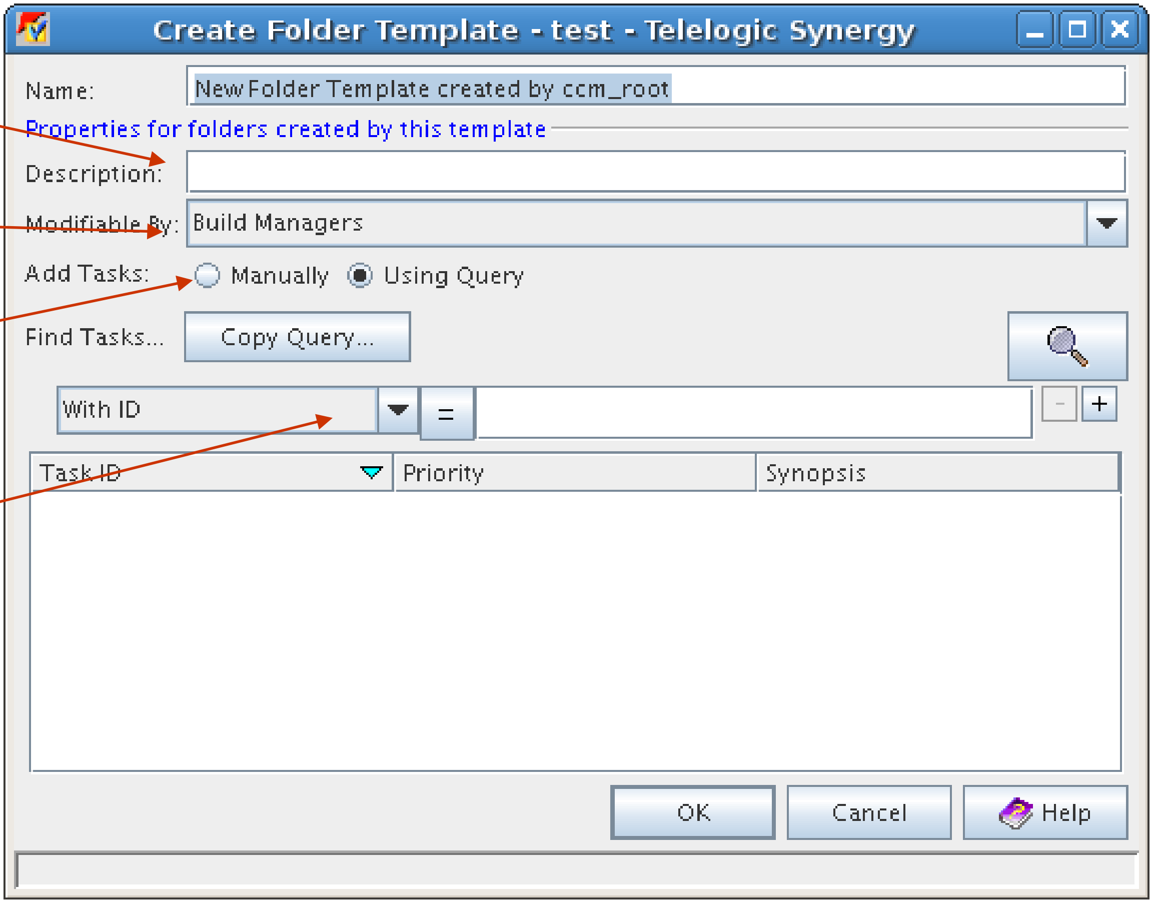
Baseline
with additional tasks

Folders and Folder Templates

- Here are some examples of folders:
  - All completed tasks for release Platform_Game/2.1
  - All Wally’s assigned and completed tasks for release common/3.0
  - Approved tasks for release Apache_Ant/1.7.0
- Folder Templates
  - Folder templates are patterns used to create the actual folders that will be used in the project grouping's update properties.
  - For example the Collaborative Development process rules contain two folder groupings:
    - all completed tasks for release %release for collaborative projects
    - assigned or completed tasks for %owner for release %release

Folders and Folder Templates

- Keywords expanded
- Security settings
- Manual for custom task selection
- Query for automatic task selection


Create Folder Template - test - Telelogic Synergy

Name: New Folder Template created by ccm_root

Properties for folders created by this template

Description:

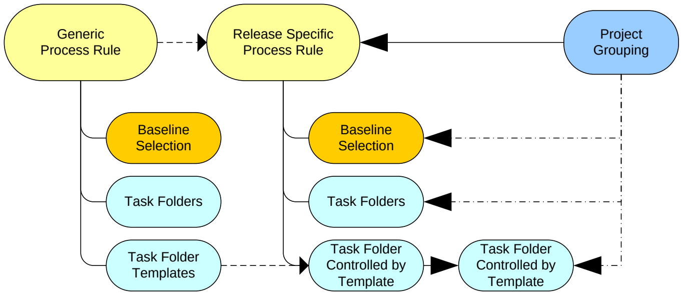
Modifiable By: Build Managers

Add Tasks: [ ] Manually [✓ Using Query]

Find Tasks... Copy Query...

With ID =

Task ID	Priority	Synopsis



Folders & Queries - Release-based Query

Description	Query
All completed tasks for release Platform_Game/2.1	status='completed' and release='Platform_Game/2.1'
Wally's assigned and completed tasks for release common/3.0	(status='assigned' or status='completed') and resolver='wally' and release='common/3.0'
All completed tasks for releases Apache_Ant/1.7.0 and Apache_Ant/1.6.1	status='completed' and (release='Apache_Ant/1.7.0' or release='Apache_Ant/1.6.1')
Wally and Bernie's tasks for Games_Package/2.0	(status='assigned' or status='completed') and (resolver='Wally' or resolver='Bernie') and release='Games_Package/2.0'

Folders & Queries - Baseline Queries

Description	Query
Tasks from published Apache_Ant/1.7.0 Integration Testing baselines	latest_baseline_tasks(‘Apache_Ant/1.7.0’, ‘Integration testing’)
Tasks from published Apache_Ant/1.7.0 Integration Testing baselines created in DCM database A.	latest_baseline_tasks_db(‘A’, Apache_Ant/1.7.0’, ‘Integration testing’

Folders & Queries - Change Request Queries

Description	Query
Resolved change requests for release CM/7.0	is_associated_task_of(cvtype=’problem’ and crstatus=’resolved’ and release=’CM/7.0’)
Maria’s assigned and completed change requests for release Change/5.1	is_associated_task_of(cvtype=’problem’ and (crstatus=’assigned’ or crstatus=’completed’) and resolver=’wally’ and release=‘Change/5.1’)

Folders & Queries - Folder-Based Queries

Description	Query
All completed tasks for release 6.0, minus tasks in Excluded Tasks folder	status=’completed’ and release=’6.0’ and not is_task_in_folder_of(cvtype=’folder’ and name=’1002’)

Component Development and Synergy 7.1

- Changes to “use version” behaviour
- Component Tasks
- Mixed Release Hierarchies and Baselines

Switch to a New Version of a Subcomponent

1. Developer sets his/her current task to integrate a new component or update an existing component by associating the new project/product to his/her current task.

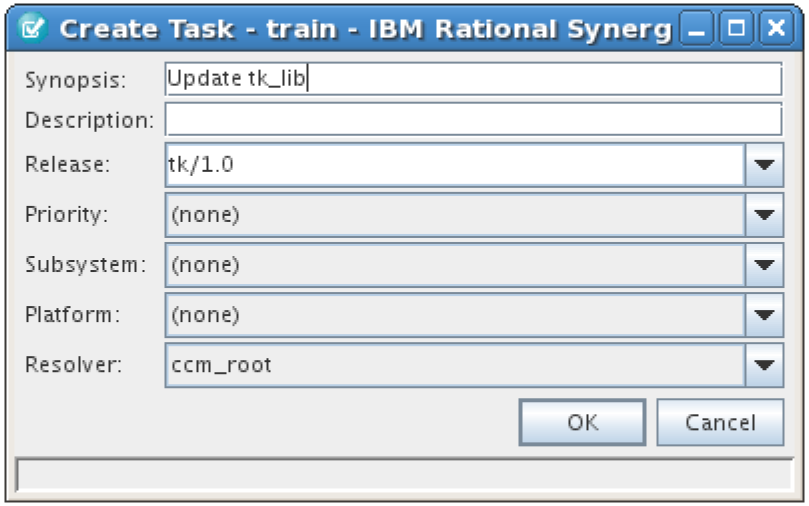
2. Developer adds the new component to the project or uses the new version of the component

3. Developer makes any changes necessary to integrate the new component

4. Developer checks in the task

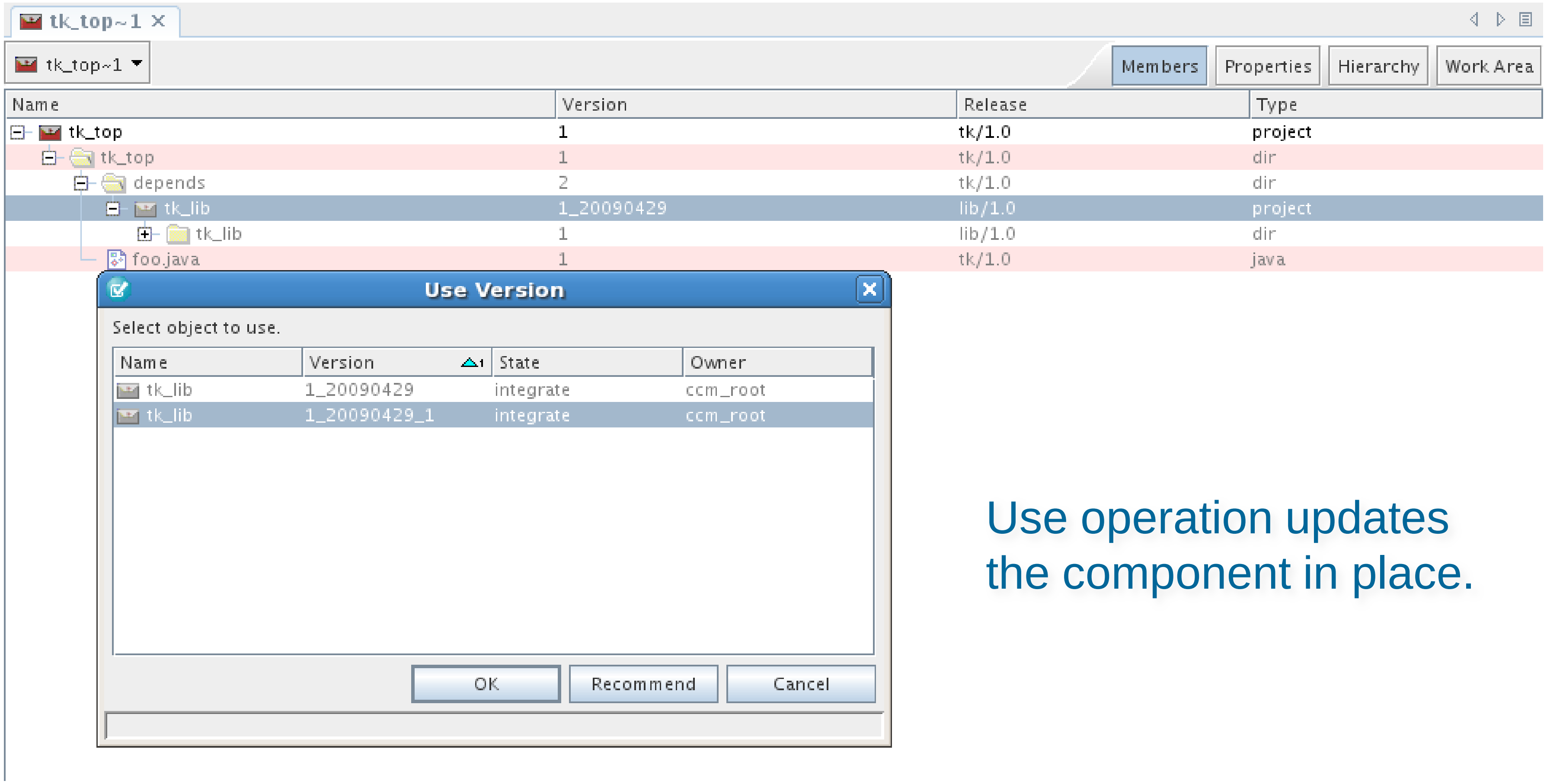
“Use Component” and Current Task

- Pre7.1 component development required manual association of component projects to tasks.
- 7.1 Automates this during the use operation.



Synopsis:	Update tk_lib
Description:	
Release:	tk/1.0
Priority:	(none)
Subsystem:	(none)
Platform:	(none)
Resolver:	ccm_root

Update the component as before …

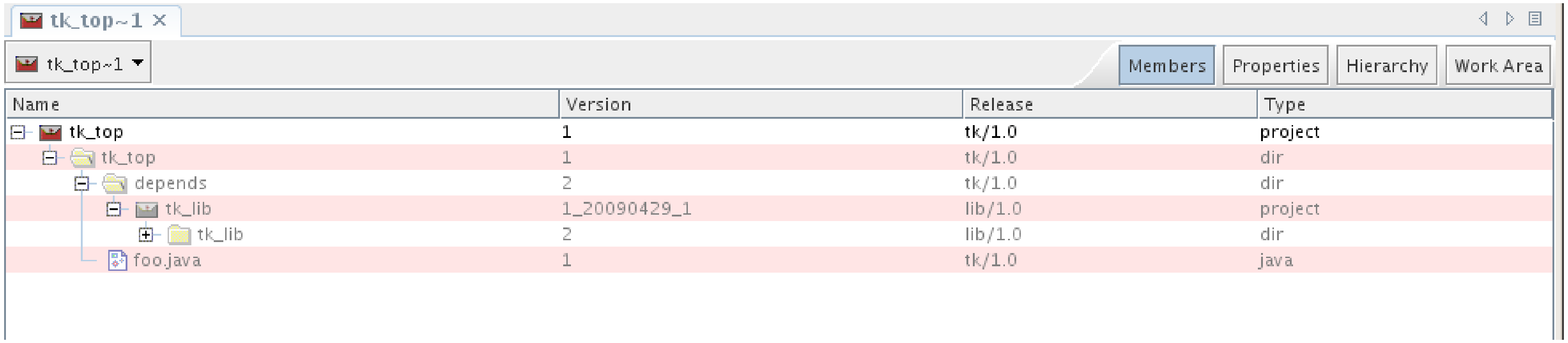


Name	Version	Release	Type
tk_top	1	tk/1.0	project
tk_top	1	tk/1.0	dir
depends	2	tk/1.0	dir
tk_lib	1_20090429	lib/1.0	project
tk_lib	1	lib/1.0	dir
foo.java	1	tk/1.0	java

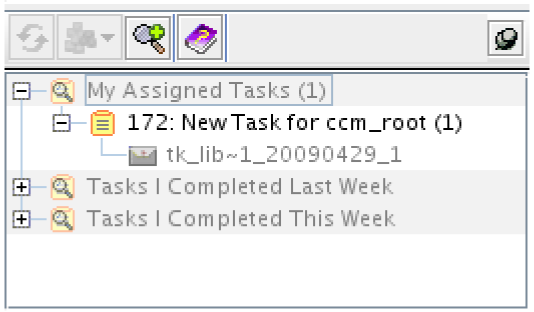
Name	Version	State	Owner
tk_lib	1_20090429	integrate	ccm_root
tk_lib	1_20090429_1	integrate	ccm_root

Use operation updates the component in place.

Task association is now automatic



Name	Version	Release	Type
tk_top	1	tk/1.0	project
tk_top	1	tk/1.0	dir
depends	2	tk/1.0	dir
tk_lib	1_20090429_1	lib/1.0	project
tk_lib	2	lib/1.0	dir
foo.java	1	tk/1.0	java



My Assigned Tasks (1)
172: New Task for ccm_root (1)
tk_lib~1_20090429_1
Tasks I Completed Last Week
Tasks I Completed This Week

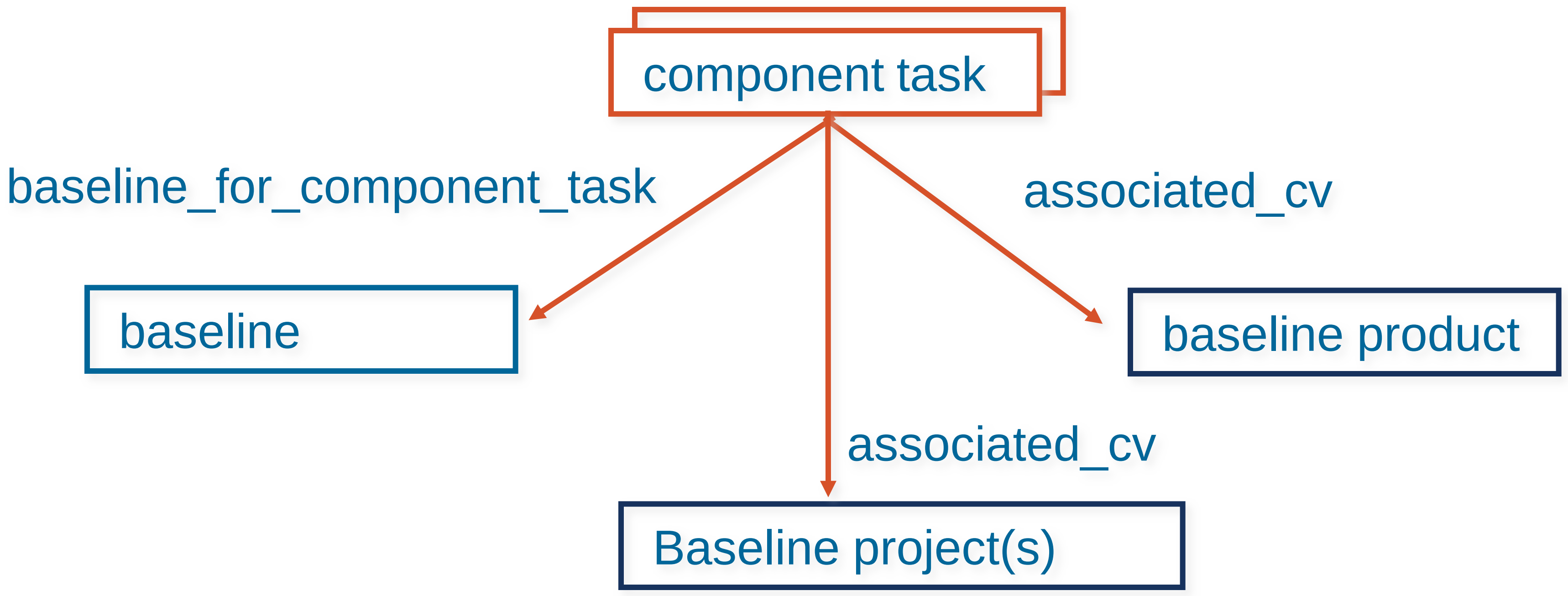
Changes then propagates through the team.

Improved CBD: Easier to Use New Subcomponents

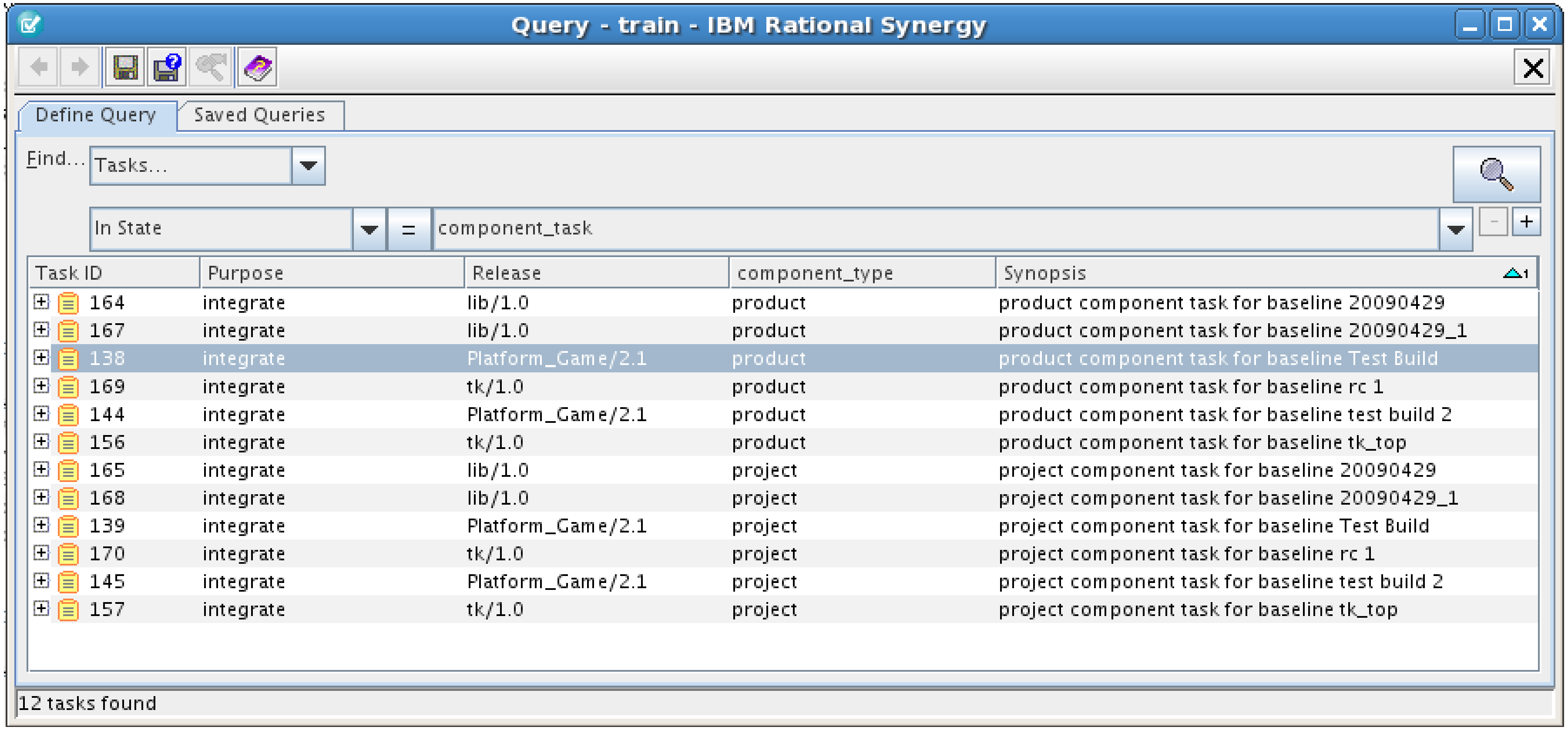
- When you create a baseline, Synergy automatically creates two component tasks
- Easily add process rules to include subcomponents using a new dialog to add component tasks to your process rules.
- Upgrade does not automatically create component tasks for old baselines
  - `ccm baseline -create_component_tasks command`
- DCM supports transfer of component tasks

Component Task Model

status=‘component task’
component_type=‘project | product’
member_status = ‘from baseline’
release = ‘from baseline’



Component Tasks



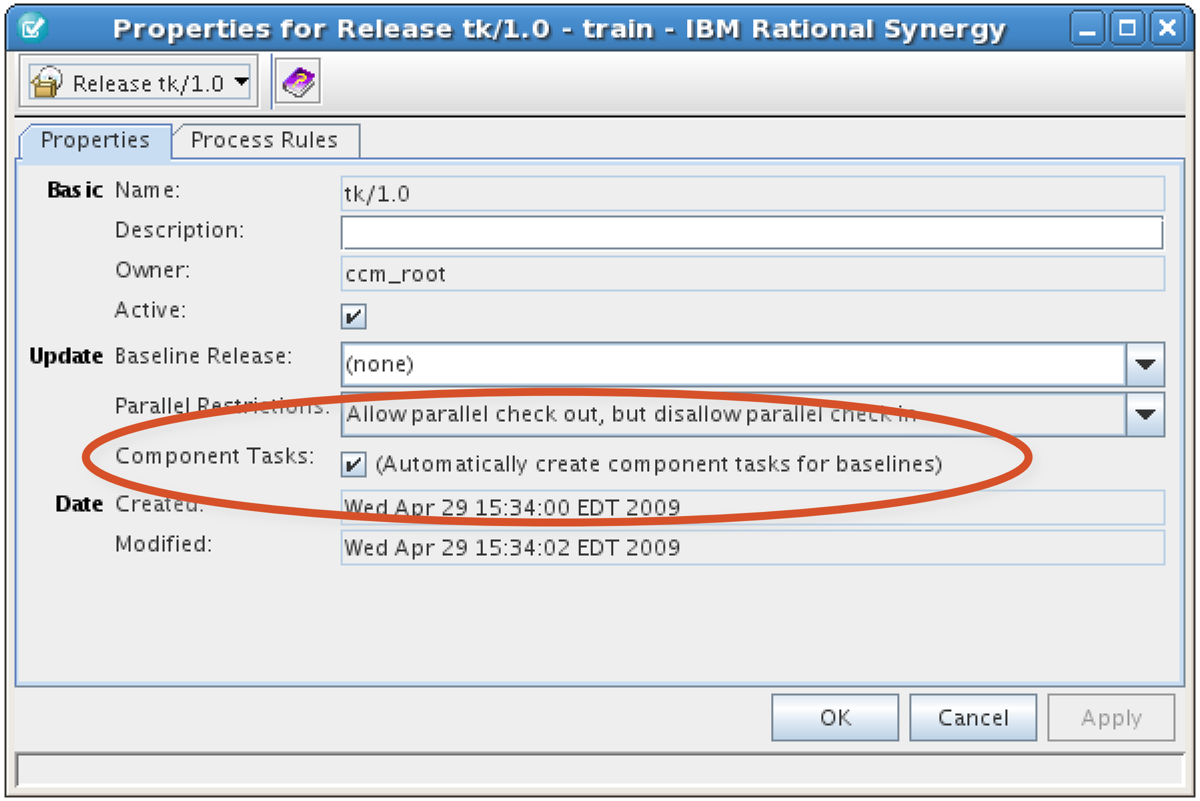
Find... Tasks...

In State = component_task

Task ID	Purpose	Release	component_type	Synopsis
164	integrate	lib/1.0	product	product component task for baseline 20090429
167	integrate	lib/1.0	product	product component task for baseline 20090429_1
138	integrate	Platform_Game/2.1	product	product component task for baseline Test Build
169	integrate	tk/1.0	product	product component task for baseline rc 1
144	integrate	Platform_Game/2.1	product	product component task for baseline test build 2
156	integrate	tk/1.0	product	product component task for baseline tk_top
165	integrate	lib/1.0	project	project component task for baseline 20090429
168	integrate	lib/1.0	project	project component task for baseline 20090429_1
139	integrate	Platform_Game/2.1	project	project component task for baseline Test Build
170	integrate	tk/1.0	project	project component task for baseline rc 1
145	integrate	Platform_Game/2.1	project	project component task for baseline test build 2
157	integrate	tk/1.0	project	project component task for baseline tk_top

12 tasks found

Release Setting for Component Task Creation


Properties for Release tk/1.0 - train - IBM Rational Synergy

Release tk/1.0

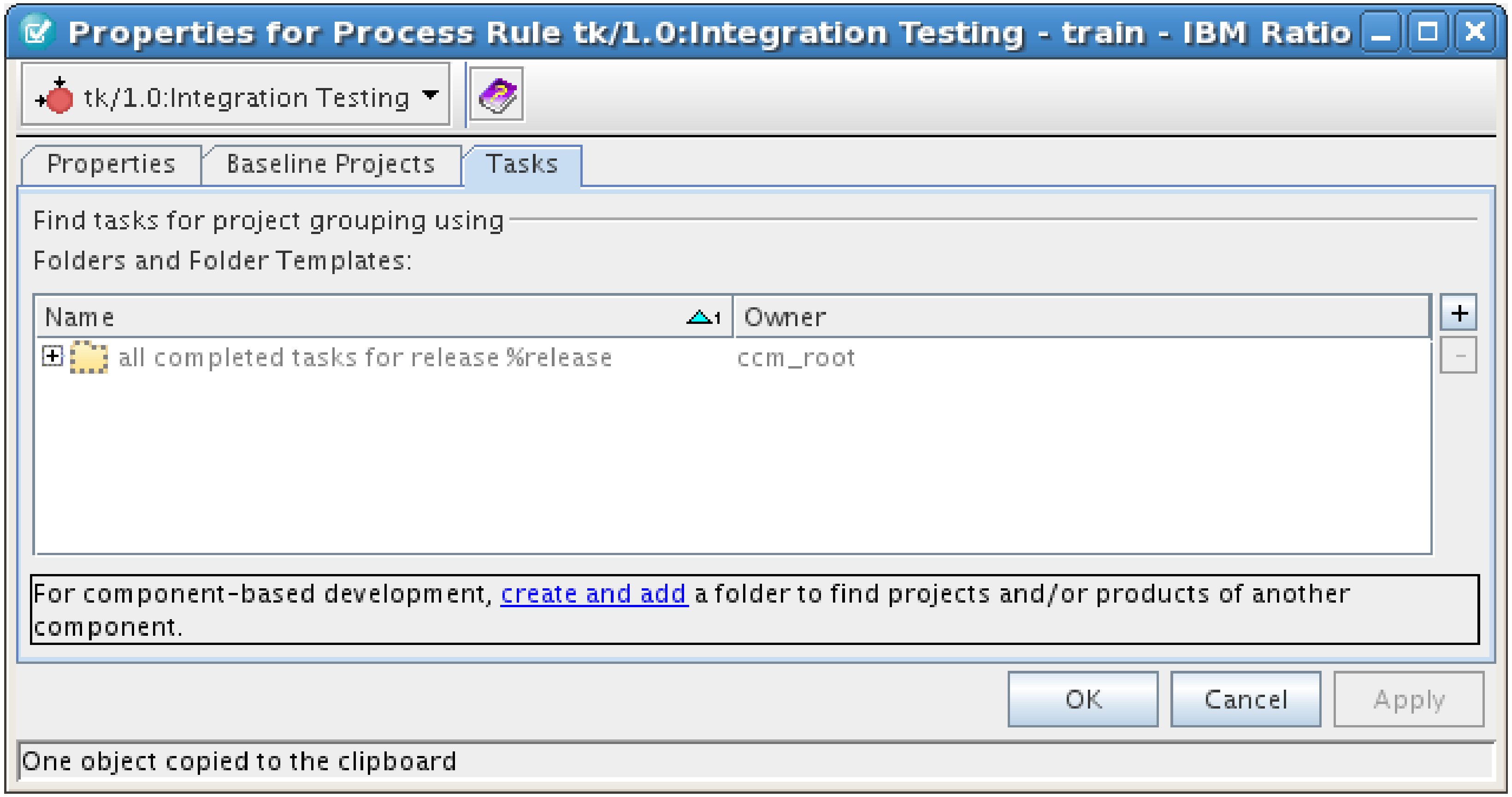
Properties | Process Rules

Basic	Name:	tk/1.0
	Description:	
	Owner:	ccm_root
	Active:	✓
Update	Baseline Release:	(none)
	Parallel Restrictions:	Allow parallel check out, but disallow parallel check in
	Component Tasks:	✓ (Automatically create component tasks for baselines)
Date	Created:	Wed Apr 29 15:34:00 EDT 2009
	Modified:	Wed Apr 29 15:34:02 EDT 2009

Component Task Query Function

-
  ```
  component_tasks(
      'scope',
      'release',
      'baseline purpose',
      'baseline state'
  )
  ```

Build Component Task Folders …



Properties for Process Rule tk/1.0:Integration Testing - train - IBM Ratio

tk/1.0:Integration Testing

Properties | Baseline Projects | Tasks

Find tasks for project grouping using

Folders and Folder Templates:

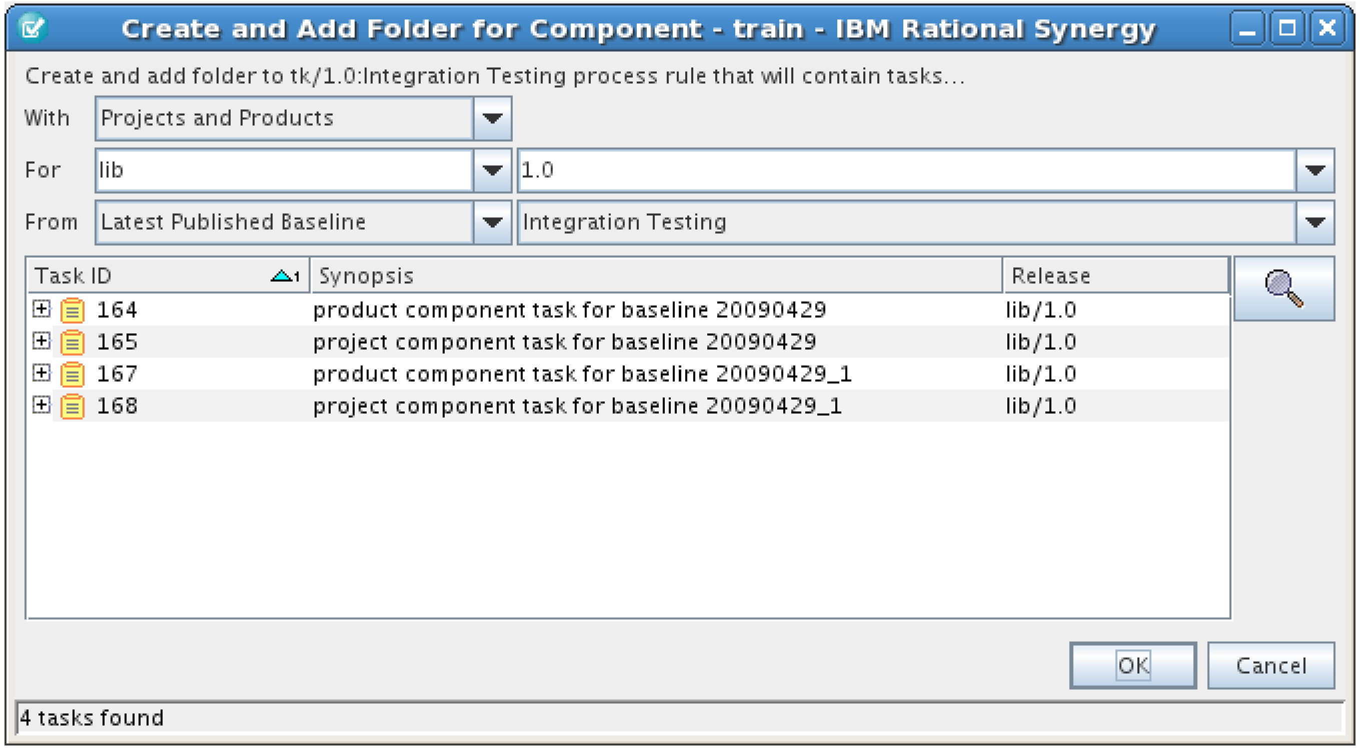
Name	Owner
all completed tasks for release %release	ccm_root

For component-based development, create and add a folder to find projects and/or products of another component.

OK | Cancel | Apply

One object copied to the clipboard

Build Component Task Folders …



Create and Add Folder for Component - train - IBM Rational Synergy

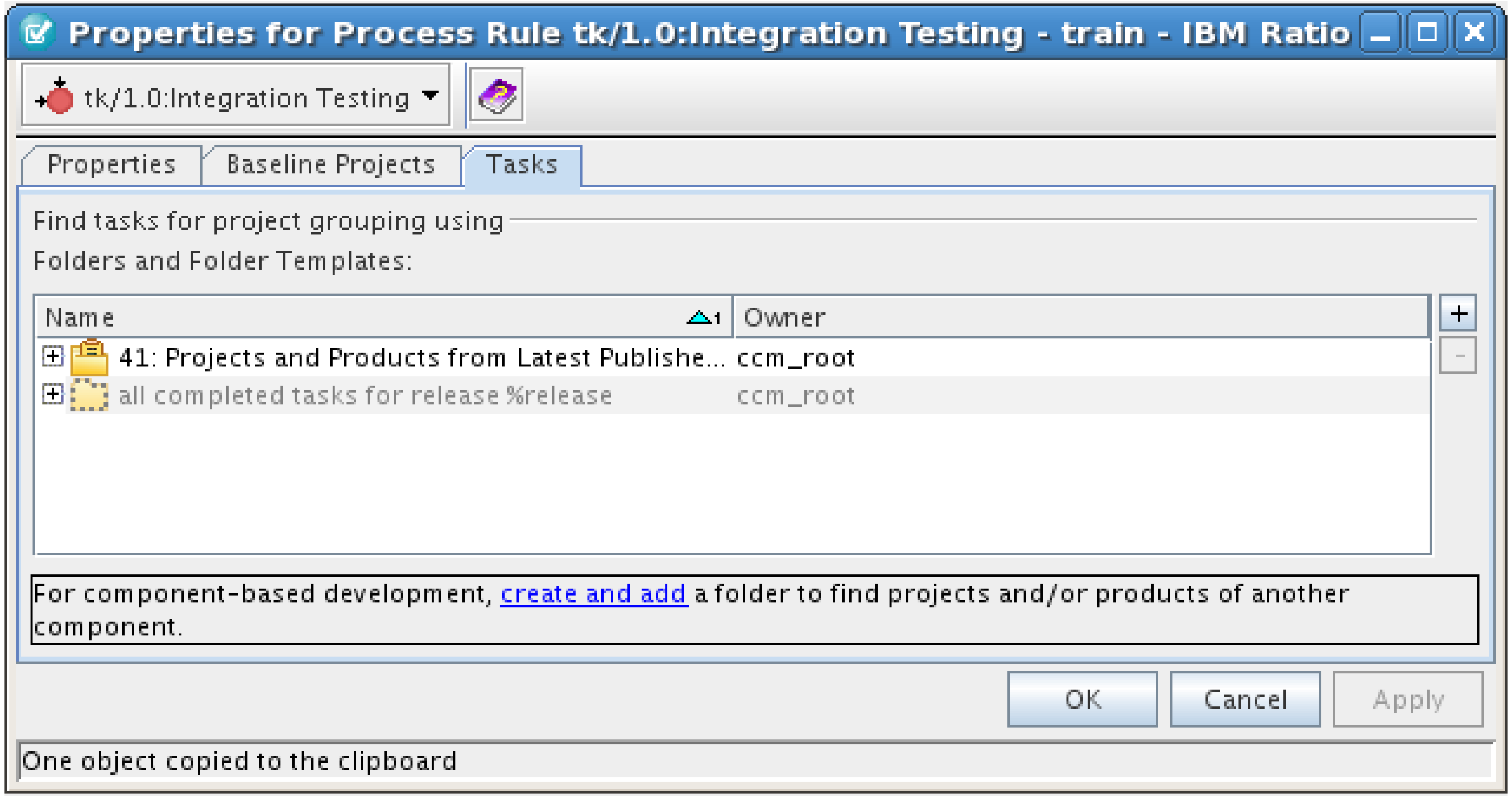
Create and add folder to tk/1.0:Integration Testing process rule that will contain tasks...

With	Projects and Products	
For	lib	1.0
From	Latest Published Baseline	Integration Testing

Task ID	Synopsis	Release
164	product component task for baseline 20090429	lib/1.0
165	project component task for baseline 20090429	lib/1.0
167	product component task for baseline 20090429_1	lib/1.0
168	project component task for baseline 20090429_1	lib/1.0

4 tasks found

Build Component Task Folders …


Properties for Process Rule tk/1.0:Integration Testing - train - IBM Ratio
tk/1.0:Integration Testing
Properties
Baseline Projects
Tasks

Find tasks for project grouping using

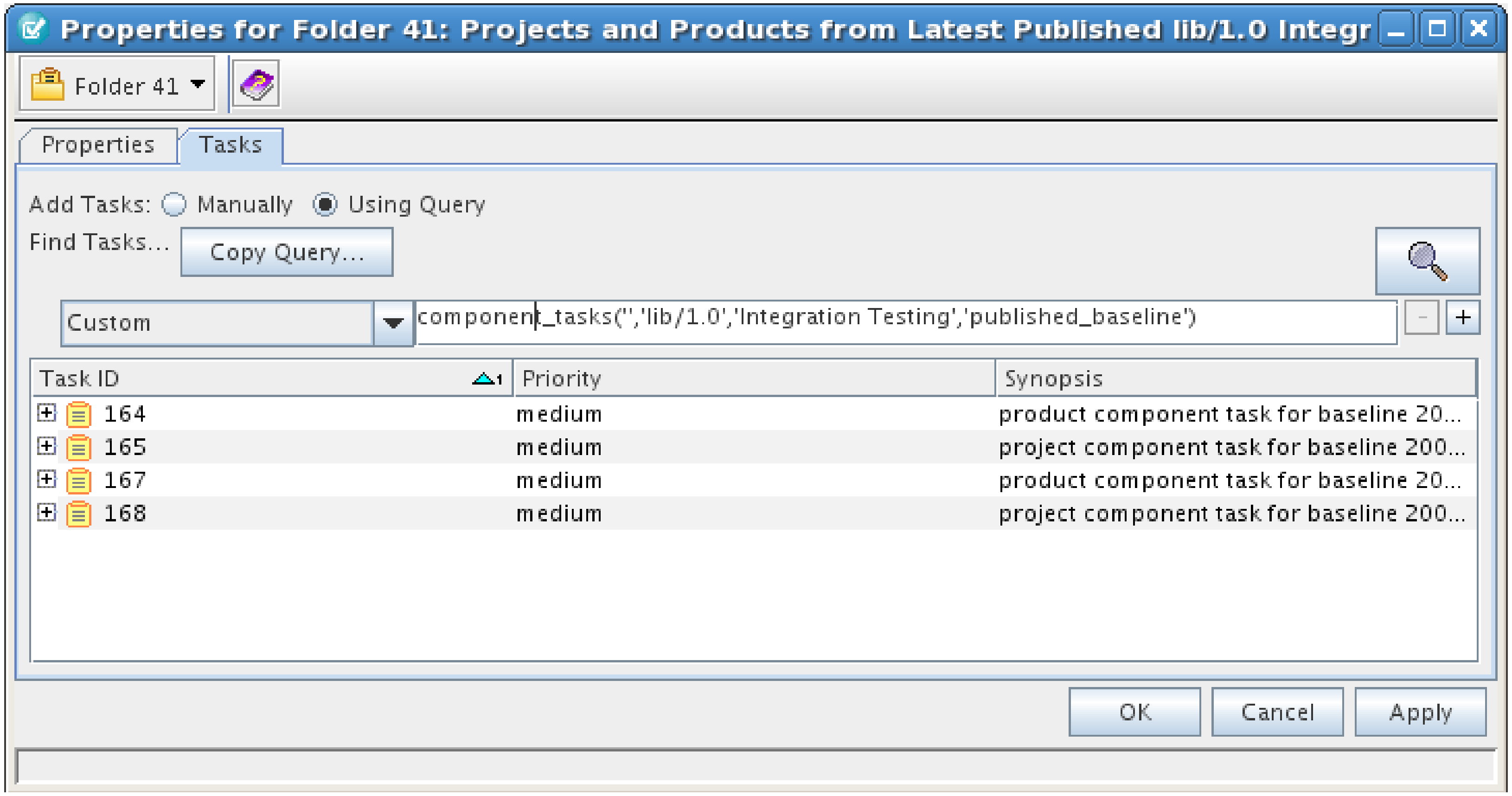
Folders and Folder Templates:

Name	Owner
41: Projects and Products from Latest Publishe...	ccm_root
all completed tasks for release %release	ccm_root

For component-based development, create and add a folder to find projects and/or products of another component.

One object copied to the clipboard

Build Component Task Folders …


Properties for Folder 41: Projects and Products from Latest Published lib/1.0 Integr
Folder 41
Properties
Tasks
Find Tasks...
Copy Query...
OK
Cancel
Apply

Add Tasks: ( ) Manually [✓ Using Query]

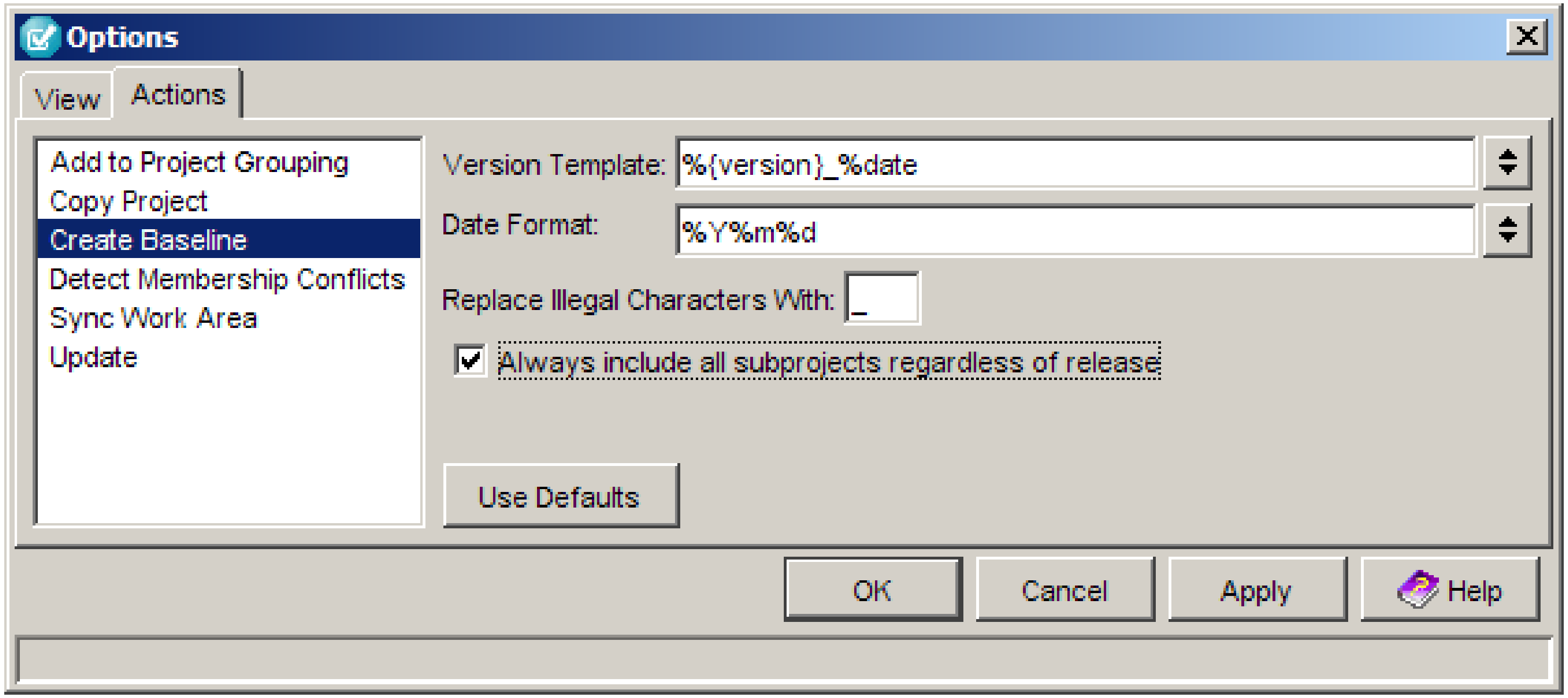
Custom: component_tasks('','lib/1.0','Integration Testing','published_baseline')

Task ID	Priority	Synopsis
164	medium	product component task for baseline 20...
165	medium	project component task for baseline 200...
167	medium	product component task for baseline 20...
168	medium	project component task for baseline 200...

“component_tasks” Selection Capabilities

- Projects from a specific release and baseline (Acme/4.3)
- Latest published products from a specific release (Foo/7.2)
- Latest released projects and products from a specific component, any release (Bar/*)
- Latest published projects from a specific release, any component (*/week42)
- Include integration prep projects from a specific baseline and release (*/week42)

Baselines will include all releases in the hierarchy (7.1)



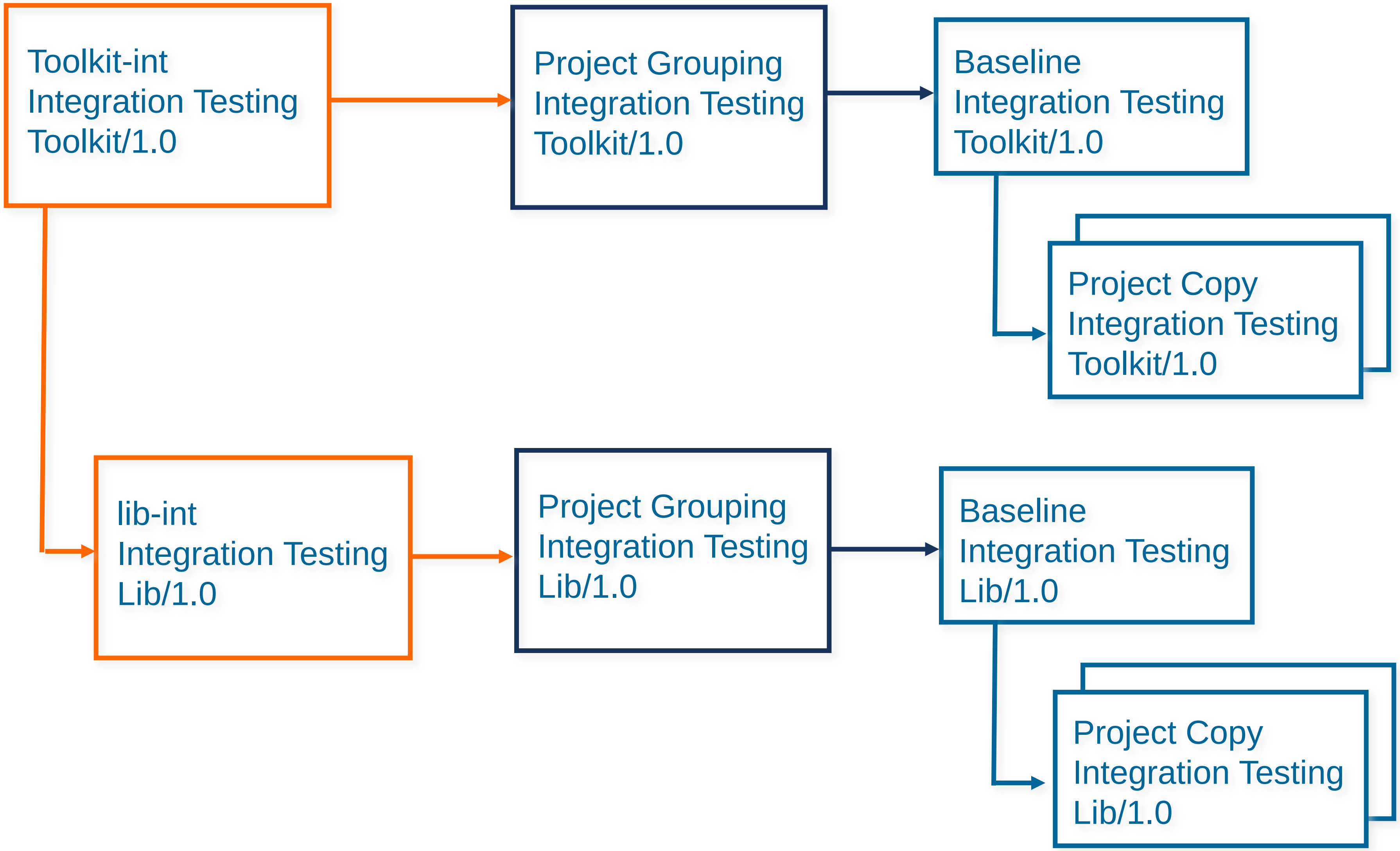
Version Template: %{version}_%date

Date Format: %Y%m%d

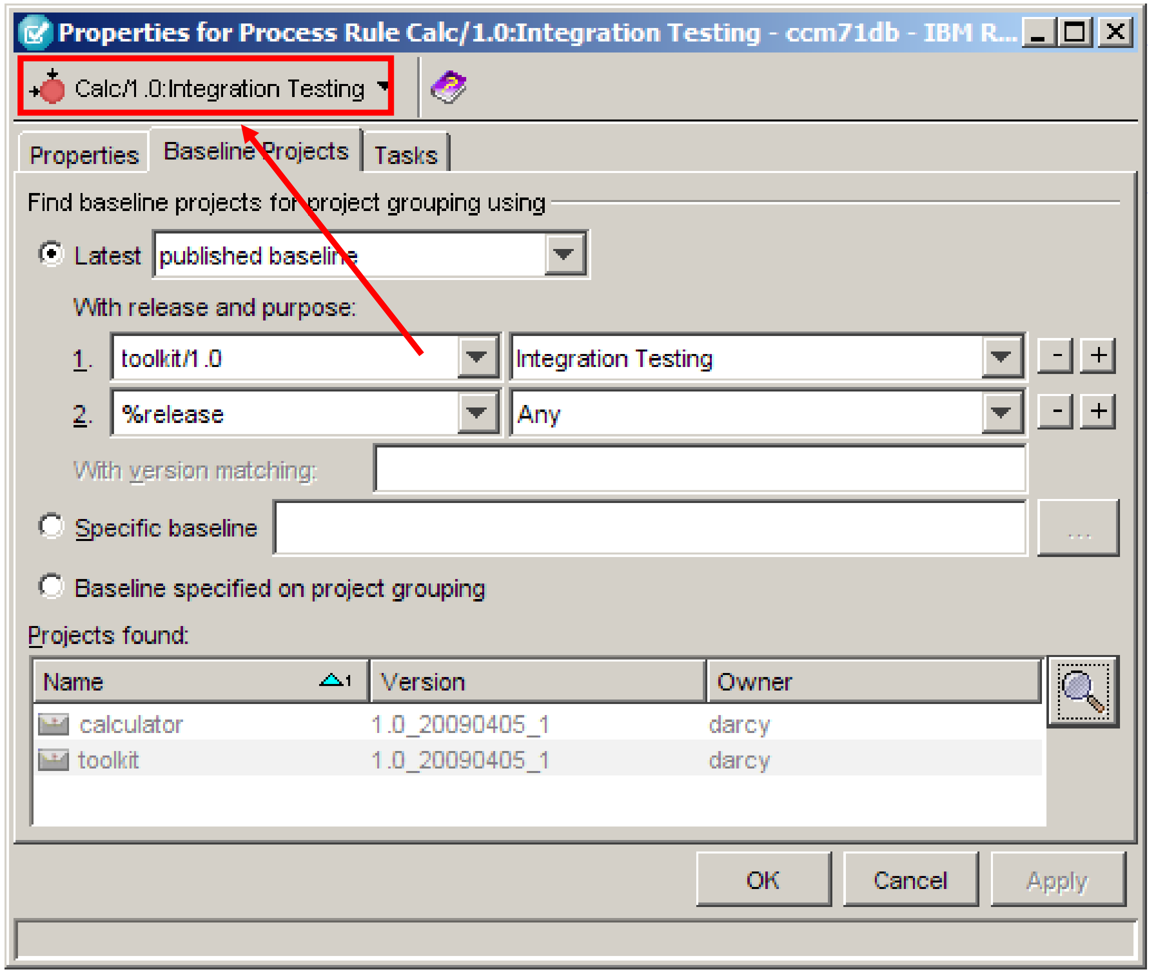
Replace Illegal Characters With: _

[✓] Always include all subprojects regardless of release

Mixed Release Hierarchies and Baselines



Components can use the hierarchy baseline


Properties for Process Rule Calc/1.0:Integration Testing - ccm71db - IBM R...

Calc/1.0:Integration Testing

Find baseline projects for project grouping using

(●) Latest: published baseline

With release and purpose:

1. toolkit/1.0 — Integration Testing
2. %release — Any

With version matching:

( ) Specific baseline

( ) Baseline specified on project grouping

Projects found:

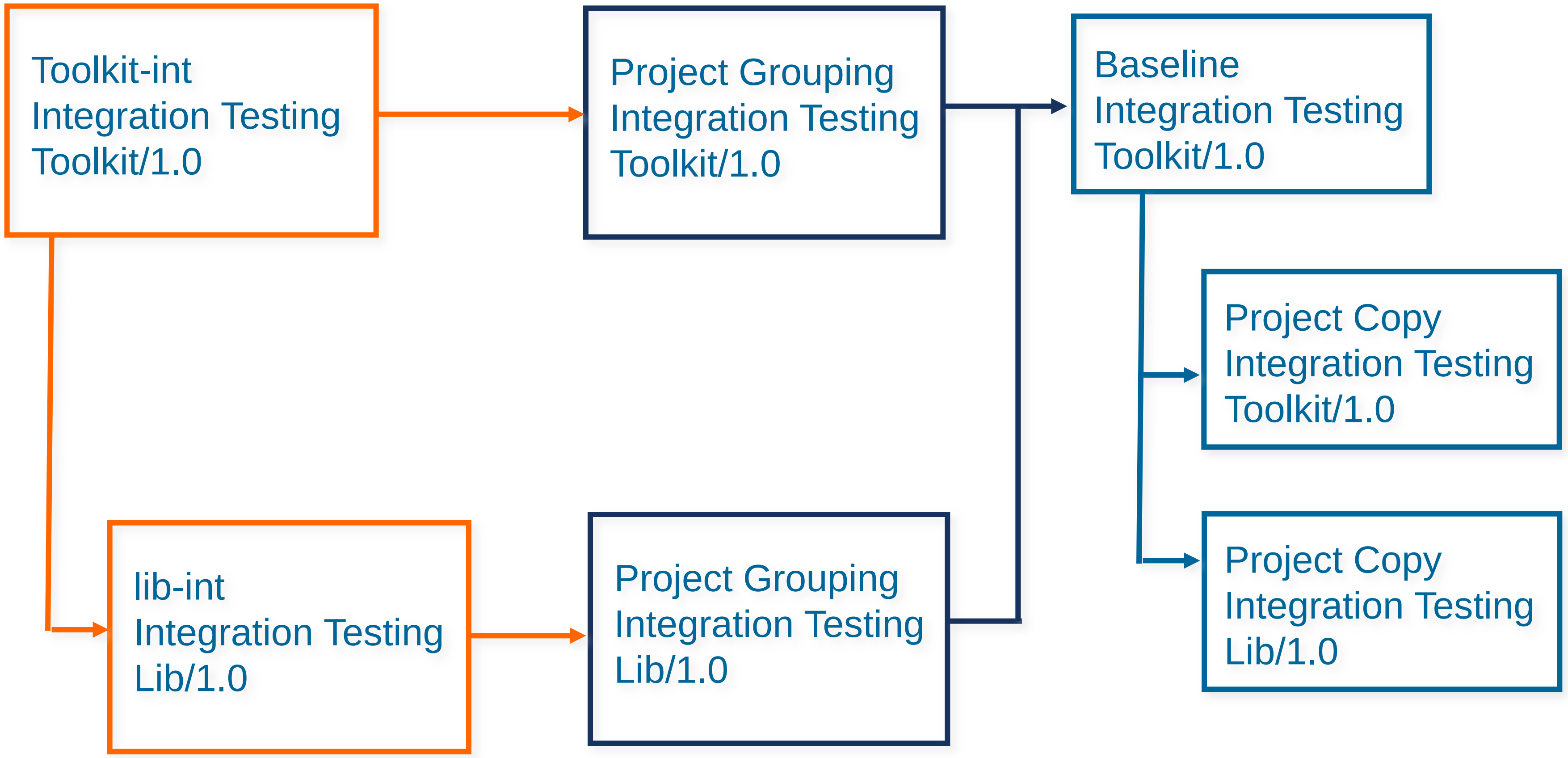
Name	Version	Owner
calculator	1.0_20090405_1	darcy
toolkit	1.0_20090405_1	darcy

Subcomponents with different releases can use the hierarchy baseline

New command line support for copying baseline rules to other process rules

- `ccm process_rule –copy –baseline_rules_only`

Mixed Releases with Single Baseline



Synergy™
Patterns for Component Based Development

Patterns

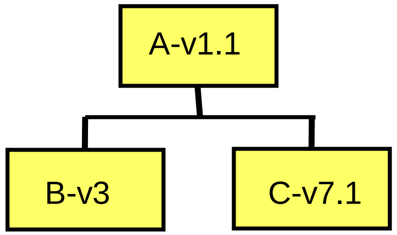
- What is a pattern?
  - A pattern is a collection of techniques for solving a particular problem or requirement.
  - For example, a pattern for parallel development would be necessary for a team that must develop two different releases of the same software application, such as a fix release and a release that contains new features.

Introduction - Why Component Based Development?

- Component Based Development (CBD) helps organizations…
- Manage Complexity
  - Complex systems
  - The skills and knowledge needed to develop system component lies in different and possibly distributed teams
- Reuse – Product Line Engineering – SW Product Lines
  - Common components shared across multiple products
- Service Oriented Architecture
  - Application integration

But really we just want to formally manage the code level interactions between teams to make it work better…

Introduction - How are reused components represented?



- As a file (a product file)
  - Library files
  - JAR files
- As a Synergy project
  - Application A version v1.1 uses 2 components
    - B version V3
    - C version V7.1
  - Component hierarchy is then represented as a Synergy project hierarchy

Introduction - Component-Based Development

- How can teams manage the components they need once they've chosen how to use CBD?
  - Use **releases** to organize components.
- A release specifies the release label of your software application.
- Releases are similar to versions, but they apply to an entire software product.
- A release can represent a product already delivered or released, or a release currently under development.
- All projects and tasks are marked for a specific release.

Introduction - Component-Based Development

- Only use different release values if it is necessary.
- Simplify the relationships between the components and the programs that use them.
- Include components using a standard task-based development approach.

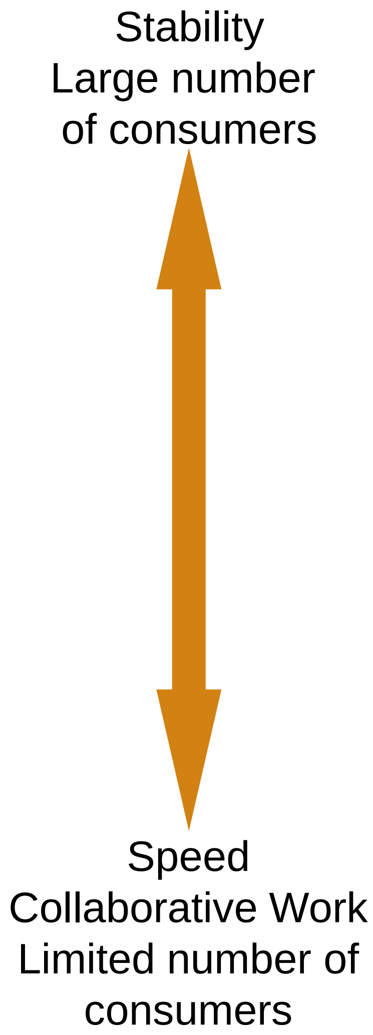
GUI Library	Gui/1.1, Gui/1.2, Gui/1.3
Editor Application	Ed/1.0, Ed/2.0, Ed/2.1, Ed/3.0, Ed/4.0
Calculator Application	Calc/1.0, Calc/2.0

Introduction - Component-Based Development

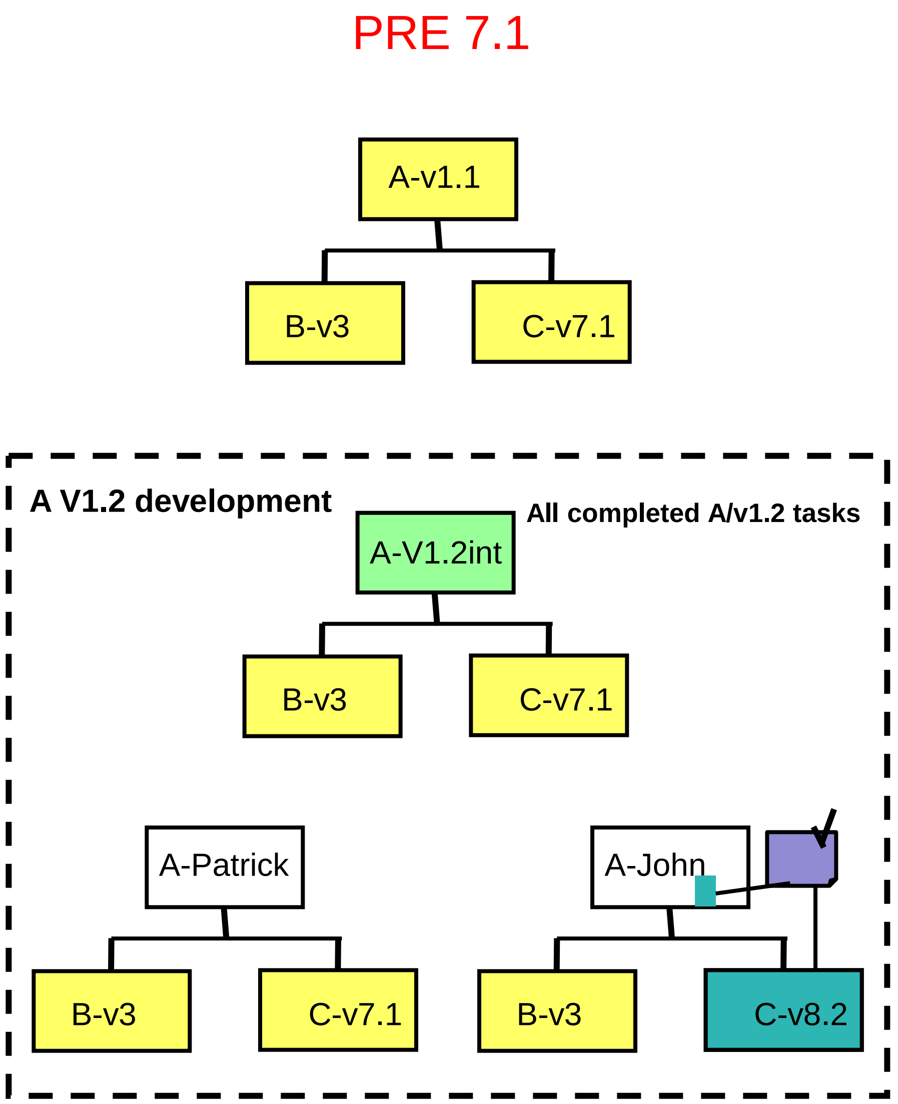
- Use a different release stream for each component to decouple the different components so they are independent of one another.
- Decoupling the different components enables them to be on different release schedules.
- Decoupling also ensures that development teams can use different processes, if they have the need.

Component Based Patterns

- **Primary:** Controlled Update
- **Primary:** Incremental Update from published baselines
- **Primary:** Incremental Update
- **Variation:** Incremental Update with active development of subcomponents

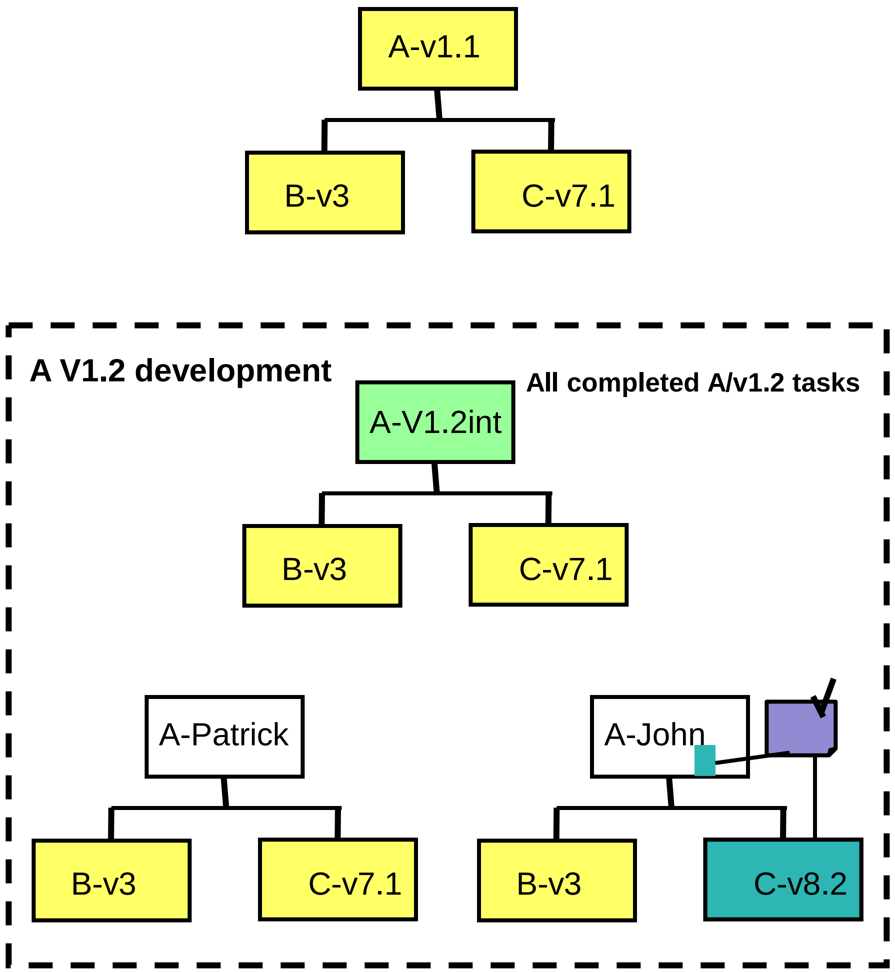


Primary Pattern : Component Based Development Controlled Update



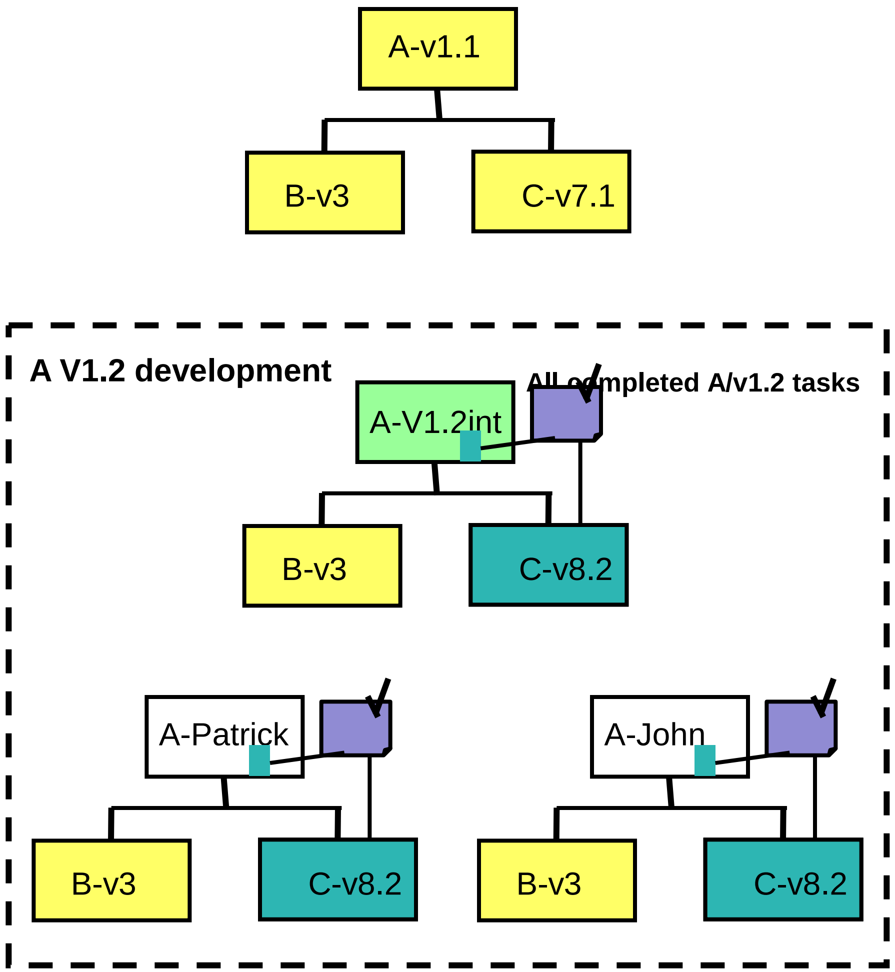
- Each component has its own development/release cycle
- Each component is maintained by a specific development team
- Component consumers reuse **static** versions of components
- Task Based workflow
  - Assign a task to use a new sub-component static version
  - The Developer…
    - selects the assigned task,
    - attaches the new sub-component version to the task,
    - performs the necessary changes and tests,
    - completes their task.

Primary Pattern : Component Based Development Controlled Update



- Each component has its own development/release cycle
- Each component is maintained by a specific development team
- Component consumers reuse **static** versions of components
- Task Based workflow
  - Assign a task to use a new sub-component static version
  - The Developer…
    - Selects the version of the component to be used,
    - performs the necessary changes and tests,
    - completes their task.

Primary Pattern : Component Based Development Controlled Update



- Each component has its own development/release cycle
- Each component is maintained by a specific development team
- Component consumers reuse **static** versions of components
- Task Based workflow
- When the task is completed it follows the workflow like any other task

Primary Pattern : Component Based Development Controlled Update Summary

- Works with standard process rules
- Good for managing systems that are configured by combining specific versions of many different components.
- Clear separation between the component committer (responsible for creating the new versions of the component) and the possibly numerous component consumers
    - Not ideal for teams who work closely on enhancing different “layers” for the same delivery, who want to integrate and test incremental changes on a regular basis.

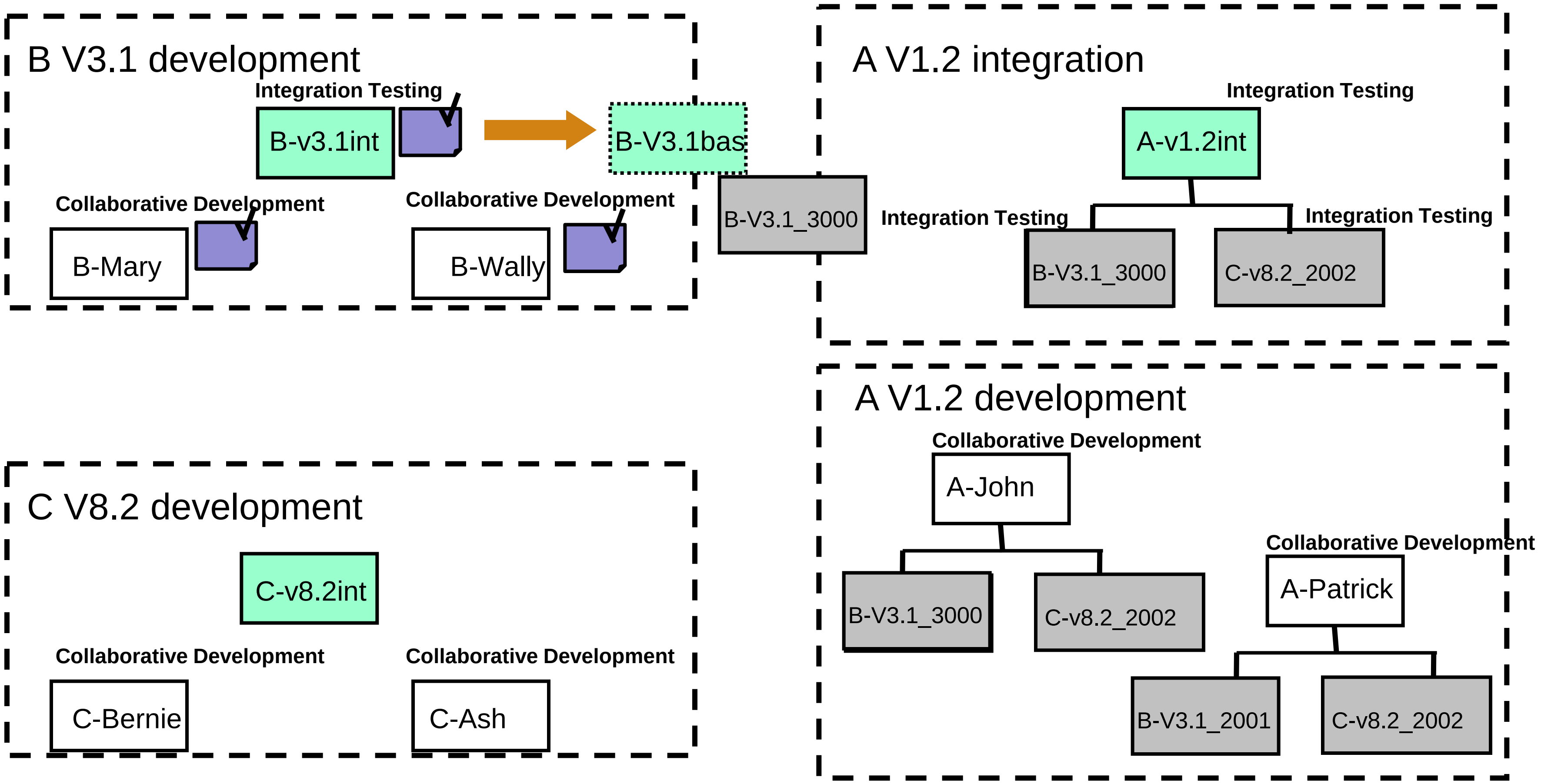
Primary Pattern : Component Based Development Incremental Update

- Closer relationship between the component committer and the component consumer(s)
  - Less likely to have many consumers

- Each component team works on their component

- One common integration space for integrating all completed changes for all components
  - Rapid integration cycles

Primary Pattern: Component Based Development Incremental Update from Published Baselines Using Component Tasks



Questions

Thank
You