

Implementing Backout and Terminate in Service Manager® Change Management

Best Practice Guide on How to Implement the Backout and Terminate Functionality to
Other Change Categories

HP® Management Software — IT Service Management



| | |
|--|----|
| Introduction | 2 |
| Requirements..... | 2 |
| Current implementation of Backout and Terminate in Release Management | 2 |
| Backout implementation | 2 |
| backout.release Process | 2 |
| Detail Format Control..... | 5 |
| Terminate Implementation | 5 |
| Detail Format Control | 8 |
| Implementing Backout Phase and Terminate processing in other change categories | 9 |
| What is a phase pointer?..... | 9 |
| Implementing Backout in other categories | 10 |
| DisplayOptions | 10 |
| Phase Definition | 11 |
| Category Definition | 11 |
| Format Control..... | 12 |
| Implementing Terminate in other categories..... | 13 |
| DisplayOptions | 13 |
| Format Control..... | 14 |
| Entering Phase - Phase Definition..... | 14 |
| Implementing conditional phases in Change Management..... | 15 |
| Conditional workflow visualization | 15 |
| Format Control | 15 |
| For more information..... | 17 |

Introduction

The Release Management change category introduces both Backout and Terminate functionalities into Service Manager. This document shows how to implement Backout and Terminate functionalities in other change categories.

Requirements

This document is intended *solely* for Service Manager administrators. Knowledge of setting up Change Management using Service Manager tailoring tools is required as well.

Important: This document is *not* intended for novice users of Service Manager; it is intended *only* for users who have administrative product knowledge in Service Manager.

Current implementation of Backout and Terminate in Release Management

The Backout and Terminate functionalities in Service Manager Release Management are implemented via the Document Engine, using display options and scripts. The Document Engine State record that handles Backout and Terminate is `cm.view`. The display options used are associated with the display screen `cm.view.display`.

Backout implementation

The Backout button moves the change into the Backout phase; upon closure of the Backout phase, the change returns to its Plan and Design phase.

The `$release.backout` variable is set to true in the pre-RAD expressions of the Backout display option of the `cm.view.display` displayscreen. A back out invokes the `backout.release` Process, where the RAD routines and final expressions described below are executed.

backout.release Process

Initial Expressions

```
$release.backout=nullsub($release.backout, false)
```

First RAD routine call

This call determines whether the current phase is the last phase defined for this change; and if so, sets a variable to indicate that this is the last phase. If the change is in its last phase and the Service Level Agreement (SLA) module is enabled, the SLA outage confirmation routine is called; otherwise this RAD routine call is bypassed.

Process Definition

Process Name:

Save Cursor Position? Run Standard Process when complete?

Run in Window? Window Title:

Initial Expressions Initial Javascript **RAD** Final Expressions Final Javascript Next Process

Expressions evaluated before RAD call

```
$L.file.vars={$L.category, $L.phase, $L.fc, $L.fc.master}
if (index(current.phase in $L.file, phases in $L.category)=lng(denull(phases in $L.category))) then ($L.last=true) else ($L.last=false)
```

RAD Application: Condition:

| Parameter Names | Parameter Values |
|-----------------|------------------|
| file | \$L.file |
| | |
| | |

Post RAD Expressions

Second RAD routine call

The second RAD routine call always executes the Service Manager scripting routine to call the backout.release script. The backout.release script brings up the backout.release form, in which you fill out the reason for the Backout operation.

Expressions evaluated before RAD call

RAD Application: Condition:

| Parameter Names | Parameter Values |
|-----------------|-------------------|
| file | \$L.file |
| name | "backout.release" |
| | |

Post RAD Expressions

Script Panel Definition

General Next Script

Script Name: Is this the first panel?

Form Name: Cluster Name:

Display Screen Name:

Enter the conditions for the execution of this script

Skip Display:

Bypass Cond:

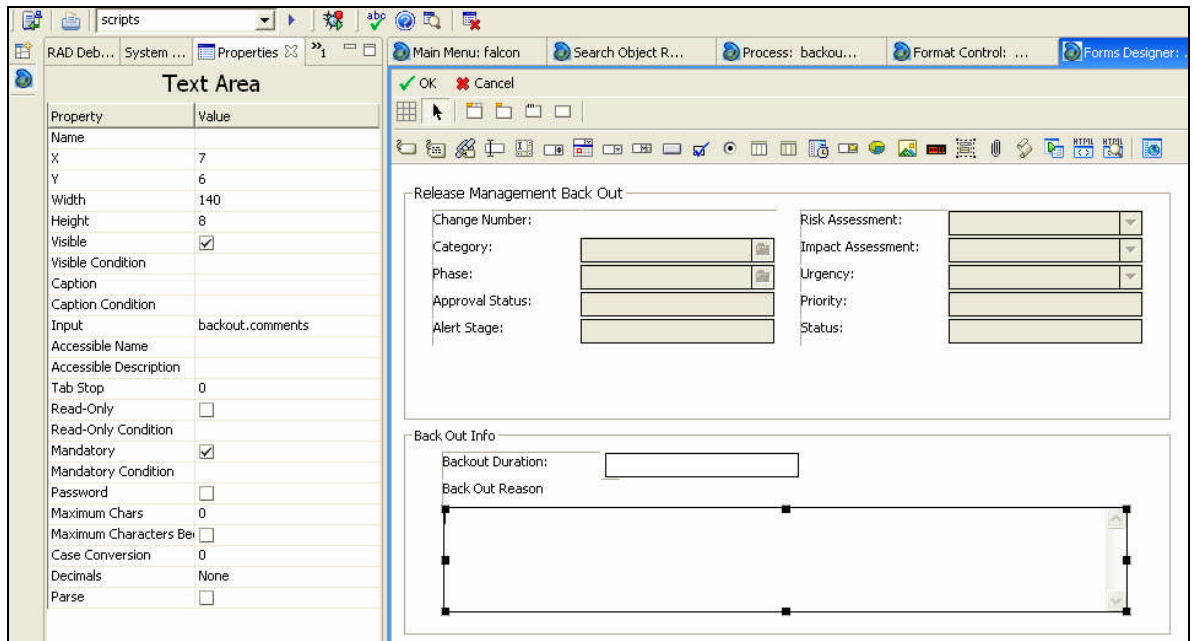
Enter=Continue?

Pre RAD Statements Pre RAD Javascript **RAD** Post RAD Statements Post RAD Javascript

Enter the statements to execute after the form executes and after the application runs

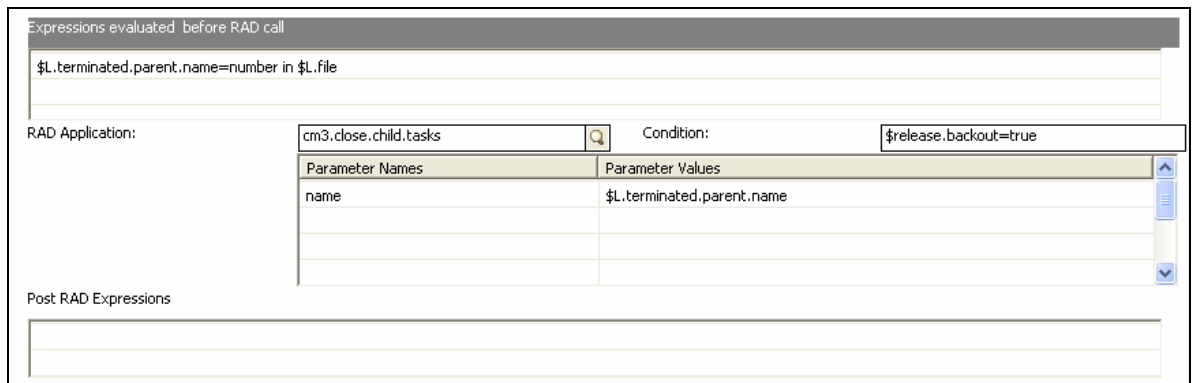
```
$terminate.ok=true
$G.backout.comments2=backout.comments in $script
```

The backout.release form contains the following fields:



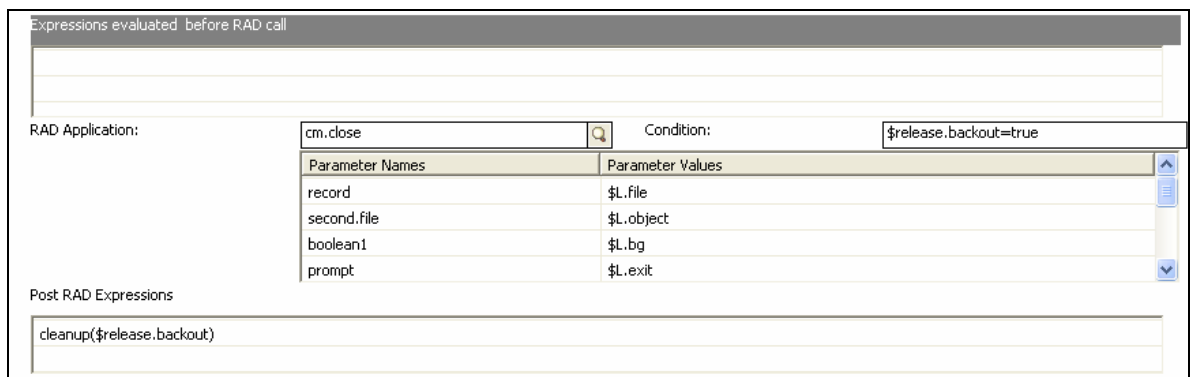
Third RAD routine call

The third RAD routine call forces the closure of any open tasks for the current phase in which the Backout option was selected. It is executed only when performing a back out.



Fourth RAD routine call

The fourth RAD routine call is called only when performing a back out. It closes the current phase of the change in which the Backout option was selected.



Final Expressions

These expressions are evaluated after the RAD routine calls are finished, every time this process is called. It sets Exit variables and resets the phase pointer to a NULL value.

```

Initial Expressions Initial Javascript RAD Final Expressions Final Javascript Next Process
if ($L.exit="normal") then ($L.exit="closestate")
if ($L.exit="bg") then ($L.exit="exit")
if ($L.exit="exit") then ($L.exit.when.done=true;if (filename($L.file)="cm3r") then ($phasepntr=NULL);if (filename($L.file)="cm3r") then ($tphasepntr=NULL))
if ($L.bg and $L.exit="exit") then ($L.exit="normal")

```

Detail Format Control

After execution of the process record completes, control is returned to the Document Engine and the standard processing. The return to the Plan and Design phase is handled in the detail Format Control (cm3r.release) by performing the statements indicated below. The first highlighted statement always executes during the Close process, and checks whether the current phase pointer is set to the Backout phase; if so the statement sets a variable that indicates that the change should go back to the Plan phase. The second highlighted statement executes during Close processing only if the variable to return to the Plan phase is set. If executed, the statement sets the phase pointer to the first phase and sets the current.phase field in the record to the value contained in the variable \$L.cur.phase.

| Format Control Maintenance - Calculations | | | | | |
|---|------|--------|-----------|-------------|---|
| Name: cm3r.release | | | | View: short | |
| add | u... | delete | display | initial | calculation |
| | | true | true | true | \$ip.array=nullsub(\$ip.array, {});\$d.assignment=nullsub(\$d.assignment, {});\$loc.array=nullsub(\$loc.array, {}) |
| | | | null(a... | true | \$ip.array={};\$d.assignment={};\$loc.array={} |
| | | true | true | true | \$do.train=false |
| | | true | true | true | if (initial.impact in \$file<"3") then (\$do.train=true) |
| | | true | | | if (\$phasepntr=3 and approval.status in \$file~="denied" and initial.impact in \$file>2) then (\$L.skip.training=true) |
| | | true | | | if (\$phasepntr=3 and approval.status in \$file="denied") then (\$L.plan.again=true) |
| | | true | | | if (\$phasepntr=5 and \$release.backout=false) then (\$L.skip.backout=true) |
| | | true | | | if (\$L.cur.pntr=6) then (\$L.plan.again=true) |
| | | true | | | if (\$phasepntr=7 and approval.status in \$file="denied") then (\$L.plan.again=true) |
| | | true | | | if (misc2 in \$file="fail" and \$L.cur.phase="Build and Test") then (\$L.plan.again=true) |
| | | | | | \$L.plan.again=true |
| | | | | | \$phasepntr=1;current.phase in \$file=\$L.cur.phase |
| | | | | | \$L.skip.training=true |
| | | | | | \$phasepntr=4;current.phase in \$file=\$L.cur.phase |
| | | | | | \$L.skip.backout=true |
| | | | | | \$phasepntr=6;current.phase in \$file=\$L.cur.phase |
| | | | true | true | \$show.rlm.testing=false |
| | | | true | true | if (\$L.cur.pntr>=3) then (\$show.rlm.testing=true) |
| | | true | | | if (\$terminate.release=true) then (\$phasepntr=7;current.phase in \$file="Verification") |

Terminate Implementation

The Terminate button moves the Release Management change to the last phase, Verification, and closes the change with a status of Terminated.

In the pre-RAD Expressions of the Terminate displayoption of the cm.view.display displayscreen, the \$terminate.release variable is set to True. The Terminate action invokes the terminate.release Process, which performs the following actions:

First RAD routine call:

The first RAD routine call determines whether the current phase is the last phase for this change; if so it sets a variable to indicate this. If the current phase is the last phase and the SLA module is enabled, the first RAD routine call then calls the SLA outage confirmation routine; otherwise it bypasses it.

Process Definition

Process Name:

Save Cursor Position? Run Standard Process when complete?

Run in Window? Window Title:

Initial Expressions Initial Javascript **RAD** Final Expressions Final Javascript Next Process

Expressions evaluated before RAD call

```
$L.file.vars={$L.category, $L.phase, $L.fc, $L.fc.master}
if (index(current.phase in $L.file, phases in $L.category)=lng(denu((phases in $L.category)))) then ($L.last=true) else ($L.last=false)
```

RAD Application: Condition:

| Parameter Names | Parameter Values |
|-----------------|------------------|
| file | \$L.file |
| | |
| | |

Post RAD Expressions

Second RAD routine call:

The second RAD routine call sets the variable \$terminate.ok to false, defines the name of the script to be executed, and overrides the script name if the change category is for Knowledge Management. The second RAD call then executes the Service Manager scripting wrapper routine and executes the script that is passed in.

Expressions evaluated before RAD call

```
$terminate.ok=false
$L.script.name="terminate.release";if (name in $L.category="KM Document") then ($L.script.name="KM Documents closed")
```

RAD Application: Condition:

| Parameter Names | Parameter Values |
|-----------------|------------------------|
| file | \$L.file |
| name | \$L.script.name |
| prompt | \$L.return.script.exit |

Post RAD Expressions





```
if ($L.return.script.exit="cancel") then ($terminate.ok=false;$terminate.release=false;$L.mode="update")
```

In the post-RAD expressions the second RAD call checks the return exit of the script; if the script returns cancel, the second RAD call ensures that the \$terminate.ok and \$terminate.release variables are both set to false and resets \$L.mode to update.

The terminate.release script first brings up the terminate.release form and then sets the \$terminate.ok variable to true:

Script Panel Definition

Is this the first panel?
 Cluster Name:

Script Name:
 Form Name:  
 Display Screen Name:  

Enter the conditions for the execution of this script

Skip Display:
 Bypass Cond:
 Enter=Continue?

Pre RAD Statements
 Pre RAD Javascript
 RAD
 Post RAD Statements
 Post RAD Javascript



Enter the statements to execute after the form executes and after the application runs




```
$terminate.ok=true
```

The terminate.release form contains two subforms, terminate.hdr and terminate.close.detail:


terminate.hdr:

Release Management Termination

Change Number:
 Category: 
 Phase: 
 Approval Status:
 Alert Stage:


Risk Assessment: 
 Impact Assessment: 
 Urgency: 
 Priority:
 Status:

terminate.close.detail:

Messages System ... Properties  1

Main Menu: falcon States: cm.view scripts: terminate.release Forms Designer: terminate.close.

OK Cancel


 Termination Reason

| Property | Value |
|------------------------|-------------------------------------|
| Name | MultiText5 |
| X | 7 |
| Y | 6 |
| Width | 140 |
| Height | 8 |
| Visible | <input checked="" type="checkbox"/> |
| Visible Condition | |
| Caption | |
| Caption Condition | |
| Input | closing.comments |
| Accessible Name | |
| Accessible Description | |
| Tab Stop | 0 |
| Read-Only | <input type="checkbox"/> |
| Read-Only Condition | |
| Mandatory | <input checked="" type="checkbox"/> |
| Mandatory Condition | |
| Password | <input type="checkbox"/> |
| Maximum Chars | 0 |
| Maximum Characters Be | <input type="checkbox"/> |
| Case Conversion | 0 |
| Decimals | None |
| Parse | <input type="checkbox"/> |

Third RAD routine call:

The third RAD routine call forces closed any open tasks for the current phase in which the Terminate option was selected. This RAD call is executed only when the \$terminate.ok variable is set to True.

Expressions evaluated before RAD call

`$L.terminated.parent.name=number in $L.file`

RAD Application: Condition:

| Parameter Names | Parameter Values |
|-----------------|----------------------------|
| name | \$L.terminated.parent.name |
| | |

Post RAD Expressions

Fourth RAD routine call:

The fourth RAD routine call closes the current phase of the change where the Terminate option was selected. As with the third RAD routine call, the fourth RAD routine call executes only when the \$terminate.ok variable is set to True.

Expressions evaluated before RAD call

RAD Application: Condition:

| Parameter Names | Parameter Values |
|-----------------|------------------|
| record | \$L.file |
| second.file | \$L.object |
| boolean1 | \$L.bg |
| prompt | \$L.exit |

Post RAD Expressions

`cleanup($terminate.ok)`

Final Expressions:

These expressions are evaluated after the RAD calls are finished, every time this Process is called. The final expressions set exit variables and reset the phase pointer to a NULL value.

◆ Initial Expressions
◆ Initial Javascript
◆ RAD
◆ Final Expressions
◆ Final Javascript
◆ Next Process

```

if ($L.exit="normal" and $terminate.release=true) then ($L.exit="closestate")
if ($L.exit="bg") then ($L.exit="exit")
if ($L.exit="exit") then ($L.exit.when.done=true;if (filename($L.file)="cm3r") then ($phaseptr=NULL);if (filename($L.file)="cm3t") then ($tphaseptr=NULL))
if ($L.bg and $L.exit="exit") then ($L.exit="normal")

```

Detail Format Control

After the execution of the Process record, control is returned to the Document Engine and the standard processing. The statement highlighted below always executes during the Close process and checks whether the \$terminate.release variable is set to true. If so, then the phase pointer is set to the last phase number, and the current.phase column in the current record is set to the name of the last phase — in this case Verification.

| Format Control Maintenance - Calculations | | | | | | |
|---|------|--------------|-----------|---------|--|----|
| Name: | | cm3r.release | | | View: | sh |
| add | u... | delete | display | initial | calculation | |
| | | true | true | true | \$ip.array=nullsub(\$ip.array, {});\$d.assignment=nullsub(\$d.assignment, {});\$loc.array=nullsub(\$loc.array, {}) | |
| | | | null(a... | true | \$ip.array={};\$d.assignment={};\$loc.array={} | |
| | | true | true | true | \$do.train=false | |
| | | true | true | true | if (initial.impact in \$file<"3") then (\$do.train=true) | |
| | | true | | | if (\$phaseptr=3 and approval.status in \$file~="denied" and initial.impact in \$file>2) then (\$L.skip.training=true) | |
| | | true | | | if (\$phaseptr=3 and approval.status in \$file="denied") then (\$L.plan.again=true) | |
| | | true | | | if (\$phaseptr=5 and \$release.backout=false) then (\$L.skip.backout=true) | |
| | | true | | | if (\$L.cur.pntr=6) then (\$L.plan.again=true) | |
| | | true | | | if (\$phaseptr=7 and approval.status in \$file="denied") then (\$L.plan.again=true) | |
| | | true | | | if (misc2 in \$file="fail" and \$L.cur.phase="Build and Test") then (\$L.plan.again=true) | |
| | | | | | \$L.plan.again=true | |
| | | | | | \$phaseptr=1;current.phase in \$file=\$L.cur.phase | |
| | | | | | \$L.skip.training=true | |
| | | | | | \$phaseptr=4;current.phase in \$file=\$L.cur.phase | |
| | | | | | \$L.skip.backout=true | |
| | | | | | \$phaseptr=6;current.phase in \$file=\$L.cur.phase | |
| | | | true | true | \$show.rm.testing=false | |
| | | | true | true | if (\$L.cur.pntr>=3) then (\$show.rm.testing=true) | |
| | | true | | | if (\$terminate.release=true) then (\$phaseptr=7;current.phase in \$file="Verification") | |

Implementing Backout Phase and Terminate processing in other change categories

The majority of the components used for the implementation of the Backout and Terminate processing can be reused in the other Change Management categories. The Detail Format Control is the only section that is specific to the category you want to implement.

What is a phase pointer?

The phase pointer (\$phaseptr) is used in the Change Management module to transition a change between the defined phases. In the out-of-the-box workflow for the change category of Hardware, the phase pointer starts with number 1 and increments through the phases to a final pointer value of 7.

For example:



\$phaseptr=1 is Change Logging

\$phaseptr=2 is Change Review

\$phaseptr=3 is Change Assessment & Planning

\$phaseptr=4 is Prepare for Change Approval

\$phaseptr=5 is Change Approval

\$phaseptr=6 is Change Implementation

\$phaseptr=7 is Evaluation & Change Closure

Implementing Backout in other categories

DisplayOptions

First, modify the Backout displayoption in the cm.view.display displayscreen to be effective for other categories of changes. To do this you set a user.condition in the displayoption. Out-of-box, the condition for the Backout button is:

```
evaluate($L.tableAccess.close) and open in $L.file=true and  
nullsub($G.ess, false)=false and ($phasepntr=5 and category in  
$L.file="Release Management")
```

This means that the Backout button is available to you if the following conditions are met:

- You have the right to close a change.
- The change is open at the time.
- You are not an ESS user.
- The current phase is phase # 5.
- You are in the Release Management change category.

As an example, one possible user condition modification could be:

If you want the Backout option available in any phase of any change, where

- The user has the right to execute a close, and
- The change is still open, and
- The user is not an ESS user,

then the user condition for the displayoption needs to be modified as follows:

```
evaluate($L.tableAccess.close) and open in $L.file=true and  
nullsub($G.ess, false)=false
```

Another possible user condition modification could be implemented as follows:

If you want the Backout option available where

- The user has the right to execute a close, and
- The change is still open, and
- The user is not an ESS user, and
- The Release Management change is in phase 5 or the Hardware change is in phase 6, then the user condition would be:

```
evaluate($L.tableAccess.close) and open in $L.file=true and  
nullsub($G.ess, false)=false and ($phasepntr=5 and category in  
$L.file="Release Management" or $phasepntr=6 and category in  
$L.file="Hardware")
```

In summary, the user condition can be modified to contain any expression that, when evaluated, allows the Backout option to be made available.

Once the user condition is set to make the Backout option available, the Backout action triggers the backout.release Process as soon as you click the Backout button. If you want the Process to remain as it is in the out-of-box system, then no changes are necessary in the Process record or the backout.release script. The detail Format Control record must be adjusted to allow the back out to return to the desired phase.

Our example will be done using the Hardware change category. The Backout button should close the Hardware Implementation phase and return the user to the Approval phase. To do so, set the user condition of the Backout displayoption to the following:

```
evaluate($L.tableAccess.close) and open in $L.file=true and  
nullsub($G.ess, false)=false and ($phasepntr=5 and category in  
$L.file="Release Management" or $phasepntr=6 and category in  
$L.file="Hardware")
```

Phase Definition

To create the Backout Hardware phase, follow these steps:

1. Go to Change Management -> Change Phases.
2. Select the **Backout** phase.
3. Change the Change Phase field to **Backout Hardware** and click **Add**.
4. Change the Description to **Hardware back out**.
5. Click the **Scripts / Views** tab and change the Default and Close Views to **cm3r.hrdw**

The screenshot shows a configuration form for a phase. The top section is titled 'Change Phase:' and contains the following fields:

- Change Phase: Backout Hardware
- Description: Hardware back out
- OperID (true) or Full Name (false): false
- Require a Start/End Date?

Below this is a tabbed interface with the following tabs: Definition, Alerts/Open & Close Behavior, Approval/Review, Model/Tasks, Auto Open Tasks, Scripts/Views, and Reports. The 'Scripts/Views' tab is selected.

The 'Scripts/Views' tab contains the following sections:

- Risk**
 - Maximum: 5
 - Calculation: false
- History**
 - Pages: true
 - Audit Records: true
- Controls**
 - Update: true
 - Approval: false
 - Close: true
 - Message: true

Category Definition

Now the Backout Hardware phase must be added to the Hardware category. To do so, follow these steps:

1. Go to **Change Management** -> **Change Categories**.
2. Select the **Hardware** category.
3. Add the **Backout Hardware** phase between the Change Implementation and Evaluation & Change Closure phases.
4. Save the record and exit.

Category Name:

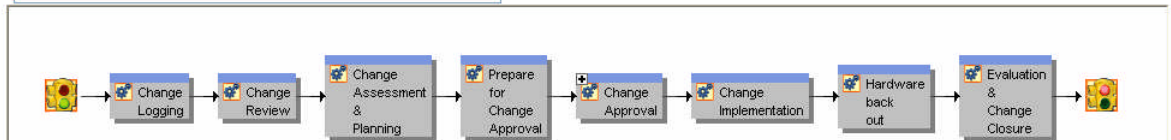
Category Description:

Availability:

Default Template:

Assign Number?

| |
|------------------------------|
| Change Phases |
| Change Logging |
| Change Review |
| Change Assessment & Planning |
| Prepare for Change Approval |
| Change Approval |
| Change Implementation |
| Backout Hardware |
| Evaluation & Change Closure |



Format Control

1. Go to **Utilities** -> **Tools** -> **Format Control**.
2. Select the record **cm3r.hardware**
3. In the calculations, make the following changes. First set the execute conditions for add, update, delete, display, and initial in the following expression to a value of **true** for the following line that needs to be added:

```
$L.cur.phase=current.phase in $file; $L.cur.pntr=$phasepntr
```

4. Set the execute condition for delete in the following expression that needs to be added to a value of **true**.

```
if ($L.cur.pntr=7) then $phasepntr=1;current.phase in $file=$L.cur.phase
```

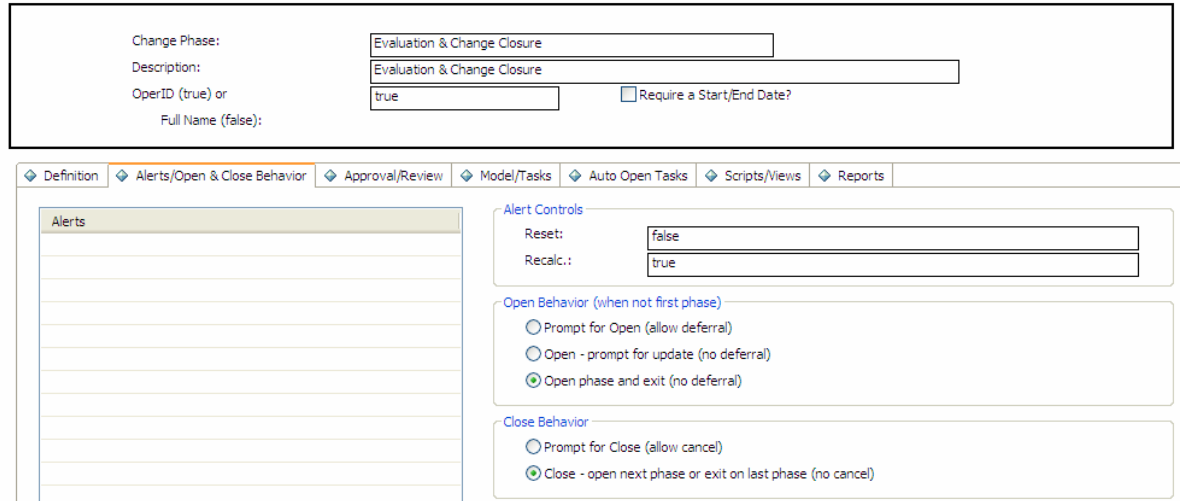
Format Control Maintenance - Calculations

Name: View:

| add | update | delete | display | initial | calculation |
|------|--------|--------|---------|---------|--|
| true | true | true | true | true | \$L.cur.phase=current.phase in \$file;\$L.cur.pntr=\$phasepntr |
| | | | | | if (\$L.cur.pntr=7) then (\$phasepntr=1);current.phase in \$file=\$L.cur.phase |

Entering Phase - Phase Definition

If you want the change to immediately move to the Backout phase rather than first prompting for the closure of the Hardware Evaluation & Change Closure phase, go to the **Hardware Evaluation & Change Closure** phase **Alerts / Open & Close Behavior** tab. In the Close Behavior frame check the **Close – open next phase or exit on last phase (no cancel)** as seen below:



Implementing Terminate in other categories

DisplayOptions

First, modify the Terminate displayoption in the cm.view.display displayscreen to be effective for other categories of changes. This is done by setting a user.condition in the displayoption.

Out-of-box, the condition for the Terminate button is:

```
evaluate($L.tableAccess.close) and open in $L.file=true and  
nullsub($G.ess, false)=false and category in $L.file="Release Management"  
and ($phasepnr=3 or $phasepnr=2 or $phasepnr=1)
```

This means that if

- You have the right to close a change, and
- The change is open at the time, and
- You are not an ESS user, and
- The current phase is a phase that is defined to occur before the Distribution phase, and
- The category of the change you are in is Release Management,

then the Terminate button is available.

To activate the button for Hardware and Release Management changes, this condition must be changed to:

```
evaluate($L.tableAccess.close) and open in $L.file=true and  
nullsub($G.ess, false)=false and (category in $L.file="Release  
Management" and ($phasepnr=3 or $phasepnr=2 or $phasepnr=1)) or  
(category in $L.file="Hardware" and ($phasepnr=5 or $phasepnr=4 or  
$phasepnr=3 or $phasepnr=2 or $phasepnr=1))
```

This means that if

- You have the right to close a change, and

- The change is open at the time, and
- You are not an ESS user, and
- The current phase is defined to occur before the Distribution or Implementation phase, and
- The category of the change is "Release Management", or the category of the change is "Hardware",

then the Terminate button is available.

Note: As a best practice, the Terminate button should be available only in the phases before the Change is implemented or distributed across the system. Since the change is implemented in phase 6, the Terminate button is available in phases 5, 4, 3, 2 and 1. Once implementation has started, the change should be backed out, not terminated.

Format Control

Next, the cm3r.hardware format control must be adjusted. Set the phase pointer to the last phase of the category (with the Backout phase in place, this is number 8 here); and set the current.phase to the name of that last phase.

Calculations:

Set the condition in the Delete column to **true** and enter the following calculation for that line:

```
if ($terminate.release=true) then ($phaseptr=8;current.phase in $file="Evaluation & Change Closure")
```

Format Control Maintenance - Calculations

Name: cm3r.hardware View: short

| add | update | delete | display | initial | calculation |
|------|--------|--------|---------|---------|---|
| true | true | true | true | true | \$.cur.phase=current.phase in \$file;\$.cur.pntr=\$phaseptr |
| | | true | | | if (\$.cur.pntr=7) then (\$phaseptr=1);current.phase in \$file=\$.cur.phase |
| | | true | | | if (\$terminate.release=true) then (\$phaseptr=8;current.phase in \$file="Evaluation & Change Closure") |

Format Control Maintenance - Validations

Name: cm3r.hardware View: long

Use Pop-up messages:

| Validations | |
|--------------|------------------------------------|
| Validation | not null(assigned.to in \$file) |
| Message | The Assigned To field is required. |
| Comments | |
| Add | |
| Update | |
| Delete | status in \$file~="terminated" |
| Display | |
| Initial | |
| Set Focus to | assigned.to |
| Message ID | 870 |

Entering Phase - Phase Definition

For the terminate functionality to work correctly, click the **Alerts / Open & Close Behavior** tab and check **Close – open next phase or exit on last phase (no cancel)** in the Close Behavior area for each of the phases that will have the Terminate button.

Change Phase: Evaluation & Change Closure
 Description: Evaluation & Change Closure
 OperID (true) or Full Name (false): true Require a Start/End Date?

Definition Alerts/Open & Close Behavior Approval/Review Model/Tasks Auto Open Tasks Scripts/Views Reports

Alerts

Alert Controls
 Reset: false
 Recalc.: true

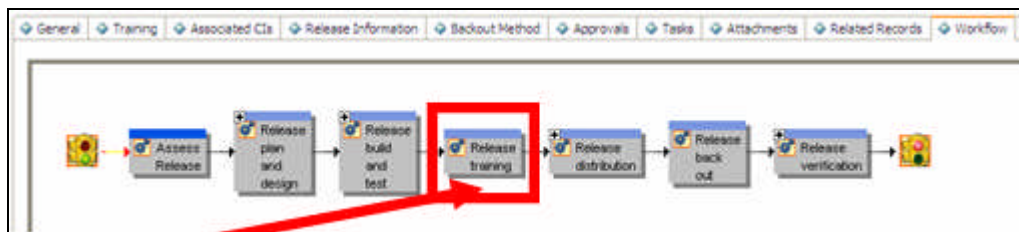
Open Behavior (when not first phase)
 Prompt for Open (allow deferral)
 Open - prompt for update (no deferral)
 Open phase and exit (no deferral)

Close Behavior
 Prompt for Close (allow cancel)
 Close - open next phase or exit on last phase (no cancel)

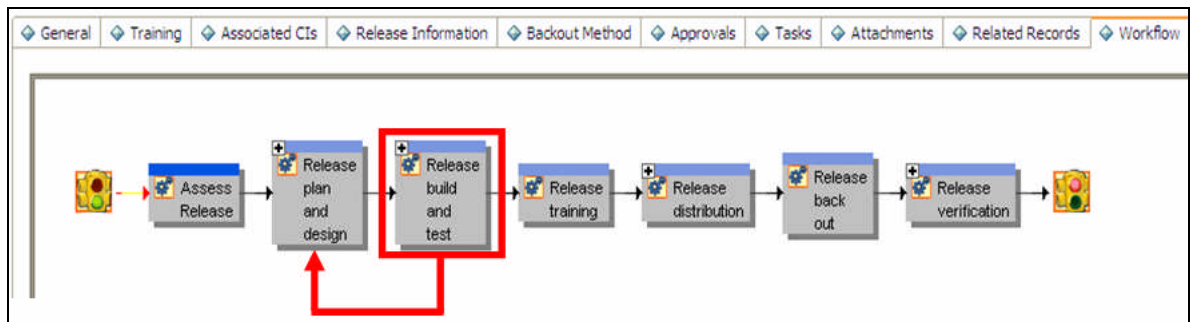
Implementing conditional phases in Change Management

Conditional workflow visualization

Release Management has two conditional phase successions implemented:



If the impact is a 2 or higher, then a training phase will be entered.



If the test fails during the test phase, the change will return to the build phase.

Format Control

The implementation of these conditional phases is done in a Format Control (cm3r.release).

The areas circled in green are used for enabling or disabling the training phase. The areas circled in red are used for returning to the plan and design phase if the test was marked as failed:

Format Control Maintenance - Calculations

Name: cm3r.release

View: short

| add | u... | delete | display | initial | calculation |
|------|------|--------|-----------|---------|---|
| true | true | true | true | true | \$L.cur.phase=current.phase in \$file |
| true | true | true | true | true | \$L.cur.pntr=\$phasepntr |
| | | true | true | true | \$ip.array=nullsub(\$ip.array, {});\$d.assignment=nullsub(\$d.assignment, {});\$loc.array=nullsub(\$loc.array, {}) |
| | | | null(a... | true | \$ip.array={};\$d.assignment={};\$loc.array={} |
| | | | | | |
| | | true | true | true | \$do.train=false |
| | | true | true | true | if (initial.impact in \$file<"3") then (\$do.train=true) |
| | | | | | |
| | | true | | | if (\$phasepntr=3 and approval.status in \$file~="denied" and initial.impact in \$file>2) then (\$L.skip.training=true) |
| | | true | | | if (\$phasepntr=3 and approval.status in \$file="denied") then (\$L.plan.again=true) |
| | | true | | | if (\$phasepntr=5 and \$release.backout=false) then (\$L.skip.backout=true) |
| | | true | | | if (\$L.cur.pntr=6) then (\$L.plan.again=true) |
| | | true | | | if (\$phasepntr=7 and approval.status in \$file="denied") then (\$L.plan.again=true) |
| | | true | | | if (misc2 in \$file="fail" and \$L.cur.phase="Build and Test") then (\$L.plan.again=true) |
| | | | | | |
| | | | | | \$phasepntr=1;current.phase in \$file=\$L.cur.phase |
| | | | | | \$phasepntr=4;current.phase in \$file=\$L.cur.phase |
| | | | | | \$phasepntr=6;current.phase in \$file=\$L.cur.phase |
| | | | | | |
| | | | true | true | \$show.rlm.testing=false |
| | | | true | true | if (\$L.cur.pntr>=3) then (\$show.rlm.testing=true) |
| | | true | | | if (\$terminate.release=true) then (\$phasepntr=7;current.phase in \$file="Verification") |

For more information

Please visit the HP Management Software support Web site at:

<http://www.hp.com/managementsoftware/support>

This web site provides contact information and details about the products, services, and support that HP Management Software offers.

HP Management Software online software support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by being able to:

- Search for knowledge documents of interest
- Submit and track progress on support cases
- Submit enhancement requests online
- Download software patches
- Manage a support contract
- Look up HP support contacts
- Review information about available services
- Enter discussions with other software customers
- Research and register for software training

Note: Most of the support areas require that you register as an HP Passport user and sign in. Many also require an active support contract.

To find more information about support access levels, go to the following URL:

http://www.hp.com/managementsoftware/access_level

To register for an HP Passport ID, go to the following URL:

<http://www.managementsoftware.hp.com/passport-registration.html>

© 2008 Hewlett-Packard Development Company, L.P. The information contained herein is subject to change without notice. The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

HP and Service Manager are registered trademarks of Hewlett-Packard Development Company, L.P. Verity is a registered trademark of Autonomy Corporation plc and its affiliates. JavaScript is a registered trademark of Sun Microsystems, Inc. in the United States and other countries.

08/2008

