ITIL Change Implementation

Applying ITIL Principles to ServiceNow

Raising Changes

Creating a Record Producer

	Functionality described here requires the **Admin** role.

Overview

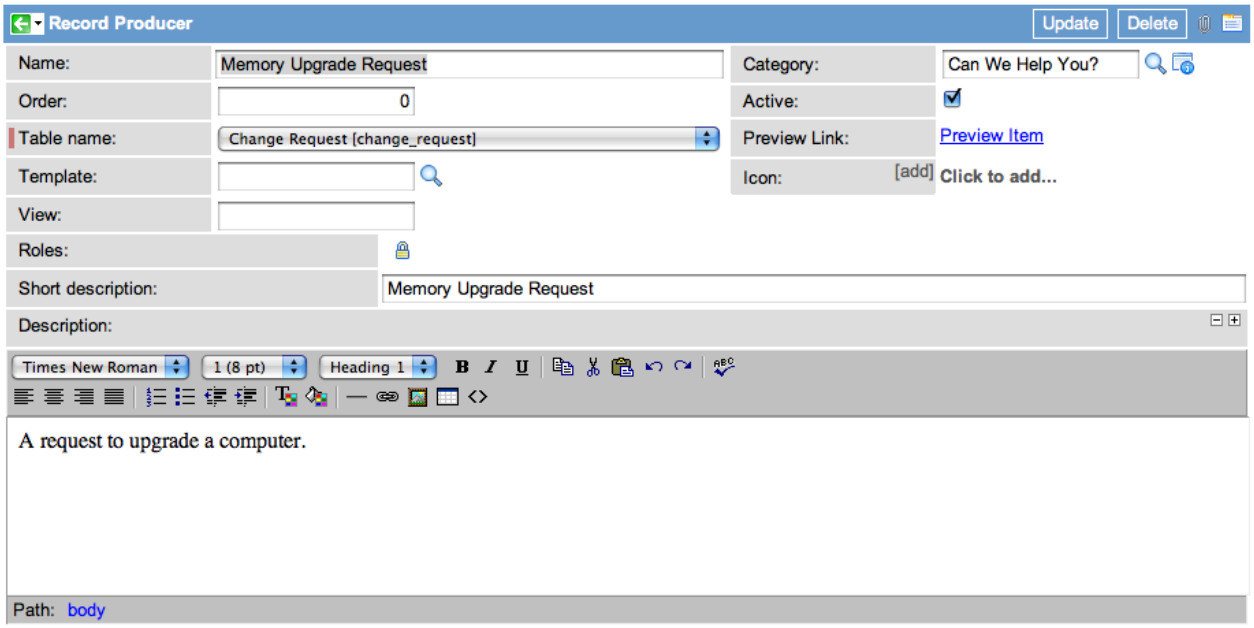
The Service Catalog can be used to request changes using a Record Producer. Record Producers appear in the Service Catalog like Catalog Items, but when they are submitted, they create a record on the specified table with the information provided either by the user, or by a template.

The examples below will demonstrate two methods of creating a Record Producer to request a memory upgrade.

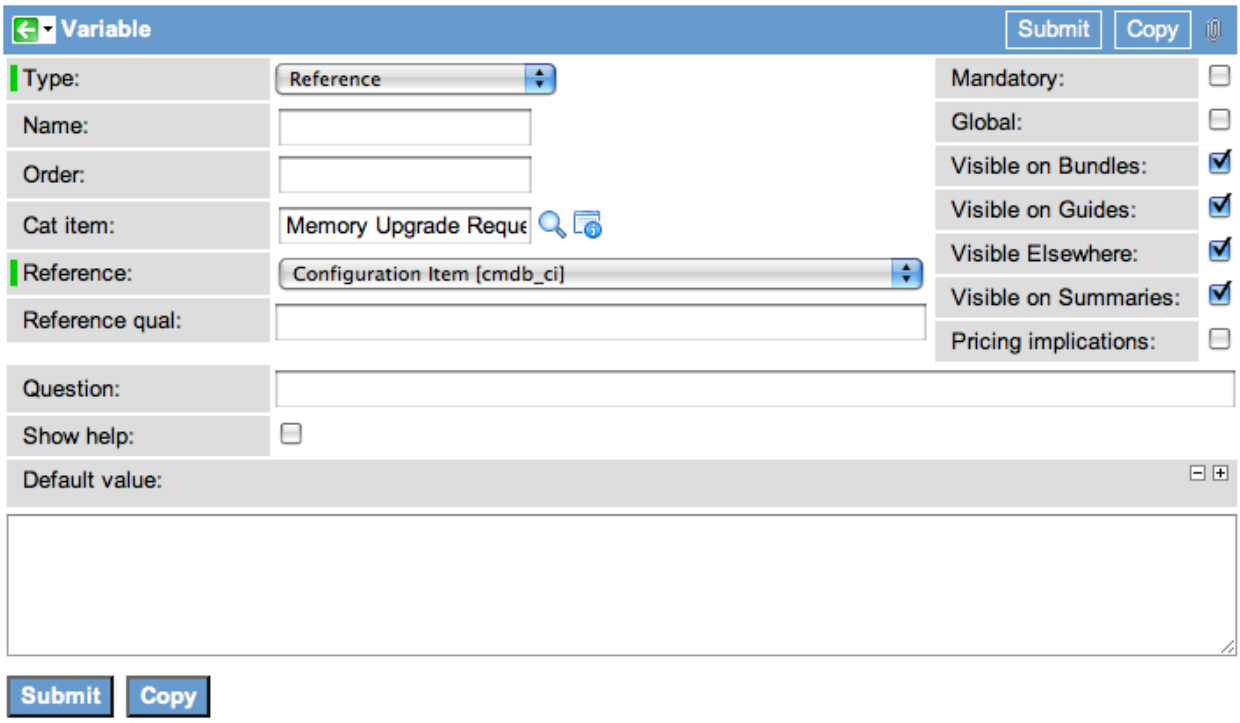
Creating a Record Producer

To create a record producer:

1. Navigate to **Service Catalog > Record Producers**.
2. Click **New**.
3. Populate the form as follows:
   - **Name** - Memory Upgrade Request.
   - **Table Name** - Change Request [change_request].
   - **Category** - Can We Help You?
   - **Roles** - itil, admin. This restricts the record producer to IT employees. Usually, outside users will file an incident rather than directly filing a request for change.

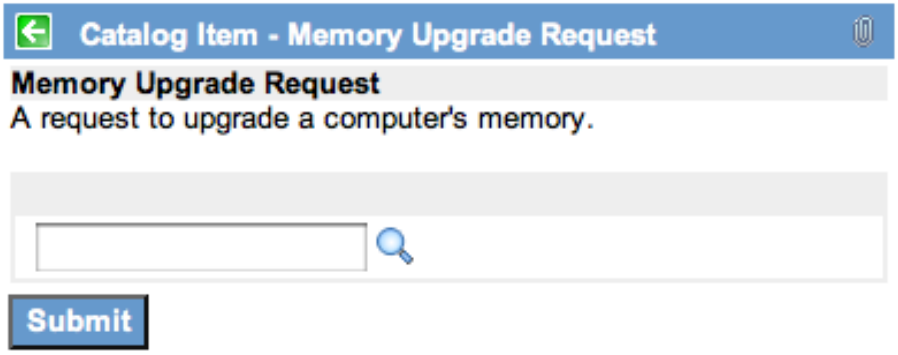


4. Right click the form and select **Save**. The related lists **Variables** and **Variable Sets** will now appear at the end of the form.
5. Scroll down to the **Variables** related list and click **New**.
6. Populate the **New Variable** form as follows:
   - **Type** - Reference.
   - **Name** - configuration_item
   - **Reference** - Configuration Item [cmdb_ci]
   - **Question:** - Which item needs a memory upgrade?



7. Click **Submit**.
8. Click **Save**.

To see how the new record producer appears to the itil users, click the **Preview Item** link:



Defining an Inbound Email Action

Overview

By default, Inbound Email Actions allow users to create or update incidents on from an email. The inbound email action parses the email and uses a script to set certain incident values. Use the following sample inbound email action to create a change request from an email.

1. Navigate to **System Policy > Inbound Actions**.
2. Click **New**.
3. Enter these values on the form:
   - **Name:** Request Change.
   - **Type:** New
   - **Active:** True
   - **Target Table:** Change Request [change_request]
   - **Condition:**

```
email.subject.indexOf("Change Request: ") == 0
```

   - **Script:**Insert the following script:

```
//Note: current.caller_id and current.opened_by are already set to the
first UserID that matches the From: email address

current.comments = email.body_text;
current.short_description = email.subject;

current.notify = 2;

if (email.body.assign != undefined)
  current.assigned_to = email.body.assign;

if (email.body.priority != undefined)
  current.priority = email.body.priority;

if (email.body.category != undefined)
  current.category = email.body.category;

current.insert();
```

Defining an Assignment Rule

	Functionality described here requires the **Admin** role.

Overview

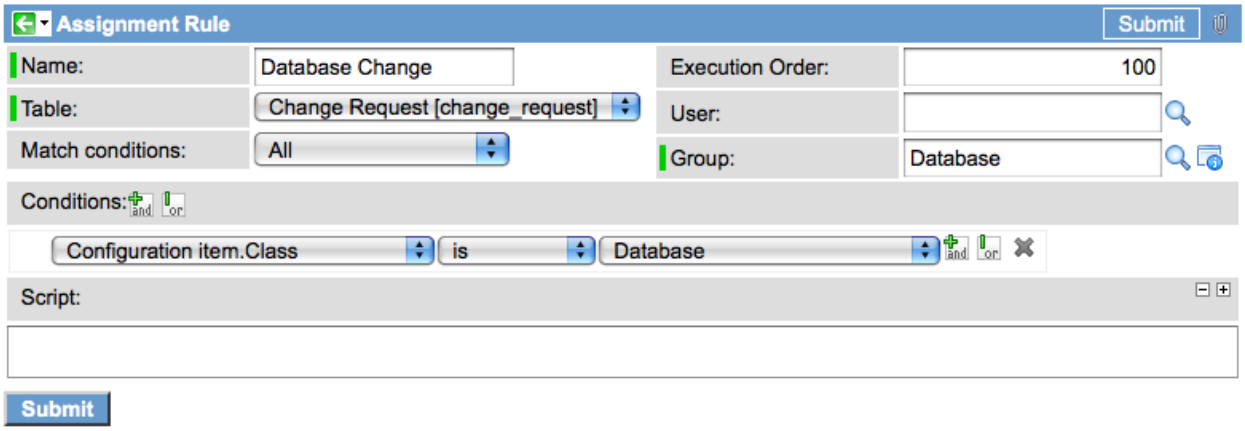
Once a change has been requested, it is important that it be assigned to the appropriate group or individual to handle the problem. Administrators can define Assignment Rules to automate the assignment process.

Note that an assignment rule can be defined either for the Change Request on a whole, or for individual Change Tasks as they get generated from Change Requests.

Defining an Assignment Rule for Changes

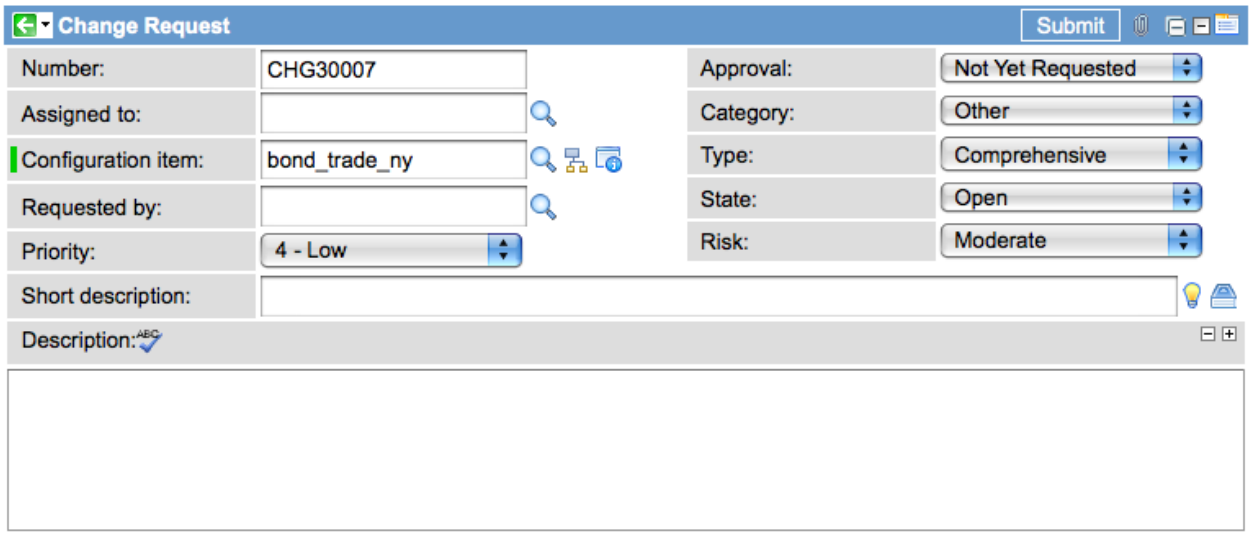
To define an Assignment Rules for Changes:

1. Navigate to **System Policy > Assignment** and click **New**.
2. Populate the field as follows:
   - **Name** - Database Change
   - **Table** - Change Request [change_request]
   - **Group** - Database
   - **Conditions** - Dot-walk to "Configuration Item.Class is Database".

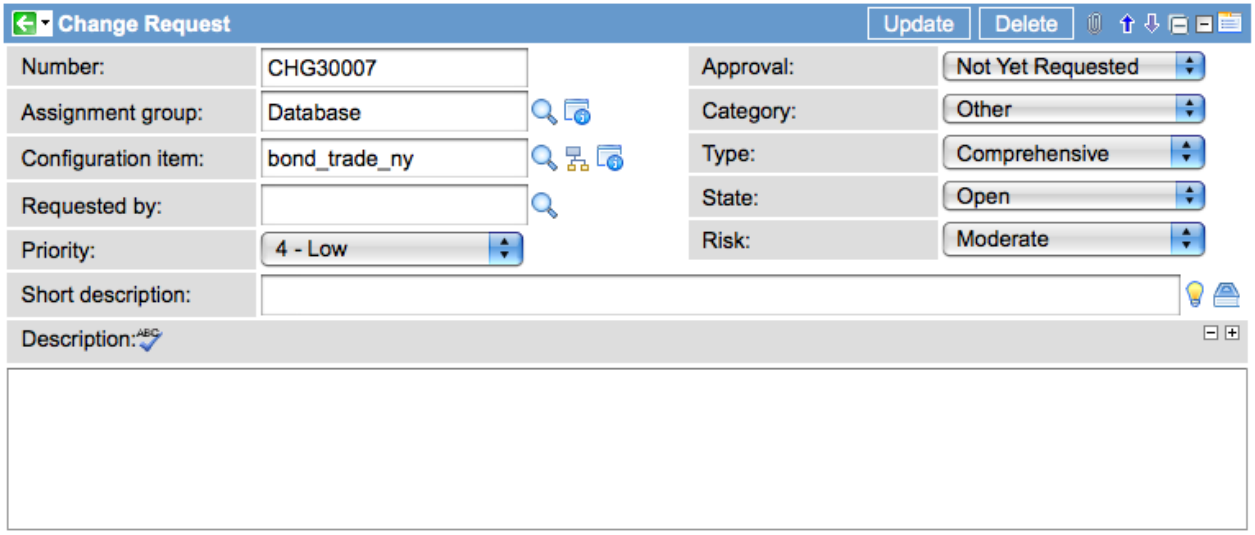


To test the Assignment Rule, navigate to **Change > Create New** and populate the form with the following:

- **Configuration Item** - bond trade ny (or any other Configuration Item with a class of Database.



When the change is saved, add the field **Assignment Group** to the form and the proper assignment group will be added:



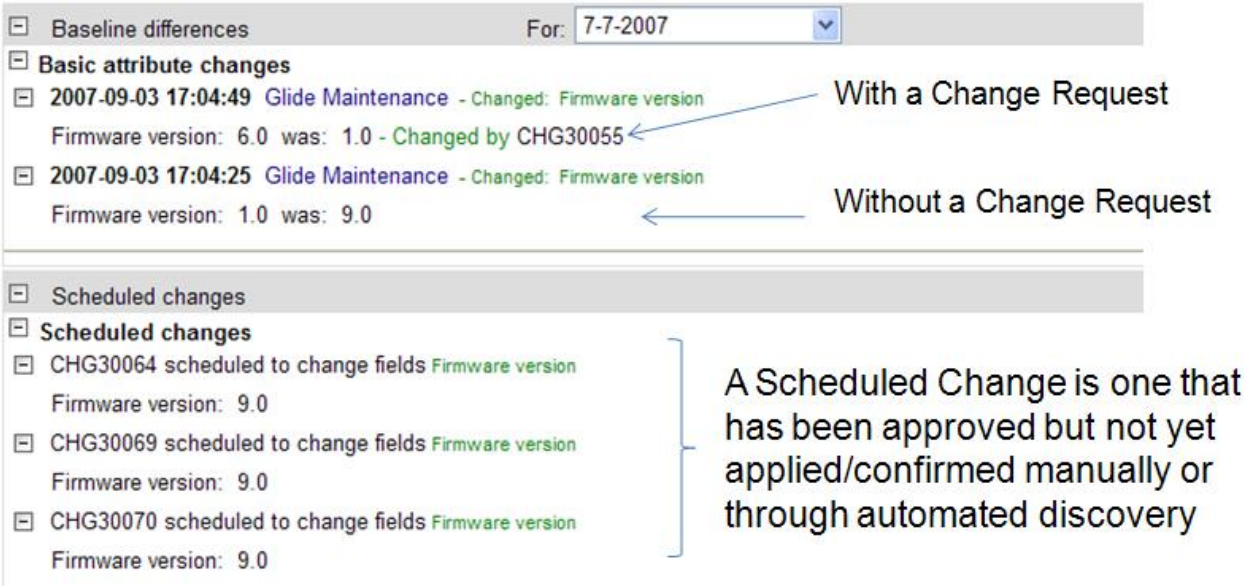
Managing CI Changes

Note: *This article applies to Fuji and earlier releases. For more current information, see Associated CIs on a Change Request* [1] *at* http://docs.servicenow.com **The ServiceNow Wiki is no longer being updated. Visit** http://docs.servicenow.com **for the latest product documentation.**

Overview

A key component of change management is tracking changes over time and ultimately the entire CI life cycle. Identification of planned and unplanned changes is also highly desirable. This is accomplished through the use of CMDB baselines. Baselines can be scheduled to occur every day, week, or month. Changes made through the change process are linked to the change request, and changes without an associated change request are also displayed. The current state of any CI in the CMDB can be compared against any baseline. Not only are basic attribute changes of a CI tracked, but also all related CIs and CI relationships.

For instructions on performing bulk CI changes, see Best Practice - Bulk CI Changes.
For information on proposed changes, see Configuring Proposed Changes to a Configuration Item.



Affected CIs and Impacted Services

You can also manage CI changes with the related lists at the bottom of the Change Request form: **Affected CIs** and **Impacted Services**. Right-click the header bar on the Change Request form and select **Refresh Impacted Services** to populate the **Impacted Services** related list, based on the currently associated CI.

The **Impacted Services** related list represents a many-to-many relationship between the Task [task] and Business Service [cmdb_ci_service] tables, and displays affected business services. You can add this related list to any task form, such as Incident or Problem, and populate it manually if desired. Adding the related list to a form also adds the **Refresh Impacted Services** option to that form's context menu. Use the menu option to auto-populate the list based on the configuration item on the record. The menu option does not remove manually-added services from the related list, but does add business services related to the CI.

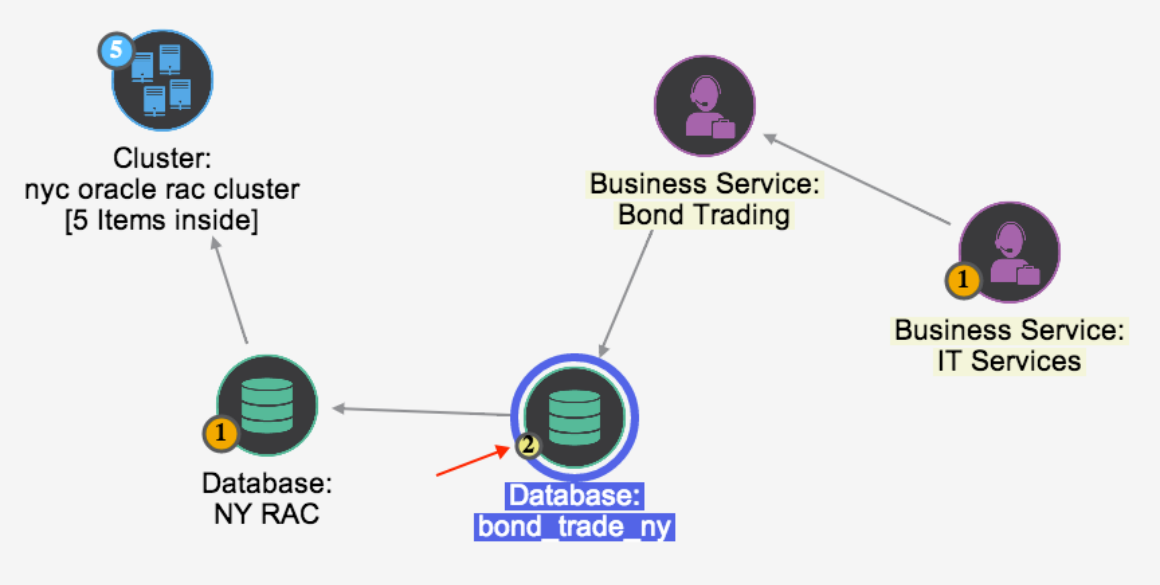
Adding Affected CIs to Change Requests Using the BSM Map

When a change request is associated to a configuration item, the change request record is accessible from the BSM map. This makes the affected services easy to assess. You can use the BSM map to identify dependent CIs affected by the change and then add them to the **Affected CIs** related list on the change request.

Note: *The BSM map shown in this procedure is available starting with the Eureka release. If you are using a version of the ServiceNow platform other than Eureka, see the related documentation in Business Service Management Map.*

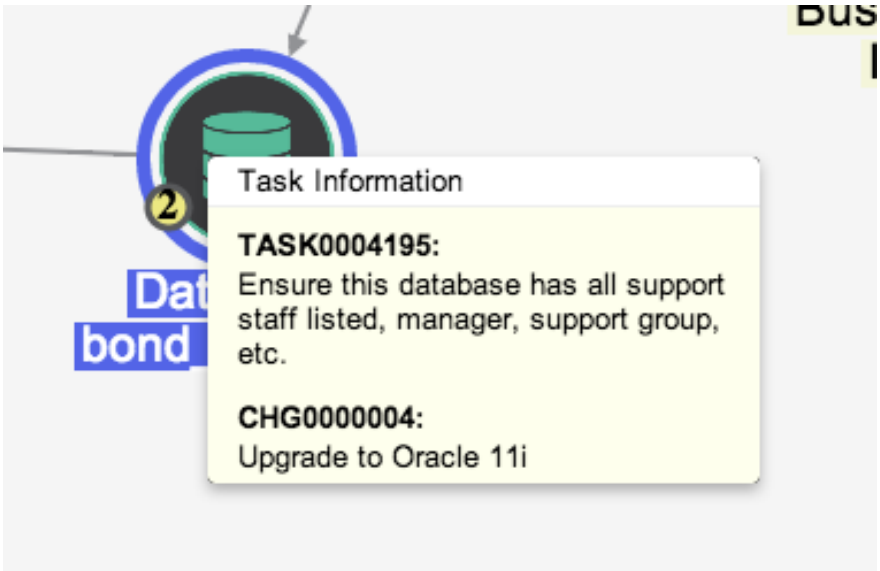
1. In the change request, click the BSM map icon ( ) beside the **Configuration item** field.

   The system displays the configuration item in the map with all its dependent CIs. In this example, there is a critical change attached to the **bond_trade_ny** database. The map includes the business services that rely on the database. The database icon has a blinking glyph on the lower left edge that indicates trouble with the node.



2. Point to the glyph to display a list of tasks and issues with the CI.

   This database has one change and a follow-on task from a compliance audit. You can open each record from this list.



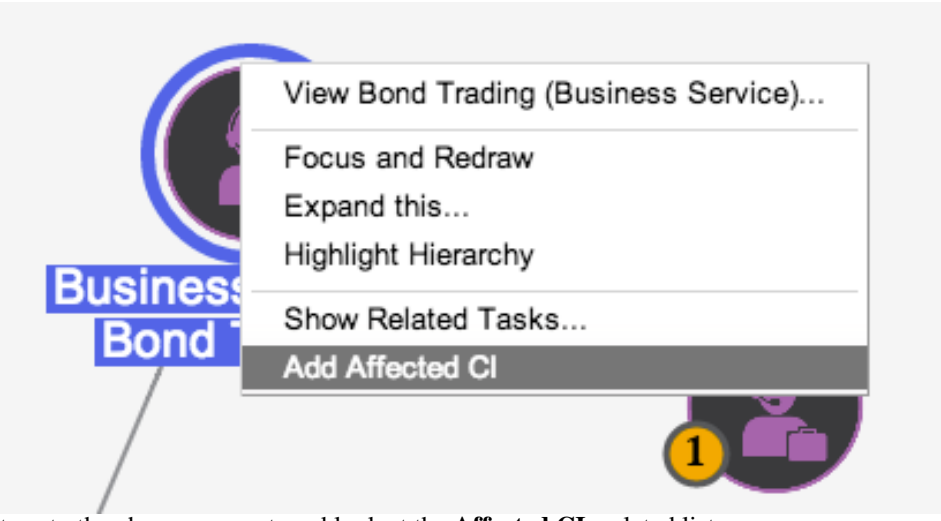
3. Click either task number to display the complete list of tasks attached to this server.

You can see who is assigned to the change and can open the record for more information.

4. To change the map configuration, select a format from the **Layout** field or use the filter panel to filter the map.

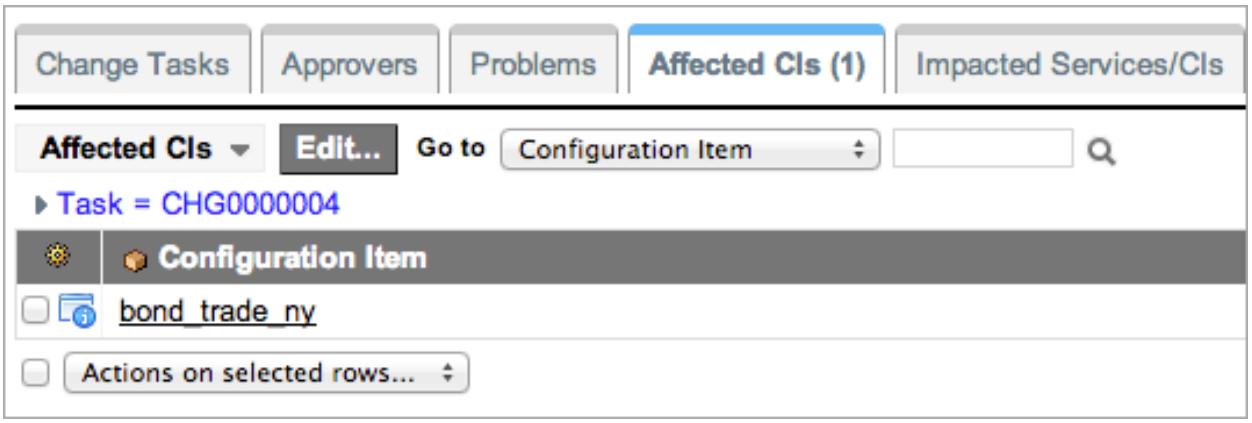
   The BSM map highlights the affected CIs, all of which are dependent on the database.

5. To add an affected CI to the change for the database, right-click a highlighted node and select **Add Affected CI** from the context menu.



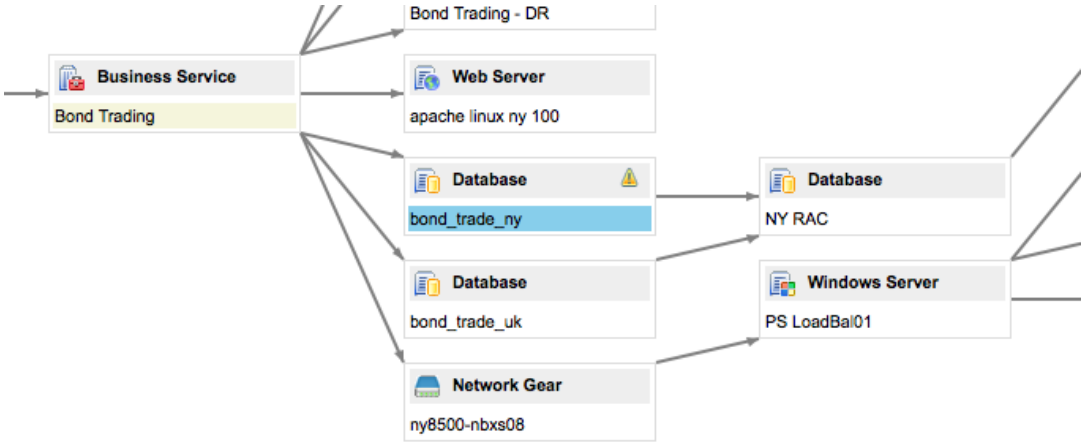
6. Return to the change request, and look at the **Affected CIs** related list.

   If the list is not visible, configure the form to display it.

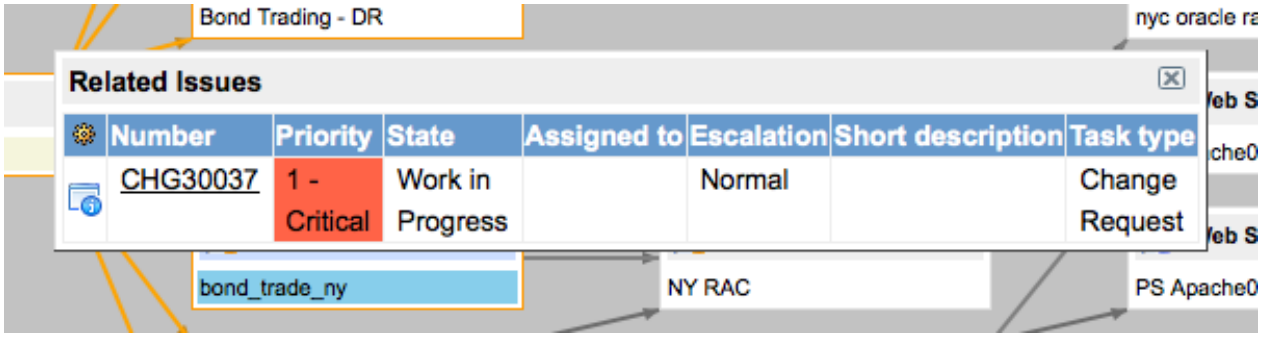


Click the plus to view the procedure for versions prior to Eureka

Once changes are attached to configuration items, they are visible in the business service map. This makes the impacted services easy to assess. The following screenshot shows a critical change attached to the **bond_trade_ny** database.



Clicking on the **Related Issues** icon displays the list of related tasks, including the change.



References

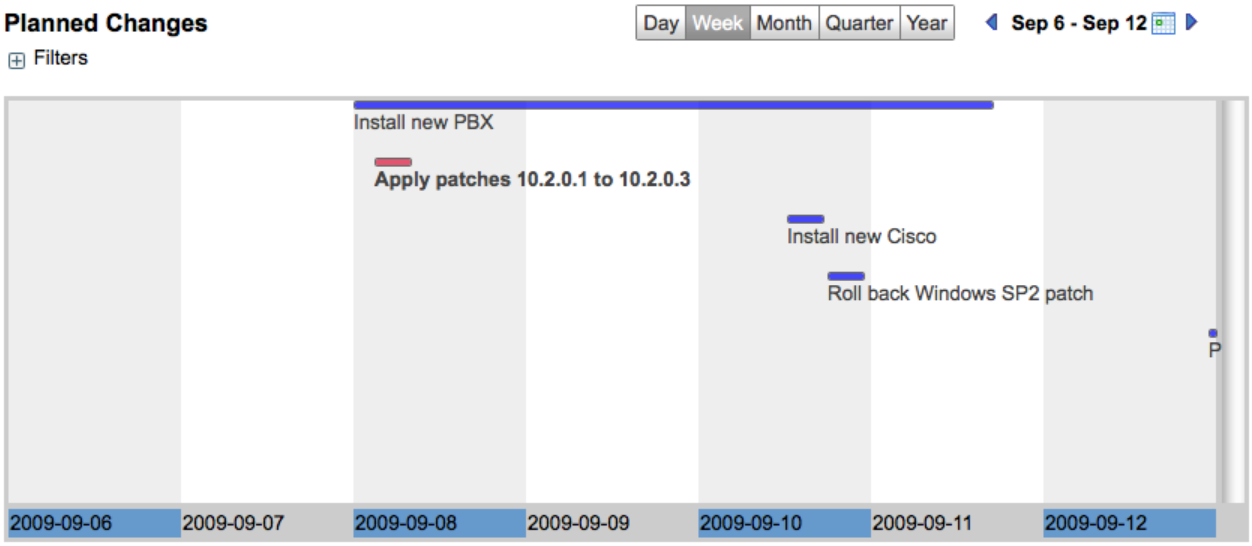
[1] https://docs.servicenow.com/bundle/jakarta-it-service-management/page/product/change-management/concept/c_AffectedCIsAndImpactedServices.html

Defining Maintenance Schedules

Note: *This article applies to Fuji. For more current information, see Create Blackout and Maintenance Schedules* [1] *at* http://docs.servicenow.com The ServiceNow Wiki is no longer being updated. Please refer to http://docs.servicenow.com for the latest product documentation.

Overview

The Maintenance Schedules feature enables you to display scheduled maintenance for configuration items in a timeline format, and marks change requests that fall outside the allowed maintenance period. Planned maintenance is displayed in a calendar format that can be configured to show daily to yearly views. Maintenance activity is represented by a *span*, which is displayed as a horizontal bar on the timeline and can appear in any color. Spans for requested changes that occur outside their allowed maintenance periods appear in red on the timeline.



Change Collision

For instances with the Collision Detector activated, blackout and maintenance schedules have condition fields. This allows blackout or maintenance schedules to be applied only to CIs that match a certain set of conditions.

Creating Maintenance Schedules

A **Weekends** schedule is included in the base system. This schedule runs from 00:00:00 on Saturday until 23:59:59 on Sunday and repeats every week. It may be helpful to refer to this example when setting up your own custom maintenance schedules.

These properties are available to help define schedule entries: `glide.schedules.repeat_nth` and `glide.schedules.fifth`. For details about the properties, see Available System Properties..

To create additional maintenance schedules:

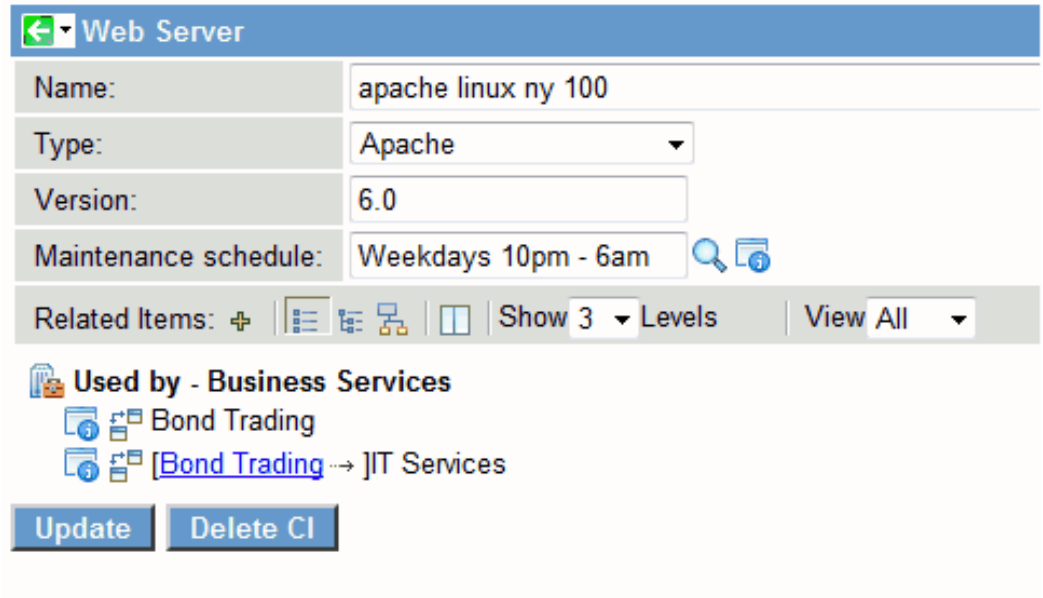
1. Navigate to **Change > Maintenance Windows** and click **New** to create a new schedule.
2. Type a unique name for the schedule.
3. Select the appropriate time zone.

4. Save the schedule form and then click **New** in the Schedule Entries related list to add times when maintenance should take place. For details on how to create schedule entries, see Schedules.
5. Use the **Condition** field to apply the maintenance schedule only to particular CIs, if desired.
6. Repeat this procedure for each different maintenance schedule needed.

When you create a new maintenance schedule from a change record, it is created in the `cmn_schedule` table. When you create a maintenance window, it is created in the `cmn_schedule_maintenance` table, which is the table that is used for checking for schedule conflicts. When selecting a maintenance window in the **CI Maintenance Schedule** field, only schedules that were added using maintenance windows are evaluated for conflicts in change.

Assigning Maintenance Schedules to Configuration Items

1. Locate the configuration item record by navigating to the appropriate module under the Configuration application menu. For example, if you want to locate a web server, look under **Configuration > Web Servers**.
2. If you have not already done so, personalize the form to add the **Maintenance schedule** field.
3. Set the value of the **Maintenance schedule** field to one of the configured maintenance schedules.
   You will be able to choose schedules of the type **maintenance**.
4. Save the CI.



Outside Maintenance Schedule on Change Requests

Personalize the change request form and add the **Outside maintenance schedule** field. This check box is informational and indicates whether planned scheduling (consisting of the planned start date and planned end date) for the change occurs outside the maintenance window. The instance sets this value and disregards any user changes to this checkbox.

This field is set to **true** if:

- The CI associated with the change, if any, is compared to the maintenance schedule, if any, and the planned dates for the change occur outside the maintenance schedule.
- Likewise Affected CIs, if any, for the change request will be checked against their own maintenance schedules, if any.

Note: *Only the maintenance window for the primary CI or affected CIs are checked; the upstream/downstream CIs are not checked.*

When you save a change request that is outside the maintenance schedule, a warning appears for each CI (primary or affected) that is outside the maintenance schedule, if the change request was previously not marked as outside the maintenance schedule.

Change Schedule

Bring up the change schedule timeline by navigating to **Change > Change Schedule**. The timeline shows all active change requests by planned start and end times. Any change requests that are outside the maintenance window are colored red on the display. Change requests that have an approval status of **Rejected** are displayed in gray.

For instructions on using timelines, see Using Timelines.

Updating Maintenance Schedules

When you update maintenance schedule entries, check the **Outside maintenance schedule** field for change requests that might have been forced outside the maintenance period and that require an update. An *async* business rule runs on the Schedule Entry table when maintenance schedules are updated to verify the **Outside maintenance schedule** field on change requests that have configuration items (primary or affected) that use that schedule. As this is done asynchronously (so as not to keep the user waiting), specific warning messages are not issued for each change request as they are when change requests are individually updated. However, if any CIs use the maintenance schedule, a single message is issued indicating that this verification will occur.

Transferring Maintenance Schedules between Instances

You must manually export maintenance schedule records because the system does not track them in update sets. See Exporting and Importing XML Files.

To transfer maintenance schedules between instances:

1. Export records from the following tables as XML files.
   - Maintenance Schedule [cmn_schedule_maintenance]
   - Schedule Entry [cmn_schedule_span].
2. Import the records on the target instance.

References

[1] https://docs.servicenow.com/bundle/jakarta-it-service-management/page/product/change-management/task/t_CreateBlkoutMaintSched.html

Assessing Changes

Calculating Risk

Note: *This article applies to Fuji. For more current information, see Risk Calculator Property* [1] *at* http://docs.servicenow.com
The Wiki page is no longer being updated. Please refer to http://docs.servicenow.com for the latest product documentation.

Overview

The Best Practice - Change Risk Calculator plugin enables dynamic calculations of the risk and impact of a change, which can be populated on the change record. The plugin also bundles some best practice risk calculations using CI attributes and time measures.

Modules

Activating the plugin adds:

- The **Risk Conditions** module in the **Change** application, to define risk and impact conditions.
- A new property to the **System Properties > Change Management** module, to specify when the conditions should apply.

Defining Risk and Impact Conditions

To write a new risk or impact condition, navigate to **Change > Risk Conditions** and click **New**.

There are three options on how to write the new rule:

- Using the **Condition** filter builder to construct the logic.
- Checking **Use advanced condition** and using the **Condition** field to script an advanced condition.
- Checking the **Use script values** option to script conditions.

In all cases a rule will only be evaluated if the 'Active' field is checked.

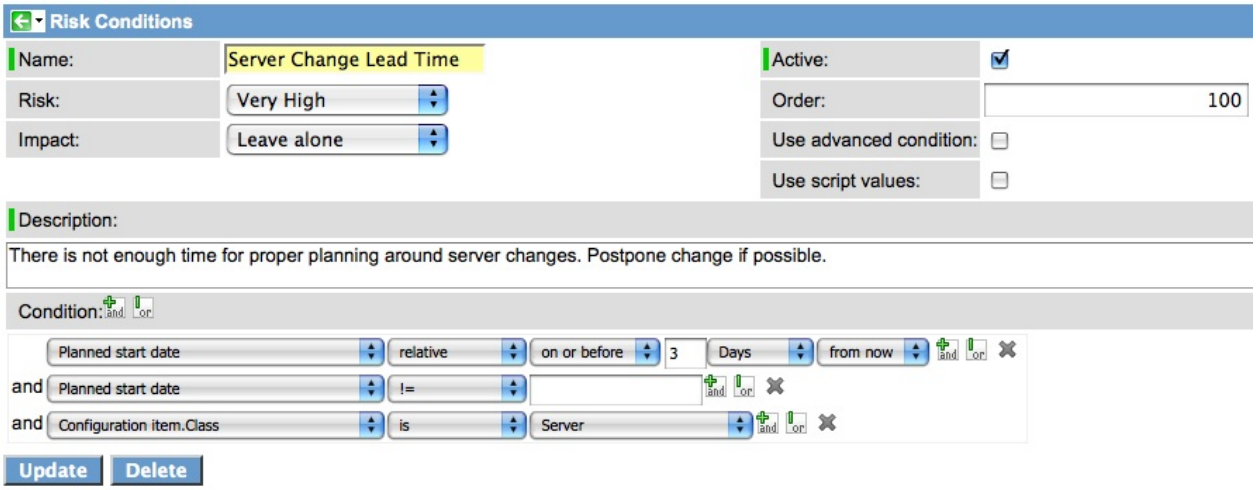
If more than one rule matches the criteria specified on a change request, the **Order** field determines the order in which rules are evaluated. Rules with a lower numbers are evaluated first, and if there are two or more rules with differing orders the rule with the lowest order is evaluated and the others are ignored.

Using the Condition Filter

The simplest way to write conditions is to use the condition filter builder field.

To define risk or impact using the condition filter:

1. Click **New** to create a new calculation.
2. Enter a name.
3. Select the **Risk** and/or **Impact** value. This determines which field is updated by this risk calculation.
4. Enter a description.
5. Create a condition using the Condition field filters.



The sample risk calculation in the screenshot evaluates lead time for change requests associated with servers.

Notice that in addition to using fields located on the change request form, the risk calculation also checks attributes from related records. This can be accomplished by dot-walking.

Using Advanced Conditions

To use advanced conditions to calculate risk or impact:

1. Click **New** to create a new calculation.
2. Enter a name.
3. Select the **Risk** and/or **Impact** value. This determines which field is updated by this risk calculation.
4. Enter a description.
5. Check the **Use advanced option** box.
6. Write the script in the **Advanced condition** field that appears.

   Rules are written using standard business rule syntax. The rule needs to set the global variable **answer** to true or false. See the screenshot below for an example:

Risk Conditions

Name:	Critical service changed	Active:	☑
Risk:	High	Order:	200
Impact:	1 - High	Use advanced condition:	☑
		Use script values:	☐

Description:

Change is for a highly critical or somewhat critical business service

Advanced condition:

```
answer = false;
if (current.cmdb_ci.sys_class_name == 'cmdb_ci_service') {
   var serv = new GlideRecord('cmdb_ci_service');
   serv.get(current.cmdb_ci);
   if (serv.busines_criticality == "1 - most critical" || serv.busines_criticality == "2 - somewhat
critical")
      answer = true;
}
```

Update Delete

The example in the screenshot above looks at the currently selected configuration item to determine whether it is a business service. If it is a Business Service then it checks a field named **Business Criticality** to see if the value is **1 - most critical** or **2 - somewhat critical**. If the condition matches, it sets 'answer = true', which will set the risk for the change request to 'High' and the impact to '1 - High'.

Another common scenario that requires scripting is determining the Business Services that will be impacted as a result of a change to one or more configuration items. A sample rule has been provided in the plugin, seen in the screenshot below:

Risk Conditions

Name:	Critical service affected	Active:	☐
Risk:	High	Order:	300
Impact:	1 - High	Use advanced condition:	☑
		Use script values:	☐

Description:

Change affects a highly critical business service

Advanced condition:

```
var ciu = new CIUtils();
var services = ciu.servicesAffectedByTask(current);

var critServices = new GlideRecord("cmdb_ci_service");
critServices.addQuery("sys_id",services);
critServices.addQuery("busines_criticality","1 - most critical");
critServices.query();
answer = critServices.hasNext(); // return true if critical service affected
```

Update Delete

In this example rule, the script uses the `CIUtils()` class to determine which Business Services will be impacted by your change. The `servicesAffectedByCI()` method is invoked, and passed the **current** change record. This method grabs the Configuration ITem entered on the current change request then locates all associated parent and child Business Services.

A list or array of Business Services is returned, and then evaluated in the script above to determine if there are any '1 - most critical' services. If there are highly critical services then the answer will be set to true.

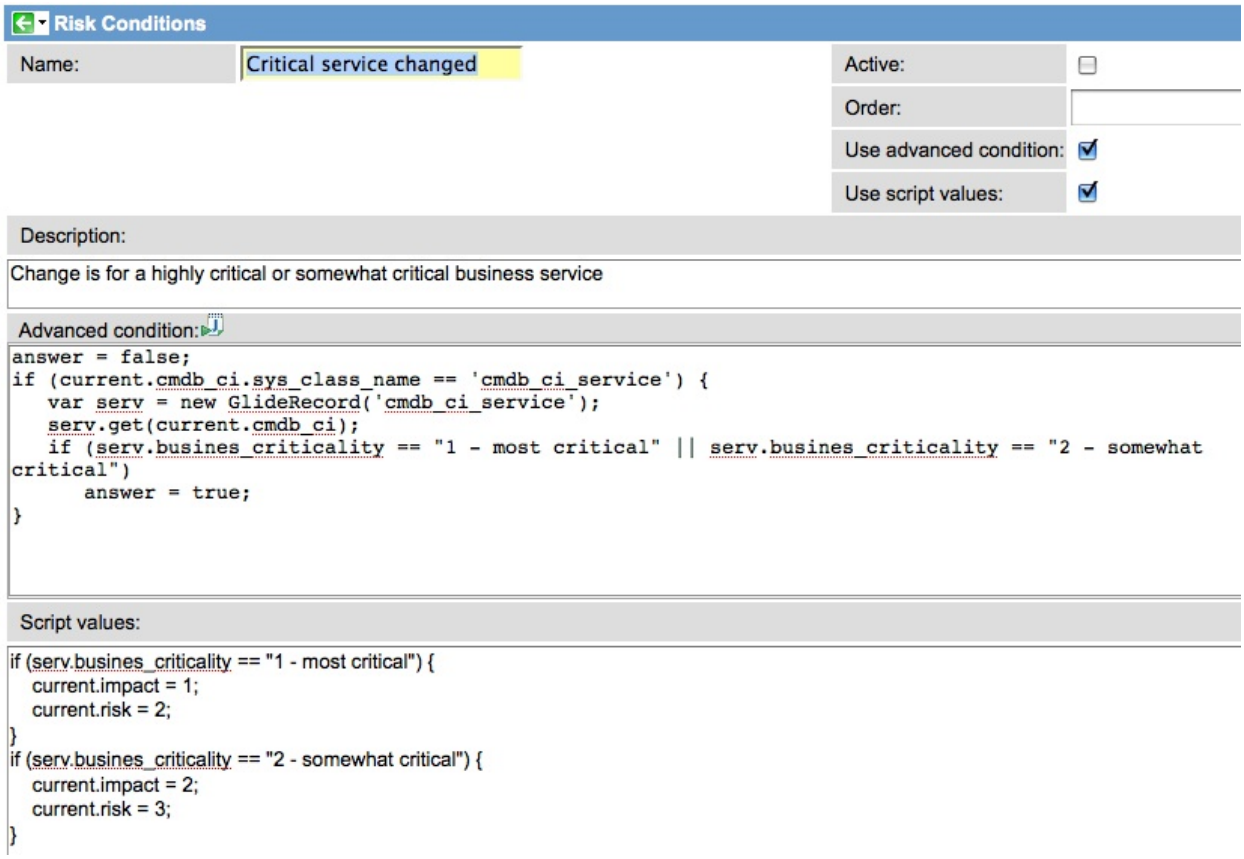
Using a Script

The **Use script values** option is used to set risk and/or impact based on variable conditions.

To use a script to calculate risk or impact:

1. Click **New** to create a new calculation.
2. Enter a name.
3. Select the **Risk** and/or **Impact** value. This determines which field is updated by this risk calculation.
4. Enter a description.
5. Check the **Use script values** box.
6. Write the script in the **Script** field that appears.

The **Critical service changed** condition, which is provided with the plugin, sets risk and impact according to the values returned by the Business Services. If the criticality is **1**, the script values are used to assign the appropriate risk and impact:

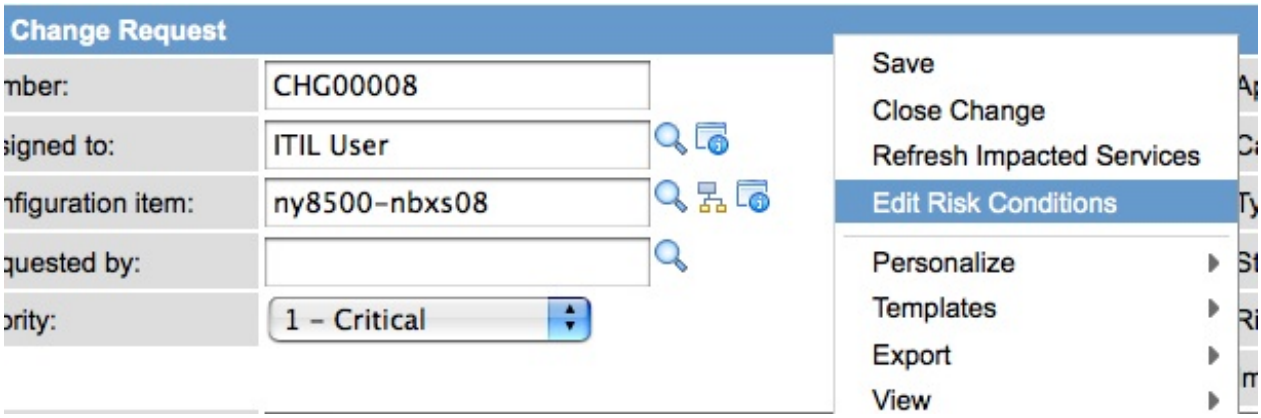


Values from the current change request can be invoked to optionally set risk and/or impact. Below is an example using current field values:

```
if (current.assignment_group.getDisplayValue == "Network") {
current.risk = 2;
current.impact = 1;
} else {
current.risk = 3;
current.impact = 2;
}
```

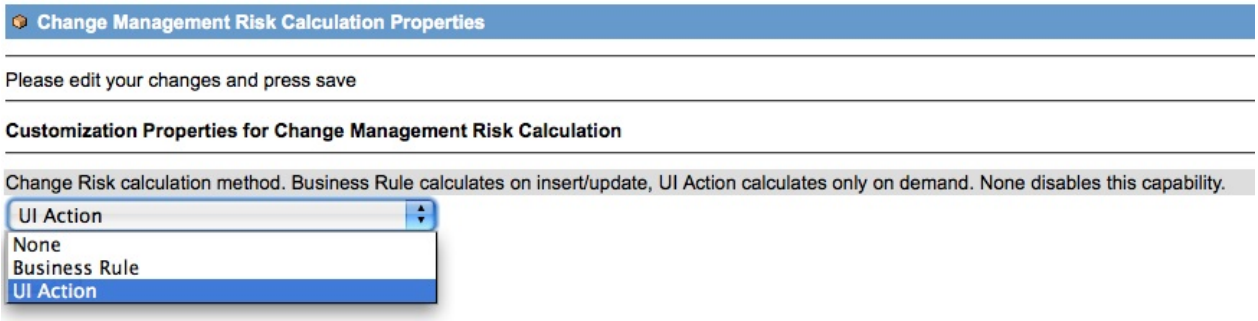
Editing Conditions from Form

In addition to navigating to the form, conditions can be edited from the change request form by right-clicking the header bar and selecting **Edit Risk Conditions** from the context menu. The **Edit Risk Conditions** option is available to users with the **itil_admin** role.



Specifying When Conditions Should Apply

The properties module (**System Properties > Change Management**) specifies when and how risk and impact rules are applied. The current options are 'None', 'UI Action' and 'Business Rule'.



None Option

Disables rules processing entirely.

UI Action Option

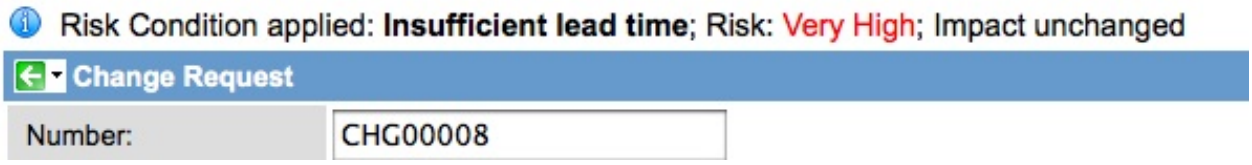
Allows users to check condition rules on demand using the **Calculate Risk** UI Action.

When the option is selected, the **Calculate Risk** UI Action will appear as a **Related Link** on the Change Form if:

- There are any conditions that apply to the current record.
- The users has the ***admin*** or ***itil*** role.

Clicking the 'Calculate Risk' button will apply any matching condition, according to order.

When a rule is applied, an alert will be displayed along the top bar that says condition was applied and the new values for risk and impact. Also, note that if hovering over the name of the applied condition displays the description for the condition.

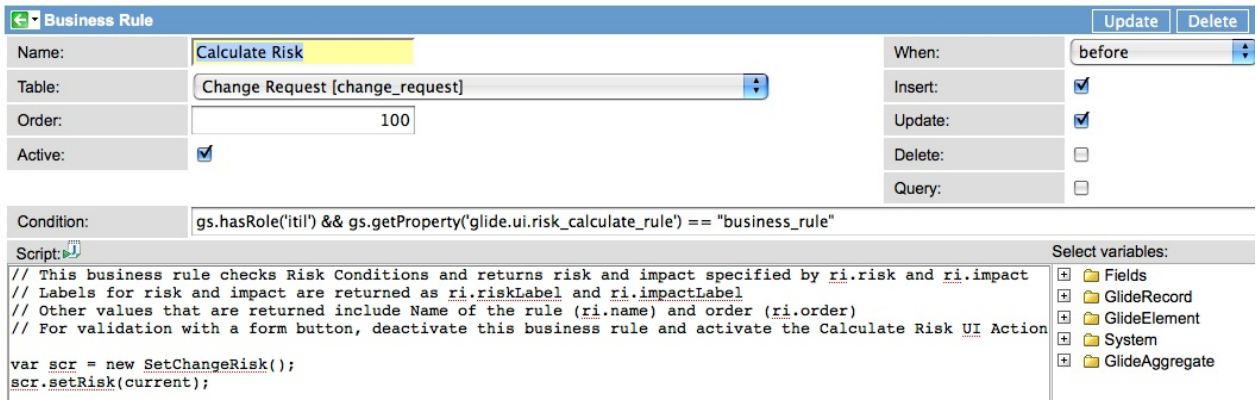


Note: *If the Change Risk Assessment plugin is activated, the* ***Calculate Risk*** *UI action is replaced by the* ***Execute Risk Calculation*** *UI action.*

Business Rule Option

Conditions will be evaluated and applied dynamically via a business rule that exists on the change request table.

The evaluation will occur before inserting a new change request record and before any update to an existing change request. Like the UI Action, this rule will only run for users with the ***admin*** and/or ***itil*** roles.



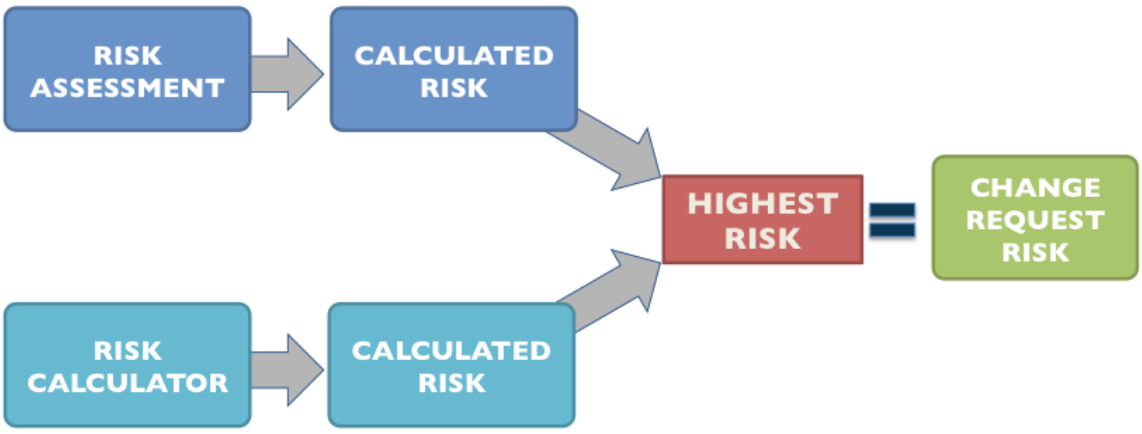
Note: *If the Change Risk Assessment plugin is activated, the* ***Calculate Risk*** *business rule is replaced by the* ***Run Risk Calculation*** *business rule.*

Using Risk Assessment and Risk Calculation

There are two methods within ServiceNow to calculate the risk of a change: Change Risk Calculator (activated by default) and Change Management Risk Assessment (an optional plugin).

- Change Management Risk Assessment uses information provided by the end user to assess a risk value.
- Change Risk Calculation uses predefined properties and conditions to calculate a risk value.

These methods can be used individually, or together, depending on your own procedures and requirements. If both methods are used together, the highest risk value from both methods is always selected:



Note: *If you have both Risk Assessment and Risk Calculation active, but only wish to use one of these methods, then simply remove all conditions for the method you do not want to use.*

References

[1] https://docs.servicenow.com/bundle/jakarta-it-service-management/page/script/server-scripting/reference/r_ChangeRiskCalculator.html

Performing Changes

Creating an Execution Plan

Overview

A common problem within change management is the generation of a sequential list of tasks to implement a specific change type. For example, in order to upgrade a production server, I might need to:

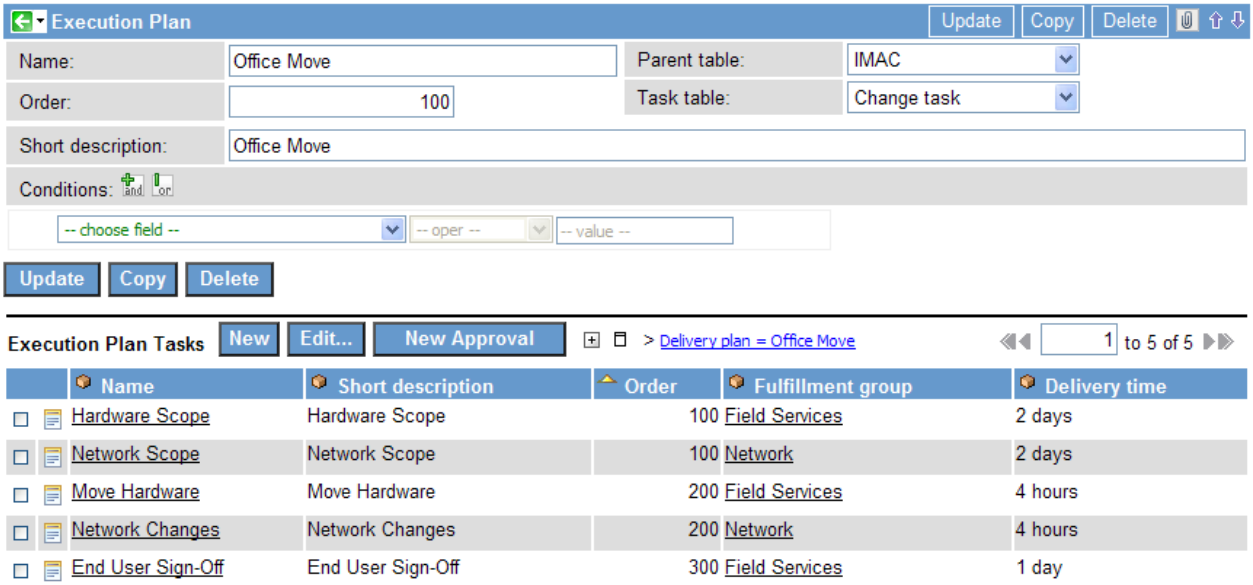
1. Application Owner Approval
2. Upgrade Test Server
3. Verify Same
4. Application Owner Verification
5. Schedule Production Outage
6. Implement Production Change
7. Verify Same

The Change Management Execution plan uses similar functionality as the Service Catalog. This functionality can be used for other tables, provided that the Attach Delivery Plan and Start Change Task business rules (which make the Execution Plan work) are re-created for the other tables.

For more complicated processes, use the Workflow Editor.

Creating an Execution Plan

From the left navigation pane, **System Policy > Execution Plan**, and click the **New** button to bring up the plan details form.



The key fields to fill in are the following:

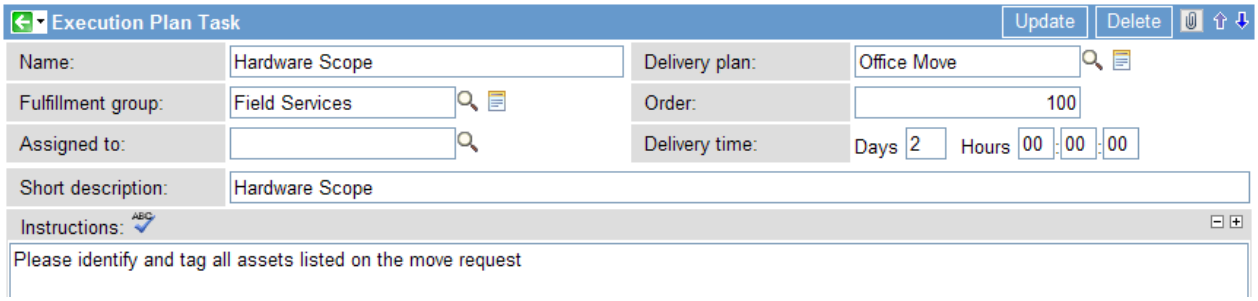
- Name -- what this plan will be called
- Parent Table -- what is the parent task for this kind of plan? For a change management plan, this is almost always going to be **Change Request** meaning that this plan starts from a Change Request.

- Task Table -- what kind of tasks should this plan create and sequence? This can be any kind of task table (incident, problem, etc), but for a change management plan, you will likely want to produce **Change Task** records.
- Order -- The order in which this plan is checked (see below)
- Condition -- Condition under which this plan is applied to a change (see below).

Creating the plan's set of tasks

Each execution plan has one or more tasks associated with it. You'll see the list of tasks at the bottom of your plan in the Execution Plan Tasks related list. When you first create your plan this list is going to be blank since we don't have any tasks defined yet.

To create a new task, we need to click the new button on the related list which will bring up a task definition. Please note that we're not actually creating a change management (or any other kind) of task at this point. We're defining a step in the execution process, rather than trying to implement it directly.



An execution plan task is defined by the following fields.

- Name -- Name of this task
- Fulfillment Group -- Group which is supposed to work this task. If filled in, will pre-populate the assignment group for this task.
- Assigned To -- User to which this task is assigned. If filled in, will pre-populate the assigned to field for this task.
- Short Description -- A short description for this task. If filled in, will pre-populate the short description field for this task.
- Instructions -- Any instructions on how to implement this task. If filled in, will pre-populate the description field for this task.
- Work Notes -- Any non customer facing instructions on how to implement this task. If filled in, will pre-populate the work notes field for this task.
- Order -- Order in which this task is to execute within this plan. Tasks execute sequentially, from lowest order number to highest order number, and tasks with the same order number execute in parallel.

Attaching a Plan to a Change

By default, if you open a change, there won't be an execution plan associated with it. Associating a particular subset of your change requests with a particular execution plan requires that we set up match conditions on our execution plans. Before we do that though, it's important to understand how the system will try to associate an execution plan with your change request.

Whenever you save a change request that **does not yet have an execution plan**, the system will attempt to select an execution plan for you via the following process:

1. Look for all execution plans with a parent task of change request.
2. Sort them based on their order value (low numbers first).
3. Test each plan's conditions against the current change request.
4. If the condition matches associate the plan with the current change request and stop checking

5. Otherwise, check the next plan

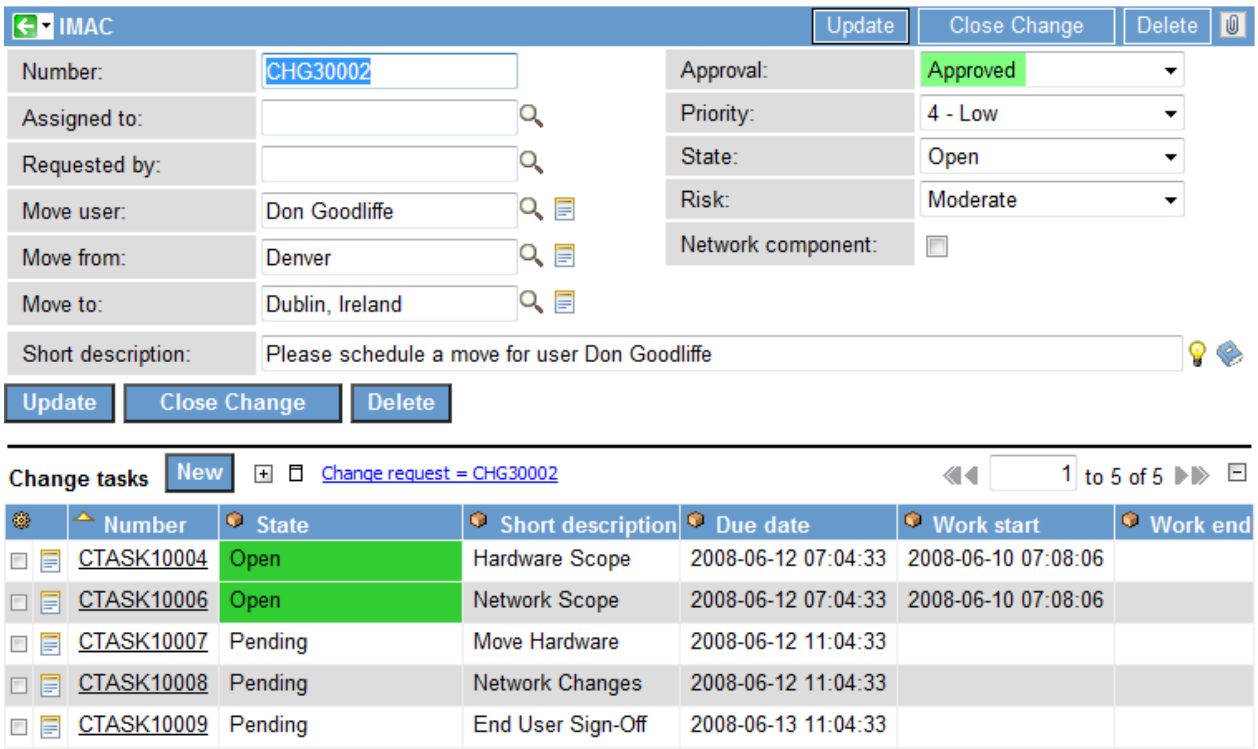
Note that it's entirely possible to check all the available execution plans, find that none match the current change request, and have a change request with no associated plan.

Automatic Task Generation

When the system determines that a change request has an execution plan, the system generates tasks automatically. This determination can occur either on the initial insert of the change request or from a subsequent update to the change request which causes it to match a plan condition. All automatic tasks are generated with a state of **Pending**, meaning that no one should begin work on them while they are in the pipeline.

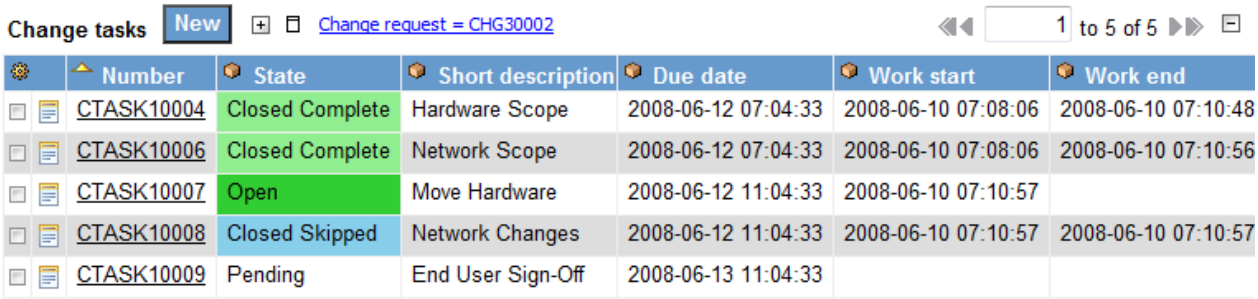
Task Initiation

As soon as the Change Request is approved (defined as the approval field changing to approved), the first set of tasks (the one(s) with the lowest order number) are automatically changed to a state of "Open" and assigned a work start time.



Task Sequencing

As tasks within the execution plan are closed, the next task(s) in sequence will automatically be started for you.



In our example (above), the first two tasks were complete, the fourth was automatically skipped (since no network component was required in this change), and the third was started automatically.

Conditional Tasks

A given step in an execution plan might not always be necessary. For example, in our office move, above, the Network Changes step is only necessary if, in the previous Network Scope step, we identified that some form of network change was required to support the move.

An execution handles cases like this via conditional tasks. Simply put, a conditional task is a task which, while always present in the execution plan, only actually starts if certain conditions are met. By default, these condition fields aren't visible on the execution plan task form so you need to configure the form and add: condition and condition script.

Once you do, your form will have two new fields on it, allowing you to express when this task should run. You can do this most simply by using the condition field to fill in a conditional expression under which this task should run.

Alternately, if your condition is too complex to express via the condition widget, you can fill in the condition script box with custom javascript that can return true, or false, depending on whether or not this tasks should execution. Within the context of that script "current" is going to be the actual change task in question.

Pre-populating Tasks

The system, by default, will copy data from a half dozen or so fields in your execution plan task into the actual generated task. You can, however, use script to populate additional data into the generated task if you so desire.

To do so, configure the form and add the "Generation Script" to your form. Then any script you type into that box will be executed as a task is being generated based on this delivery plan task. Within the context of this script, current will be the (not yet inserted) new change task. Note that you do **not** need to call current.insert() in your script as the system will automatically take care of that for you.

Accessing Variable Pool in Generation Script

In the generation script, "current" does not yet have a variable pool, but you have access to the variable pool on its parent (the sc_req_item record). You can access that variable pool in the following way (this example simply logs the values in the variable pool, but you can use it as needed):

```
gs.log('Table: ' + current.getTableName()); gs.log('Parent: ' + current.parent.sys_id);

var item = new GlideRecord('sc_req_item'); item.addQuery('sys_id', current.parent.sys_id); item.query(); if(item.next()){

for(var variable in item.variable_pool) { gs.log(variable); var answer = eval ("item.variable_pool." + variable + ".getDisplayValue()");

 gs.log(answer);}

}
```

Creating a Template for Change Tasks

Note: *This article applies to Fuji. For more current information, see Change Management* [1] *at* http://docs.servicenow.com The ServiceNow Wiki is no longer being updated. Please refer to http://docs.servicenow.com for the latest product documentation.

Overview

Administrators can create a template that can be used to create change requests with pre-defined supporting tasks.

The process is as follows:

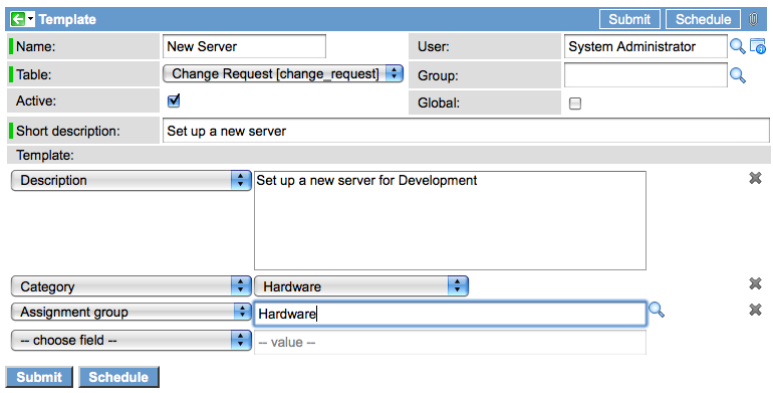
1. Create the master template and child templates.
2. Link the templates.
3. Use the template in a module.

Note: *Child templates are only applied if the parent template is applied from a module. Child templates are not applied by simply applying a template to a new form.*

Creating the Master Template and Child Templates

Create the master template.

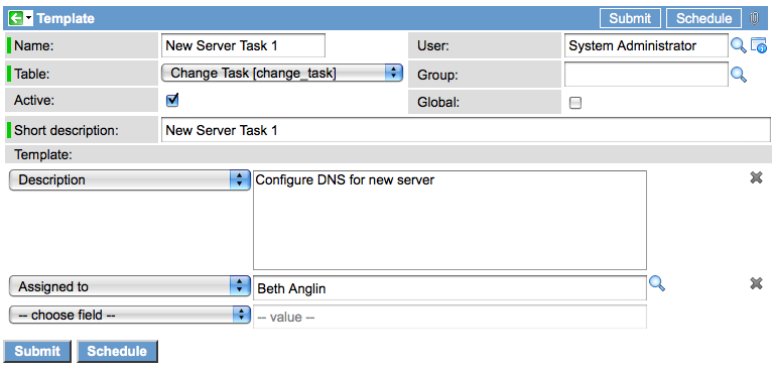
1. Navigate to **System Definition > Templates**.
2. Click **New**.
3. Give the template a name and set the table in this case [change_request].
4. Add fields to be set automatically in the change request.
5. Save the record by right-clicking the header bar and selecting **Save**.



The New Server template.

Next, create more than one child tasks for this change request. Each task will be its own new template.

1. Navigate to **System Definition > Templates**, and then click **New**.
2. Give the template a name and set the table to [change_task].
3. Add the necessary fields that you want the template to auto-populate for new change requests.
4. Save the template after you add each field by right-clicking the form header selecting **Save**.



The template for child tasks.

5. Repeat these steps to create as many change task templates as needed.

Linking the Templates

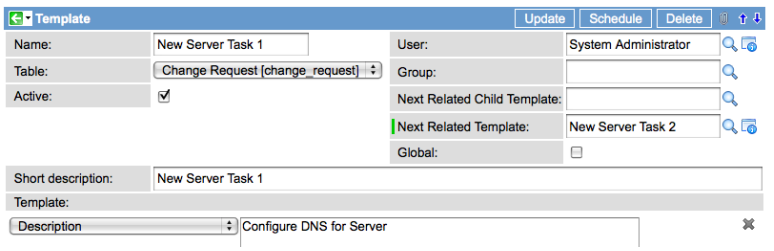
Open the first template, which is the master for the change request.

1. Configure the form and add the **Next Related Child Template** and **Next Related Template** fields.
2. Link the first template to the child template using the **Next Related Child Template** field. In this reference field, enter the name of the first change task template that you created, which in this example is named New Server Task 1.
3. Click **Update**.

The New Server template with the **Next Related Child Template** field.

Open this first child task template, which is called New Server Task 1 in this example.

1. Click the lookup icon next to the **Next Related Template** field and select the next child template. This field links the current template to the next child template.
2. Click **Update**.



The child task template with the **Next Related Template** field.

3. Repeats these steps until all templates are linked together.

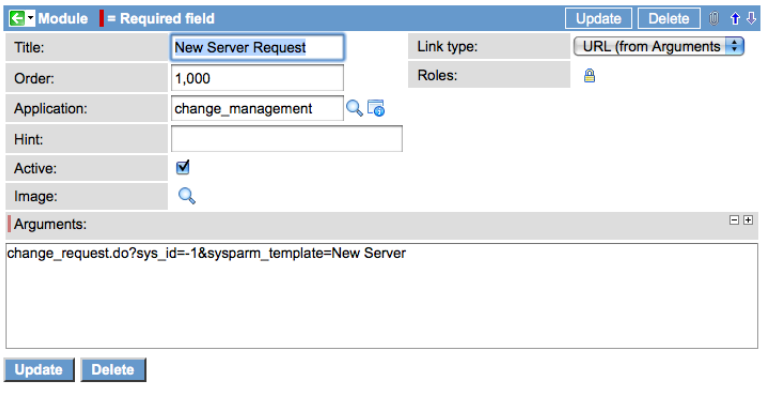
Using the Template in a Module

Add a new module to the Change Management application to create a new change request that uses the templates.

1. Right-click the **Change** application and select **Edit Application Menu**.
2. Click **New** in the **Modules** related list.

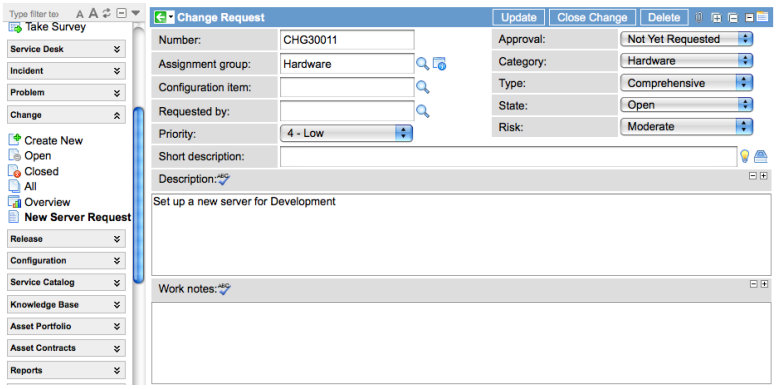
   A UI policy for New Record type modules removes the **Arguments** field from the form. To use the templates, it is necessary to construct a URL that references the templates.
3. Give the module a name.
4. Select the [change_request] table.
5. Select **URL (from arguments)** for the link type.
6. Enter the following in the **Arguments** field, but replace the link with the name of the appropriate master template.

```
change_request.do?sys_id=-1&sysparm_template=New Server
```



A module that uses the template to create a new change request.

Using the module will create a new record, populated with the template as shown in this screenshot:



A new Change Request record using the template.

References

[1] https://docs.servicenow.com/bundle/jakarta-it-service-management/page/product/change-management/concept/c_ITILChangeManagement.html

Change Management Workflows Plugin

Note: *This article applies to Fuji. For more current information, see Getting Started with Workflows* [1] *at* http://docs.servicenow.com The Wiki page is no longer being updated. Please refer to http://docs.servicenow.com for the latest product documentation.

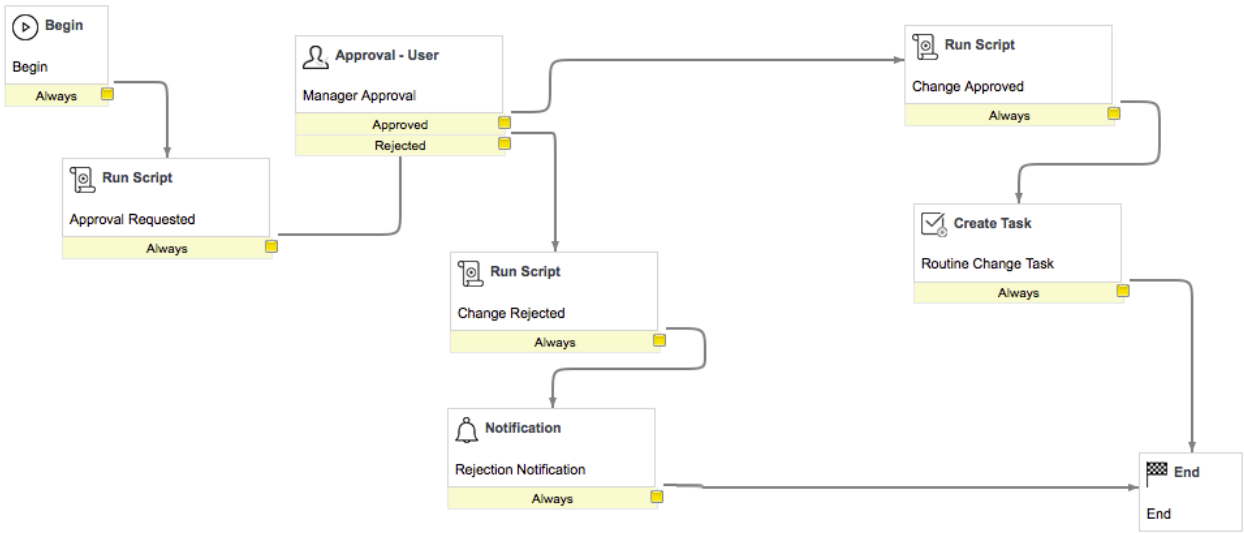
Overview

Install the Change Management Workflows plugin to add three new template workflow versions which are designed for use with the Change Management process. These workflows behave exactly like any workflow: they can be associated with any task, they can be edited using the graphical workflow editor, or edited using the Workflow Versions form.

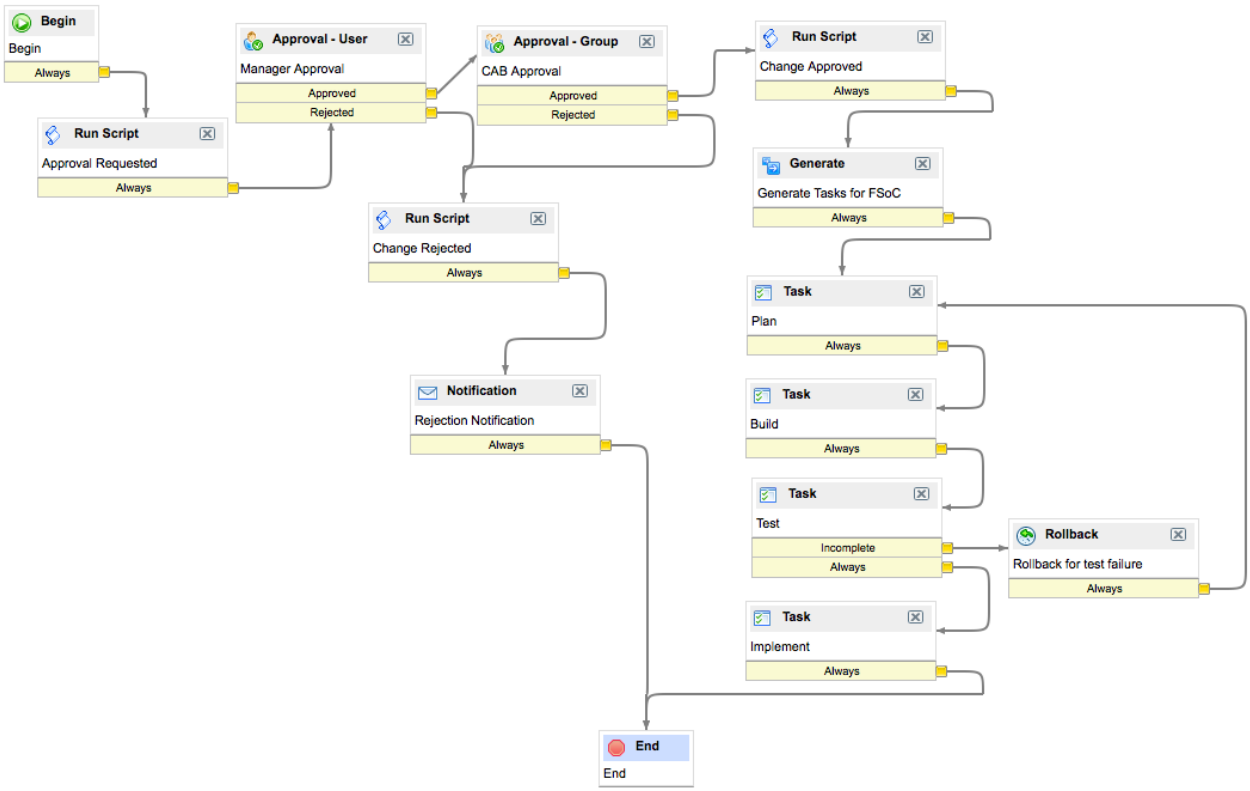
These three workflows are:

- Routine Change
- Comprehensive Change
- Emergency Change

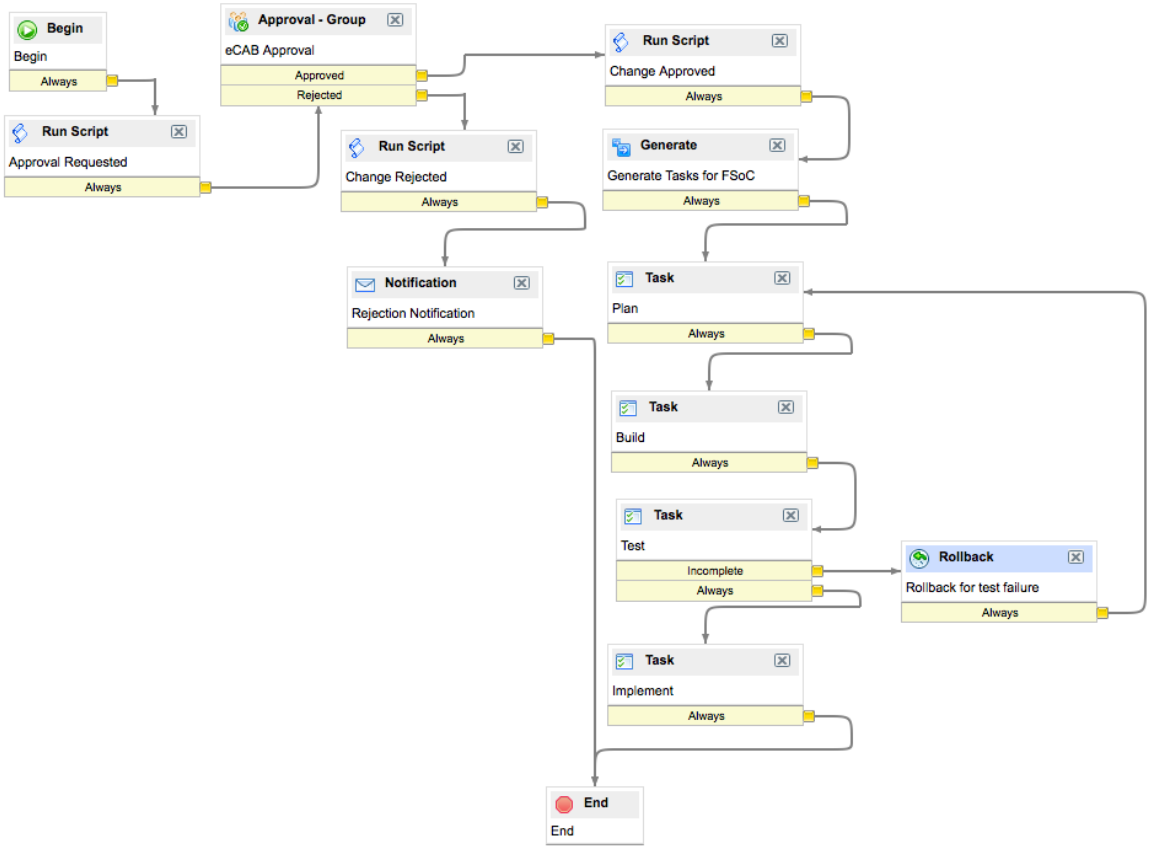
Routine Change



Comprehensive Change



Emergency Change



References

[1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/administer/workflow/reference/getting-started-workflows.html

Defining a Workflow

Note: *This article applies to Fuji. For more current information, see Change Management* [1] *at* http://docs.servicenow.com The ServiceNow Wiki is no longer being updated. Please refer to http://docs.servicenow.com for the latest product documentation.

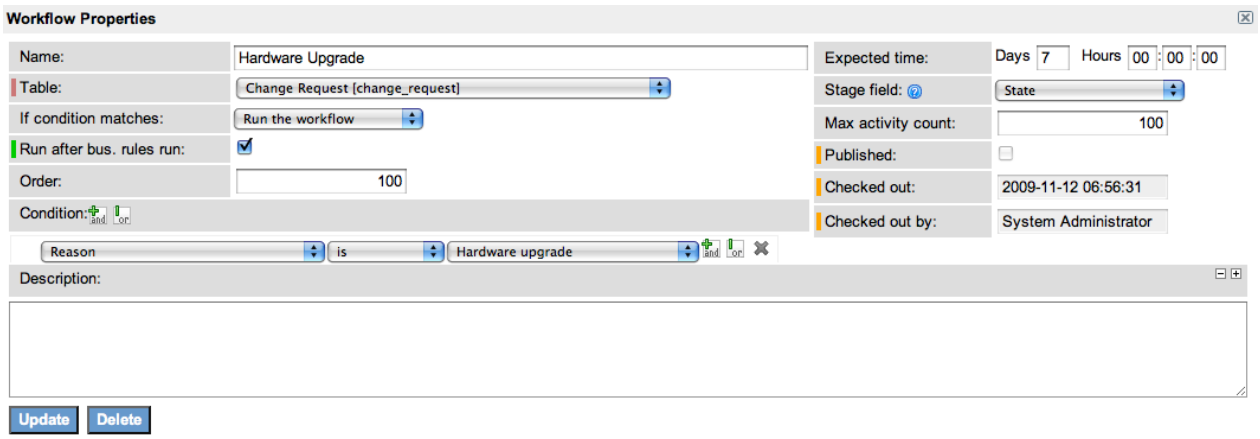
Overview

The Graphical Workflow Editor allows administrators to automate and define standard Change Management processes. The example workflow here performs the common task of upgrading hardware. The workflow can be reused any time that hardware is added to a configuration item.

In addition to manually defined workflows, you can install three basic Change Management workflows with the Change Management Workflows Plugin.

Defining a Change Management Workflow

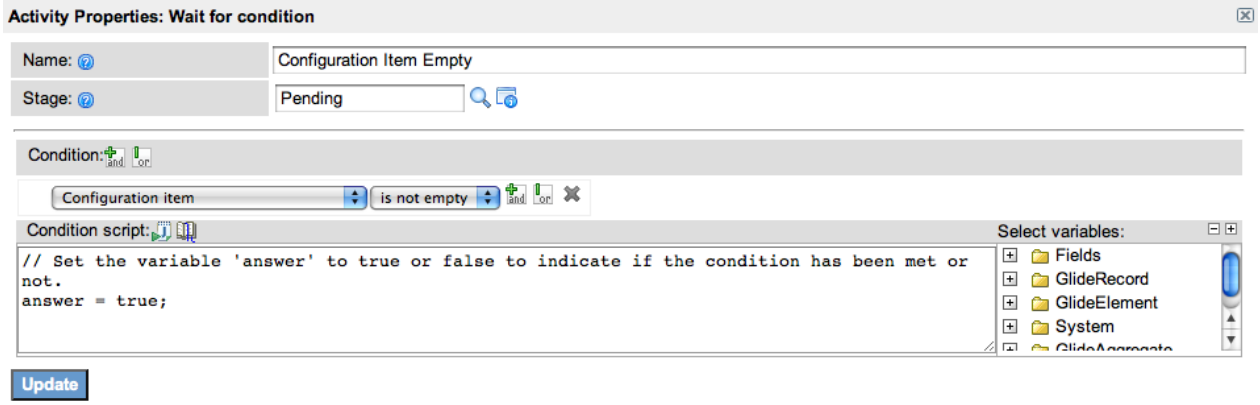
1. Navigate to **Workflow > Workflow Editor** and click **New**.
2. Populate the form as follows:
   - **Name** - Hardware Upgrade
   - **Table** - Change Request [change_request]
   - **If condition matches** - Run the Workflow
   - **Run after bus. rules run** - True. This field needs to be checked before any workflow that uses approvals, or the business rules will conflict with the workflow and fail to run properly. If this field does not appear on the Workflow Properties form, it will need to be added.
   - **Expected Time** - Days 7 Hours 00:00:00.
   - **Stage Field** - State. As the workflow passes from activity to activity, activites can update the state field accordingly.
   - **Conditions** - Reason is Hardware upgrade



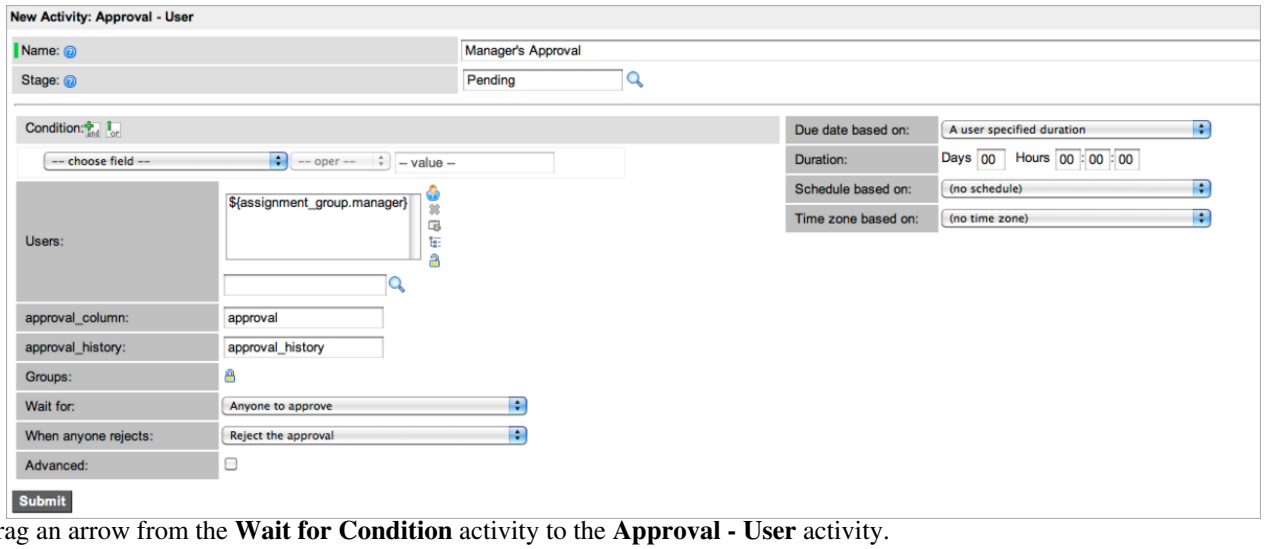
3. Drag the **If** activity onto the arrow between **Begin** and **End**. This activity will verify that a configuration item has been specified. Complete the fields as follows:

- **Name** - Configuration Item
- **Stage** - Pending
- **Condition** - Configuration Item is not empty.

4. Drag the **Wait for Condition** activity into the space below the **If** activity. This activity will wait for a configuration item to be supplied if none was originally supplied. Complete the fields as follows:
   - **Name** - Configuration Item Empty
   - **Stage** - Pending
   - **Condition** - Configuration Item is not empty.

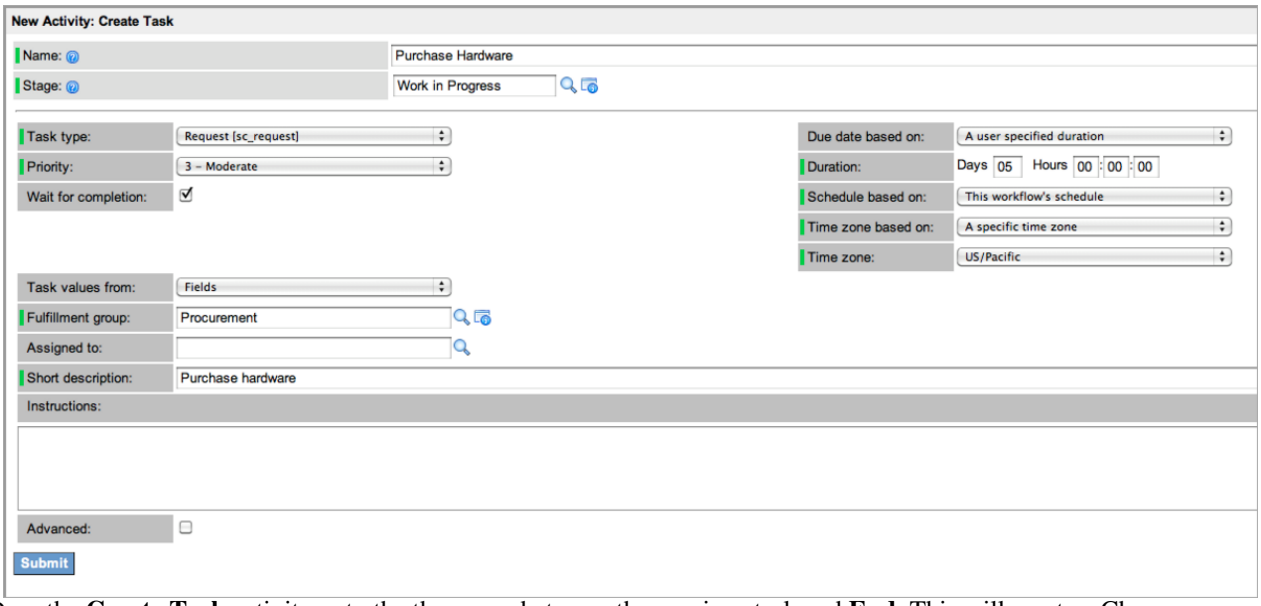


5. Drag an arrow from the **No** tab of the **If** activity to the end of the **Wait for Condition** activity.
6. Drag the **Approval - User** activity onto the arrow between **If** and **End**. This activity will request an approval from the assignment group's manager. Complete the fields as follows:
   - **Name** - Manager's Approval
   - **Stage** - Pending
   - **User** - Dot-walk to **${assignment_group.manager}**

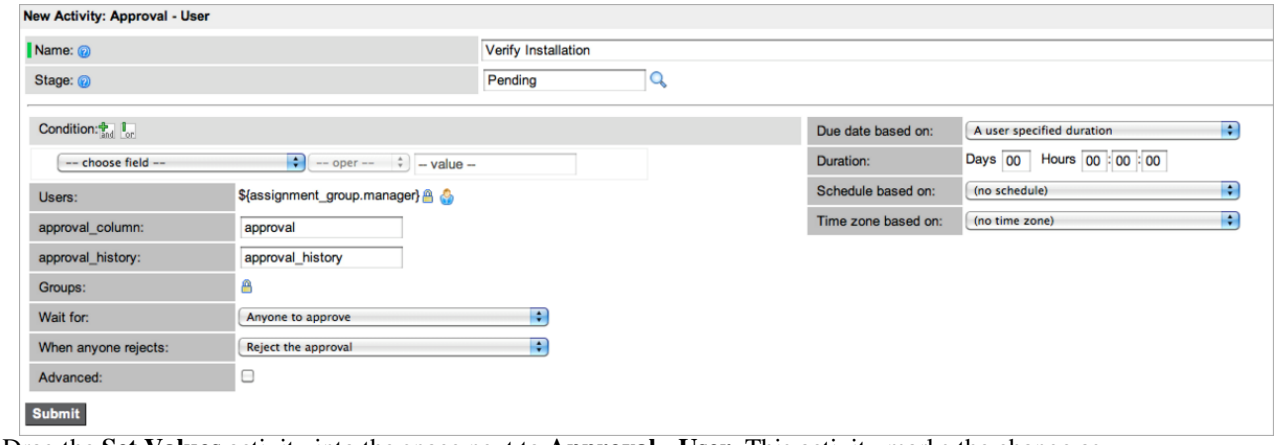


7. Drag an arrow from the **Wait for Condition** activity to the **Approval - User** activity.
8. Drag the **Set Value** activity into the space next to **Approval - User**. This activity will mark the change as Rejected. Complete the fields as follows:
   - **Name** - Rejection
   - **Stage** - Closed Skipped
   - **Set These Values:** - Approval Rejected.

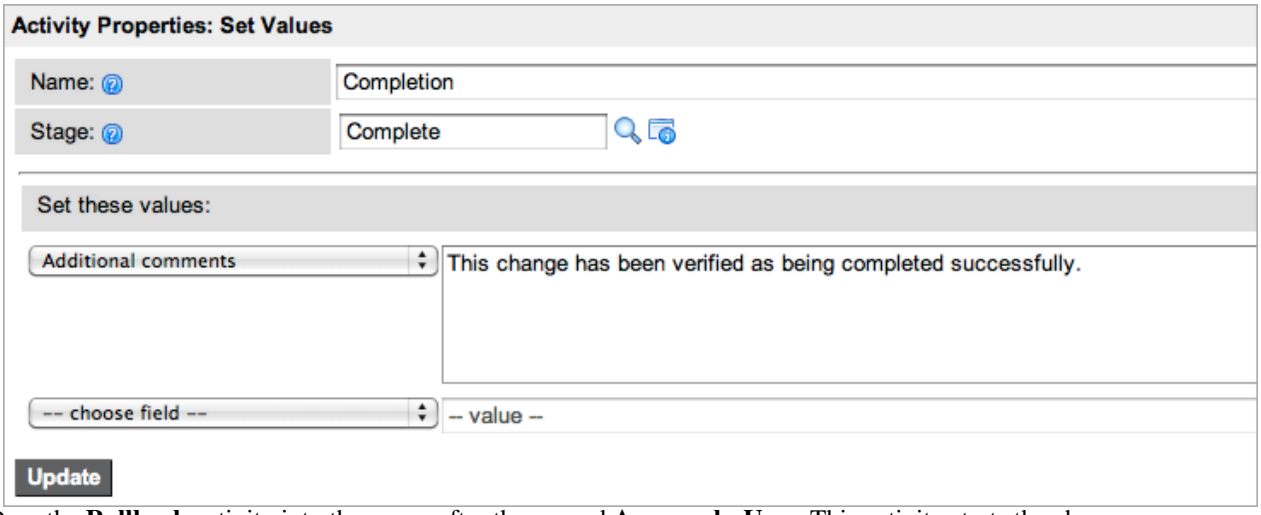
9. Drag an arrow from the **Rejected** tab at the bottom of **Approval - User** to **Rejection** and drag the tab at the bottom of **Rejection** to **End**.
10. Drag the **Create Task** activity onto the the arrow between **Approval - User** and **End**. This will create a catalog request for Procurement to acquire the hardware. Note that if there is a catalog fulfillment workflow that applies to this task, it will run. Complete the fields as follows:
    - **Name** - Purchase Hardware
    - **Stage** - Work in Progress
    - **Task Type** - Request [sc_request]
    - **Priority** - 3 - Moderate
    - **Fulfillment Group** - Procurement



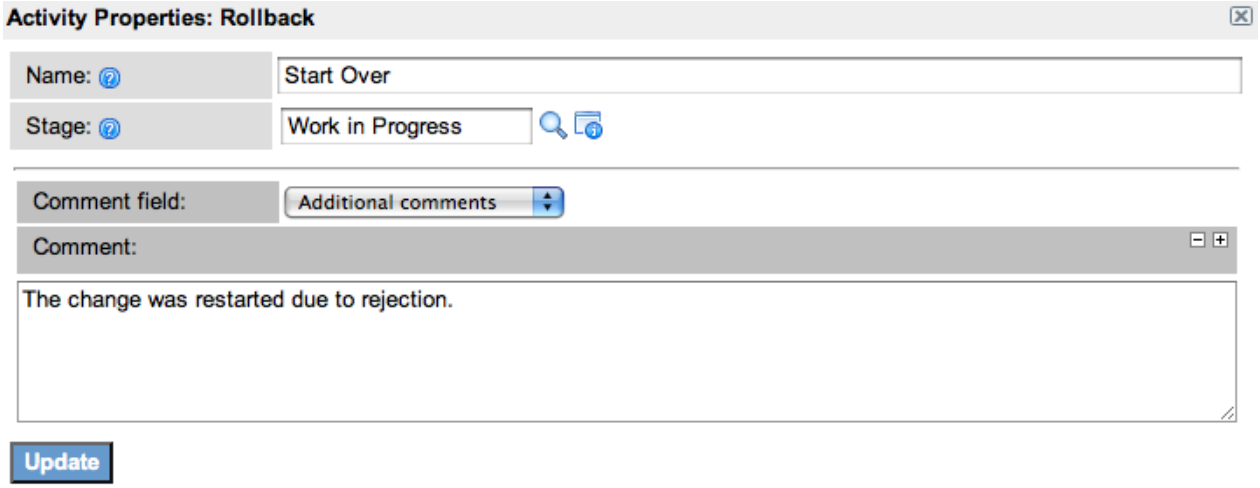
11. Drag the **Create Task** activity onto the the arrow between the previous task and **End**. This will create a Change Task for Hardware to install the hardware. Complete the fields as follows:
    - **Name** - Install Hardware
    - **Stage** - Work in Progress
    - **Task Type** - Change Task [change_task]
    - **Priority** - 3 - Moderate
    - **Fulfillment Group** - Hardware
12. Drag the **Create Task** activity onto the arrow between the previous task and **End**. This activity will generate a task to verify the installation. Complete the fields as follows:
    - **Name** - Verify Proper Installation
    - **Stage** - Work in Progress
    - **Fulfillment Group** - Software
13. Right click the **Approval - User** activity and select **Copy Activity**. Drag the arrow from the last task to the copied approval, and drag another arrow from the copied approval's **Approved** tag to **End**. This will ask the manager to verify the installation again. Click the copied **Approval** and update the form as follows:
    - **Name** - Verify Installation



14. Drag the **Set Values** activity into the space next to **Approval - User**. This activity marks the change as Completed. If you also want the corresponding Change record to automatically close upon workflow completion, mark the **Stage** as **Closed Complete** instead. Complete the fields as follows:
    - **Name** - Completion
    - **Stage** - Complete or Closed Complete
    - **Set These Values:** - Additional Comments: This change has been verified as being completed successfully.

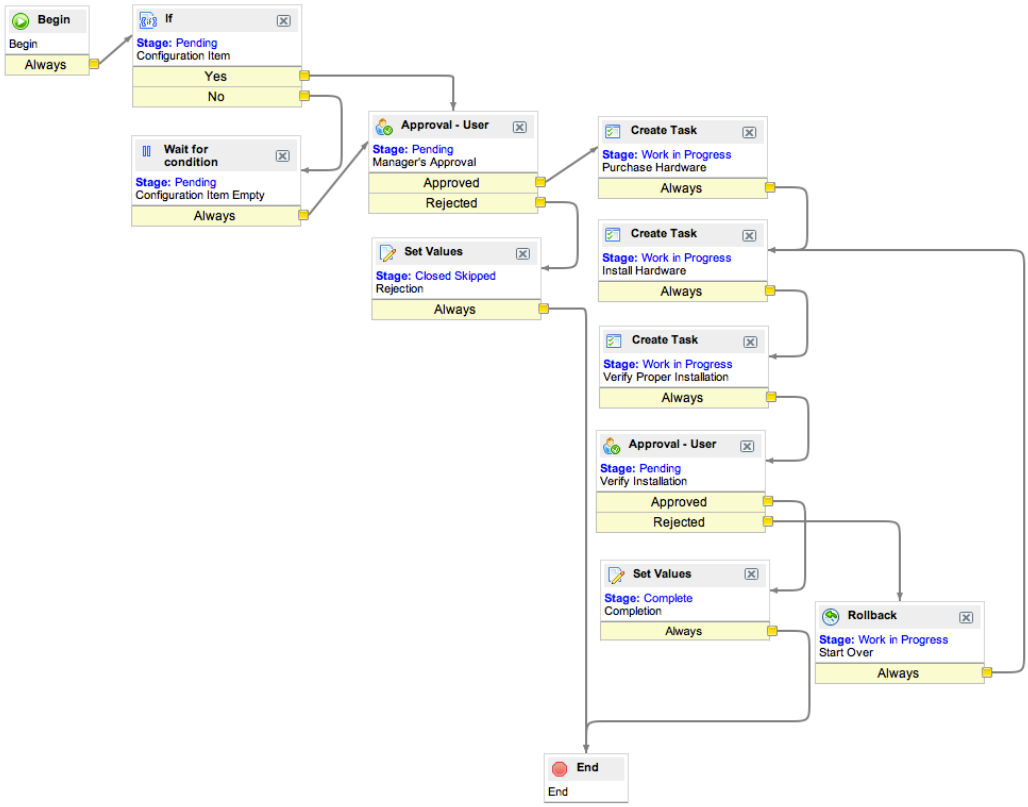


15. Drag the **Rollback** activity into the arrow after the second **Approval - User**. This activity starts the change over if the manager rejects it. Complete the fields as follows:
    - **Name**: *Start Over*
    - **Stage**: *Work in Progress*
    - **Comment Field**: *Additional Comments*
    - **Comments**: *The change was restarted due to rejection.*



16. Drag the arrow from the **Rejected** tab of the second **Approval - User** to the **Rollback** activity, and from the **Rollback** activity to **Install Hardware**.
17. If you want the workflow to automatically close the change request after the workflow completes, do one of the following:
    - Have the change workflow run as a subflow activity. After the change subflow completes, add a **Set Values** activity in the main workflow to set the desired state of the task.
    - Add a **Set Values** or **Run Script** activity to set the desired state of the task.
18. Publish the workflow in the Workflow Actions menu.

    The resulting workflow should be as follows:



Case Study - Advanced Approval Workflow

Part One

Overview

In the ServiceNow platform, there is often more than one way to approach implementing a process. Understanding how to translate a complex business requirement into an efficient in-product process may be tricky. This case study explores one example of approvals for a change process to demonstrate how to take advantage of some of the more advanced functionality to implement a more complex business case.

Business Case

For example, suppose a hypothetical organization wanted to implement the following business case for change requests:

- If the change request is for a routine change, then the usual routine change workflow occurs without extra approvals.
- If the change request is for a comprehensive or emergency change, then the Change Advisory Board (CAB) must approve the change before the usual comprehensive change workflow can occur.
- If the change request is for a comprehensive or emergency change with high or very high risk, then it requires the following approvals:

1. The Infrastructure team at the CI's location, the CI's owner, and the CAB
2. If the above parties approve, the VP of Infrastructure and the CFO must also both approve

Approvals can be handled either by approval rules or by the workflow engine. The approval rules offer a simpler method for creating approvals and is suitable for one-stage approvals with clear conditions. However, this business case is more suitably executed by a workflow because it includes an approval with multiple stages. All of the approval processes can be defined in one workflow, rather than in three separate approval rules, which will be easier to review and alter in the future.

Part one of this case study demonstrates how to satisfy the first two requirements of the business case, and part two demonstrates how to satisfy the last requirement.

Building the Advanced Approval Workflow (Part One)

Preparing the Routine and Comprehensive Change Subflows

The workflow must trigger the processes for implementing a routine change and a comprehensive change. It is possible to specify these processes in the workflow, but this can lead to an overcomplicated workflow that is difficult to follow visually. Instead, use the predefined **Routine Change** and **Comprehensive Change** workflows as subflows. These predefined workflows are included in the Change Management Workflows plugin.

To ensure that the subflows are not triggered twice, set their **If Condition Matches** property to **None**.

1. Open and check out the **Routine Change** workflow.
2. In the title bar, click the menu icon (gear icon in releases prior to Fuji) and select **Properties**.
3. From the **If Condition Matches** list, select **None**.
4. Click **Update**.

5. Repeat these steps for the **Comprehensive Change** workflow.

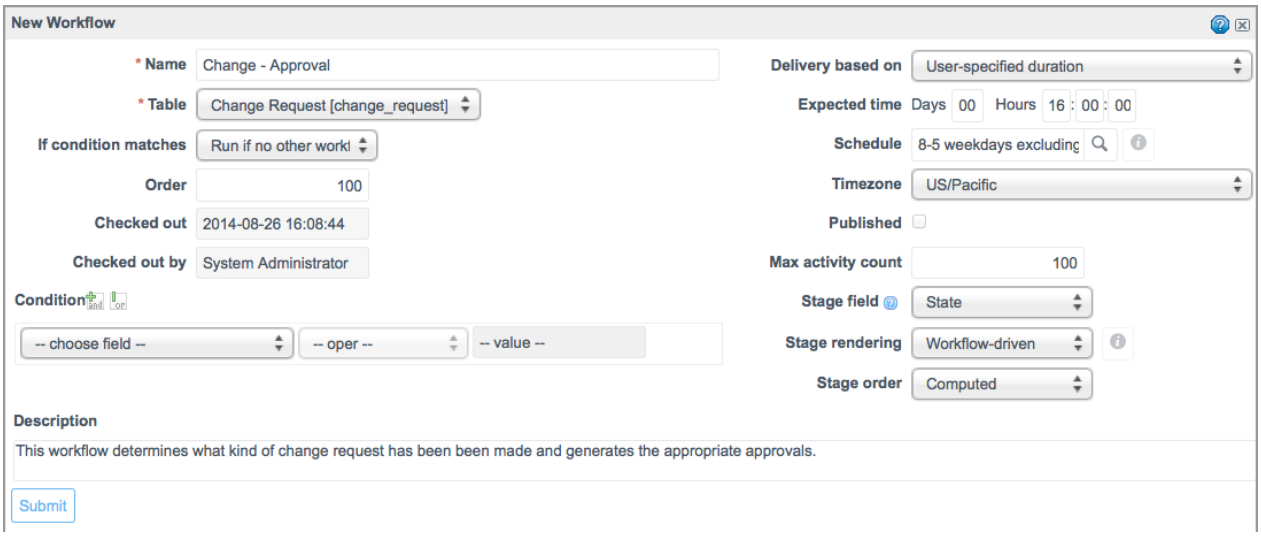
Defining the Workflow

The first step in creating the **Advanced Approval** workflow is to create the workflow and define its properties. The workflow you want to create is a default workflow that takes new change requests and sorts them based on both the type of change (routine vs. comprehensive) and risk of the change (low/medium vs. high/very high). The workflow runs on the Change Request table unless another workflow is attached to the change; this means that when a new change request is saved, this workflow attaches to the record unless the user has specified a particular workflow to process the change.

To define the workflow for the business case above:

1. Navigate to **Workflow > Workflow Editor** and create a new workflow.
2. Populate the form with the following information:
   - **Name**: *Change - Approval*
   - **Table**: *Change Request [change_request]*
   - **If condition matches**: *Run if no other workflows are matched yet*
   - **Stage field**: *State*

     In Fuji, you must submit the workflow, reopen the properties, and click the Stages tab to see the **Stage field** list.
3. Click **Submit**.



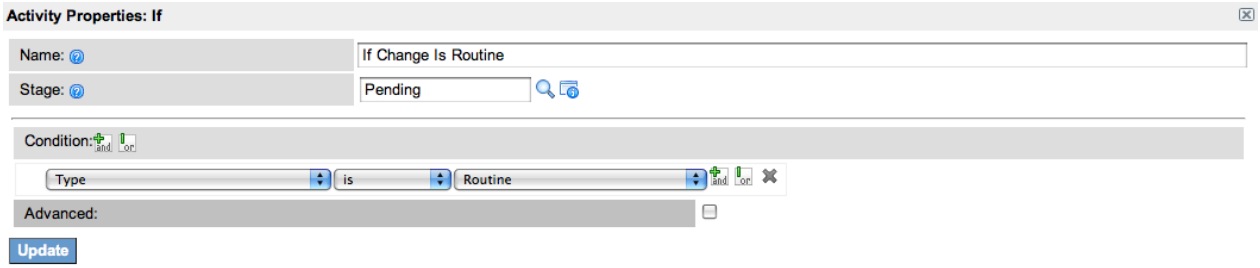
   This workflow is triggered by any new change requests that do not fulfill the conditions of a different workflow. As the workflow passes from activity to activity, it updates the **State** field, allowing users to track the change request's progress.
4. Add activities to define the workflow's behavior.

Defining the Routine Change Case

The first business case requirement is that a routine change triggers the **Routine Change** workflow. The workflow must determine the type of workflow based on the change request's **Type** field. If the field is a match, it triggers the process for routine changes without any approvals. This requires an **If** activity to identify routine changes, and then an activity to trigger the **Routine Change** workflow.

To create the **If** activity:

1. Drag the **If** activity onto the arrow between **Begin** and **End**.
2. Populate the form with the following information:
   - **Name**: *If Change Is Routine*
   - **Stage**: *Pending*
   - **Condition**: *[Type] [is] [Routine]*
3. Click **Submit**.



To add the **Routine Change** workflow:

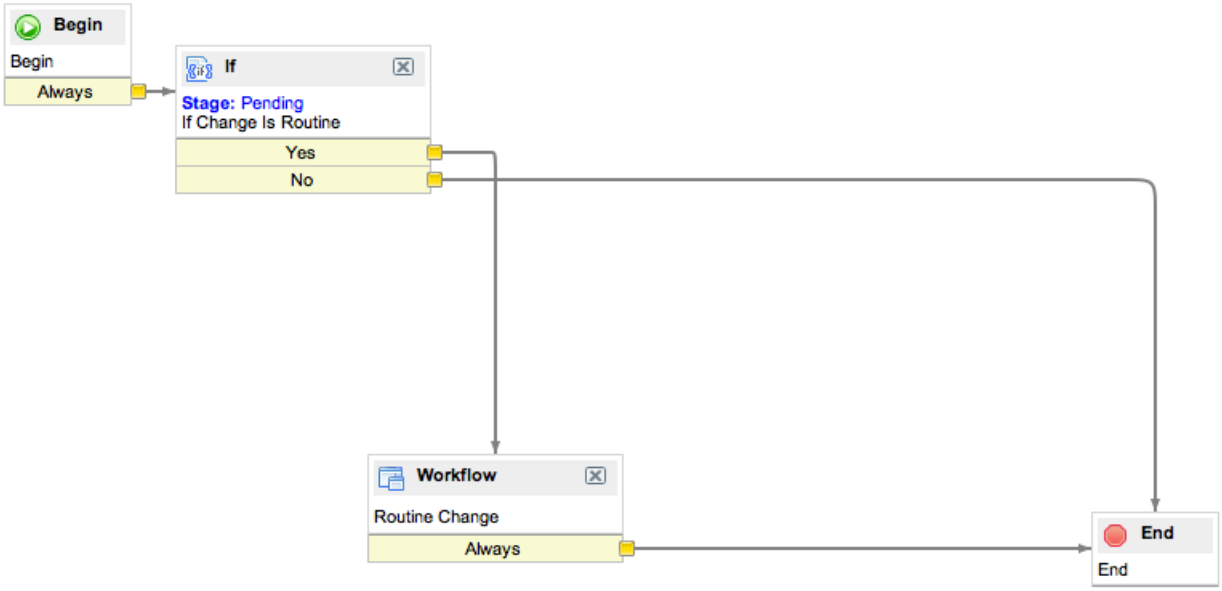
1. From the **Workflows** list in the palette, drag the **Routine Change** workflow into the area below the **If Change is Routine** activity.
2. Submit the form that appears.
3. From the **If Change Is Routine** activity:

   a. Delete the line from the **Yes** tab to the **End** activity.

   b. Click the **Yes** tab and drag to the **Routine Change** workflow.

   c. Click the **No** tab and drag to the **End** activity.
4. From the **Routine Change** workflow, click the **Always** tab and drag to the **End** activity.

   The workflow appears like this:



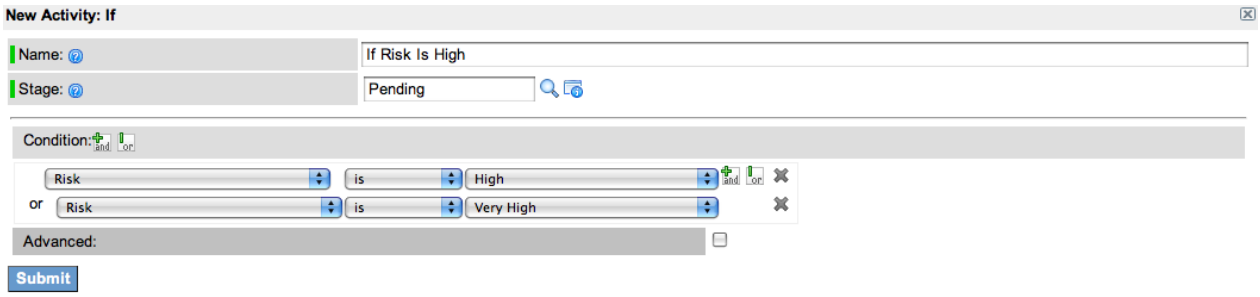
Defining the Comprehensive Change Workflow

The next step in the advanced approval process is handling non-routine changes. Comprehensive changes are identified as high risk or non-high risk. If the change is not high risk, an approval for it can be requested from the CAB. If the CAB approves the change, the workflow triggers the **Comprehensive Change** workflow. If the CAB rejects the change, the change is marked as **Rejected** and **Incomplete** before the workflow ends.

The **CAB** group in this example was created as a new group to illustrate the process. Any group can be used in its place.

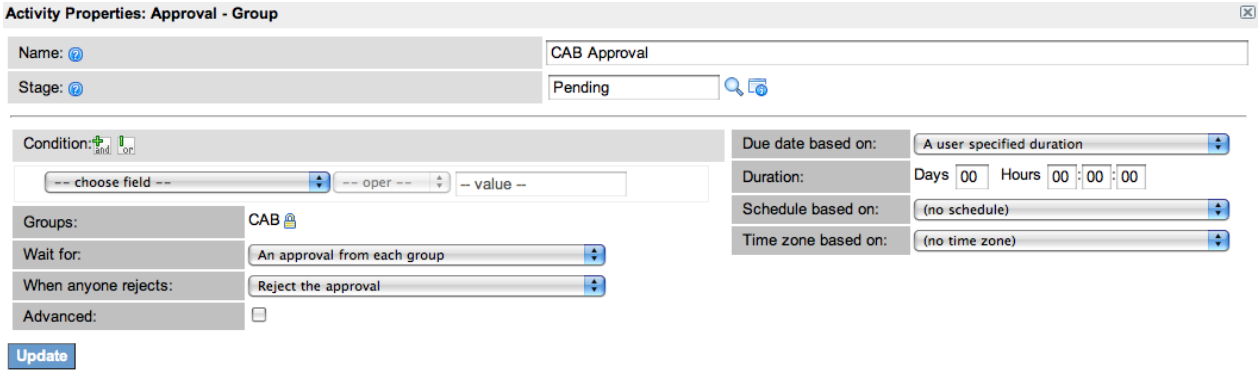
To create the **If** activity that determines if the change is high risk:

1. Drag the **If** activity onto the arrow between the **If Change Is Routine** and **End** activities.
2. Populate the form with the following information:
   - **Name**: *If Change Is High Risk*
   - **Stage**: *Pending*
   - **Condition**: *[Risk] [is] [High]* OR *[Risk] [is] [Very High]*
3. Click **Submit**.



To create the **CAB** approval activity:

1. Drag the **Approval - Group** activity into the area below the new **If Change Is High Risk** activity.
2. Populate the form with the following information:
   - **Name**: *CAB Approval*
   - **Stage**: *Pending*
   - **Groups**: *CAB* (or the name of the group used to identify your Change Approval Board members)
3. Click **Submit**.



4. To trigger the **CAB Approval** activity for non-high risk activities, click the **No** tab from the **If Change is High-Risk** activity and drag to the **CAB Approval** activity.

To trigger the **Comprehensive Change** workflow for changes approved by the CAB:

1. From the **Workflows** list in the palette, drag the **Comprehensive Change** workflow into the area below the **If Change is High Risk** activity.
2. Submit the form that appears.

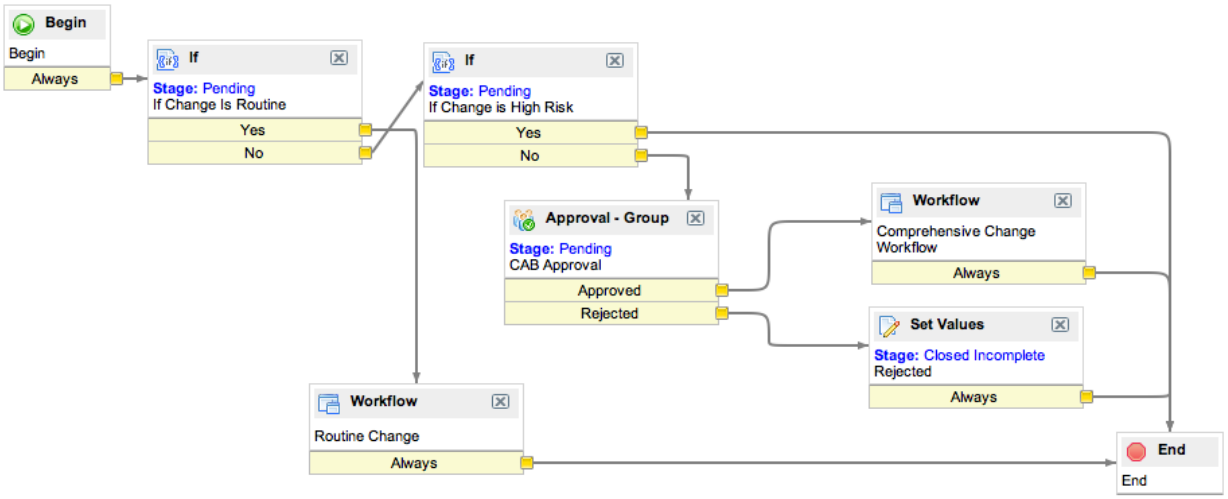
3. From the **CAB Approval** activity, click the **Approved** tab and drag to the new **Comprehensive Change** workflow.

To set the value of changes rejected by the CAB to **Rejected**:

1. Drag the **Set Values** activity into the area below the new **CAB Approval** activity.
2. Populate the form with the following information:
   - **Name**: *Rejected*
   - **Stage**: *Complete*
   - **Set these values**: From the first choice list, select *Approval*. From the second choice list, select *Rejected*.
3. Click **Submit**.
4. From the **CAB Approval** activity, click the **Rejected** tab and drag to the new **Rejected** activity.
5. From the **Rejected** activity, click the **Always** tab and click to the **End** activity.

Result

The workflow should now look like this:



This workflow fulfills the first two points of the business case: it triggers the proper approvals and workflow for both routine changes and low-to-medium risk comprehensive changes. To fulfill the third point of the business case, see Part Two.

Part Two

Overview

The workflow engine can provide a powerful and flexible method of automating a complex approval process. This case study demonstrates how to design a complex approval workflow for a hypothetical business case.

These points of the business case were fulfilled in Part One of this case study:

- If the change request is for a routine change, then the usual routine change workflow needs to take place without extra approvals.
- If the change request is for a comprehensive or emergency change, then the CAB needs to approve the change, and then the usual comprehensive change workflow should take place.

The last point of the business case is explained in this part of the case study. If the change request is for a comprehensive or emergency change with high or very high risk, then it requires the following approvals:

1. The Infrastructure team at the CI's location, the CI's owner, and the Change Advisory Board.
2. If the above parties approve, then the VP of Infrastructure and the CFO must also both approve.

Building an Advance Approval Workflow (Part 2)

The approval process for high risk changes is more complicated than the approvals for the other two categories of changes. As such, the approvals will be requested in two rounds:

1. One round of approvals will get approvals from any member of the Infrastructure Department at the configuration item's location, AND the owner of the configuration item, AND any member of the Change Advisory Board. These are the individuals who are involved in the change -- the affected party and the people who will implement the change.
2. The second round of approvals will only be issued if all of the previous approvals are approved and will request approvals from the VP of Infrastructure and the CFO. Presumably, because this is a high-risk change, there is a requirement that it be approved by senior executives.

This means that all of the parties involved in the change (which are the members of the first approval round) will have to finish their approvals before the executives are involved in round 2. This is a way of reducing the amount of approvals requested of executives, since only change requests which have been approved of by all of the involved parties will proceed to the executives.

Requesting the First Round of Approvals

If all of the members of the first round were of the same type (e.g. all users, or all groups), then it could be accomplished using an **Approval - Group** or an **Approval - Activity**. To issue multiple approvals to different types of parties, this example will use the **Approval Coordinator** activity. First, the approval coordinator itself is defined, and then any number of approvals within the approval coordinator can be defined as well. To power the first round of approvals, create an **Approval Coordinator** with two **Approval - User**s and one **Approval - Group** within it.

For the purposes of this example, a department called **Infrastructure** has been created.

To define the Approval Coordinator:

1. Drag an **Approval Coordinator** activity onto the arrow between the **If Change is High Risk** and **End** activities.
2. Populate the form as follows:
   - **Name** - Involved Parties Approval
   - **Stage** - Pending

- **Wait For** - All Child activities to be approved. This means that *all* of the approvals within the approval coordinator must be marked **Approved** to continue.

The first approval within the **Approval Coordinator** is going to be the approval for members of the Infrastructure Department at the location of the configuration item. Because the users are going to be found based on both location and department, it will be necessary to use a simple script to filter users.

This script queries the user table for any users whose location is the same as the Change record's configuration item's location, AND whose department is **Infrastructure** (in this example, the sys_id **e29201830a0a3c1e01554027c17b1593**). The **sys_id** for **Infrastructure** may differ, so be sure to replace that sys_id with the correct sys_id. (For information on looking up the sys_id, see here).

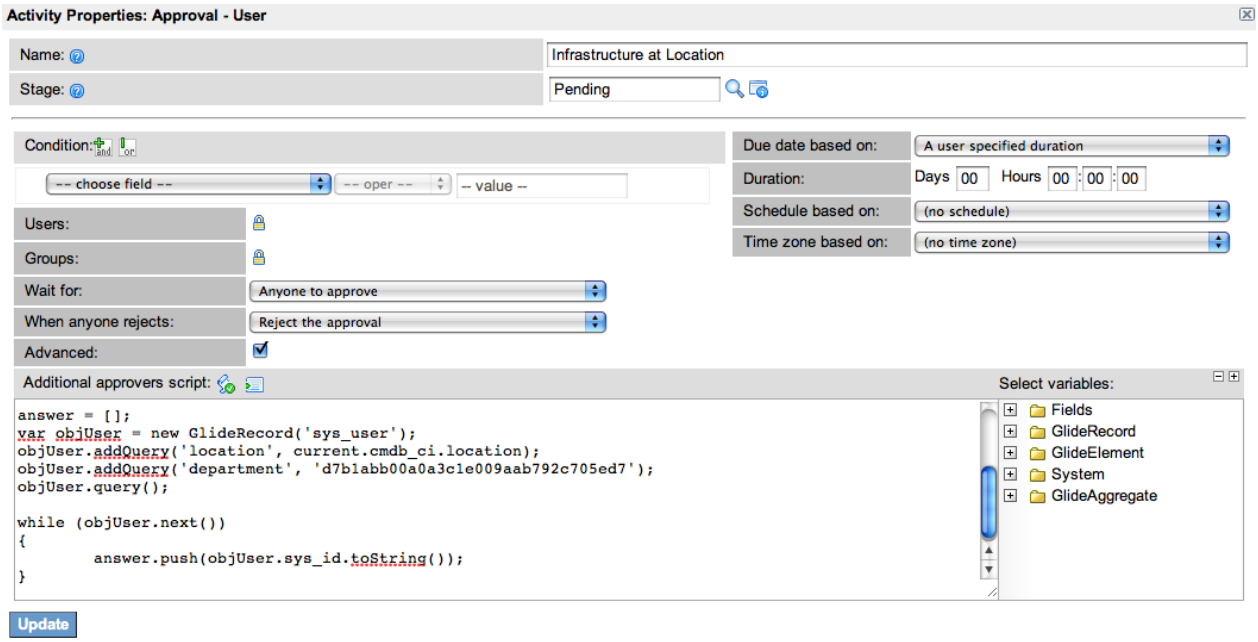
Note that if this was being implemented in a business, it would be useful to make Location a mandatory field on Configuration Items. If a configuration item has no location specified, this approval will be skipped.

To define the Infrastructure Department's approval:

1. Within the **Approval Coordinator** activity in the workflow, click the **Add Approval - User** link.
2. Populate the form as follows:
   - **Name** - Infrastructure at Location
   - **Stage** - Pending
   - **Wait For** - Anyone to approve
   - **Advanced** - True
   - **Script** -

```
answer = [];
var objUser = new GlideRecord('sys_user');
objUser.addQuery('location', current.cmdb_ci.location);
objUser.addQuery('department', 'e29201830a0a3c1e01554027c17b1593');
objUser.query();

while (objUser.next())
{
      answer.push(objUser.sys_id.toString());
}
```

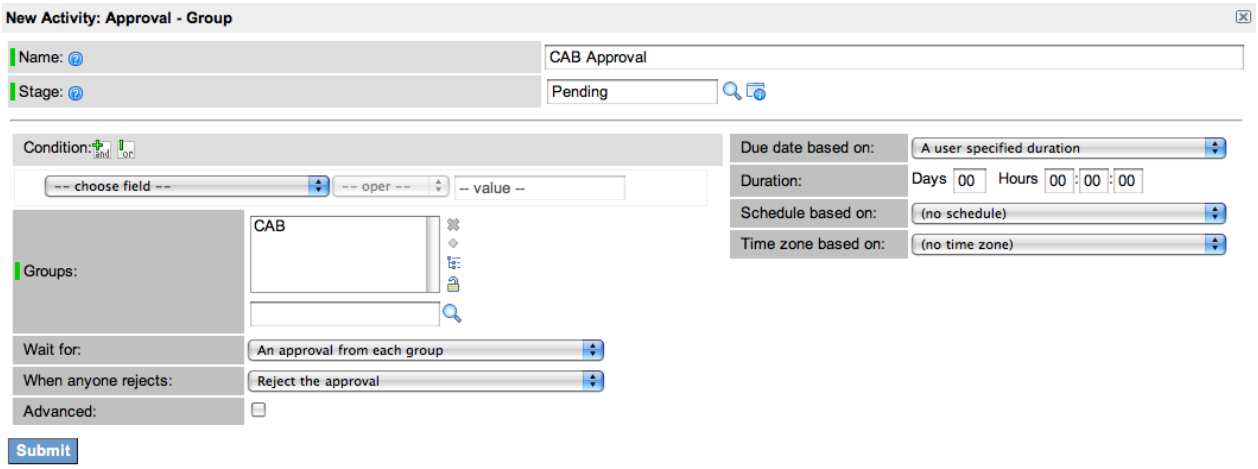


To define the CI's owner's approval:

1. Within the **Approval Coordinator** activity in the workflow, click the **Add Approval - User** link.
2. Populate the form as follows:
   - **Name** - Owner Approval
   - **Stage** - Pending
   - **Users** - Use the Variable Picker ( ) to select the Configuration Item's **Assigned To** and **Managed By** fields.

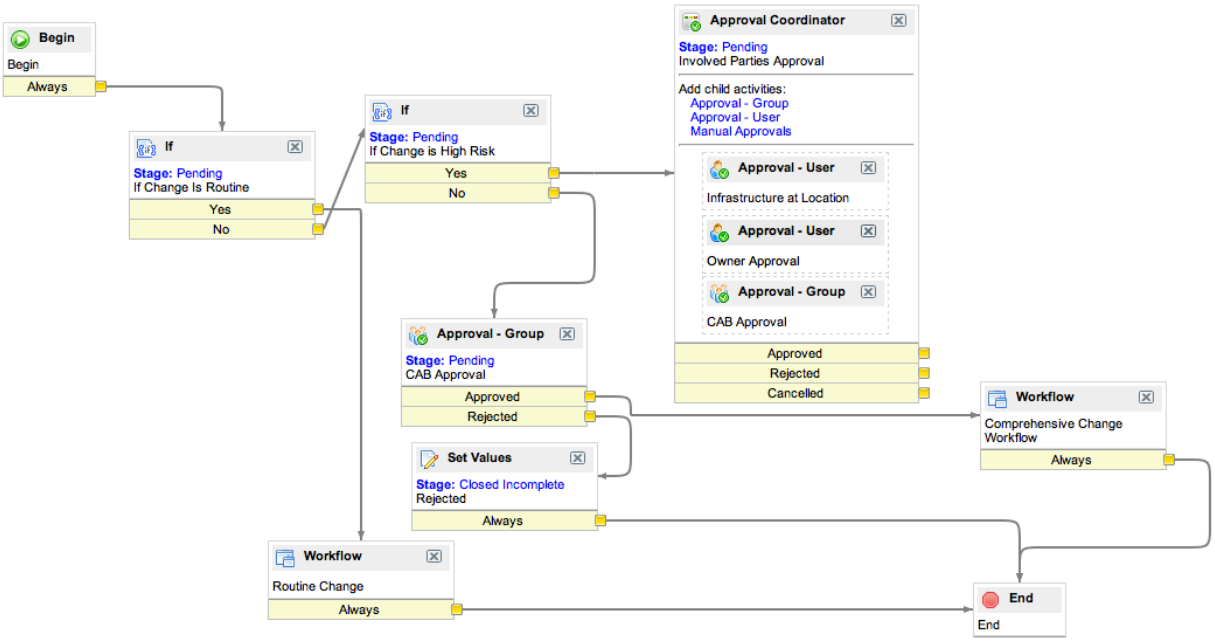
To define the CAB's approval:

1. Within the **Approval Coordinator** activity in the workflow, click the **Add Approval - Group** link.
2. Populate the form as follows:
   - **Name** - CAB Approval
   - **Stage** - Pending
   - **Groups** - CAB



Result

The workflow with the finished Approval Coordinator should look like this:



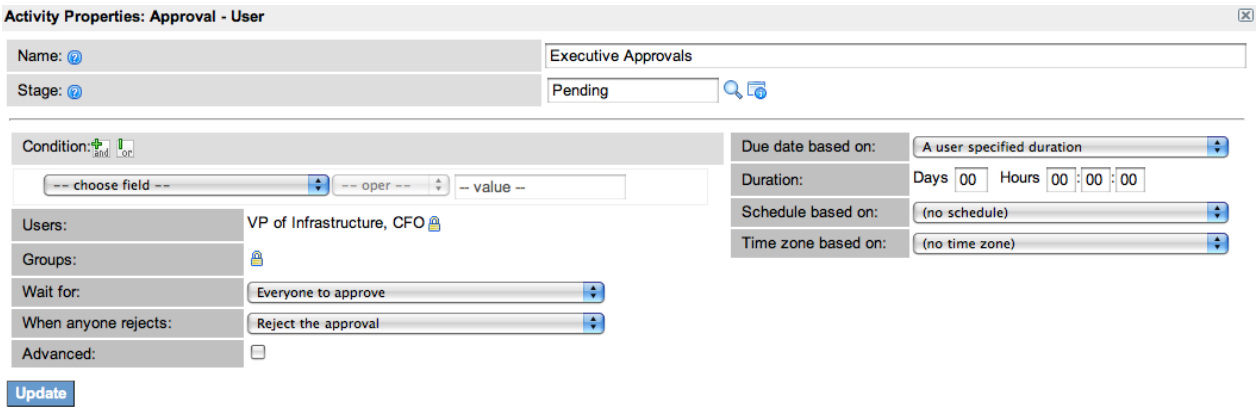
Requesting the Second Round of Approvals

If the first round of approvals are approved, a second round of approvals need to be issued to the VP of Infrastructure and CFO. This example is manually issuing the approvals to both the VP of Infrastructure and CFO, so that we can use one **Approval - User** activity to issue both approvals.

For the purposes of this example, the users **CFO** and **VP of Infrastructure** have been created and do not exist out-of-box.

To define the Executives' approval:

1. Drag the activity **Approval - User** to the space to the right of the Approval Coordinator.
2. Populate the form as follows:
   - **Name** - Executive Approvals
   - **Stage** - Pending
   - **Users** - VP of Infrastructure, CFO
   - **Wait For** - Everyone to Approve



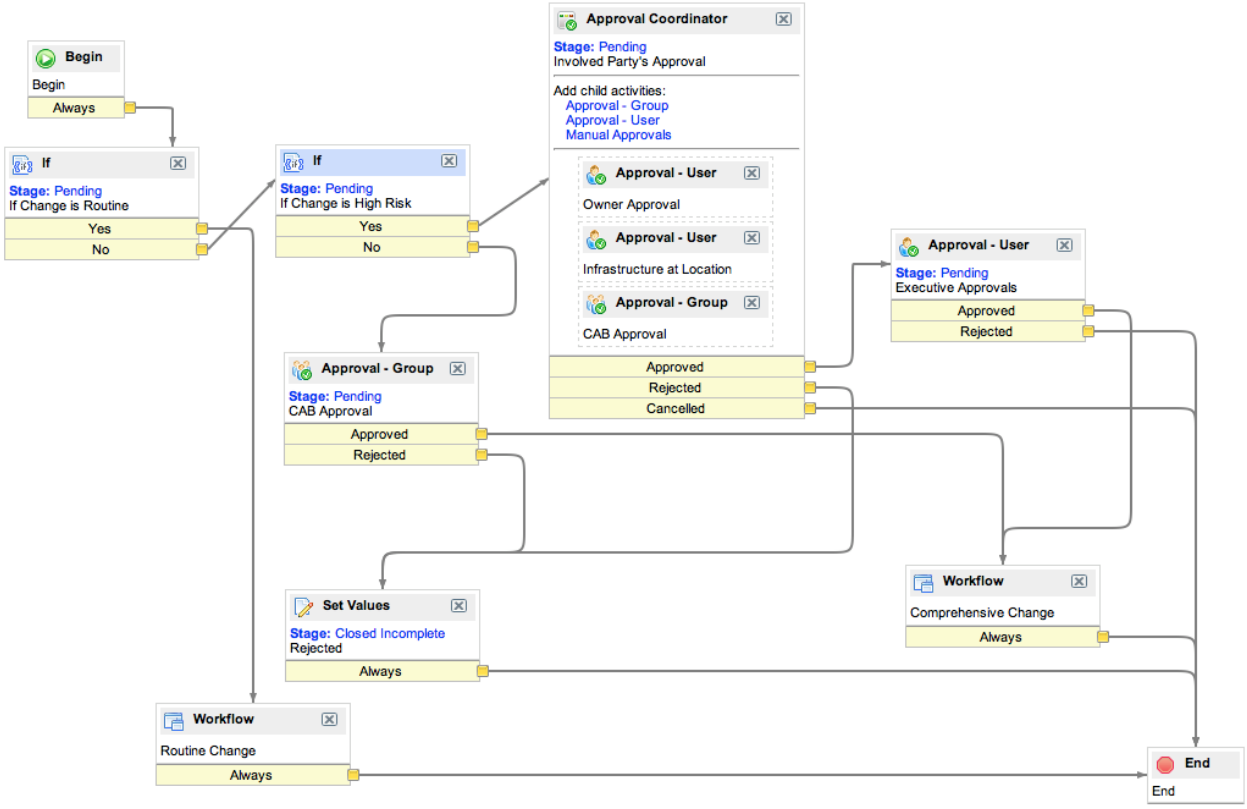
Tying the Approvals Together

Now all that remains is to connect the new activities to the correct outcome:

1. Connect the **Approved** transition of the **Approval - User** activity to the **Comprehensive Change** workflow.
2. Connect the **Rejected** transition of the **Approval - User** activity to the **End** point.
3. Connect the **Approved** transition of the **Approval Coordinator** activity to the **Approval - User** activity.
4. Connect the **Rejected** transition of the **Approval Coordinator** activity to the **Set Value** activity.
5. Connect the **Cancelled** transition of the **Approval Coordinator** activity to the **End** point.

Result

This example workflow (downloadable as an update set here) is the result, and satisfies the approval business case outlined at the start of this case study:



Continual Service Improvements

Reporting

Note: *This article applies to Fuji and earlier releases. For more current information, see Reporting* [1] *at* http://docs.servicenow.com **The ServiceNow Wiki is no longer being updated. Visit** http://docs.servicenow.com **for the latest product documentation.**'

Overview

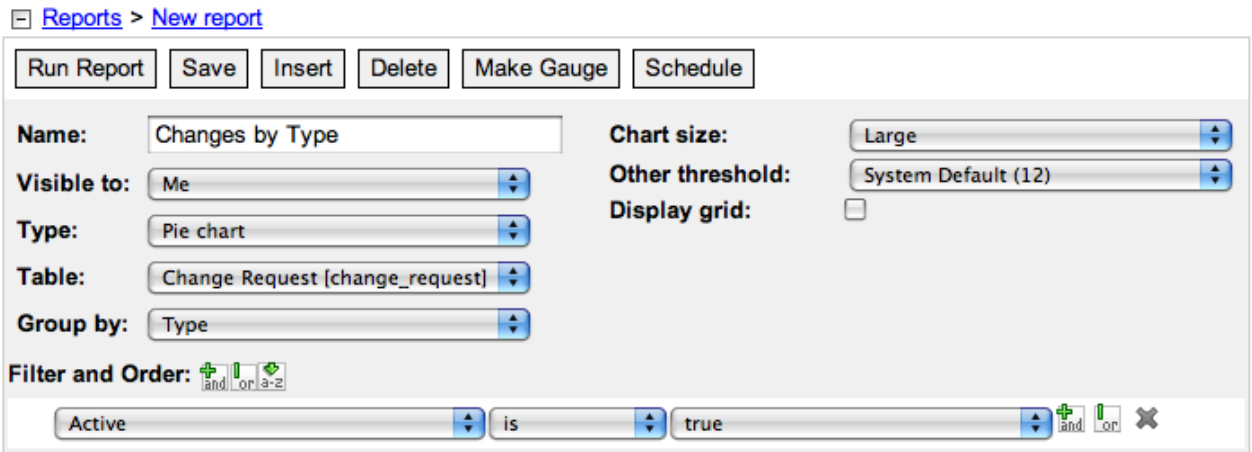
In order to continually improve the Change Management process, it is possible to gather the information collected by the platform and present the data in reports. Below are instructions on how to create a pair of sample Change Management reports.

Changes by Type

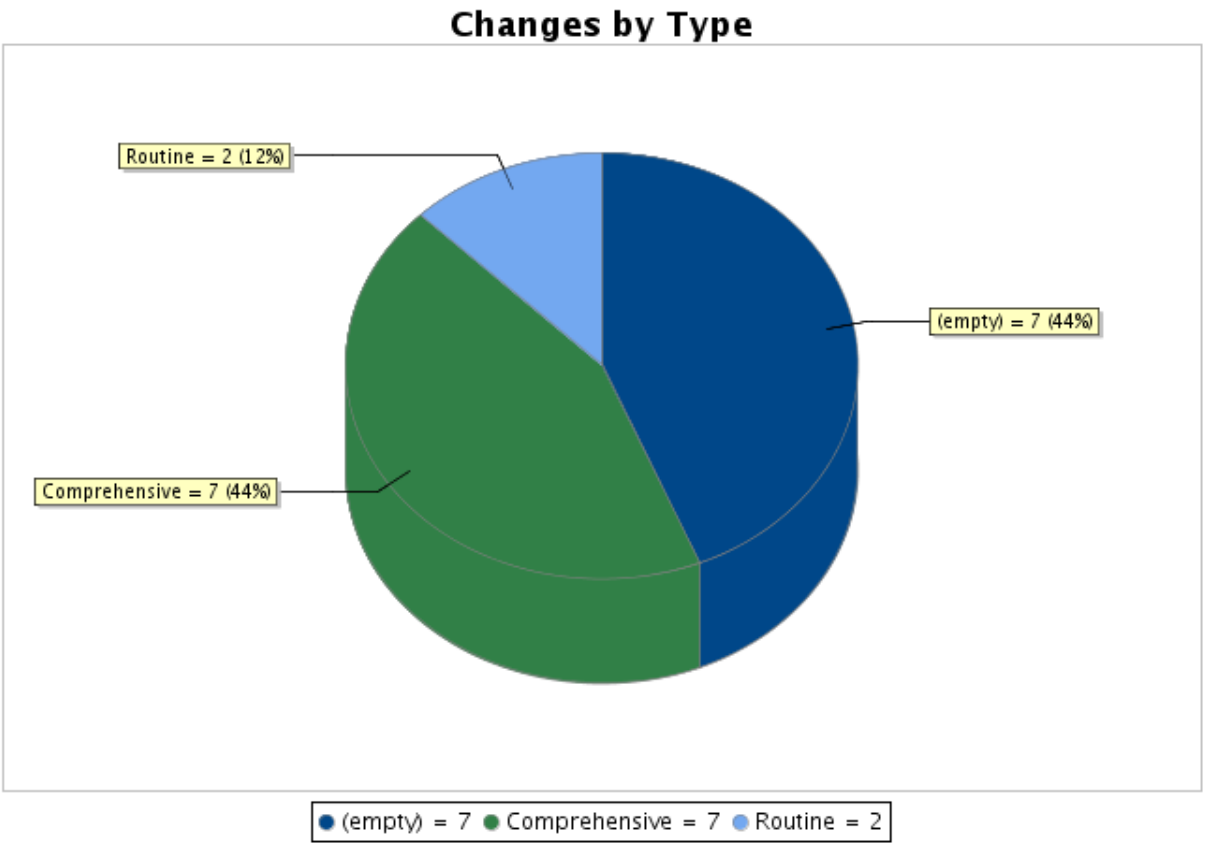
This pie chart displays active changes, grouped by type of change.

To report on Incidents by Caller's Company:

1. Navigate to **Reports > View / Run** and click **New**.
2. Populate the form as follows:
   - **Name** - Changes by Type.
   - **Type** - Pie Chart
   - **Table** - Change Request [change_request]
   - **Group By** - Type
   - **Filter and Order** - Active is True



3. Select **Run Report**. The bar chart should appear as follows:



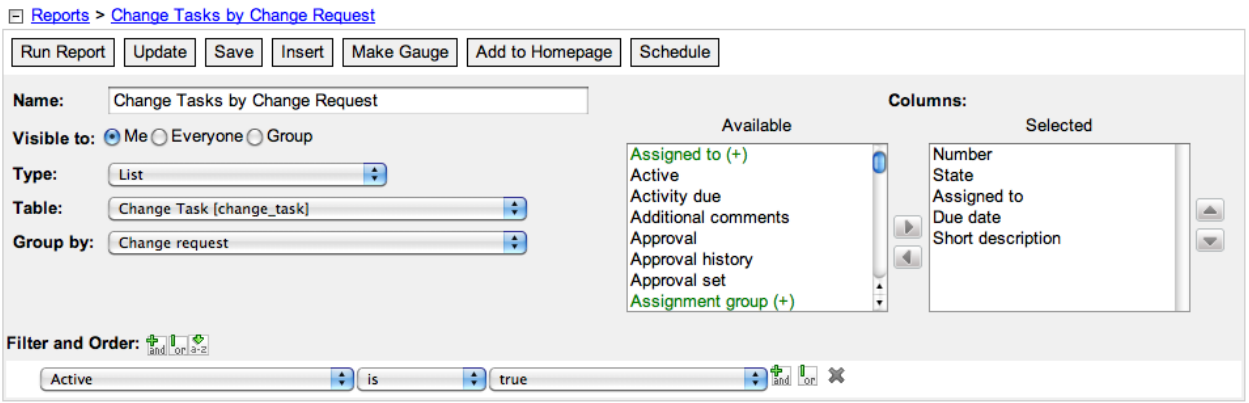
4. Click **Save**.

Change Tasks by Change Request

The following example is a list that displays active Change Tasks grouped by Change Requests.

To report on open Change Tasks by Change Request:

1. Navigate to **Reports > View / Run** and click **New**.
2. Populate the form as follows:
   - **Name** - Change Tasks by Change Request.
   - **Type** - List
   - **Table** - Change Task [change_task]
   - **Filter and Order** - Active is True.



3. Select **Run Report**. The line chart should appear as follows:

Change Tasks

1 to 2 of 2

6 total Change Tasks

Number	State	Assigned to	Due date	Short description
Change request: CHG0000001 (4)				
CTASK0010001	Open	Don Goodliffe	2011-03-08 14:19:41	Back-Up Database
CTASK0010002	Open	David Loo	2011-03-08 14:20:17	Take VmWare snapshot of Environment
CTASK0010003	Open	Bow Ruggeri	2011-03-08 18:00:00	Install Oracle 10g onto SD1
CTASK0010004	Open	Don Goodliffe	2011-03-09 14:21:56	Configure Siebel Software for new DB
Change request: CHG0000002 (2)				
CTASK0010005	Open	Bow Ruggeri	2011-03-09 14:22:39	Preliminary System Testing
CTASK0010006	Open	Bow Ruggeri	2011-03-09 16:00:00	Multi-User Testing and QA

1 to 2 of 2

4. Click **Save**.

Changes by Configuration Item and State

To report on Changes by Configuration Item and State:

1. Navigate to **Reports > View / Run** and click **New**.
2. Populate the form as follows:
   - **Name** - Changes by Configuration Item and State.
   - **Type** - Pivot Table
   - **Table** - Change Request [change_request]
   - **Row** - Configuration Item.
   - **Column** - State.
   - **Filter and Order** - Assigned To is the name of the current user. In this example, the current user will be ITIL User.

Reports > New report

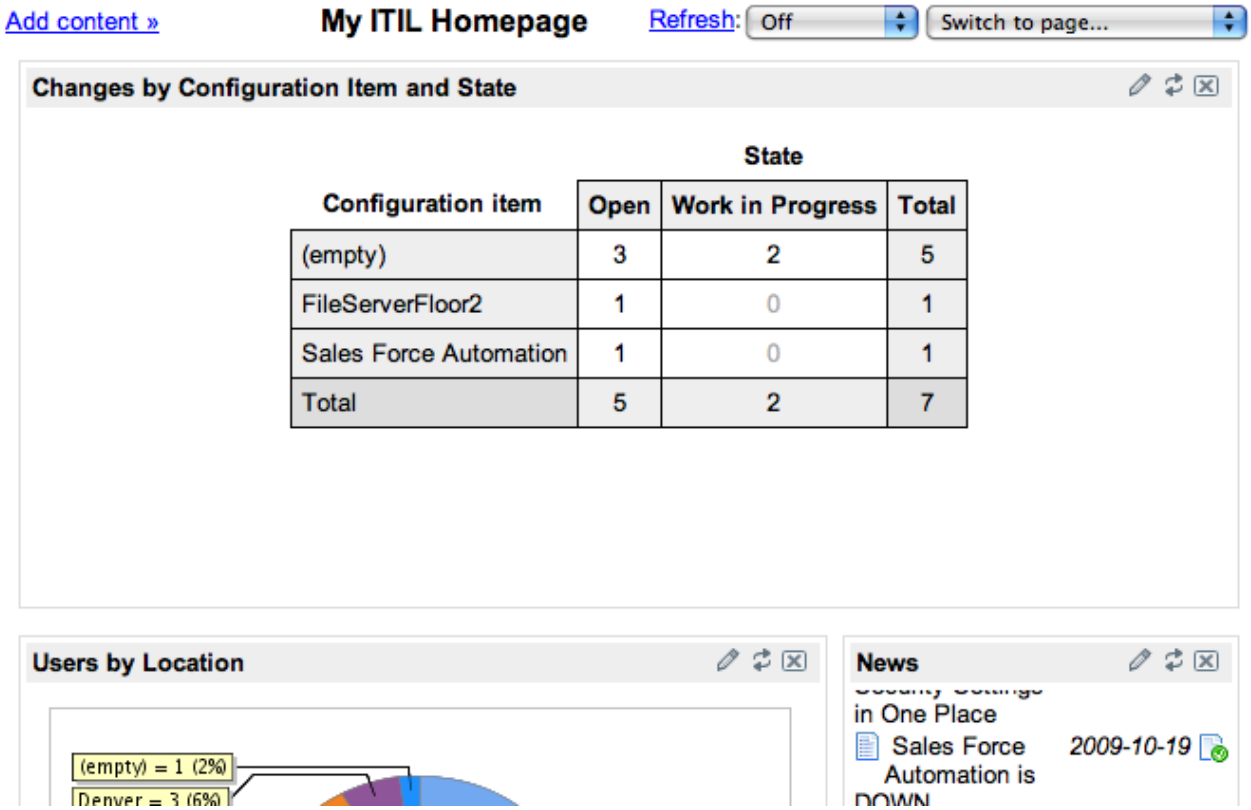
Run Report | Save | Insert | Delete | Make Gauge

Name: Changes by Configuration Item and Sta
Visible to: Me
Type: Pivot Table
Table: Change Request [change_request]
Row: Configuration item
Column: State
Aggregation: Count
Other threshold: System Default (12)
Filter:
Assigned to | is | ITIL User

3. Select **Run Report**. The table should appear as follows:

	State		
Configuration item	**Open**	**Work in Progress**	**Total**
(empty)	3	2	5
FileServerFloor2	1	0	1
Sales Force Automation	1	0	1
Total	5	2	7

4. Click **Make Gauge** and then **Add to Homepage**. Select a particular dropzone. The report will now appear on the last homepage viewed.



References

[1] https://docs.servicenow.com/bundle/jakarta-performance-analytics-and-reporting/page/use/reporting/reference/reporting-landing-page.html

Plugins

Best Practice - Bulk CI Changes Plugin|Bulk CI Changes

Overview

This plugin provides functionality to record a single change proposal that will be linked to all affected CIs. The fields **CI Class** and **Proposed Change** were added to the Change Request form with this plugin.

This plugin is useful for change processes that use:

- The Proposed Change feature to update CI data.
- Change requests based on a single CI Class.
- The Affected CI related list to track impacted CIs.

Plugin Contents

Below are a list of the elements added by installation of the **Best Practice - Bulk CI Changes Plugin**.

Database Table Structure Changes

change_request (Change Request) Table

Configure the form to add these fields:

Field	Input Value
ci_class	A **Table Name** type field to identify the class of CIs that the change applies to.
proposed_change	A template value type field to capture the field change values that could be applied to each affected CI.

task_ci (Affected CIs) Table

Field	Input Value
ci_item	Adds a reference qualifier to filter the **Affected CI** lookup to the CI Class defined in the change request.

Business Rules

The following business rules are added when the plugin is installed:

Business Rule	Table	Description
Clear Proposed Change on Class Change	change_request	Runs when the **ci_class** field is changed. Clears the proposed change field value.
Delete Affected CIs on Class Change	change_request	Runs when the **ci_class** field is changed. Deletes *task_ci* records since they may no longer match the **ci_class**.
Deploy Proposed Changes to CIs	change_request	Runs on update when proposed change value changes. Copies the current proposed change from the change request to the *task_ci* record.
Deploy Proposed Changes to new CIs	task_ci	Runs on all inserts where task class is **change_request**. Copies the current proposed change from the change request to the *task_ci* record.
affectedCIClassFilter	global	A special global rule script called by the new attribute on the **task_ci.ci_item** field to filter the lookup of CIs based on the **change_request.ci_class** field value.

Client Scripts

The following client script is added when the plugin is installed:

Client Script	Table	Description
Alert on Change of CI Class	change_request	Triggered by a change in the **ci_class** field. Alerts the user that the affected CI's will be deleted, then forces a form Submit so the business rules run.

UI Actions

The following UI actions are disabled since they are designed to work with individual affected CI propose changes.

- **Save Proposed Changes**
- **Propose Changes**

UI Policies

The following UI policy is included in the plugin:

Table	Conditions	Description
change_request	ci_class=^EQ	Hide Proposed Change if CI Class is blank.

Using the Bulk CI Changes Plugin

Perform the following procedures on the change request form:

1. Configure the change request form as follows:

   a. Add the **CI Class** and **Proposed Change** fields, if they are not already visible.

      A UI policy hides the **Proposed Change** field until a **CI class** is selected.

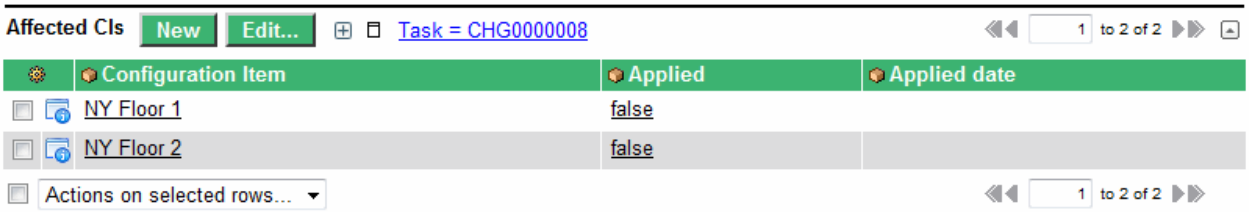
   b. Add the *Affected CI* related list, if it doesn't already exist.

   c. Remove the **Configuration Item** field from the form, since all CIs should be tracked through the affected CI related list.
2. Select a **CI class**.

   The form will submit and save if all the mandatory fields are completed.
3. Click **Edit** on the affected CI related list.

The selection is filtered to show only CIs from the selected class.

4. Add the CIs that are involved in the change.

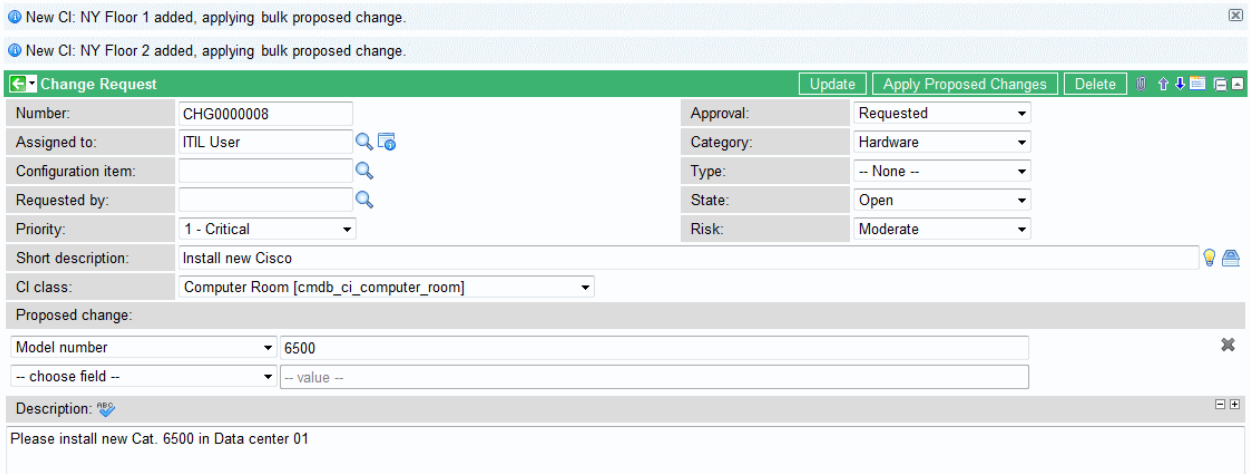


5. Populated the **Proposed change** field with the intended changes.

   This process is that same as proposing a single affected CI change. Whenever the **Proposed change** field is modified or affected CIs are added, the saved changes are linked to all affected CIs. There are no changes to how applying proposed changes works.

6. Update the record.

   The resulting changes are listed at the top of the form. These messages appear on the form only if there are CIs listed in the *Affected CIs* list.



Activating the Plugin

Click the plus to expand instructions for activating a plugin.

If you have the admin role, use the following steps to activate the plugin.

1. Navigate to **System Definition > Plugins**.
2. Right-click the plugin name on the list and select **Activate/Upgrade**.

   If the plugin depends on other plugins, these plugins are listed along with their activation status.

3. [Optional] If available, select the **Load demo data** check box.

   Some plugins include demo data—sample records that are designed to illustrate plugin features for common use cases. Loading demo data is a good policy when you first activate the plugin on a development or test instance. You can load demo data after the plugin is activated by repeating this process and selecting the check box.

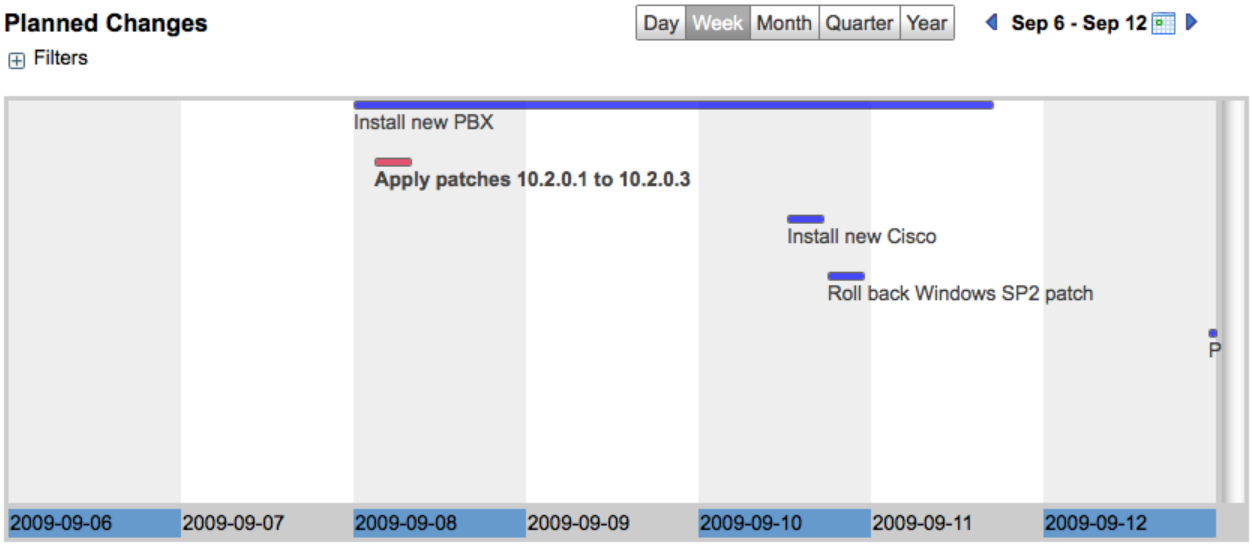
4. Click **Activate**.

Maintenance Schedules

Note: *This article applies to Fuji. For more current information, see Create Blackout and Maintenance Schedules* [1] *at* http://docs.servicenow.com The ServiceNow Wiki is no longer being updated. Please refer to http://docs.servicenow.com for the latest product documentation.

Overview

The Maintenance Schedules feature enables you to display scheduled maintenance for configuration items in a timeline format, and marks change requests that fall outside the allowed maintenance period. Planned maintenance is displayed in a calendar format that can be configured to show daily to yearly views. Maintenance activity is represented by a *span*, which is displayed as a horizontal bar on the timeline and can appear in any color. Spans for requested changes that occur outside their allowed maintenance periods appear in red on the timeline.



Change Collision

For instances with the Collision Detector activated, blackout and maintenance schedules have condition fields. This allows blackout or maintenance schedules to be applied only to CIs that match a certain set of conditions.

Creating Maintenance Schedules

A **Weekends** schedule is included in the base system. This schedule runs from 00:00:00 on Saturday until 23:59:59 on Sunday and repeats every week. It may be helpful to refer to this example when setting up your own custom maintenance schedules.

These properties are available to help define schedule entries: `glide.schedules.repeat_nth` and `glide.schedules.fifth`. For details about the properties, see Available System Properties..

To create additional maintenance schedules:

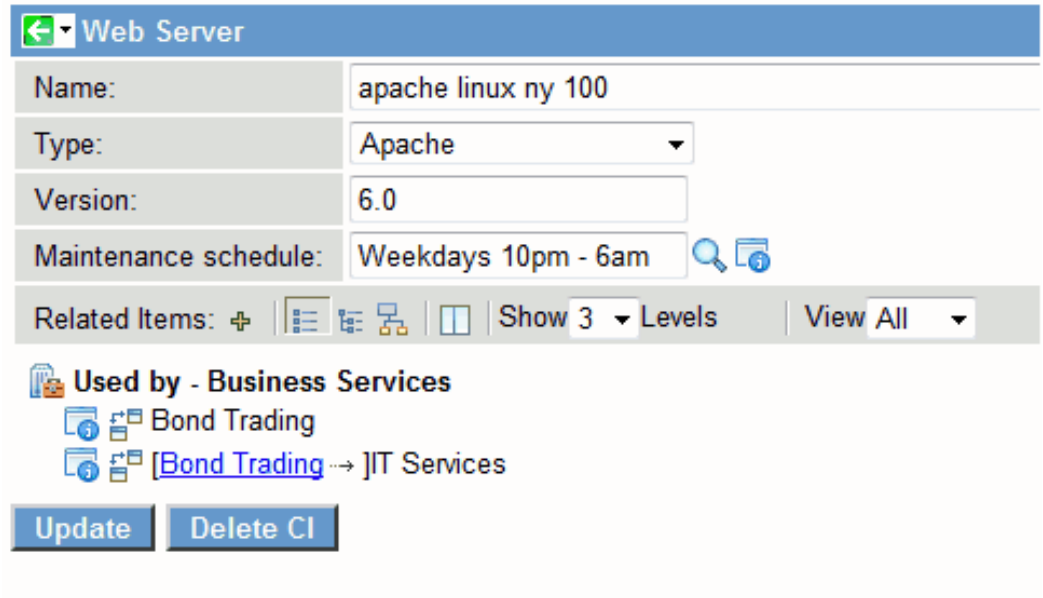
1. Navigate to **Change > Maintenance Windows** and click **New** to create a new schedule.
2. Type a unique name for the schedule.
3. Select the appropriate time zone.

4. Save the schedule form and then click **New** in the Schedule Entries related list to add times when maintenance should take place. For details on how to create schedule entries, see Schedules.
5. Use the **Condition** field to apply the maintenance schedule only to particular CIs, if desired.
6. Repeat this procedure for each different maintenance schedule needed.

When you create a new maintenance schedule from a change record, it is created in the `cmn_schedule` table. When you create a maintenance window, it is created in the `cmn_schedule_maintenance` table, which is the table that is used for checking for schedule conflicts. When selecting a maintenance window in the **CI Maintenance Schedule** field, only schedules that were added using maintenance windows are evaluated for conflicts in change.

Assigning Maintenance Schedules to Configuration Items

1. Locate the configuration item record by navigating to the appropriate module under the Configuration application menu. For example, if you want to locate a web server, look under **Configuration > Web Servers**.
2. If you have not already done so, personalize the form to add the **Maintenance schedule** field.
3. Set the value of the **Maintenance schedule** field to one of the configured maintenance schedules.
   You will be able to choose schedules of the type **maintenance**.
4. Save the CI.



Outside Maintenance Schedule on Change Requests

Personalize the change request form and add the **Outside maintenance schedule** field. This check box is informational and indicates whether planned scheduling (consisting of the planned start date and planned end date) for the change occurs outside the maintenance window. The instance sets this value and disregards any user changes to this checkbox.

This field is set to **true** if:

- The CI associated with the change, if any, is compared to the maintenance schedule, if any, and the planned dates for the change occur outside the maintenance schedule.
- Likewise Affected CIs, if any, for the change request will be checked against their own maintenance schedules, if any.

Note: *Only the maintenance window for the primary CI or affected CIs are checked; the upstream/downstream CIs are not checked.*

When you save a change request that is outside the maintenance schedule, a warning appears for each CI (primary or affected) that is outside the maintenance schedule, if the change request was previously not marked as outside the maintenance schedule.

Change Schedule

Bring up the change schedule timeline by navigating to **Change > Change Schedule**. The timeline shows all active change requests by planned start and end times. Any change requests that are outside the maintenance window are colored red on the display. Change requests that have an approval status of **Rejected** are displayed in gray.

For instructions on using timelines, see Using Timelines.

Updating Maintenance Schedules

When you update maintenance schedule entries, check the **Outside maintenance schedule** field for change requests that might have been forced outside the maintenance period and that require an update. An *async* business rule runs on the Schedule Entry table when maintenance schedules are updated to verify the **Outside maintenance schedule** field on change requests that have configuration items (primary or affected) that use that schedule. As this is done asynchronously (so as not to keep the user waiting), specific warning messages are not issued for each change request as they are when change requests are individually updated. However, if any CIs use the maintenance schedule, a single message is issued indicating that this verification will occur.

Transferring Maintenance Schedules between Instances

You must manually export maintenance schedule records because the system does not track them in update sets. See Exporting and Importing XML Files.

To transfer maintenance schedules between instances:

1. Export records from the following tables as XML files.
   - Maintenance Schedule [cmn_schedule_maintenance]
   - Schedule Entry [cmn_schedule_span].
2. Import the records on the target instance.

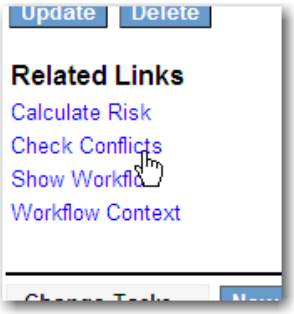
Change Management Collision Detector Plugin

Overview

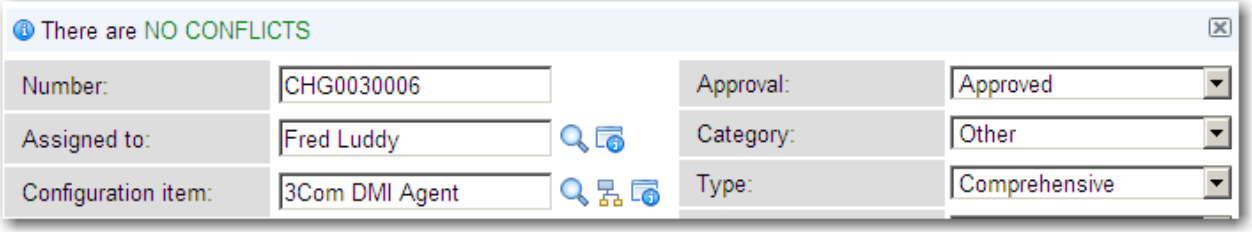
The Change Management Collision Detector plugin provides the ability to detect whether planned changes conflict with other changes or have other scheduling issues. When the Collision Detector is run, any issues which are detected are added to the new **Conflicts** related list with an explanation as to what issues were detected.

Detecting Conflicts

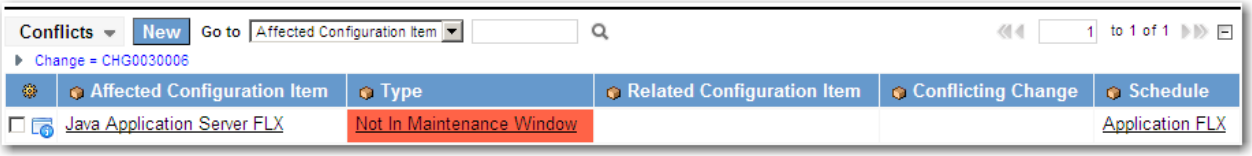
To detect collisions, open a change request and use the related link **Check Conflicts**. The itil role is required.



If there are no collisions, an alert displays **NO CONFLICTS** at the top:



If there are collisions, they are added to the related list **Conflicts** (you may need to configure the form to add this related list):



The following information is captured in the **Conflicts** related list:

Field	Input Value
Change	A reference to the scheduled change that has a conflict.
Affected Configuration Item	The affected CI associated with the change.
Conflicting Change	The change that is in conflict with the scheduled change, if any.
Related Configuration Item	The parent CI or child CI of the current CI, if it has caused a conflict.
Schedule	The maintenance window or blackout window that is causing the conflict, if any.

Type	The issue that caused the conflict. One of: • **CI Already Scheduled** • **Parent CI Already Scheduled** • **Child CI Already Scheduled** • **Not in Maintenance Window** • **Parent Not In Maintenance Window** • **Child Not In Maintenance Window** • **Blackout**
Last Checked	The last time the conflicts were checked. The default value is **now**. The **Last Checked** field is automatically updated.

Blackout Schedules

The Change Management Collision Detector plugin also introduces the concept of blackout schedules, times during which changes cannot be scheduled.

Blackout schedules are defined by navigating to **Change > Blackout Window**. They are defined the same as other Schedules but with the type field set to **blackout**.

Note: *A schedule containing blackout schedule entries cannot also contain maintenance schedule entries.*

Applying Blackout and Maintenance Schedules to Certain CIs

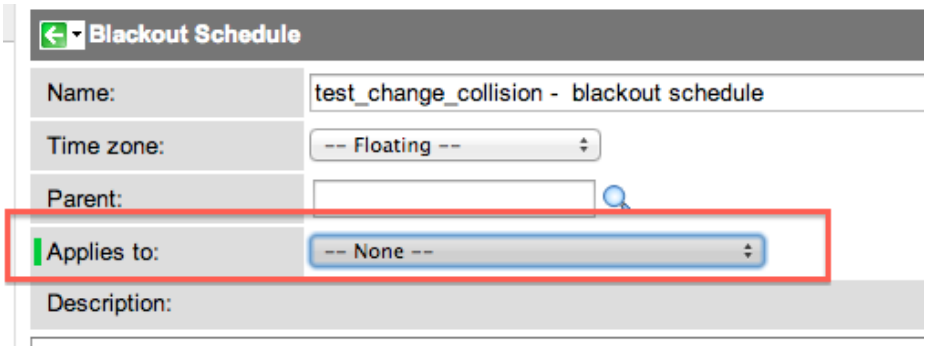
Both blackout and maintenance schedules are **Condition Schedules**, providing a **Condition** field which can be used to apply the blackout schedule only to CIs which match predetermined conditions.

For example, blackout schedules can be defined to apply only to CIs in a particular location or of a particular class.

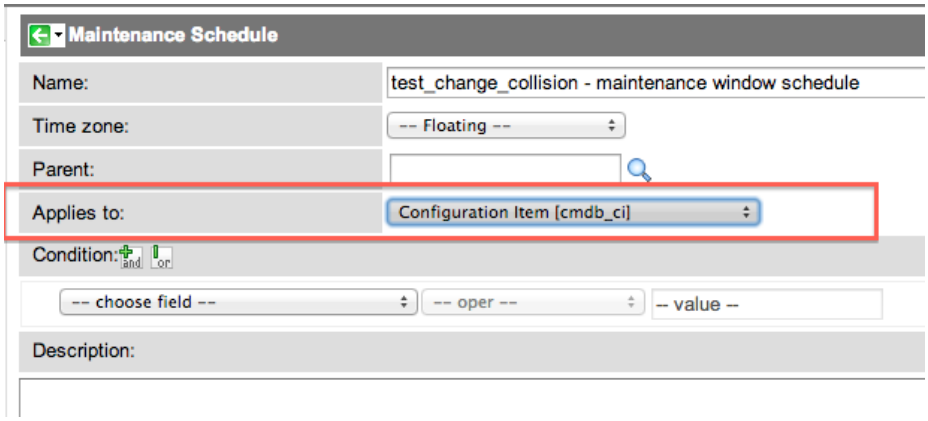
Note: *By default, the condition builder used when checking conflicts is not case sensitive, to ensure all conflicts are identified. To use case sensitive conditions, add a system property of type **boolean** called **change.conflict.filter.case_sensitive**. Set this property to **True** to support case sensitivity or to **False** to disable case sensitivity.*

Checking No CIs or All CIs

To check conflicts against no CIs, set the **Applies To** field to **None**. No conflicts will then be triggered.



To check conflicts against all CIs, select the top-level **cmdb_ci** item.



Installed Components

Database Table Structure

The following tables are added with the Change Management Collision Detector plugin:

Table	Description
Blackout Schedule [cmn_schedule_blackout]	A table extending cmn_schedule. Allows a schedule to be defined in which no changes are allowed to take place. The cmn_schedule_blackout.description column is available on the Schedule [cmn_schedule] table.
Conflict [conflict]	The conflicts which are detected by the collision detector.
Condition Schedules [cmn_schedule_condition]	Schedules with a condition field. Condition schedules apply a schedule only to CIs which match predetermined conditions.
Maintenance Schedules [cmn_schedule_maintenance]	Schedules defining maintenance for configuration items, including condition fields. See Maintenance Schedules Support for more details.

Properties

The following properties are accessible through the new module **Change > Administration > Conflict Properties**.

Property	sys_property Name	Description
Blackout Window Checking	change.conflict.blackout	When enabled, it checks to see if there are any blackout windows at the same time.
Change Already Scheduled	change.conflict.currentci	When enabled, it checks to see if there are other changes scheduled at the same time for the selected configuration item (or the configuration items in the Affected CI List if in "Advanced" mode).
Change in Maintenance Window for CI	change.conflict.currentwindow	When enabled, it performs a lookup on the currently selected configuration item (or the configuration items in the Affected CI List if in "Advanced" mode), get the associated maintenance window, then determine if the planned start and end dates fall within the maintenance window. You can customize any configuration item (CI), pull out the Maintenance Window field, create a new schedule, then place that schedule on the CI.
Mode	change.conflict.mode	Choices are: • **Basic:** When enabled, only checks only the current change request's CI against other change requests' CI and affected CIs. • **Advanced:** When enabled, checks both the current change request's CI and affected CIs against other change requests' CI and affected CIs.

Check Related Child CI Maintenance Window	change.conflict.relatedchildwindow	When enabled, this will perform a look up on the currently selected Configuration Item (or the Configuration Items in the Affected CI List if in "Advanced" mode), get the maintenance window of any related child Configuration Item's associated maintenance windows then determine if the planned start and end dates fall within the maintenance windows. You can customize any Configuration Item (CI) and pull out the Maintenance Window field, create a new schedule then place that schedule on any CI.
Check Related Parent CI Maintenance Window	change.conflict.relatedparentwindow	When enabled, it performs a lookup on the currently selected configuration item (or the configuration items in the Affected CI List if in "Advanced" mode), get the maintenance window of any related parent Configuration Item associated, then determine if the planned start and end dates fall within the maintenance windows. You can customize any configuration item (CI), pull out the Maintenance Window field, create a new schedule, then place that schedule on any CI.

Scripts

The following business rules are added to **sys_script**:

- **Add CI in affected CIs List** - When in Advanced mode, this checks whether the change request's CI is in the change request's Affected CI List. If not, it adds the CI to the list.

The following client scripts are added to **sys_script_client**:

- **Notify Conflict** - Notifies the user if a conflict exists.

The following script includes are added to **sys_script_include**:

- **ChangeConflict** - Class that holds conflicts' details.
- **ChangeCheckConflicts** - Class methods for Change Management Collision Detection. ChangeCheckConflicts.check() is the entry point from 'Change Conflicts' sys_ui_action
- **ChangeConflictHandler** - Class for handling change conflicts.
- **ChangeCollisionHelper** - Class for collision helper methods.

Change Collision API

Overview

The Change Management Collision Detector Plugin includes three Script Includes, each of which contain helper functions specific to the plugin, when scripting on the server side or using AJAX calls on the client.

The three script includes are:

- ChangeCollisionHelper
- ChangeConflict
- ChangeConflictHandler

ChangeCollisionHelper

Method Summary	
Return Object	**Details**
boolean	**isDateInCiMaintenanceWindows (`startDate, endDate, maintenanceWindow`)** **Checks if the time span defined by *startDate* and endDate *falls in the CI's maintenance window*** **Parameters:** `startDate` - The beginning date of a time span. `endDate` - The ending date of a time span. `maintenanceWindow` - A Configuration Item's sys_id. **Returns:** `boolean` - An array of CIs.
	getCiMaintenanceSchedule (`ci`) **Gets the Maintenance Schedule for a CI.** **Parameters:** `ci` - A Configuration Item's sys_id.
array	**getBlackoutsByDate (`startDate, enDate`)** **Gets any blackout that overlap the period defined by *startDate* and *endDate*.** **Parameters:** `startDate` - The beginning date of a time span. `enDate` - The ending date of a time span. **Returns:** `array` - An array of blackouts (blackoutId:stringSpan).

array changeIds	**getChangesWithAffectedCi (`ci, startDate, enDate`)** **Get changes scheduled in the timespan (defined by *startDate* and *endDate*) that have the given CI in their *Affected CIs* list.** **Parameters:** `ci` - A Configuration Item's sys_id. `startDate` - The beginning date of a time span. `enDate` - The ending date of a time span. **Returns:** `array changeIds` - An array of sys_ids for Change records.
array changeIds	**getChangesWithCi (`ci, startDate, enDate`)** **Get the changes that are in the timespan (*startDate*, *endDate*) and that are linked to the given CI.** **Parameters:** `ci` - A Configuration Item's sys_id. `startDate` - The beginning date of a time span. `enDate` - The ending date of a time span. **Returns:** `array changeIds` - An array of sys_ids for Change records.
array	**getAffectedCisByChangeId (`changeId`)** **Gets the *Affected CI* sys_ids for the given change.** **Parameters:** `changeId` - A Change Record's sys_id **Returns:** `array` - An array of sys_ids of Affected CIs.
	addCiToChangeAffectedCis (`ci, changeid`) **Add CI to the change's *Affected CI* list.** **Parameters:** `ci` - A Configuration Item's sys_id. `changeid` - A Change Record's sys_id.
boolean	**isCiInAffectedCis (`ci, changeid`)** **Check if an CI is already in the change's *Affected CIs* list.** **Parameters:** `ci` - A Configuration Item's sys_id. `changeid` - A Change Record's sys_id. **Returns:** `boolean` - True if the CI already in the change's *Affected CIs* list, false if not.

array	**getDependants (`ci`)** **Get all the CIs that depend on the given CI.** **Parameters:** `ci` - A Configuration Item's sys_id. **Returns:** `array` - An array of CIs.
array	**getDependencies (`ci`)** **Get all the CIs that the given CI depends on.** **Parameters:** `ci` - A Configuration Item's sys_id. **Returns:** `array` - An array of CIs.

ChangeConflict

Method Summary

Return Object	Details
	initialize (`configurationItemId, changeId, type`) **Initializes a ChangeConflict object.** **Parameters:** `configurationItemId` - A Configuration Item's sys_id. `changeId` - A Change Request's sys_id. `type` - A value from the conflict's Type choice list. **Example:** `var cc = new ChangeConflict(82992eb60ad337024fbb6d06a866c636,8799385b0a0a2c3e14c041a7f5414266,not_in_maintenance_window);`
string	**toString ()** **Returns a String representation of the conflict** **Returns:** `string` - The string representation of the conflict.

ChangeConflictHandler

Method Summary	
Return Object	**Details**
	initialize () **Initializes a Change Conflict Container array.**
	addChangeConflict (`changeConflict`) **Adds the Change Conflict to a Change Conflict Container.** **Parameters:** `changeConflict` - The sys_id of a Change Conflict.
changeConflictContainer	**getConflicts** () **Gets an array of Change Conflicts from a Change Conflict Container.** **Returns:** `changeConflictContainer` - An array of Change Conflicts.
	saveConflicts () **Writes out the Change Conflicts in a Change Conflict Container array to individual Change Conflict records.**
	deleteConflictsByChangeId (`changeID`) **Deletes conflicts that are associated with the same Change Request (by sys_id).** **Parameters:** `changeID` - A sys_id of a Change Request.

Change Risk Assessment Plugin

Overview

Change Management Risk Assessment provides a flexible way to capture information from the end user, in order to derive the risk of the associated change. Libraries of questions can be used to derive the risk of a change based on criteria contained within the change record. For example, a different set of questions could be set for a hardware change versus a software change.

The assessment uses a weighted score approach for each question. The overall score for an assessment is evaluated against thresholds to determine the risk of the change.

Activating Change Management Risk Assessment

Click the plus to expand instructions for activating a plugin.

If you have the admin role, use the following steps to activate the plugin.

1. Navigate to **System Definition > Plugins**.
2. Right-click the plugin name on the list and select **Activate/Upgrade**.

   If the plugin depends on other plugins, these plugins are listed along with their activation status.
3. [Optional] If available, select the **Load demo data** check box.

   Some plugins include demo data—sample records that are designed to illustrate plugin features for common use cases. Loading demo data is a good policy when you first activate the plugin on a development or test instance. You can load demo data after the plugin is activated by repeating this process and selecting the check box.
4. Click **Activate**.

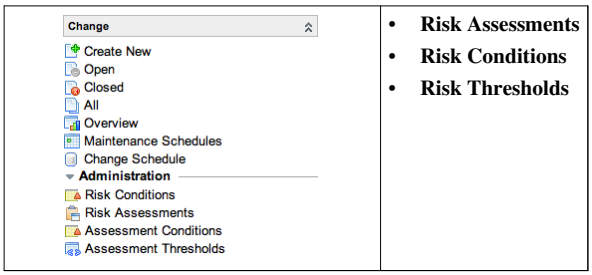
Dependencies

The following plugins are activated with Change Management Risk Assessment:

- Assessment Components
- Best Practice - Change Risk Calculator
- Best Practice - Task Survey Management

Applications and Modules

Change Management Risk Assessment adds the following modules to the Change application.



Installed with Change Management Risk Assessment

Demo Data

These risk assessments are provided as demo data:

- **Hardware Risk Assessment**
- **Software Risk Assessment**

Defining Risk Assessment

A risk assessment presents the user with a series of questions to determine risk. The composite weighted score that is derived from the user's answers is used to calculate risk, based on the thresholds associated to the risk assessment.

To define a new risk assessment, navigate to **Change > Risk Assessment** and click **New**. Populate the following fields:

Field	Input Value
Name	A name for the risk assessment, which will be displayed to the user.
Introduction	If desired, an introduction for the user.

The **Assessment Questions** related list determines what questions the user will answer.

Field	Input Value
Question	The question asked of the user. **Note:** *Mandatory choice list questions are not supported, use a select list instead.*
Weight	Determines the weighting applied to each question. This will be multiplied by the score of the answer to calculate the weighted score.
Related List: Assessment Questions > Assessment Question Choices	
Order	Determines where in the question the choices will be displayed. Ordered from lowest order to highest.
Value	The choice as presented to the user.
Score	The score of the choice.

The **Assessment Thresholds** related list determines the risk that will be set depending on the calculated composite score for any assessment that is taken. The composite score is the sum of all weighted scores for the assessment. Take care to ensure that the thresholds are set correctly based on the questions and answer combinations.

Field	Input Value
Score Greater Than	If the score totalled from all of the user's answers is greater than this number, the risk in the **Risk** field will be applied to the change.
Risk	The risk to apply if the threshold is met.

The **Assessment Conditions** related list determines which risk assessment is attached to each change. When the system evaluates which assessment to attach, it uses the first matching one. When defining multiple questionnaires, take care to ensure that the conditions result in the correct assessments being attached. Keep the conditions simple and mutually exclusive to make this easier to understand and maintain.

If a default questionnaire is required, simply add a condition record with no conditions and set the order to a suitably high number to ensure that other conditions will be evaluated first.

Field	Input Value
Active	If the check box is selected, this condition will be evaluated.
Condition	A condition builder field to determine what changes will use this Risk Assessment.
Description	A description of this condition.
Order	If multiple conditions apply, the risk assessment with the lowest order will be used.
Table	The table on which the risk assessment will be run. **Note:** if this risk assessment is being used on the **Change** table, the table needs to be **Change [change_request]**.

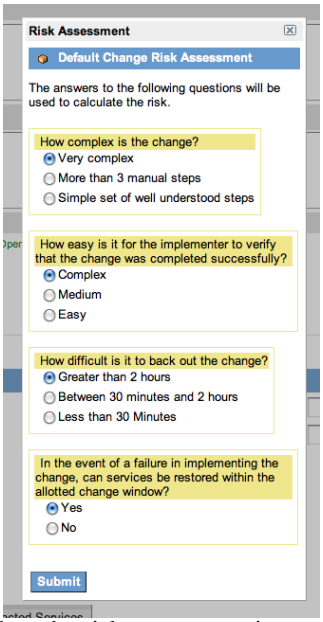
Assessing Risk

Once risk assessments are defined, an ITIL user can assess risk.

1. On any submitted change form, click the **Fill Out Risk Assessment** related link:

   Related Links
   Fill Out Risk Assessment

2. The ITIL user is presented with the most appropriate risk assessment, based on the definition. Complete the risk assessment.



3. When the risk assessment is complete, click the **Execute Risk Calculation** related link to calculate the risk based on the assessment:

Related Links
Fill Out Risk Assessment
Execute Risk Calculation

4. When the form reloads, information messages at the top describe the results of the calculation:

Risk assessment evaluated. Risk: Very High

Risk Condition applied: **Insufficient lead time**; Risk: Very High; Impact unchanged

Risk set to: Very High

Viewing Risk Assessment Responses

To view the assessment responses, administrators can add the **Task Assessment > Task** related list to the Change Request form. The related list displays risk assessments associated with the change request. Click the reference icon for an assessment to view the responses.

Alternatively, administrators or users with the survey_admin or survey_reader role can view responses in the Survey application:

1. Navigate to **Survey > Survey Responses**.
2. Filter the list accordingly to view the responses you want to see.

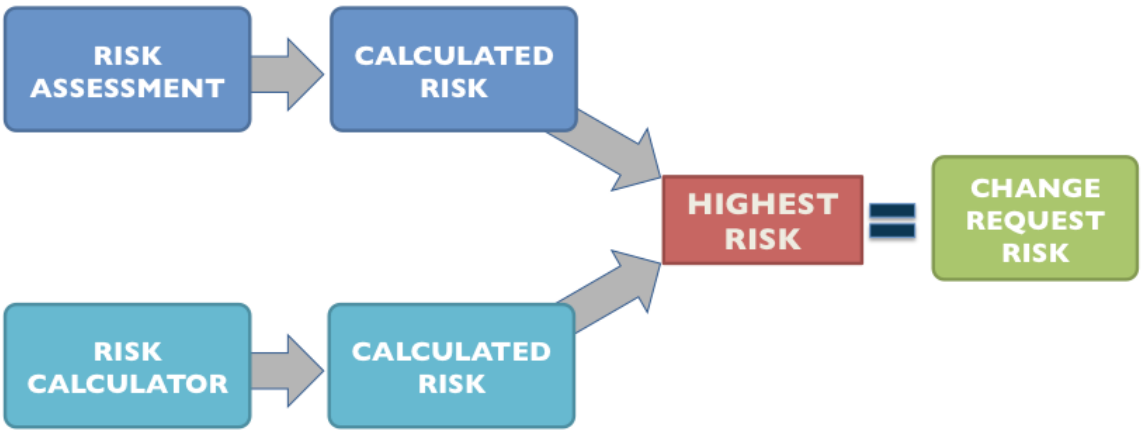
   You may want to filter by **Instance**. Survey instances are individual assessments and are distinguished by the date and time at which they are taken.

Using Risk Assessment and Risk Calculation

There are two methods within ServiceNow to calculate the risk of a change: Change Risk Calculator (activated by default) and Change Management Risk Assessment (an optional plugin).

- Change Management Risk Assessment uses information provided by the end user to assess a risk value.
- Change Risk Calculation uses predefined properties and conditions to calculate a risk value.

These methods can be used individually, or together, depending on your own procedures and requirements. If both methods are used together, the highest risk value from both methods is always selected:



Note: *If you have both Risk Assessment and Risk Calculation active, but only wish to use one of these methods, then simply remove all conditions for the method you do not want to use.*

Article Sources and Contributors

Creating a Record Producer *Source*: http://wiki.servicenow.com/index.php?oldid=100123 *Contributors*: Davida.hughes, Guy.yedwab, Joseph.messerschmidt, Steven.wood

Defining an Inbound Email Action *Source*: http://wiki.servicenow.com/index.php?oldid=134822 *Contributors*: Davida.hughes, G.yedwab, Guy.yedwab, Joseph.messerschmidt, Neola, Steven.wood, Vaughn.romero

Defining an Assignment Rule *Source*: http://wiki.servicenow.com/index.php?oldid=100159 *Contributors*: Davida.hughes, Guy.yedwab, Joseph.messerschmidt

Managing CI Changes *Source*: http://wiki.servicenow.com/index.php?oldid=250701 *Contributors*: Anat.kerry, CapaJC, Dan.sherwin, Davida.hughes, Dawn.bunting, Fuji.publishing.user, G.yedwab, Guy.yedwab, John.ramos, Joseph.messerschmidt, Myla.jordan, Steven.wood, Vhearne

Defining Maintenance Schedules *Source*: http://wiki.servicenow.com/index.php?oldid=89870 *Contributors*: Anat.kerry, Cheryl.dolan, Davida.hughes, Emily.partridge, Eric.jacobson, G.yedwab, Guy.yedwab, John.ramos, Joseph.messerschmidt, Michael.randall, Rachel.sienko, Steven.wood, Suzanne.smith, Vaughn.romero

Calculating Risk *Source*: http://wiki.servicenow.com/index.php?oldid=89834 *Contributors*: CapaJC, Cheryl.dolan, David.Bailey, Davida.hughes, Eric.schroeder, G.yedwab, Guy.yedwab, John.ramos, Joseph.messerschmidt, Mark.stanger, Neola, Steven.wood, Vhearne

Creating an Execution Plan *Source*: http://wiki.servicenow.com/index.php?oldid=244600 *Contributors*: Boonetp, CapaJC, Davida.hughes, Fuji.publishing.user, G.yedwab, Guy.yedwab, Joseph.messerschmidt, Pat.Casey, Rachel.sienko, Steven.wood, Vhearne

Creating a Template for Change Tasks *Source*: http://wiki.servicenow.com/index.php?oldid=250423 *Contributors*: CapaJC, Danijel.stanojevic, David.Bailey, Davida.hughes, Emily.partridge, Fuji.publishing.user, G.yedwab, Guy.yedwab, John.ramos, Joseph.messerschmidt, Phillip.salzman, Steven.wood, Vaughn.romero, Vhearne

Change Management Workflows Plugin *Source*: http://wiki.servicenow.com/index.php?oldid=89836 *Contributors*: Davida.hughes, Emily.partridge, G.yedwab, Guy.yedwab, John.ramos, Joseph.messerschmidt, Steven.wood

Defining a Workflow *Source*: http://wiki.servicenow.com/index.php?oldid=250486 *Contributors*: Davida.hughes, Fuji.publishing.user, G.yedwab, Guy.yedwab, Ishrath.razvi, John.ramos, Joseph.messerschmidt, Rachel.sienko, Steven.wood, Virginia.kelley

Part One *Source*: http://wiki.servicenow.com/index.php?oldid=244818 *Contributors*: Davida.hughes, Fuji.publishing.user, G.yedwab, Guy.yedwab, Ishrath.razvi, Jennifer.ball, Joe.Westrich, Joseph.messerschmidt, Julie.phaviseth, Peter.smith, Publishing.user, Rachel.sienko, Russ.sarbora, Steven.wood, Vaughn.romero

Part Two *Source*: http://wiki.servicenow.com/index.php?oldid=239954 *Contributors*: Davida.hughes, Fuji.publishing.user, G.yedwab, Guy.yedwab, Ishrath.razvi, Joe.Westrich, Joseph.messerschmidt, Rachel.sienko, Steven.wood

Reporting *Source*: http://wiki.servicenow.com/index.php?oldid=250814 *Contributors*: Davida.hughes, Guy.yedwab, John.ramos, Joseph.messerschmidt, Michael.randall

Best Practice - Bulk CI Changes Plugin|Bulk CI Changes *Source*: http://wiki.servicenow.com/index.php?oldid=89832 *Contributors*: Anat.kerry, David.Bailey, Davida.hughes, Emily.partridge, Fuji.publishing.user, G.yedwab, Guy.yedwab, John.roberts, Joseph.messerschmidt, Mark.stanger, Michael.randall, Rachel.sienko, Steven.wood, Suzanne.smith, Wallymarx

Maintenance Schedules *Source*: http://wiki.servicenow.com/index.php?oldid=89870 *Contributors*: Anat.kerry, Cheryl.dolan, Davida.hughes, Emily.partridge, Eric.jacobson, G.yedwab, Guy.yedwab, John.ramos, Joseph.messerschmidt, Michael.randall, Rachel.sienko, Steven.wood, Suzanne.smith, Vaughn.romero

Change Management Collision Detector Plugin *Source*: http://wiki.servicenow.com/index.php?oldid=70890 *Contributors*: Cheryl.dolan, David.Bailey, Davida.hughes, Emily.partridge, Fuji.publishing.user, G.yedwab, Guy.yedwab, Joseph.messerschmidt, Maneesha.Nanda, Michael.randall, Pat.Casey, Rachel.sienko, Steven.wood, Suzanne.smith, Vaughn.romero

Change Collision API *Source*: http://wiki.servicenow.com/index.php?oldid=248190 *Contributors*: David.Bailey, Davida.hughes, Emily.partridge, G.yedwab, Guy.yedwab, Joseph.messerschmidt, Julie.komrosky, Neola, Rachel.sienko

Change Risk Assessment Plugin *Source*: http://wiki.servicenow.com/index.php?oldid=82835 *Contributors*: David.Bailey, Emily.partridge, Guy.yedwab, Joseph.messerschmidt, Rachel.sienko, Ross.weir, Roy.lagemann, Sydney.nickell

Image Sources, Licenses and Contributors

Image:Role.gif *Source*: http://wiki.servicenow.com/index.php?title=File:Role.gif *License*: unknown *Contributors*: CapaJC

Image:ChngRecordProducer1.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChngRecordProducer1.png *License*: unknown *Contributors*: Guy.yedwab

Image:ChngRecordProducer2.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChngRecordProducer2.png *License*: unknown *Contributors*: Guy.yedwab

Image:ChgRecordProducer3.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChgRecordProducer3.png *License*: unknown *Contributors*: Guy.yedwab

Image:DBChangeAssgn.png *Source*: http://wiki.servicenow.com/index.php?title=File:DBChangeAssgn.png *License*: unknown *Contributors*: Guy.yedwab

Image:DBChangemAssgn2.png *Source*: http://wiki.servicenow.com/index.php?title=File:DBChangemAssgn2.png *License*: unknown *Contributors*: Guy.yedwab

Image:DBChangeAssgn3.png *Source*: http://wiki.servicenow.com/index.php?title=File:DBChangeAssgn3.png *License*: unknown *Contributors*: Guy.yedwab

Image:Warning.gif *Source*: http://wiki.servicenow.com/index.php?title=File:Warning.gif *License*: unknown *Contributors*: CapaJC

Image:Basline schedule.jpg *Source*: http://wiki.servicenow.com/index.php?title=File:Basline_schedule.jpg *License*: unknown *Contributors*: Dan.sherwin

Image:BSMicon.png *Source*: http://wiki.servicenow.com/index.php?title=File:BSMicon.png *License*: unknown *Contributors*: Guy.yedwab

Image:BSM_Change_Map.png *Source*: http://wiki.servicenow.com/index.php?title=File:BSM_Change_Map.png *License*: unknown *Contributors*: Dawn.bunting

Image:BSM_Change_Task_Info.png *Source*: http://wiki.servicenow.com/index.php?title=File:BSM_Change_Task_Info.png *License*: unknown *Contributors*: Dawn.bunting

Image:BSM_Change_Affected_CI_Option.png *Source*: http://wiki.servicenow.com/index.php?title=File:BSM_Change_Affected_CI_Option.png *License*: unknown *Contributors*: Dawn.bunting

Image:BSM_Change_Affected_CI_List.png *Source*: http://wiki.servicenow.com/index.php?title=File:BSM_Change_Affected_CI_List.png *License*: unknown *Contributors*: Dawn.bunting

Image:BSMChange.png *Source*: http://wiki.servicenow.com/index.php?title=File:BSMChange.png *License*: unknown *Contributors*: Guy.yedwab

Image:RelatedIssues.png *Source*: http://wiki.servicenow.com/index.php?title=File:RelatedIssues.png *License*: unknown *Contributors*: Guy.yedwab

Image:Planning Timeline.png *Source*: http://wiki.servicenow.com/index.php?title=File:Planning_Timeline.png *License*: unknown *Contributors*: Eric.jacobson

Image:Maintenance Schedule CI.gif *Source*: http://wiki.servicenow.com/index.php?title=File:Maintenance_Schedule_CI.gif *License*: unknown *Contributors*: Steven.wood

Image:Risk calc2.jpg *Source*: http://wiki.servicenow.com/index.php?title=File:Risk_calc2.jpg *License*: unknown *Contributors*: Eric.schroeder

Image:Risk calc3.jpg *Source*: http://wiki.servicenow.com/index.php?title=File:Risk_calc3.jpg *License*: unknown *Contributors*: Eric.schroeder

Image:Risk calc4.jpg *Source*: http://wiki.servicenow.com/index.php?title=File:Risk_calc4.jpg *License*: unknown *Contributors*: Eric.schroeder

Image:Risk calc5.jpg *Source*: http://wiki.servicenow.com/index.php?title=File:Risk_calc5.jpg *License*: unknown *Contributors*: Eric.schroeder

Image:Risk calc10.jpg *Source*: http://wiki.servicenow.com/index.php?title=File:Risk_calc10.jpg *License*: unknown *Contributors*: Eric.schroeder

Image:Risk calc6.jpg *Source*: http://wiki.servicenow.com/index.php?title=File:Risk_calc6.jpg *License*: unknown *Contributors*: Eric.schroeder

Image:Risk calc8.jpg *Source*: http://wiki.servicenow.com/index.php?title=File:Risk_calc8.jpg *License*: unknown *Contributors*: Eric.schroeder

Image:Risk calc9.jpg *Source*: http://wiki.servicenow.com/index.php?title=File:Risk_calc9.jpg *License*: unknown *Contributors*: Eric.schroeder

Image:Risk_Assessment_and_Calculation.png *Source*: http://wiki.servicenow.com/index.php?title=File:Risk_Assessment_and_Calculation.png *License*: unknown *Contributors*: David.Bailey

Image:Execution plan detail.png *Source*: http://wiki.servicenow.com/index.php?title=File:Execution_plan_detail.png *License*: unknown *Contributors*: CapaJC, Pat.Casey

Image:Execution task.png *Source*: http://wiki.servicenow.com/index.php?title=File:Execution_task.png *License*: unknown *Contributors*: CapaJC, Pat.Casey

Image:Change with running tasks.png *Source*: http://wiki.servicenow.com/index.php?title=File:Change_with_running_tasks.png *License*: unknown *Contributors*: CapaJC, Pat.Casey

Image:Change with tasks2.png *Source*: http://wiki.servicenow.com/index.php?title=File:Change_with_tasks2.png *License*: unknown *Contributors*: CapaJC, Pat.Casey

Image:ChangeTemplate1.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChangeTemplate1.png *License*: unknown *Contributors*: CapaJC, Guy.yedwab

Image:ChangeTemplate2.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChangeTemplate2.png *License*: unknown *Contributors*: CapaJC, Guy.yedwab

Image:ChangeTemplate3.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChangeTemplate3.png *License*: unknown *Contributors*: CapaJC, Guy.yedwab

Image:ChangeTemplate4.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChangeTemplate4.png *License*: unknown *Contributors*: CapaJC, Guy.yedwab

Image:NewChangeTemplate.png *Source*: http://wiki.servicenow.com/index.php?title=File:NewChangeTemplate.png *License*: unknown *Contributors*: Guy.yedwab

Image:NewChangeRecord.png *Source*: http://wiki.servicenow.com/index.php?title=File:NewChangeRecord.png *License*: unknown *Contributors*: Guy.yedwab

Image:Routine Change.png *Source*: http://wiki.servicenow.com/index.php?title=File:Routine_Change.png *License*: unknown *Contributors*: Fuji.publishing.user, Guy.yedwab

Image:Comprehensive Change.png *Source*: http://wiki.servicenow.com/index.php?title=File:Comprehensive_Change.png *License*: unknown *Contributors*: Guy.yedwab

Image:Emergency Change.png *Source*: http://wiki.servicenow.com/index.php?title=File:Emergency_Change.png *License*: unknown *Contributors*: Guy.yedwab

Image:ChgWorkflowForm.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChgWorkflowForm.png *License*: unknown *Contributors*: Guy.yedwab

Image:ChgWorkflowForm01.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChgWorkflowForm01.png *License*: unknown *Contributors*: Guy.yedwab

Image:ChgWorkflowForm1.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChgWorkflowForm1.png *License*: unknown *Contributors*: Guy.yedwab, Steven.wood

Image:ChgWorkflowForm2.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChgWorkflowForm2.png *License*: unknown *Contributors*: Guy.yedwab

Image:ChgWorkflowForm3.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChgWorkflowForm3.png *License*: unknown *Contributors*: Guy.yedwab, Steven.wood

Image:ChgWorkflowForm5.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChgWorkflowForm5.png *License*: unknown *Contributors*: Guy.yedwab, Steven.wood

Image:ChgWorkflowForm7.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChgWorkflowForm7.png *License*: unknown *Contributors*: Fuji.publishing.user, Guy.yedwab, Steven.wood

Image:ChgWorkflowForm6.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChgWorkflowForm6.png *License*: unknown *Contributors*: Guy.yedwab

Image:ChgWorkflowExample.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChgWorkflowExample.png *License*: unknown *Contributors*: Guy.yedwab, Steven.wood

Image:CaseWkflw1.png *Source*: http://wiki.servicenow.com/index.php?title=File:CaseWkflw1.png *License*: unknown *Contributors*: Fuji.publishing.user, Guy.yedwab, Publishing.user, Steven.wood

Image:CaseWkflw2.png *Source*: http://wiki.servicenow.com/index.php?title=File:CaseWkflw2.png *License*: unknown *Contributors*: Guy.yedwab, Publishing.user

Image:CaseWkflw3.png *Source*: http://wiki.servicenow.com/index.php?title=File:CaseWkflw3.png *License*: unknown *Contributors*: Guy.yedwab, Publishing.user

Image:CaseWkflw4.png *Source*: http://wiki.servicenow.com/index.php?title=File:CaseWkflw4.png *License*: unknown *Contributors*: Guy.yedwab, Publishing.user

Image:CaseWkflw5.png *Source*: http://wiki.servicenow.com/index.php?title=File:CaseWkflw5.png *License*: unknown *Contributors*: Guy.yedwab, Publishing.user

Image:CaseWkflw7.png *Source*: http://wiki.servicenow.com/index.php?title=File:CaseWkflw7.png *License*: unknown *Contributors*: Guy.yedwab, Publishing.user

Image:CaseWkflw6.png *Source*: http://wiki.servicenow.com/index.php?title=File:CaseWkflw6.png *License*: unknown *Contributors*: Guy.yedwab, Publishing.user

Image:CaseWkflw8.png *Source*: http://wiki.servicenow.com/index.php?title=File:CaseWkflw8.png *License*: unknown *Contributors*: Guy.yedwab

Image:CaseWkflw9.png *Source*: http://wiki.servicenow.com/index.php?title=File:CaseWkflw9.png *License*: unknown *Contributors*: Guy.yedwab

Image:Picker.png *Source*: http://wiki.servicenow.com/index.php?title=File:Picker.png *License*: unknown *Contributors*: G.yedwab

Image:CaseWkflw10.png *Source*: http://wiki.servicenow.com/index.php?title=File:CaseWkflw10.png *License*: unknown *Contributors*: Guy.yedwab

Image:CaseWkflw11.png *Source*: http://wiki.servicenow.com/index.php?title=File:CaseWkflw11.png *License*: unknown *Contributors*: Guy.yedwab, Steven.wood

Image:CaseWkflw12.png *Source*: http://wiki.servicenow.com/index.php?title=File:CaseWkflw12.png *License*: unknown *Contributors*: Guy.yedwab

Image:CaseWkflw13.png *Source*: http://wiki.servicenow.com/index.php?title=File:CaseWkflw13.png *License*: unknown *Contributors*: Guy.yedwab, Steven.wood

Image:ChgReport1.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChgReport1.png *License*: unknown *Contributors*: Guy.yedwab

Image:ChgReport2.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChgReport2.png *License*: unknown *Contributors*: Guy.yedwab

Image:ChgReport4.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChgReport4.png *License*: unknown *Contributors*: Guy.yedwab

Image:ChReport5.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChReport5.png *License*: unknown *Contributors*: Guy.yedwab
Image:ChgReport7.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChgReport7.png *License*: unknown *Contributors*: Guy.yedwab
Image:ChgReport8.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChgReport8.png *License*: unknown *Contributors*: Guy.yedwab
Image:ChgHomepage.png *Source*: http://wiki.servicenow.com/index.php?title=File:ChgHomepage.png *License*: unknown *Contributors*: Guy.yedwab
Image:Bulk CI Changes2.gif *Source*: http://wiki.servicenow.com/index.php?title=File:Bulk_CI_Changes2.gif *License*: unknown *Contributors*: Steven.wood
Image:Bulk CI Changes3.gif *Source*: http://wiki.servicenow.com/index.php?title=File:Bulk_CI_Changes3.gif *License*: unknown *Contributors*: Steven.wood
Image:CHG-showconflicts.png *Source*: http://wiki.servicenow.com/index.php?title=File:CHG-showconflicts.png *License*: unknown *Contributors*: G.yedwab
Image:CHG-noconflict.png *Source*: http://wiki.servicenow.com/index.php?title=File:CHG-noconflict.png *License*: unknown *Contributors*: G.yedwab
Image:CHG-relatedconflicts.png *Source*: http://wiki.servicenow.com/index.php?title=File:CHG-relatedconflicts.png *License*: unknown *Contributors*: G.yedwab
Image:Maintenance-schedule.png *Source*: http://wiki.servicenow.com/index.php?title=File:Maintenance-schedule.png *License*: unknown *Contributors*: David.Bailey
Image:Blackout-schedule.png *Source*: http://wiki.servicenow.com/index.php?title=File:Blackout-schedule.png *License*: unknown *Contributors*: David.Bailey
Image:RSKASS-module.png *Source*: http://wiki.servicenow.com/index.php?title=File:RSKASS-module.png *License*: unknown *Contributors*: Guy.yedwab
Image:RSKASS.png *Source*: http://wiki.servicenow.com/index.php?title=File:RSKASS.png *License*: unknown *Contributors*: Guy.yedwab
Image:RSKASS2.png *Source*: http://wiki.servicenow.com/index.php?title=File:RSKASS2.png *License*: unknown *Contributors*: Guy.yedwab
Image:RSKASS3.png *Source*: http://wiki.servicenow.com/index.php?title=File:RSKASS3.png *License*: unknown *Contributors*: Guy.yedwab
Image:RSKASS4.png *Source*: http://wiki.servicenow.com/index.php?title=File:RSKASS4.png *License*: unknown *Contributors*: Guy.yedwab