

Using Tasks

Managing Tasks in ServiceNow

Task Tables

Task Table



Note: *This article applies to Fuji and earlier releases. For more current information, see Task Table ^[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* **The ServiceNow Wiki is no longer being updated.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

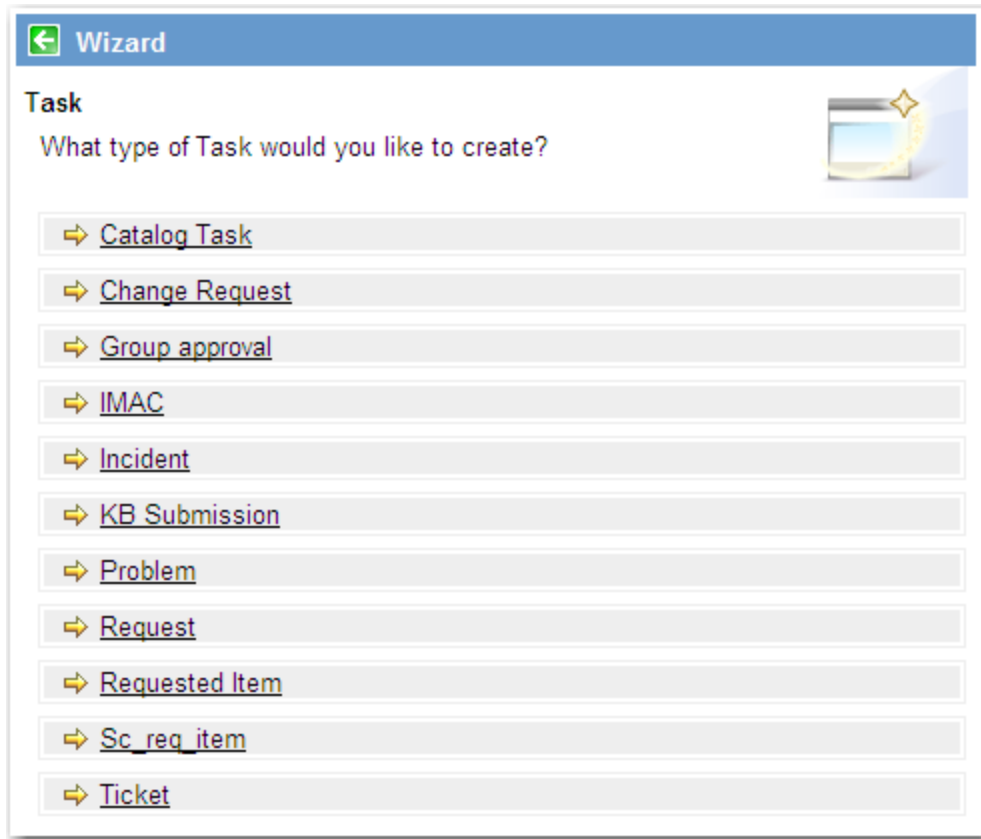
Task [task] is one of the core tables provided with the base system. It provides a series of standard fields used on each of the tables that extend it, such as the **Incident [incident]** and **Problem [problem]** tables. In addition, any table which extends task can take advantage of task-specific functionality for driving tasks.

Planned Task

The Planned Task Plugin provides the **Planned Task [planned_task]** table, which extends the **Task [task]** Table to provide more fields for tasks to measure duration and effort.

Creating a Task

Tasks are not created directly on the task table. Instead, tasks are created on task child tables. Clicking the **New** button on the Task record list will launch the Task Interceptor, which guides the user to the desired Task-extending table:



Modifying the Task Interceptor

To modify the task interceptor:

1. Navigate to **System Definition > Interceptors** (this module may need to be activated).
2. Select the **Task** Interceptor.
3. The Related List **Answers** specifies what choices are presented, and where the user will be redirected to once the choice is selected. Modify the list as desired.
4. After making changes, test the interceptor by clicking **Try It**.



Note: To disable the Task Interceptor, rename it to something other than **task.do**. This will disable it without deleting it.

State	state	Integer	A choice list for status of the task: <ul style="list-style-type: none"> • Pending • Open • Work in Progress • Closed Complete • Closed Incomplete • Closed Skipped
Sys ID	sys_id	GUID	The Unique Record Identifier for all records.
Task Type	sys_class_name	sys_class_name	A field which specifies the type of task, and adds a child class to extend the record. Dynamically populated when the record is created on the child table. For more information on extending tables, see Tables and Classes.
Time Worked	time_worked	timer	A timer which measures how long a record is open in the form view. For more information, see Adding a Time Tracking Field.
Watch list	watch_list	glide_list	A list of users who will receive email notifications when the record is updated. Note: the watch list will only receive notifications if Email Notifications are defined. For more information, see Configuring Watch Lists.
Work notes	work_notes	journal_input	Allows comments to be put on a record, viewable only by <i>util</i> users. Each comment is inserted into the Activity field. For more information, see Journal Fields, below.

Journal Fields

Journal fields work together to create a log of changes and comments as tasks are worked on.



Note: *Journal Fields only work on audited tables.*

Fields of the type **journal_input** are multi-line text boxes which, upon save, add the comments into the **Activity** field with a notation. The following fields accept input into the journal:

- **Additional Comments:** can be updated by any user.
- **Work Notes:** can be updated by *util* users.

These comments, as well as any changes to the record or email notifications sent out because of the record, will be displayed in the **Activity** formatter, which can be added to the form like a field:

Activity

2009-03-18 13:10:05 bow.ruggeri - Changed: Additional comments, Work notes, Incident state
Believe we have this resolved. Can you confirm?
cable got unplugged
Incident state: Awaiting User Info was: Active

2009-03-18 13:08:21 admin - Changed: Priority
Priority: 3 - Moderate was: 1 - Critical

2009-03-18 13:07:42 admin - Changed: Additional comments, Assigned to, Work notes, Impact
Checking it out now. Dropping impact to low since it only affects one office of users, and they have the day off.
Assigned to: Bow Ruggeri was: Howard Johnson
Not sure what's going on yet, but we're on it. Something to do with the VPN. Bow, can you take a look?
Impact: 3 - Low was: 1 - High

2009-03-18 13:03:42 beth.anglin - Changed: Additional comments, Incident state, Priority
Network file shares are indeed inaccessible. Hope they can be restored soon!
Incident state: Active was: Awaiting Problem
Priority: 1 - Critical was: 4 - Low

For more information, see Using Journal Fields

Driving Tasks

There are a number of different tools available to drive tasks to completion. These tools can be run on any table which extends task.

Approvals

Approvals can be generated to a list of Approvers, either manually or automatically, according to Approval Rules. Approvals can be incorporated into Workflows or can stand alone. For more information, see Approvals.

Approvals can be used on non-task tables.

Assignments

Assignment Rules can automatically assign tasks to users or groups, ensuring that tasks are handled by the most appropriate team members. For more information, see Defining Assignment Rules.

Service Levels

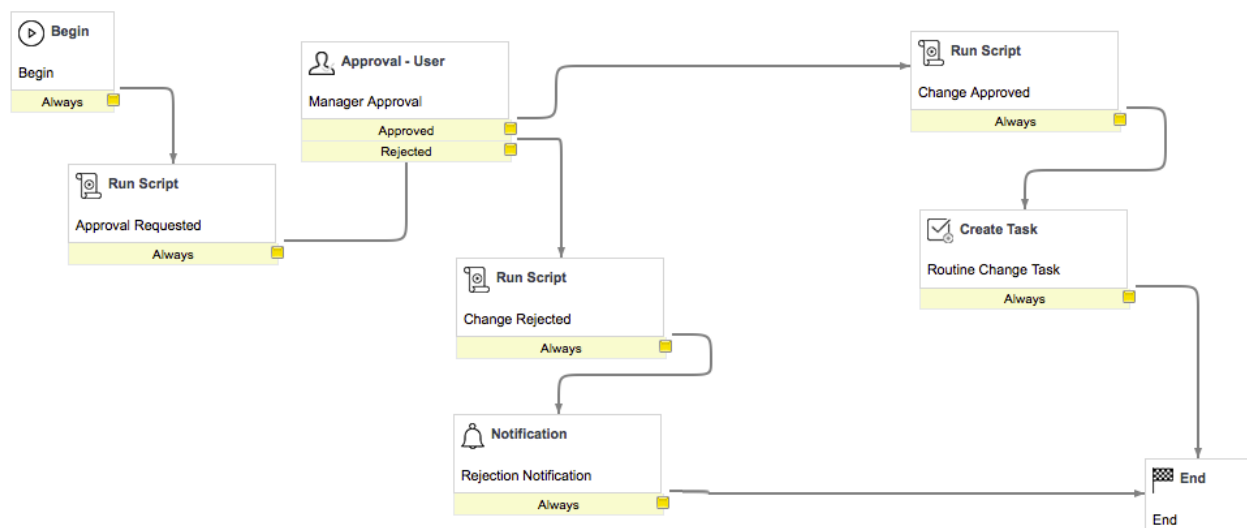
Service Level Agreements can track the amount of time that a task has been open, to ensure that tasks are completed within an allotted time. For more information, see Service Level Agreements.

Inactivity Monitors

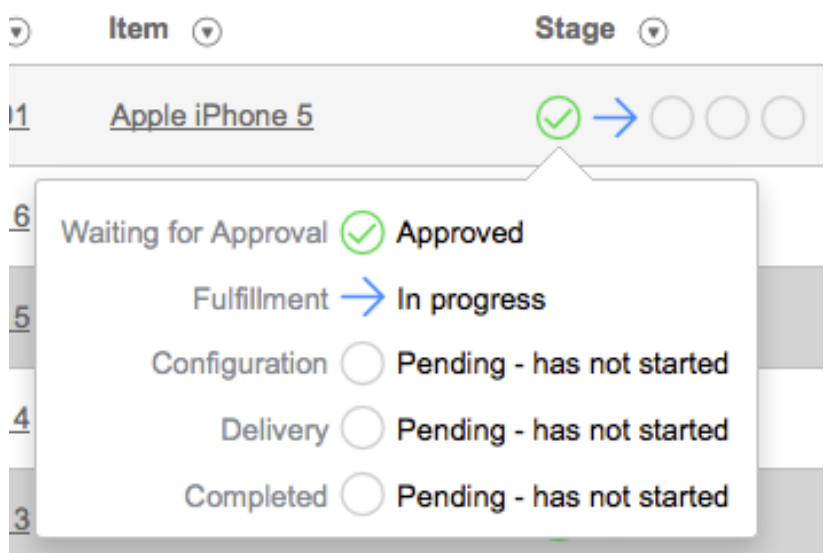
Inactivity monitors ensure that tasks do not fall by the wayside by notifying users when tasks have been untouched for a predefined period of time. For more information, see Setting Inactivity Monitors.

Workflow

An administrator can specify a specific workflow process to apply to tasks that meet certain conditions. Once a task is created that meets the conditions, the workflow will apply an automated process to the task. The process is defined in the graphical workflow editor:



As the process takes place, it will update any field designated as a workflow field:



Workflows are not specific to the tasks, but there are task-specific Workflow Activities (such as Task Activities and Approval Activities). For more information, see Workflow Overview.

Modifying the Task Table

Modifications made to the task table will be applied to all child tables. Be sure that the changes being made should apply to all of the child tables. Adding fields is a low-impact change, because the field can be hidden on tables that do not need it; however, deleting fields may cause unwanted data loss if the field is being used across tables. Note, when adding choice list entries to a choice list on the Task table, ensure the entry value is unique.

Some parts of a field definition can be changed in such a way as to not apply to all child tables using Dictionary Overrides.

References

[1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/administer/task-table/concept/c_TaskTable.html

Many to Many Task Relations Plugin



Note: This article applies to Fuji. For more current information, see *Many to Many Task Relations* ^[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

By default, tasks can be related to each other using a parent/child relationship, such as a Problem with a group of child Incidents or a Catalog Request with a group of child Catalog Tasks. However, it may prove useful to record exactly the nature of the relationship between the task records. When activated, the Many to Many Task Relations plugin allows administrators to define relationships between different tasks.

Activating the Plugin

The Many to Many Task Relations plugin is included with the following plugins:

- Planned Task
- Work Management
- Project Management
- Governance Risk and Compliance



Note: Contact ServiceNow to activate the plugin by itself.

Click the plus to expand instructions for requesting a plugin.

1. Navigate to [HI ^[2]].
2. Click **Service Catalog**.
3. Click **Request Plugin Activation**.
 - [Required] In **Target Instance**, select the instance on which to activate the plugin.
 - [Required] In **Plugin Name**, enter the name of the plugin to activate.
 - [Optional] In **Date and time would you like the plugin to be enabled?**, specify a date and time at least 12 hours in the future. Leave this field empty if you want the plugin activated as soon as possible.

Note: Plugins are generally activated during business hours in the Pacific time zone, but can be scheduled for a different time with advance notice.

 - [Optional] In **Reason/Comments**, add any information that would be helpful for the ServiceNow technical support engineer activating the plugin.
4. Click **Submit**.

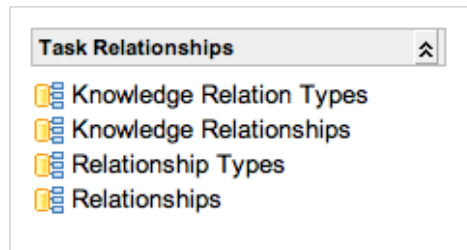


Note: References to Work Management indicate that the information on this page is valid for ServiceNow versions prior to Fuji. In the Fuji release, Work Management was renamed Field Service Management.

Plugin Manifest

When the plugin is activated, the **Task Relationships** application is available with the following modules:

- **Knowledge Relation Types:** contains the definitions for the possible types of relationships between knowledge articles and tasks.
- **Knowledge Relationships:** displays all of the defined relationships between existing tasks and knowledge articles.
- **Relationship Types:** contains the definitions for the possible types of relationships between tasks.
- **Relationships:** contains all of the defined relationships between existing tasks.



The following Relation Types are available by default:

- Knowledge Relation Types
 - Solution is documented in::Documenting solution for
- Relationship Types
 - Caused by::Causes
 - Contains::Task of
 - Documenting Solution for::Solution is documented in
 - Investigated by::Investigates
 - Permanent correction for::Permanently corrected by
 - Related to::Related to
 - Requesting::Requested by
 - Solved by::Solves

Defining a Relationship Type

To define a new Relationship Type:

1. Navigate to **Task Relationships > Relation Types** and click **New**.
2. Populate the **Parent Descriptor** and **Child Descriptor** fields with a short description of the relationship between the two tasks, such as a parent descriptor of **Caused By** and a child descriptor of **Causes**.
3. Right-click the form header bar and select **Save**. The **Name** field automatically populates with the Parent and Child descriptors.

 A screenshot of a web form titled "Task Relationship Type". At the top, there is a header bar with a back arrow icon, the title "Task Relationship Type", and three buttons: "Update", "Delete", and a circular icon with a plus sign. Below the header, there are three input fields. The first is labeled "Parent descriptor:" and contains the text "Depends on". The second is labeled "Child descriptor:" and contains the text "Is depended on by". The third is labeled "Name:" and contains the text "Depends on::Is depend". At the bottom of the form, there are two buttons: "Update" and "Delete".

Furthermore, it is possible to define Task Relationships Allowed from the Task Relationship Type record.

1. Scroll down to the related list and click **New**.
2. Populate the parent and child tables to define which tables are able to accept the relationship.

3. If desired, it is possible to define scripts to run in the **Parent** and **Child** script fields. These scripts are run when a parent or child record is run to automatically generate the other task (child or parent). These scripts use the **current** value of the new record, as opposed to the **source** record which triggered the script.

Task Relationships Allowed

Submit

Parent task:

Problem [problem]

Child task:

Release Phase [release_phase]

Type:

Permanent correction fc

Parent script:

Child script:

Submit

Viewing Task Relations

The **Task Relations** field can now be added to the task form:

Related Items: +

☒ Investigated by Incidents

INC0000001 - Charlie Whitherspoon - - Can't read email

INC0000002 - Howard Johnson - - Can't get to network file shares

☒ Solved by Change Requests

CHG0000003 - Roll back Windows SP2 patch

☒ Investigates Incidents

INC0000002 - Howard Johnson - - Can't get to network file shares

Update

Close Incident

Create Knowledge

Delete

The plus icon can be used to add or remove tasks from the list:

Add/Remove relationships with INC0000002

Active true

and Number INC0000002

Current Relationships

- Investigated by
- INC0000001
- INC0000002
- Solved by
- CHG0000003
- Investigates
- INC0000002
- Caused by
- Related to
- Solution is documented in

Available Tasks

Knowledge

- KB0000001
- KB0000002**
- KB0000003
- KB0000004
- KB0000005
- KB0000006
- KB0000007
- KB0000008
- KB0000009
- KB0000011
- KB0000012
- KB0000013
- KB0000014
- KB0000015
- KB0000016
- KB0000017
- KB0000018
- KB0000019
- KB0000020
- KB0000021
- KB0000022
- KB0000023
- KB0000024

Remove

Add

Number	KB0000002
Topic	General
Short description	Who needs to use the VPN?

Modifying the Displayed Fields

The fields displayed in the **Task Relations** field and editing interface are defined by the Reference Lookup's list view.

To modify the displayed fields:

1. Navigate to a form that has a reference to the table whose display values you would like to modify within Task Relations field.
2. Click the magnifying glass to display the Reference Lookup list view.
3. Right-click the list header and select the appropriate option for your version:
 - **Fuji or later:** Configure > List Layout
 - **Eureka or earlier:** Personalize > List Layout


The selected List layout will be used as the display value for records within the Task Relations field.

Mark as Solution Button

This button gets added to the KB popup view and is displayed when you search the knowledge base from a task record. Clicking the button creates a record in the Task / KB Relationships [task_rel_kb] table to associate the KB article with a task. Its functionality has been replaced in the KCS plugin and is no longer necessary. You can disable this button by marking the active field false on the 'solution_button' UI macro.

UI Actions

Once the task relationships are defined, it is possible to use UI Actions to define the task relationship as a new task is being created from an old task. Below are a few examples.

	Warning: These examples may not work on all instances. They are provided as illustrative examples.
---	---

Example 1 - Cause an Incident

This UI Action allows the change management team to log an incident directly from the change request and records that the incident was caused by the change.

Create a new UI Action on the Change Request [change_request] table and place the following into the script:

```
var inccaus = new GlideRecord("incident");
inccaus.short_description = current.short_description;
inccaus.comments = current.comments.getHTMLValue();
// inccaus.parent = current.sys_id;
inccaus.insert();
CauIncident();

gs.addInfoMessage("Incident " + inccaus.number + " created");
action.setRedirectURL(current);
action.setReturnURL(inccaus);

function CauIncident() {
var m2m = new GlideRecord('task_rel_task');
m2m.initialize();
m2m.child = current.sys_id;
m2m.parent = inccaus.sys_id;
m2m.type.setDisplayValue("Caused by::Causes");
m2m.insert();
}
```

Example 2 - Cause a Problem

This is a UI Action to allow the change management team to record a problem from a change request and record that the problem was caused by the change.

Create a new UI Action on the Change Request [change_request] table and paste the following script:

```
var probcaus = new GlideRecord("problem");
probcaus.short_description = current.short_description;
probcaus.comments = current.comments.getHTMLValue();
// probcaus.parent = current.sys_id;
probcaus.insert();
CauProblem();

gs.addInfoMessage("Problem " + probcaus.number + " created");
action.setRedirectURL(current);
action.setReturnURL(probcaus);
```

```
function CauProblem() {  
var m2m = new GlideRecord('task_rel_task');  
m2m.initialize();  
m2m.child = current.sys_id;  
m2m.parent = probcaus.sys_id;  
m2m.type.setDisplayValue("Caused by::Causes");  
m2m.insert();  
}
```

Example 3 - Fix a Problem

This UI Action allows a change request to be generated from a problem, recording that the change will fix the problem.

Create a new UI Action on the Problem [problem] table, and paste the following code:

```
var fixchg = new GlideRecord("change_request");  
fixchg.short_description = current.short_description;  
fixchg.comments = current.comments.getHTMLValue();  
// fixchg.parent = current.sys_id;  
fixchg.insert();  
FixChange();  
  
gs.addInfoMessage("Change " + fixchg.number + " created");  
action.setRedirectURL(current);  
action.setReturnURL(fixchg);  
  
function FixChange() {  
var m2m = new GlideRecord('task_rel_task');  
m2m.initialize();  
m2m.child = current.sys_id;  
m2m.parent = fixchg.sys_id;  
m2m.type.setDisplayValue("Fixes::Fixed by");  
m2m.insert();  
}
```

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/administer/task-table/concept/c_ManyToManyTaskRelations.html
- [2] <http://hi.service-now.com>

Process Flow Formatter Plugin

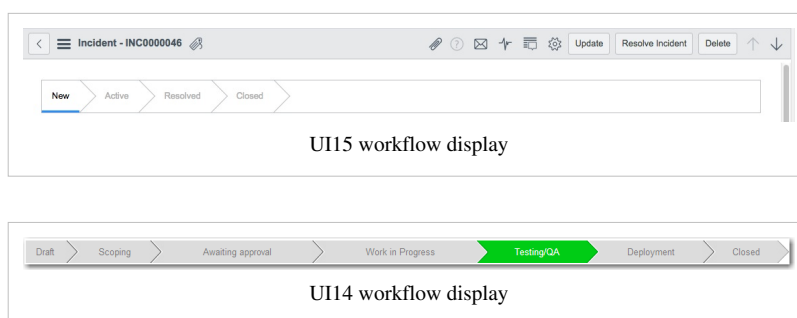


Note: This article applies to Fuji and earlier releases. For more current information, see *Process Flow Formatter* ^[1] at <http://docs.servicenow.com>. **The ServiceNow Wiki is no longer being updated. Visit <http://docs.servicenow.com> for the latest product documentation.**

Overview

The process flow formatter ^[2] quickly summarizes multiple pieces of information about a process and displays the stages graphically at the top of a form. Each record on the Flow Formatter [sys_process_flow] table represents a process stage and can have a different condition applied to it. When specified conditions are fulfilled, the formatter highlights the current stage and, starting with the Fuji release, all previous stages.

These examples show a workflow in the UI15 and UI14 interfaces. See Navigation and the User Interface for more information on these interfaces.



As soon as any formatter stages are defined for a table, they appear on the form associated with that table in the order specified, assuming the formatter has been added to the form.

You can see examples of preconfigured process flow formatter stages in Work Management, Release Management, and Sales Force Automation.

Adding a Process Flow Formatter

To add a process flow formatter, complete these tasks.

Task 1: Create a Process Flow Formatter

Users with the admin role can create a process flow formatter stage:

1. Navigate to **System UI > Process Flow**.
2. Click **New**.
3. Complete the form, as appropriate (see table).
4. Repeat as necessary.

Activating the Process Flow Formatter

Users with the admin role can activate the Process Flow Formatter 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**.

Enhancements

Fuji

- The **Active** field on the Flow Formatter [sys_process_flow] table determines whether formatter stages appear in the flow display.

References

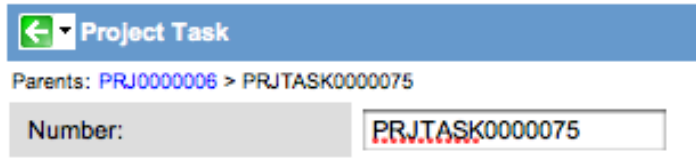
[1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/administer/form-administration/reference/r_ProcessFlowFormatter.html

[2] https://docs.servicenow.com/bundle/helsinki-servicenow-platform/page/administer/form-administration/reference/r_ProcessFlowFormatter.html

Task Parent Breadcrumbs Formatter

Overview

The parent breadcrumbs formatter on the Task table provides breadcrumbs that show the parent or parents of the current task. This formatter can be used on any table that extends Task, as well.



Adding the Parent Breadcrumbs Formatter

To add the parent breadcrumbs formatter to a form, configure the form and add **Parent Breadcrumbs** to the desired location. The breadcrumbs show only six levels of parents. If more levels exist, the breadcrumbs display an ellipsis (...).

Customizing the Parent Breadcrumbs Formatter

To customize the parent breadcrumbs formatter:

1. Navigate to **System UI > Formatters**.
2. Select **Parent Breadcrumbs**.
3. Click **View UI Macro for this Formatter** to view and/or alter the underlying formatter.

By default, the breadcrumb uses the default display field, `gr.getDisplayValue()`, as the link in the breadcrumb. To customize this, add the following line, replacing the **fieldName** parameter with the desired field name (not the field label):

```
pc.setLabelField("fieldName")
```

If a user points to a breadcrumb, the short description for that record appears as a hint by default. To display alternate hover text for the breadcrumb, add the following line, replacing the **fieldName** parameter with the desired field name (not the field label)::

```
pc.setTitleField("fieldName")
```

Be sure to add these lines in the proper location, as shown in the following example:

```
//parent crumb functions - script include
var pc = new ParentCrumbs(gr);
pc.setLabelField("short_description"); //override the default display
field to be used for label
pc.setTitleField("number"); //override default short_description hover
text
var crumbs = pc.getCrumbs();
```



Using the Parent Breadcrumbs Formatter on Non-Task Tables

The parent breadcrumbs formatter can be used on non-Task tables as long as the table has a reference to itself through a field called parent.

To make the formatter available for a different table, duplicate the formatter used by the Task table:

1. Navigate to **System UI > Formatters**.
2. Select **Parent Breadcrumbs**.
3. Set the **Table** field to the appropriate table.
4. Right-click the form header and choose **Insert**.
5. Add the new formatter to the appropriate form.

Task Outage Plugin

Overview

The Task-Outage Relationship plugin allows users to create an outage from an Incident or Problem form. This plugin creates a many-to-many relationship between the Task [task] table and the Outage [cmdb_ci_outage] table, and includes the UI Action **Create Outage Record**.

Creating an Outage from a Task

To create an outage from a task:

1. Navigate to the task record.
2. Right-click the header bar and select **Create Outage Record** from the context menu.

The screenshot shows the 'Incident' form in ServiceNow. The form includes fields for Number (INC0000031), Caller (Joe Employee), Location (Hamburg, Germany), Configuration item (Email), Impact (1 - High), Urgency (1 - High), Priority (1 - Critical), and Short description (EMAIL Server Down). A context menu is open over the form header, showing options like Save, Create Change, Create Knowledge, **Create Outage Record**, Create Problem, and Create Request. A tooltip for 'Create Outage Record' is visible, stating: 'Create outage record, create m2m between task and outage, redirect to new outage record'. The right side of the form shows fields for Opened (2010-09-16 17:18:03), Opened by (Don Goodliffe), Incident state (New), Assigned to (David Loo), and Knowledge.

3. Populate the Outage record as appropriate.

If a **Configuration Item** was specified on the task, it is associated with the outage.

The screenshot shows the 'Outage' form in ServiceNow. The form includes fields for Configuration Item (Email), Type (Outage), Begin (2011-02-04 08:21:10), End, and Duration (Days 00, Hours 00, 00:00). There are 'Update' and 'Delete' buttons at the bottom.

When you save the outage, it appears in the **Outage** related list on the task form.

Associating Tasks to Outages

A defined outage can also have tasks associated with it.

To associate tasks with an outage:

1. Navigate to the outage record.
2. Add the **Task** related list to the form.
3. Click **Edit** on the related list to add or remove tasks.

Adding the UI Action to Other Task Forms

By default, the **Create Outage Record** is available on the Incident and Problem tables. It can be added to any task's form.

To add the UI Action to other task forms:

1. Navigate to **System Definitions > UI Actions**.
2. Select the UI Action **Create Outage Record**
3. Alter the **Conditions** field. By default, the condition is:

```
current.getRecordClassName() == 'incident' || current.getRecordClassName() == 'problem'
```

The UI Action can be added or removed from tables using this query string. For example, to add this UI Action to the **Ticket** table:

```
current.getRecordClassName() == 'incident' || current.getRecordClassName() == 'problem' || current.getRecordClassName() == 'ticket'
```

To use the UI Action on only the **Incident** table:

```
current.getRecordClassName() == 'incident'
```

Enabling the Plugin

The Task Outage plugin is admin-installable.

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**.

Installed Components

Database Table Structure

The following tables will be added:

Display Name [Table Name]	Description
Task Outage [task_outage]	A Many-to-Many table storing references to the Task and Outage [cmdb_ci_outage] tables.

Scripts

The business rule **Add Short Description** is included in the plugin, to add a short description to outages in the following format "<CI Name> Outage".

The UI Action **Create Outage Record** is added to the Task table, to create outages directly from the task form.

Time Cards



Note: This article applies to Fuji. For more current information, see *Time Cards*^[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 time card management feature works with the Task table to record time worked on Projects, Incidents, Problems, and Change Requests. Task assignees can record time worked in the **Time worked** field on a task record or enter hours directly into their time card. Some tables support automatic time card creation based on start and end date fields.

Time cards also have an optional approval mechanism for project managers. Administrators and other roles that act as approvers can see all the time cards for the week. All users who are in a role that is responsible for working on tasks also can access their personal time cards. Time cards can have any of the following states.

- **Pending**
- **Approved**
- **Submitted**
- **Rejected**



Note: The Time card management plugin is required to use time cards. Some of the procedures on this page require the project management feature, which activates time cards automatically.

Recording Time Worked

Time accrued on a project or spent working on any record in the Task table is retrieved by the time card from the **Time worked** field. This field is present on Project Task records by default, but does not appear on the Incident, Problem, and Change forms and must be added by configuring the form. Time recorded in this field is used to populate an existing time card or to create a new time card if one does not already exist. This behavior is controlled by a time card **property**. The **Time worked** field has a counter that acts like a stopwatch for the duration of the time spent in the record. The counter can be stopped and started by a button in the field. By default, the **Time worked** counter is enabled and begins recording the elapsed time when the record is opened. Stop the counter with the red stop sign button and restart it with the green *play* button.

Time counter started:

Time counter stopped:

If you are creating time cards from time worked entries, you can add the related list to display the time worked records on the time card form.

You will also notice an informational message on the time card to let you know that changes to time worked records will override values in the time card. This is displayed using a formatter, which can be added or removed by

configuring the form.

← Time Card | = Required field

Time Worked property enabled, values may be overwritten by time worked records

Week starts on:	2011-01-02		
State:	Pending		
Category:	Task work		
 Task:	INC0000003		
User:	John Roberts		

Project Management User Resources

When the project management feature is enabled, the total time from each time card is displayed in the User Resource record for that Project Management task. To view a resource record, navigate to either of the following locations:

- **Project > Views > Resources by Project:**

← User resource

Allocation %:	100		
Planned hours:	293		
Planned task:	PRJ0000006		
Responsibility:	Project resource		
User:	Eric Schroeder		
Actual hours:	64		

Update
Delete


- **Project > Views > Project by Resources:**

User: Eric Schroeder (2)					
Responsibility	User	Short description	Allocation %	Planned hours	Actual hours
Project resource	Eric Schroeder	Demo project (small)	45	30.6	12
Project resource	Eric Schroeder	Demo project (medium)	100	293	64

Creating a Time Card

Time cards can be created automatically or manually:

- **Automatic:** Configure time cards to be created when a user updates a task record. This behavior is controlled by a time card property that is set to *false* by default. See the table of properties in this page for details. In Incident, Problem, and Change records, the **Time worked** field must be added to the form.
- **Manual:** Create a new time card for each task and enter the times manually.



Note: Time cards cannot be created automatically when you use the mobile interface. Use the desktop interface if you want to use the automatic time card feature.


Users with the timecard_admin role can create a time card manually:

1. Navigate to **Time Cards > All** and click **New**.

The **Week starts on**, **State**, and **Category** fields are completed automatically. The category defaults to **Task work**, but can be any of the following:
 - Task work
 - Admin
 - Meeting
 - KTLO (maintenance of existing system)
 - Out of office
 - Training
2. Select a **Task** from the pop-up list.

This can be anything from the Task table.
3. Select your name from the list in the **User** field.
4. Click **Submit**.




After the time card is created, the hours for that task can be incremented automatically from the **Time worked** field in the task record. This is controlled by a time card property, which is set to *true* by default. See the table of properties in this page for details. If automatic updates are not configured, the time card must be updated manually by the user or an administrator.








 Time Card

= Required field

Update

Delete



Week starts on:	2010-05-23		Sunday:	<input type="text" value="0"/>
State:	Submitted		Monday:	<input type="text" value="4"/>
Category:	Task work		Tuesday:	<input type="text" value="2"/>
Task:	PRJTASK0000013	 	Wednesday:	<input type="text" value="8"/>
User:	Bow Ruggeri	 	Thursday:	<input type="text" value="3"/>
			Friday:	<input type="text" value="4"/>
			Saturday:	<input type="text" value="0"/>
			Total:	<input type="text" value="21"/>

Update

Delete

Managing Time Cards

The **My Time Cards > Current** module presents a page showing all of your time cards for the current week. There is also a control to **Generate Task Cards**. This button will search for all planned tasks that are scheduled for the current time card period, if you don't already have a time card for the task.

Properties

Users with the `timecard_admin` role can set time card properties by navigating to **Time Cards > Administration > Properties**.

Name	Description	Default
<code>com.snc.time_card.autocreate</code>	Auto-create a user's time card when they update a task	No
<code>com.snc.time_card.time_worked</code>	Auto-fill a user's time card with time from their 'Time worked' entries	No
<code>com.snc.time_card.update.effort</code>	Update the task's 'Actual effort' based on the hours entered in the time card	No
<code>com.snc.time_card.update.resource</code>	Update the project/user's resource allocation record based on the hours entered in the time card	No
<code>com.snc.time_card.start_day</code>	What day should time cards start on, default is Sunday. Changing this value may create duplicate time cards for the week of the change, since time card queries are based on this value.	Sunday

Managing Costs

When the cost management feature is enabled, time cards can be used to manage the cost of labor in the **Financial Management** application.

When a time card is marked **Approved**, the user's rate (listed in the **Labor Rate Card**) is used to generate a one-time Expense Line for the time worked. If no Labor Rate Cards apply to the user, the property `com.snc.time_card.default_rate` defines a default rate.

Roles

The `timecard_admin` role enables users to approve, modify, and delete the time cards of other users.

Activating Time Card Management

Administrators can activate the Time card management 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**.

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/administer/task-table/concept/c_TimeCards.html
-

Planned Task Tables

Planned Task Plugin

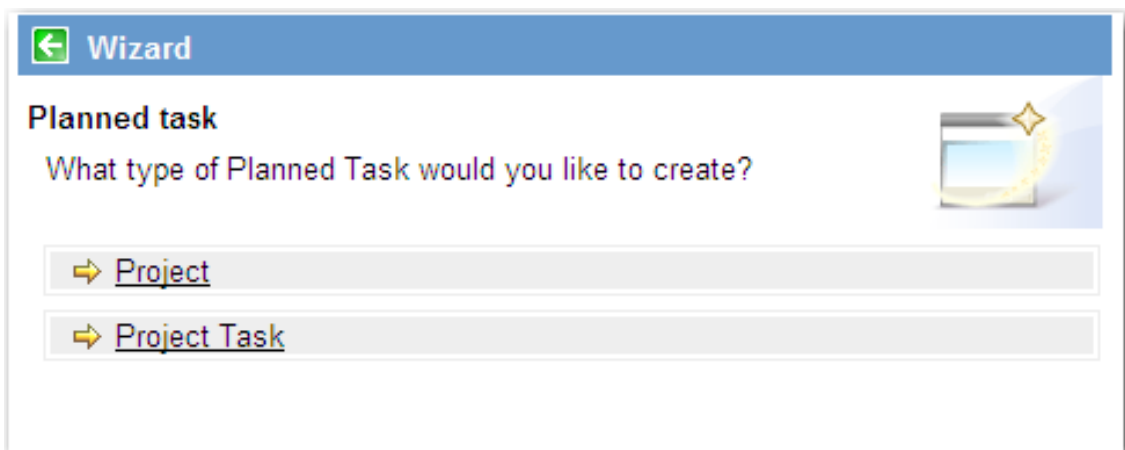
Overview

The Planned Task plugin provides a Planned Task [planned_task] table that extends the Task [task] table. Planned tasks provide additional fields for tasks pertaining to time and effort as part of a planned, multi-stage process.

The Planned Task plugin is included with the Project Management plugin. The Project [pm_project] table extends planned task, but any custom table can be created to extend Planned Task and leverage its fields.

Creating a Planned Task

Planned Tasks are not created directly on the Planned Task [planned_task] table. Instead, planned tasks are created on planned task child tables. Clicking the **New** button on the Planned Task record list will launch the Planned Task Interceptor, which prompts the user to select a child table to create the planned task on:



Modifying the Planned Task Interceptor

To modify the planned task interceptor:

1. Navigate to **System Definition > Interceptors** (this module may need to be activated).
2. Select the **Planned Task** Interceptor.
3. The Related List **Answers** specifies what choices are presented, and where the user will be redirected to once they select the choice. Modify the list as desired.

The screenshot shows the 'Interceptor' plugin interface. At the top, there's a form with fields for 'Name' (Planned task), 'Question' (What type of Planned Task would you like to create?), and 'Intercepts' (planned_task.do). There are 'Update' and 'Delete' buttons. Below the form is a table with columns: Name, Active, User Prompt, Next question, Order, and Target URL. The table contains two rows: 'Project' (Active: true, User Prompt: Project, Order: 100, Target URL: pm_project.do) and 'Project Task' (Active: true, User Prompt: Project Task, Order: 200, Target URL: pm_project_task.do). There are also 'Update' and 'Delete' buttons for the table.

Measuring Time and Effort

The Planned Task [planned_task] table provides standard fields for tracking duration and effort.

Duration measures time from start to end date. Effort measures hours of work exerted on the project.

Duration

- **Planned duration:** the projected length of time for the planned task.
- **Actual duration:** the actual length of time so far for the planned task.
- **Remaining duration:** the **Planned duration** minus the **Actual duration**, which represents the projected length of time left.

Effort

- **Planned effort:** the projected amount of time that will be spent on the planned task.
- **Actual effort:** the actual amount of time that has already been spent on the planned task.
- **Remaining effort:** the **Planned effort** minus the **Actual effort**, which represents the project amount of work left.
- **Percent complete:** the **Actual effort** divided by the **Planned effort**, which estimates the percentage of planned work which has been completed.

Important Planned Task Table Fields

The following table contains a list of important Planned Task fields:

Label	Name	Type	Description
Actual cost	work_cost	currency	The actual cost of the planned task, to be compared with the Estimated cost .
Actual duration	work_duration	glide_duration	The actual length of time (from start time to end time) of work on the planned task, to be compared with the Planned duration .
Actual effort	work_effort	glide_duration	The actual time spent working, to be compared to the Planned effort .
Critical Path	critical_path	boolean	
Estimated cost	cost	currency	An estimation of the cost of the planned task, to be compared with the actual cost.
HTML Description	html_description	html	A description field that accepts HTML mark-up.
Percent Complete	percent_complete	decimal	A percentage of the completed effort. Generated using the Planned effort and Actual effort fields.
Planned duration	duration	glide_duration	The estimated length of time (from start time to end time) of the planned task.

Planned effort	effort	glide_duration	The estimated amount of time spent working on the planned task.
Planned end date	end_date	glide_date_time	The estimated date and time for the planned task to end.
Planned start date	start_date	glide_date_time	The estimated date and time for the planned task to start.
Remaining duration	remaining_duration	glide_duration	The difference in planned and actual duration, representing the time left for the planned task.
Remaining effort	remaining_effort	glide_duration	The difference in planned and actual effort, representing the amount of work time left for the planned task.
Rollup	rollup	boolean	Read-only field managed by the system that identifies the task as having child tasks. A rollup task will have a number of its fields calculated from the children so those fields will be read-only.
Time constraint	time_constraint	string	A description of time constraints that apply to the planned task.
Top Task	top_task	reference (planned_task)	When different planned tasks are stacked in a hierarchy, this field populates with the highest-level parent task. For example, if Project A has a child Project B, and Project B has a child Project C, then Project C's Top Task is Project A. Project A's Top Task field will be blank.

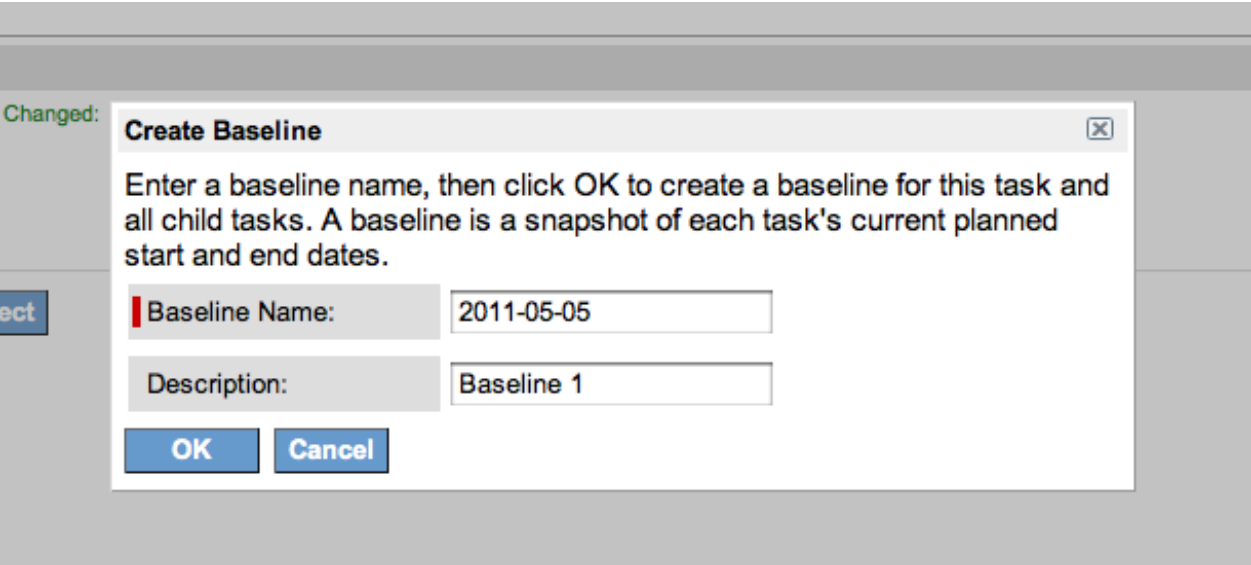
Planned Task Scripts

A number of business rules and one script include power the dynamic calculation of crucial Planned Task fields:

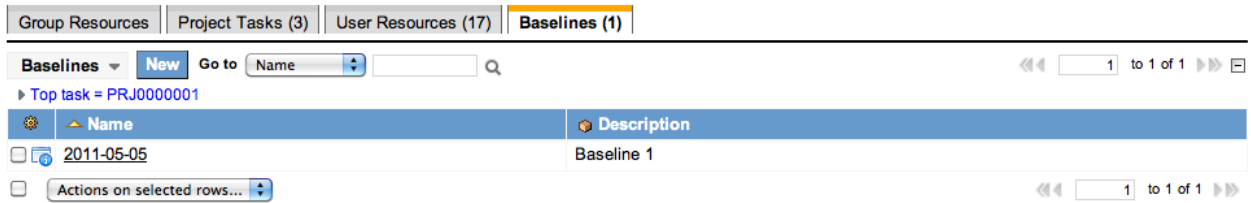
Business Rule	Description
Set Actual Work Start Value	Sets the planned task's Actual Start Date when State is set to the default work state.
Set Close Data on Inactive	Sets the planned task's close data when task becomes inactive.
Recalculate	Recalculates the planned task schedule fields when one of the schedule fields changes.
Auto close milestones	Automatically closes milestones when they are passed.

Creating a Baseline

A Planned Task Baseline is a record of the planned task's start and end times at a particular moment in time. To create a baseline, navigate to the top planned task's form and select the **Create a Baseline** related link:



To view the baseline, personalize the related lists to add a related list of baselines:



Each baseline has a related list of Baseline Items that records every task associated with the top planned task, as well as its start and end times:



Related Links

[View Baseline on Gantt Chart](#)

Baseline Items			New	Go to	Task		
Baseline = 2011-05-05							
Task	Start	End					
PRJ0000001	2011-04-26 13:00:00	2011-05-08 13:00:00					
PRJTASK0000001	2011-04-26 13:00:00	2011-05-01 13:00:00					
PRJTASK0000002	2011-05-01 13:00:00	2011-05-04 13:00:00					
PRJTASK0000003	2011-05-04 13:00:00	2011-05-08 13:00:00					

The baseline can be viewed on a Gantt Chart using the related link.

Activating the Plugin

The plugin is included in the Project Management plugin.

Planned Task Hierarchy



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

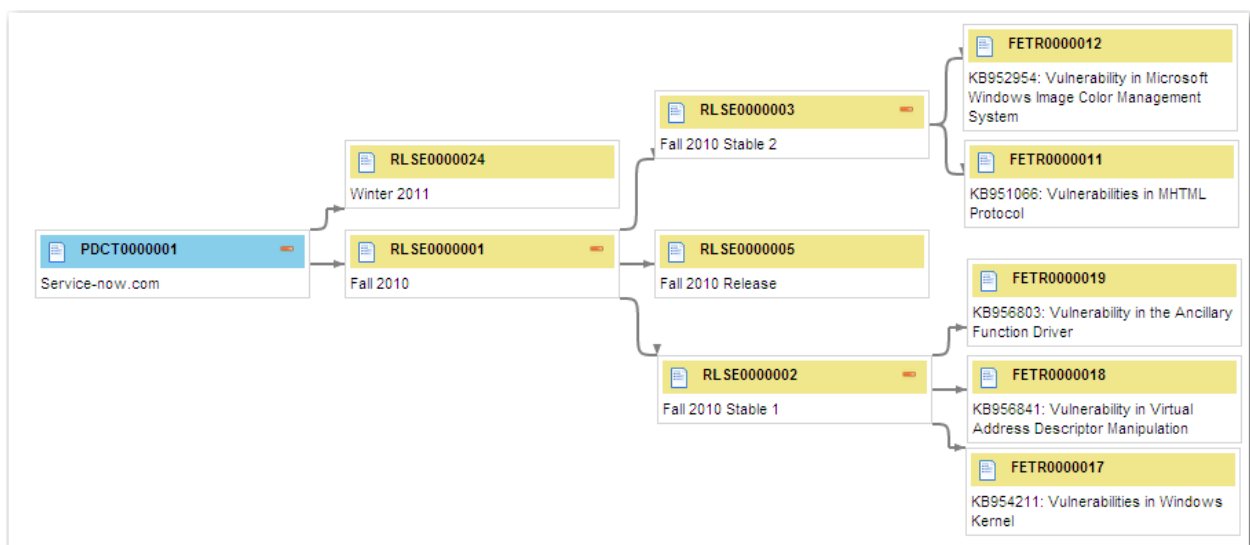
Overview

The **Task Hierarchy** tool available for Planned Task displays the relationship between parent and child planned tasks. Out-of-box, the **Task Hierarchy** tool is available in both Project and Release management.

Using the Planned Task Hierarchy

Different **Planned Task** tables have different UI actions to launch the task hierarchy:

- To view a Project's hierarchy, navigate to the use the **Task hierarchy** context menu action.
- To view a Product's hierarchy in Release v2, navigate to the product and click the **Product hierarchy** related link.
- To view a Release's hierarchy in Release v2, navigate to the release and click the **Release hierarchy** related link.



Adding the Hierarchy to Other Planned Task Tables

The Task Hierarchy can be added to any planned task table by:

1. Navigating to **System UI > UI Actions**.
2. Selecting one of the existing Task Hierarchy UI Actions (e.g. **Task hierarchy** if Project Management is activated).
3. Change the table to the desired table and rename the UI Action if appropriate, and insert.

The hierarchy should now be available as a UI Action on the new table's form.

References

[1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/administer/task-table/task/t_PlannedTaskHierarchy.html

Gantt Chart



Note: This article applies to Fuji and earlier releases. For more current information, see *Gantt Chart*^[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

The Gantt chart is a visual representation of a project timeline that shows start and end dates of tasks, and the dependencies between tasks. Use the Gantt chart to add and delete tasks, change task dates and dependencies, and assess the progress of the overall project.

Key Concepts and Terms for This Topic

- **Critical path:** linked project tasks that determine how long the project takes to complete.
- **Project portfolio Gantt chart:** a read-only Gantt chart available for a group of projects organized into a portfolio.

Using the Gantt Chart

The Gantt chart is a visual representation of the project showing tasks, the amount of time to complete the tasks, and dependencies between the tasks.

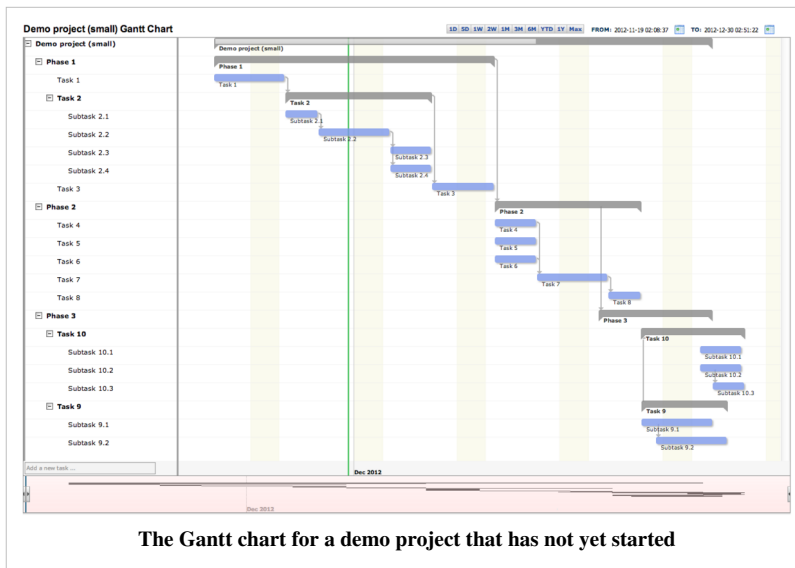
Navigating Through the Gantt Chart

1. Open a project.
2. Under **Related Links**, click **Gantt Chart**.

The Gantt chart shows the project and tasks. Any nested projects (and associated tasks) are also shown within the hierarchical structure. Each task is labeled by its short description and is represented by a line proportional to the task's duration.

3. Change the perspective:
 - Move the slider at the bottom of the chart right or left to scroll across the chart, or adjust the end points of the slider to change the magnification. A narrow slider zooms in on the tasks and provides a more detailed view of the task relationships. A wide slider pulls the view out and makes more of the Gantt chart visible on the screen.

- Click a time interval, such as **1M** for one month, from the options above the chart. The time interval zooms the chart in or out starting with the current time and date, which is signified by a vertical green line.
- Specify a time interval by selecting **From** and **To** dates above the chart.



On the Gantt chart, the following items are also available:

- The Project outline: a pane that lists all project tasks in a hierarchy.
- The project schedule: vertical light-brown bars in the background that show time periods that are not part of the work schedule. For example, the graphic above shows the Gantt chart for a project with the **8-5 weekdays excluding holidays** schedule specified. The vertical light-brown bars are weekends and holidays. Use this information when setting task start dates, end dates,

and dependencies so that you do not schedule a task to begin or end on non-work days.

Creating a Dependency in the Gantt Chart

To create a dependency between two tasks:

1. Click and hold a task.
2. Drag to a new task. A line appears showing the new relationship.
3. In the message that appears, confirm the new dependent relationship.
4. Continue connecting tasks for all dependencies in the project.

To edit a relationship:

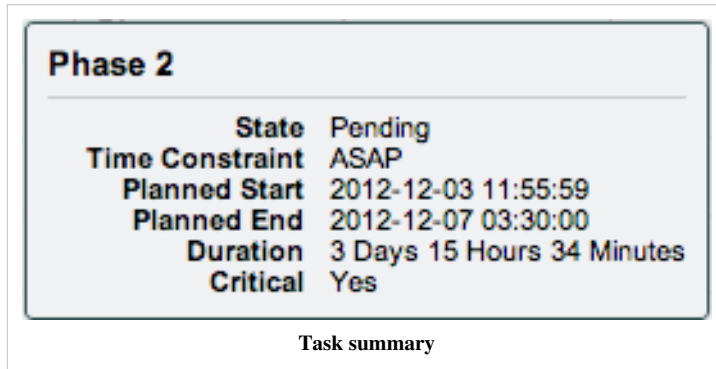
1. Double-click the connector between tasks.
2. Use the **Planned Task Relationship** dialog box to define the order of tasks and the lag time between tasks.
 - The relationship **Type** for planned tasks is **Predecessor of::Successor of** and should not be changed.
 - When a **Lag** value exists, ServiceNow recalculates the critical path accordingly.

Viewing Information About a Task in the Gantt Chart

View full task information in either of these ways:

- Double-click a task bar to open the project task record and view or edit the form.
- For a brief summary of the task, point to a task bar to display a tooltip.

Administrators can configure the information shown in the tooltip.



Phase 2

State	Pending
Time Constraint	ASAP
Planned Start	2012-12-03 11:55:59
Planned End	2012-12-07 03:30:00
Duration	3 Days 15 Hours 34 Minutes
Critical	Yes

Task summary

Adding Tasks

To add tasks to a project in the Gantt chart:

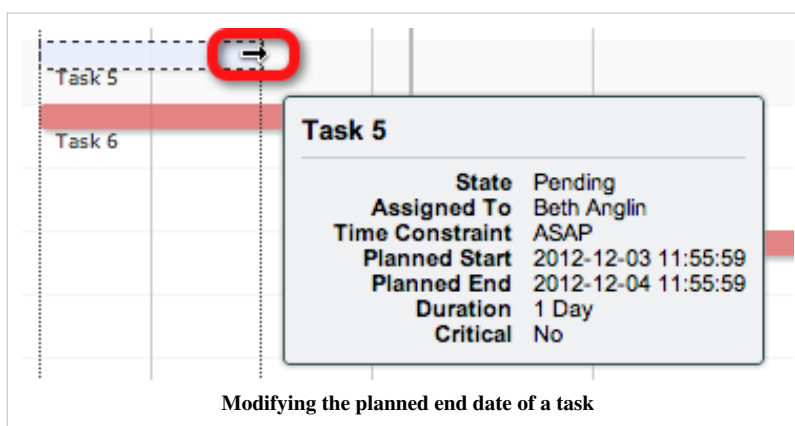
1. Navigate to the Gantt chart for the appropriate project.
2. Enter a name for the new task in the **Add a new task** field and press **Enter**.
3. From the left tab, drag the task name to the desired location in the task hierarchy.

4. Double-click the task bar and configure a duration in the Project Task form.
5. Click **Update**.
6. Create dependent relationships between tasks as appropriate.

Editing Tasks

Use the Gantt chart to quickly change task attributes, such as start and end time, rather than opening every Task form and modifying field values one by one. From the Gantt chart, you can modify the following:

- **Planned start date:** move the task along the timeline to change the start time and to impose a **Time constraint of Start on a specific date**. You can also drag a task to change its start date if the task **Time constraint** is set to **Start on a specific date** (not **Start ASAP**) and the task has not yet started. The start date of a task cannot be modified if the task already started (has an actual start date), the task has already ended (has an actual end date), or the task time constraint is set to **Start ASAP**.
- **Planned end date:** drag the right edge of the task bar to extend the planned end date. You can only extend the planned end date for tasks that are not parent tasks and have not yet ended.



Modifying the planned end date of a task

Task 5

State	Pending
Assigned To	Beth Anglin
Time Constraint	ASAP
Planned Start	2012-12-03 11:55:59
Planned End	2012-12-04 11:55:59
Duration	1 Day
Critical	No

- **Dependencies:** To edit or delete a dependency, double-click an existing dependency connector line between two tasks and make the changes in the Planned Task Relationship dialog box. The relationship **Type** for planned tasks is **Predecessor of::Successor of** and should not be changed.
- **Lag time:** Lag time is the interval between the end of a predecessor task and the start of a successor task. The default setting is 0. To edit the lag time, double-click a connector and make the changes in the Planned Task Relationship dialog box.

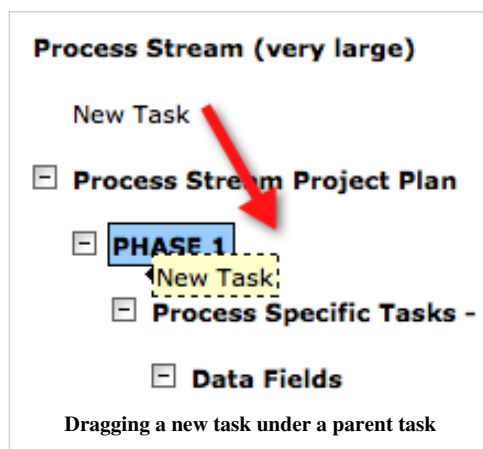
- **Resources:** To change a user resource for an existing task or add a resource to a new task, double-click the task bar and edit the **Assigned to** field in the Project Task form.

Editing Parent-Child Relationships

On the left pane, drag a task onto another task to change the parent-child relationship. The task that you dragged becomes the child task. Edit the task just like any other task.

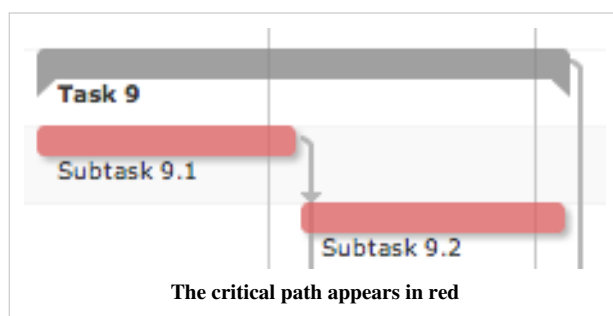
Note the following:

- New tasks have the **Time Constraint** field set to **Start ASAP**. If the task becomes a child task, the **Time Constraint** value stays **Start ASAP**, but the **Planned start date** becomes the start date of the parent task.
- You can further adjust the child task's start or end date with the dragging tool, but you can adjust the start date of the task only if the **Time Constraint** value is *not* **Start ASAP**.
- If a child task is scheduled to start on a date earlier than the parent task, the **Time Constraint** changes to **Start ASAP**.
- If you drag the end date of the child task past the end date of the parent task, the end date of the parent task automatically extends to the same end date.



The Critical Path

The critical path is highlighted in red on the Gantt chart to differentiate critical path tasks from standard tasks in blue and rollup tasks in gray. Not all tasks are part of the critical path, only those tasks that directly affect the finish date. Use the critical path to determine which tasks are driving the finish date. If schedule adjustments are necessary, consider making resource or other changes to those tasks on the critical path. In the following screenshot, the two child tasks are part of the critical path.



The Portfolio Gantt Chart

In addition to the project Gantt chart, the Project application offers a read-only project portfolio Gantt chart that shows the projects in each portfolio. One bar appears for each project in the portfolio.

References

- [1] https://docs.servicenow.com/bundle/jakarta-it-business-management/page/product/project-management/concept/c_GanttChart.html

Task Resources



Note: This article applies to Fuji and earlier releases. For more current information, see *Task Resources*^[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

Resources are the individuals assigned to perform tasks and subtasks in Project Management. You can manage your resources with resource plans in the Resource Management application, starting with the Dublin release.

In versions prior to the Dublin release, or if you are not using the Resource Management application, you can select resources from users or groups. **Skills** can be defined for individuals and for entire groups, ensuring that capable people are available for each project task. Because of the way in which Project Management is integrated into the task table, resources use basic ITIL task management processes. SLAs, approvals, and reporting are built in, just as they are for Incident Management and Change Management. A project task appears in a user's queue like any other task and does not indicate that it is a part of a project.

Prerequisites

Before you perform the procedures in this page, be sure you have completed the **Gantt Chart** for your project, showing task relationships and durations.

Adding User Resources

The User Resources record enables an administrator to associate a user with the project, with a project responsibility, and a percentage allocation. This percentage allocation checks against the project's schedule and calculates the amount of hours the percentage allocation represents. These hours are then used to determine whether this resource can continue to work project tasks, or if the resource is *out of time* for the project.

You add resources to a project in the **User resources** Related List in a project form. After selecting the project's resources, you attach those resources to their tasks in the **Resource Timeline**. Only one resource is assigned to a task at a time.



Note: You can add resources to **tasks** at the time the tasks are created. However, the best practice is to add multiple resources to the project through the Related List and then use the Timeline to make assignments, particularly for large projects. Resource allocation on large projects usually involves frequent adjustments.

To create user resources from a project:

1. Click **Edit** in the **User resources** Related List.
2. In the slushbucket of users that appears, select the resources for the project and save the choices.

The **User resource** list is populated with starting values of 100% allocation and zero hours.

Project Tasks (4) | User resources (4)

User resources

Edit...

Planned task = PRJ0010004

Responsibility	User	Allocation %	Planned hours	Actual hours
<input type="checkbox"/> Project resource	Beth Anglin	100	48	0
<input type="checkbox"/> Project resource	Howard Johnson	100	48	0
<input type="checkbox"/> Project resource	Jim Ranzetti	100	48	0
<input type="checkbox"/> Project resource	Luke Wilson	100	48	0

Actions on selected rows...

The User Resources related list

3. Click a link in the **Responsibility** column to open the User resource record.
4. Configure the resource's **Planned hours** and **Responsibility** (if different from **Project resource**).

User resource

Allocation %:

Planned hours:

Planned task:

Responsibility:

User:

Actual hours:

50

48

PRJ0010004

Project manager

Beth Anglin

0

Update

Delete

Configuring a user resource

5. Update the record and configure the next resource.
- The totals in the **Actual hours** column can be updated manually or automatically from the resource's **Time Card**.

User Resources with Costing Add-on

The **Project Management v2 Costing Add-on plugin** adds an additional column called **Actual Cost** to the **User Resources Related List**. The platform

calculates the **Actual Cost** of a resource by multiplying the **Actual Hours** consumed by the *labor rate* for the resource on the **Labor Rate Card** that applies to the user.

Adding Group Resources

The **Group Resources Related List** in the Project form enables an administrator to associate ServiceNow groups with a project to facilitate resource planning. The following process illustrates how a large organization might use group resources to plan a project:

1. The project is created and the group resources are estimated, providing details for project review and approvals.
2. The project is approved, and the tasks are created without user resources being assigned. All that is necessary at this stage is to assign each task to a group that has the necessary skills.
3. Group managers review the work requirements and select the appropriate person to work on the task.

!

Note: When a project task has a group in the **Assignment group** field, the list of users in the **Assigned to** field in the Project Task form is filtered on the members of the assigned group.

Add group resources to projects at the task level and select the user resource to assign to the task at the same time. All group resources assigned to project tasks appear in the **Group Resources Related List** in the Project form.

1. Navigate to *Project > Projects > Pending* and select a Project.
2. In the Project record, select the **Project Tasks Related List**.
3. Select a task from the list.
4. Select a group from the **Assignment group** field.

If the **Skills Management Plugin** is activated, an assignment group with pre-configured skills can be assigned to this task. For details, see **Assigning Skills to Tasks**.

5. Select a user from the **Assigned to** field.

The selection list in this field is filtered on the members of the assignment group selected, ensuring that the user has the necessary skills required for the task.

6. Click **Update**.

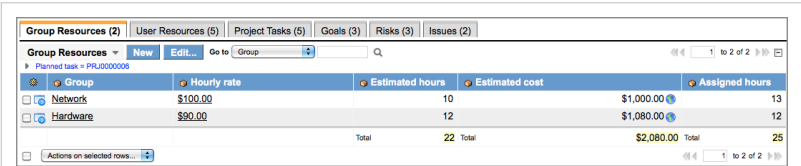
In the Project form, the selected groups appear in the Group Resource Related List.

Group Resource Related List

The fields provided in the **Group Resources** Related List depend on whether or not the Project Management v2 Costing Add-on plugin is installed.

Field	Input Value
Group	Name of an assignment group defined for a task in this project. All members of an assignment group are available to work on a task and inherit any skills assigned to the group.
Hourly rate	Defines the hourly rate for members of this group, as defined in the Group record. This field requires the Project Management v2 Costing Add-on plugin . The hourly rate is multiplied by the Estimated hours for the group to estimate project costs.
Estimated hours	Estimated number of effort hours needed from this group to complete the project. This field is used during project planning to estimate resource requirements. The estimated hours are multiplied by the Hourly rate of the group to estimate project costs. This field is found on the Group resource form. Use one of the following methods to access this form: <ul style="list-style-type: none">If the Project Management v2 Costing Add-on plugin is installed, click the link in the Hourly rate column.If the Project Management v2 Costing Add-on plugin is NOT installed, click the link in the Estimated hours column.
Estimated cost	Total of a group's Hourly rate multiplied by the Estimated hours . The total of each group's estimated cost is the estimated cost of the project. This field requires the Project Management v2 Costing Add-on plugin . For more information, see Managing Project Costs .
Assigned hours	The total of the hours assigned to each group from the Planned effort field in a Project Task form. This field is only visible in the Advanced View. Assigned hours represent specific time estimates on the task level that will differ from the Estimated hours .

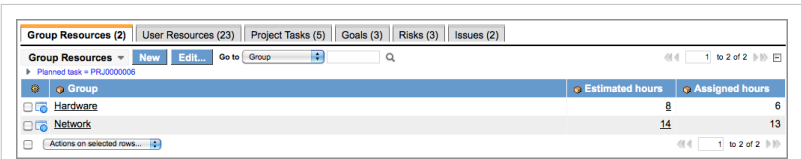
The following illustrates the Group Resources related list when the Costing Add-on is activated:



Group	Hourly rate	Estimated hours	Estimated cost	Assigned hours
Network	\$100.00	10	\$1,000.00	13
Hardware	\$90.00	12	\$1,080.00	12
Total		22	\$2,080.00	25

Group Resources related list with Costing Add-on

The following illustrates the Group Resources related list without the Costing Add-on:



Group	Estimated hours	Assigned hours
Hardware	8	6
Network	14	13

Group Resources related list without the Costing Add-on

Resource Timelines

The following updates were added to the User Resources timeline in the Spring 2010 Stable 2 release:

- Improved rendering performance, particularly for projects with a large number of items.
- Visual improvements including summary pane overlay bevel, timeline drop shadow, font adjustments, and preview mask coloring tweaks.
- Capability to split the timeline into multiple frames.
- Quick range selection buttons and start/end time view adjustment.
- Inner segments for Timeline Span elements on the timeline, useful for % complete on Gantt Charts or travel time for Field Service Management visual dispatching.

- Auto-refresh capability.



Note: For more details and API documentation on the new features please see: *Schedule Page Spring 2010 Stable 2 Updates*

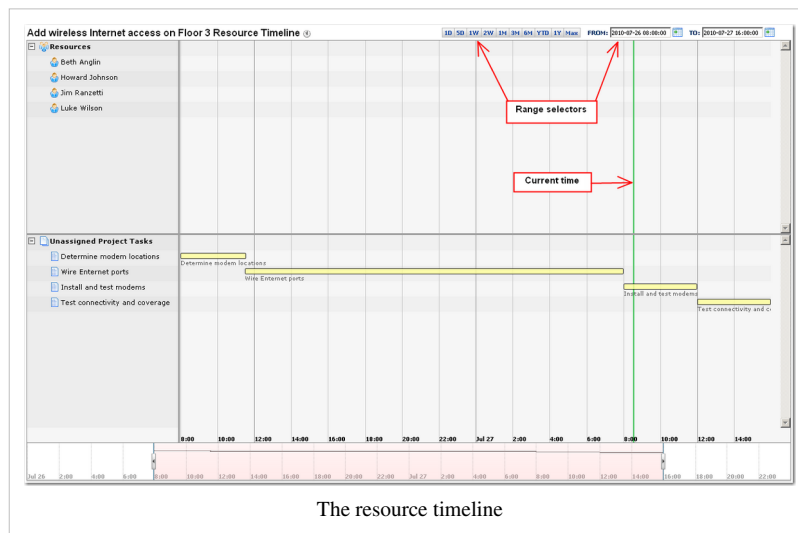
The *Resource Timeline* is a graphical, interactive map that establishes the relationships of tasks and sub-tasks to user resources for a project. Create the timeline after you identify the resources and tasks for the project. You must establish the relationships manually between the unassigned tasks and the free resources.

At this point, you should have a project record containing tasks and a list of resources to associate with those tasks.

1. Click the **Resource Timeline** Related Link in the project record.

The Timeline displays two panes: *Resources* and *Unassigned Project Tasks*. You can resize the panes by dragging the dividing line that separates them. The tasks are arranged in the proper order (as defined in the **Gantt Chart**) and with the proper durations. In this example, the duration for the second task, **Wire Ethernet ports**, extends over two days, yet is configured for 8 hours. This is because we specified a project schedule of **8-5 weekdays**, and the time slot for that task must span the off-work hours between the days.

NOTE: The **Schedule** field in the Project form is optional and visible only in the **Advanced** view of the form.



2. Use the pink slider at the bottom of the timeline to change the perspective.

- a. Move the slider from right to left to view all the tasks on a long timeline.
- b. Adjust the end points of the slider to change the magnification.

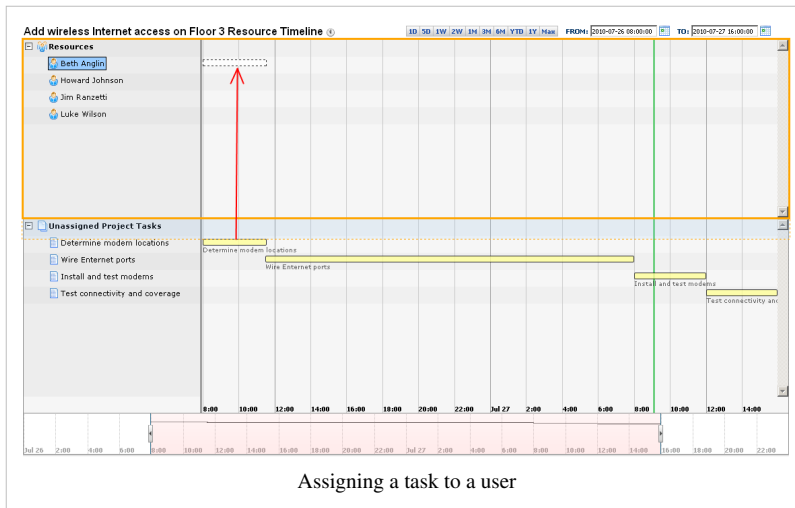
A narrow slider zooms in on the tasks and provides a more detailed view of complex timelines. A wide slider pulls the view out

and makes more of the timeline visible on the screen.

3. Use the Range Selectors at the top of the timeline to change the perspective.

The increments go from one day to one year. To limit the timeline to the length of the current project, click **Max**. The green, vertical line is the current time and sweeps across the resource timeline automatically.

4. To assign a task to a resource, drag the task bar from the *Unassigned Project Tasks* pane up until the resource's name is highlighted, and then release the task bar.

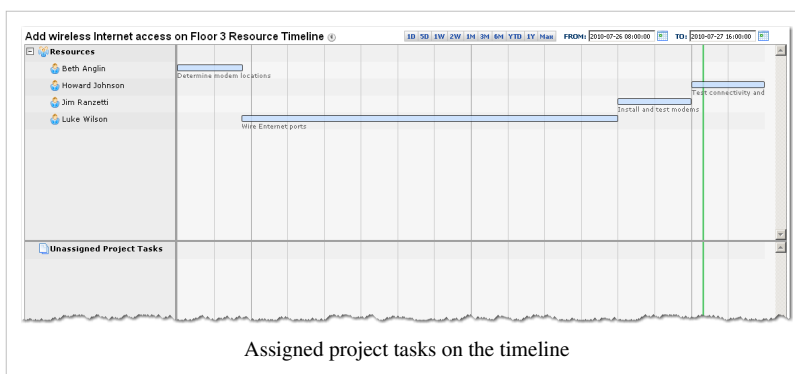


5. To *unassign* a task to a resource, grab the task bar with the mouse and move it back down into the *Unassigned Project Tasks* list. A dialog box appears asking you to confirm the action.

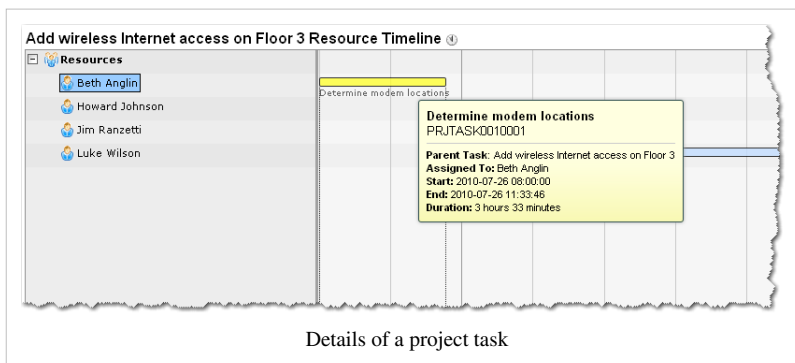
6. Click **OK**.

7. Assign the remaining tasks to resources.

When you are done, the *Unassigned Project Tasks* pane is empty, and all the tasks are aligned with their resources.



8. To view the specifics of a task, hover the cursor over the task bar in the Timeline.



9. To edit or delete a task record, double-click on the bar.

The task record shows a log of all resource assignments for that task in the **Activity** field.

What Do I Do Next?

If you have finished selecting user resources for your project and have

assigned tasks to the resources in the timeline, you are ready to **start the project**.

Using Task Resources with Other Planned Tasks

If the Project Management v2 Plugin is activated, the Task Resources related lists and timeline are available on any planned task, using the method detailed above.

References

- [1] https://docs.servicenow.com/bundle/jakarta-it-business-management/page/product/project-management/concept/c_TaskResources.html

Managing Planned Tasks in Project



Note: This article applies to Fuji and earlier releases. For more current information, see *Project Task Relationship and Dependencies* ^[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

Project management enables you to create child tasks that are nested under a parent task and successor tasks that are dependent on the completion of a predecessor task. This page explains how to create such relationships and dependencies.

To create and edit portfolios, projects, and tasks, users must have the `project_manager` role in their user profile record. See *Installed with Project Management* for more information on roles in the Project application.

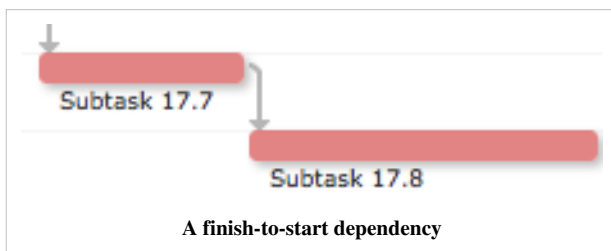
Key Concepts and Terms for This Topic

- **Predecessor task:** a project task that, upon completion, is followed by another task. A predecessor task has a dependent relationship with its successor task.
- **Successor task:** a project task that cannot start until another task finishes. The successor task has a dependent relationship with its predecessor task.
- **Lag time:** a manually specified time break between predecessor and successor tasks. Configure a lag time, if necessary, when creating a predecessor-successor dependency.

The system applies the project schedule in the v3 application (available starting with the Dublin release). For the v2 application, the system converts lag time to hours and does not consider the project schedule when applying lag time.

- **Parent task:** a project task with smaller tasks, referred to as child tasks, underneath it. Child tasks break down the work of a parent task into more manageable subsets. Certain fields for child tasks, such as planned end date, roll up and affect the same field in the parent task.
- **Child task:** a project task that is a subset of a larger task. Child task start dates cannot occur before the start date of the parent task.
- **Rollup task:** another term for a parent task in the context of aggregating child task items, such as effort or resources, into a larger parent task calculation. All fields on rollup task forms are read-only in the v3 application.
- **Roll down:** state changes roll down from the project to project tasks, and from parent tasks to child tasks in the v3 application.

Task Dependencies



A task dependency is created when one task is forced to start after another task finishes. For example, subtask 17.8 can only start when the **State** field of subtask 17.7 changes to **Closed**. The Project application only supports this kind of dependency, referred to as a finish-to-start dependency.

Creating a Task Dependency

The easiest way to create a task dependency is with the Gantt chart.

Another option is to use related lists to create dependencies:

1. If the successor task does not already exist, navigate to the project form and create it.
Do not create the task from the predecessor task form. Doing so creates a parent-child relationship.
2. Navigate to the predecessor task.
3. Configure the related lists for the Project Task form and add **Planned Task Relationship > Parent**. Do not select **Predecessor of** or **Successor of**.

This adds the **Planned Task Relationships** related list to the Project Task form. This related list shows successor tasks.

4. In the **Planned Task Relationships** related list, click **New**.
5. On the Planned Task Relationship form, click the lookup icon and select the appropriate successor task.
6. Verify that the relationship **Type** is **Predecessor of::Successor**. *Do not* change this relationship type.
7. Enter the **Lag** time, if any, in either days or hours.
 - In the v3 application, the lag time takes the project schedule into consideration. If the lag time is 10 hours and the default schedule of an 8-hour work day is in use, the lag time pushes the task to the following day to cover the additional hours.
 - In the v2 application, the system converts the lag time to total hours. Therefore, a lag value of **1** day is equivalent to 24 hours. The lag time does not take the project schedule into consideration.
8. Click **Submit**.

Task Time Constraints

The Project Task form includes a the **Time Constraint** field: either **Start ASAP** or **Start on specific date**.

- If the successor task is set to **Start ASAP**:
The successor task appears on the Gantt chart as starting immediately after the predecessor completes without any lag time. However, the successor task can start on a later date if it has a value in the **Lag** field. To enter a lag value, double-click the relationship line in the Gantt chart and enter a **Lag** value. Alternatively, open the predecessor task and enter a **Lag** value for the successor task in the **Planned Task Relationships** related list.
- If the successor task is set to a **Start on Specific Date** that is *later* than the finish date of the predecessor:
The successor task starts at the time specified. On the Gantt chart, a lag appears just as if you set the **Lag** value on the relationship. However, the actual **Lag** value is not actually modified.
- If the successor task is set to a **Start on Specific Date** that is *earlier* than the finish date of the predecessor:
ServiceNow changes the successor task time constraint to **Start ASAP** and the task starts immediately after the predecessor finishes, unless a **Lag** value exists.

State Changes on Tasks in Dependencies

Dependencies do not affect the ability to change the state of predecessor or successor tasks. For example, if a project is already in progress, you can still change a successor task to **Work in Progress** even if the predecessor task has not finished. Also modify the successor task to start on specified date that is earlier than the planned end date of the predecessor. Although this would violate the dependency for planning purposes, ServiceNow provides this kind of flexibility in modifying the project. You can also perform actions like closing a successor task, and then opening a predecessor task. Although you are allowed to make these kinds of modifications to predecessors and successors, the related project tasks and the way they are represented in the Gantt chart might show unexpected results.



Modifying Dependencies

To modify an existing dependency from the Gantt chart:

- 1. Double click the relationship line in the Gantt chart.
- 2. Enter a different task in the **Successor** task field.

To modify an existing dependency from a related list:

- 1. Open the Project Task form for the successor task and click the existing predecessor task in the **Planned task relationships** related list.
- 2. Enter a different task in the **Successor** field.

Removing Dependencies

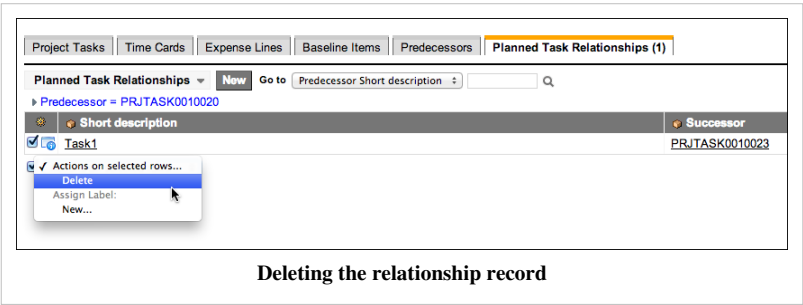
Remove a dependency that is no longer necessary from either the Gantt chart or the Project Task form. Removing the dependency also deletes the dependency record in the Planned Task Relationship table.

To remove a dependency from the Gantt chart:

- 1. Double-click the relationship.
- 2. In the **Planned Task Relationship** form, click **Delete**.

To remove a dependency from the Project Task form:

- 1. Open the predecessor task in the Project Task form and go to the **Planned Task Relationships** related list.
- 2. Select the check box beside the relationship being removed.
- 3. On the **Actions on selected rows** menu, select **Delete**.



Parent-Child Task Relationships

If a task is relatively large and requires several users with different skills to manage, break the task into subtasks and create parent-child relationships. A child task should be a relatively

smaller, manageable size of work. When you group child tasks together under a parent, values such as **Estimated cost** aggregate and roll up to the parent task. So the parent task takes on the form of a *summary task* or *rollup task* for its child tasks. **Planned start date** and **Planned end date** rollup occurs when you create child tasks: the duration of the parent automatically adjusts to *cover* its child tasks.

A parent-child relationship is different from a dependency relationship. In a dependency, one task must finish before another begins. In a parent-child relationship, any number of tasks can be nested under a parent task with or without any dependencies. When you create a parent-child relationship, the parent task number is saved in the **Parent** field in the Project Tasks table. All project management tasks have a parent: either another project task or the project itself.

Creating a Parent-Child Task Relationship

The easiest method to create parent-child relationships is on the Gantt chart.

To create parent-child relationships with related lists:

1. Navigate to the parent task in the relationship.
2. In the **Project Tasks** related list, click **New**.

The same Project Task form appears for all tasks regardless of the parent-child relationship.

3. Create the task and click **Submit**.

The newly created task becomes the child task in the relationship.

To help remember what the parent of any task is, view the breadcrumb at the top of the Project Task form. It is also helpful to configure the form layout to include the **Parent** field.



Parent: PRJ99999999

The Parent field. In this example the project is the parent.

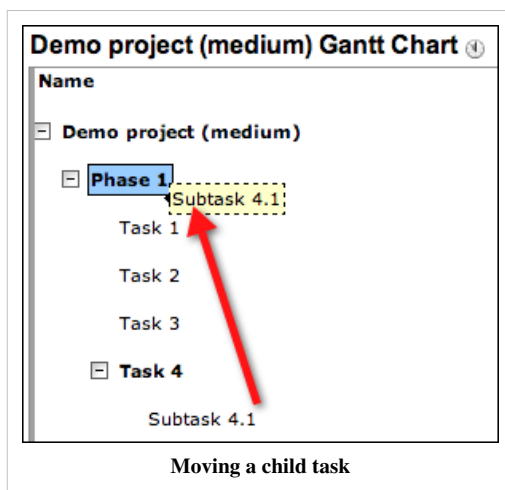
Changing the Parent Task

To change the parent of a child task:

1. Navigate to the child task in the relationship.
2. Configure the form to add the **Parent** field, if needed.
3. In the **Parent** field, select the new parent task for this child task. To have the task stand alone in the project, select a project instead of a task.

Modifying Parent-Child Relationships

To modify a parent-child relationship by using the Gantt chart, drag a child task to any other task (no matter whether it is a child or parent task). If you drag a child task up to the project name, it becomes a standalone task and is no longer considered a child task.



To modify a parent-child relationship from a related list:

1. Navigate to the child task in the relationship.
2. Configure the form to add the **Parent** field if needed.
3. In the **Parent** field, select the new task that you want the child task assigned to. To have the task stand alone in the project, select a project instead of a task.

Unlike a dependency, a parent-child relationship is not saved as a record in any table. The only modification that takes place when a parent-child relationship is modified is the **Parent** field in the child task record.

Time Constraints in Parent-Child Relationships

- If a child task is set to **Start ASAP**, the child task starts at the same time the parent task starts (as long as it does not have dependencies with other child tasks).
- If the parent task is set to **Start ASAP** and child tasks are set to **Start on specific date**, the earliest child task start date determines the start date of the parent (assuming no other dependencies). In this case, the parent's **Time constraint** field remains **Start ASAP** but the actual start date is changed to the start date of the earliest child task.
- If both the parent and first child task are set to **Start on specific date** but the first child starts later than the parent, the parent start date remains **Start on Specific Date** but the actual start date is pushed to the start date of the child. For example, if the parent task starts on October 1 and the earliest child task starts on October 2, the **Planned start date** of the parent is changed to October 2.
- Child precedence also applies to end dates. If the child task's estimated end date is later than the parent task's end date, the parent task's estimated end date extends to cover the child. For actual values, a parent has the same start date as the earliest start date of its children and the latest actual end date as the latest end date of its children, assuming the child tasks are **Closed Complete**. If the child tasks are not in the **Closed Complete** state, the actual end date of the parent is empty.
- For the planned start date of the parent task:
 - The planned start date is the earliest planned start date of all the children that do not have an actual start date.
 - If all child tasks have actual start dates, the parent task's planned start date is set to the actual start date.
- For the planned end date of the parent task, the latest planned end date or actual end date of the child tasks determines the parent's planned end date.

Dependencies with Parent or Child Tasks

Options exist to create predecessor-successor relationships between child tasks with different parents, between two different parent tasks, or between a child task and another parent task. However, if the predecessor task finishes after the successor task starts, creating a dependency between child tasks that have different parents is not allowed.

Parent-Child (Rollup) Task Calculations

Rollups involve date changes, state changes, and value calculations.

- Date changes involve modifying the planned start or end date of a parent task based on those values in child tasks.
- State changes involve modifying the state of the project record or parent task records if all child records are set to a certain state.
- Calculations involve summing the values of child tasks and then automatically updating the parent to reflect a new total.

Rollups work differently on these fields in the v3 application (available starting with the Dublin release):

- **Planned Start date:** set to read only for parent tasks. Remains editable for the project record (also considered the top-level task).
- **Planned End Date:** becomes read only.
- **Planned Duration:** becomes read only.
- **Actual Start Date:** becomes read only.
- **Actual end date:** becomes read only.
- **State:** becomes read only.

Duration Rollups

Rollups are calculated for the following:

- **Planned duration and planned effort:** the sum of all planned duration and planned effort values for all child tasks.
- **Actual duration and actual effort:** the sum of all actual duration and actual effort values. Actual duration and actual effort values are calculated when all child tasks are in the **Closed Complete** state. Actual effort values can include rollups from time cards.



Note: Verify that the time card property *Update the task's 'Actual effort' based on the hours entered in the time card* is enabled. Navigate to **Time cards > Administration > Properties** to enable this property.

Cost Rollups

Cost calculations roll up when the the Project Management Costing Add-on is active.

- **Estimated cost:** the sum of all cost estimates at the beginning of a project. Estimated costs of child tasks roll up to parent tasks and to the project.
- **Actual cost:** by default for the project, the sum of all costs of all the project's expense lines, which are typically associated with a a time card and a labor rate. To track costs, define rate cards for the task and labor expenses. These rate cards automatically generate expense lines showing actual expenditures, which are associated with the projects. If rate cards are defined, the task expense lines are generated as each project task closes, and labor expense lines are generated when time cards are approved. Expense lines are visible in the **Expense Lines** related list, which requires the **Advanced view** on a both Project and Project Task forms.

For actual costs of child tasks to properly roll up to the project and be added to project expense lines, the following must be true:

- The com.snc.project.rollup.cost property must be set to **true**. To enable this property, navigate to **Project > Administration > Properties** and select the **Enable project cost rollup** check box.
- The glide.cost_mgmt.process_task_top_task property must be set to false. To enable this property, navigate to **Financial Management > Admin > Properties** and select the **When creating a task expense line should the system also create expense lines for the task's top task** check box.
- The glide.cost_mgmt.calc_actual_cost property must be set to true. To enable this property, navigate to **Financial Management > Admin > Properties** and select the **For planned tasks types, calculate the actual cost field using the total of expense lines for the task** check box.

Enabling Cost Rollup Calculations

To use rollup calculations:

1. Navigate to **Project > Administration > Properties**.
2. **Select Enable project cost rollup** and click **Save**.

Rollup values are read-only on forms. Point to the icon beside the field for a tooltip message.

Estimated cost: \$1,099.00

Net value: 0.00

A rollup task for an estimated cost. The field is not directly editable on the parent.

Project State Rollups and Roll Downs

Project task states roll up. The state of parent tasks becomes read only in the v3 application, and changes automatically when you change the states of child tasks.

Project task states can roll up if:

- The state of the child task is manually changed and there are no other conditions on the parent task.
- The state of the child task is changed to **Work in Progress** or **Closed**. These states roll up to the parent. **Pending** and **Open** do not roll up to the parent task.

Project states can also roll down in the v3 application. If you change the state of a project to closed, all tasks under it change to the default closed value (**Closed Complete**). If a closed project or closed task is reopened, all tasks under it change as follows:

- Project or parent changed from closed to **Pending** or **Open**: child tasks change to **Open**.
- Project or parent changed from closed to **Work in Progress**:
 - Child tasks with a **Start on** date that has passed are changed to start **ASAP** and the state is changed to **Work in Progress**.
 - Child tasks with a **Start on** date that has not yet passed retain the same start on date but the state is changed to **Open**.

References

- [1] https://docs.servicenow.com/bundle/jakarta-it-business-management/page/product/project-management/concept/c_ProjectTaskRelationDepend.html

Assignment Rules

Assignments



Note: This article applies to Fuji. For more current information, see *Define Assignment Rules* ^[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

ServiceNow can automatically assign a task to a user or group based on pre-defined conditions by using:

- Data lookup rules
- Assignment rules

Data Lookup Rules

Data lookup rules offer a generic way to change any field value, not just assignment fields. Data lookup rules offer the following improvements over the **Assignment** module:

- Ability to change any field value not just an assignment field
- More options to define when a rule runs:
 - On form change (Allows assignment rules to apply to unsaved changes on a form)
 - On record insert
 - On record update
- Option to replace existing values (including default values)



Note: You can define data lookup and **Assignment** rules at the same time. The system ignores any duplicate rules after an incident has been assigned unless you are using a data lookup definition option to replace existing values.

Assignment Rules Module

The **Assignment** rules module allows you to automatically set a value in the **assigned_to** and **assignment_group** fields when a set of conditions occurs. An assignment rule must also meet these additional criteria to run:

- The task record has been created or updated. Assignment rules do not apply to unsaved changes on a form.
 - The task record must be unassigned. The record cannot have an existing value for either the **assigned_to** or **assignment_group** fields. Assignment rules cannot overwrite existing assignments (including assignments set by a default value or a previously run assignment rule).
 - The assignment rule is the first rule that matches the table and conditions. If more than one assignment rule matches the conditions, only the rule with the lowest order value runs.
-

Precedence between Assignment Rules and Business Rules

When creating new assignment rules, keep in mind that business rules can take precedence over assignment rules in certain circumstances. Assignment rules and business rules run in the following order:

- 1. All before business rules that run on a record insert with an order value less than 1000.
- 2. The first assignment rule with the lowest execution order and matching condition.
- 3. All before business rules that run on a record insert with an order value greater than or equal to 1000.
- 4. All after business rules that run on record insert.

Defining Assignment Rules

You can define assignment rules either from Data Lookup and Record Matching Support, from the **Assignment** module, or from workflow Task Activities.

Data Lookup Rules

To define an assignment rule with Data Lookup and Record Matching Support:

- 1. Navigate to **System Policy > Rules > Assignment Lookup Rules**.
- 2. Click **New**.
- 3. Populate the assignment data lookup fields (see table).
- 4. Click **Submit**.

Assignment Data Lookup

Category:Software

Subcategory:-- None --

Configuration Item:

Location:

Assignment Group:Software

Assigned To:

Active:☒

Order:100

Submit

The Assignment Data Lookup form

Field	Input Value
Category	Select the category the data lookup matches against.
Subcategory	Select the subcategory the data lookup matches against.
Configuration Item	Select the configuration item the data lookup matches against.
Location	Select the location the data lookup matches against.
Assignment Group	Select the assignment group to assign the incident to.
Assigned To	Select the user to assign the incident to.
Active	Set to Yes to run the rule or No to deactivate the rule.
Order	Enter the order in which the rule runs compared to other rules on the same table. The Data Lookup Plugin runs the rule with the lowest order and matching values.



Note: The assignment lookup rule assigns incidents matching the values in the matcher fields (**Category**, **Subcategory**, **Configuration Item**, and **Location**) to the values in the setter fields (**Assignment Group** and **Assigned To**). A valid assignment lookup rule requires at least one matcher field and one setter field.

Assignment Module Rules

To define an assignment rule:

- 1. Navigate to **System Policy > Rules > Assignment** and click **New**.
- 2. Complete the fields on the form (see table).

Assignment Rule

UpdateDelete

Use Assignment Rules to automatically assign tasks to users and groups.

More Info

Name:Database or SoftwareActive:☐

Applies To

Select a Table and specify the Conditions that must be met before the task is assigned to the user or group. The rule is applied only if the task is not already assigned to another user or group.

Table:Incident [Incident]

Conditions:

Category: is one of: RequestInquiry / HelpSoftwareHardware

Assign To

User:

Group:Software

Script

Enter a script to further customize the assignment rule. Scripts provide access to current.variable_pool variables.

Script:

UpdateDelete

The Assignment Rule form for software requests



Note: You might need to configure the form to see all the fields below.

Field	Description
Name	The descriptive name for the assignment rule.
Active	An indicator of whether the assignment rule is active. Only active assignment rules take effect.
Applies to	
Table	The table with the records that the assignment rule applies to.
<div><div></div><div>Note: The list shows only tables and database views that are in the same scope as the assignment rule (starting with the Fuji release).</div></div> <p>If you select a custom table that extends the task table, you must clear the instance cache by navigating to <a href="https://<instance_name>.service-now.com/cache.do">https://<instance_name>.service-now.com/cache.do in order for the assignment rule to work.</p>	
Conditions	The conditions in which the assignment rule will apply.
Assign to	
User	The user which will be assigned the event.
Group	The group which will be assigned the event.
Script	
Script	A script to determine advanced assignment rule functionality. The current.variable_pool set of variables is available.
Other fields	

Match	Choices are:
Conditions	<ul style="list-style-type: none">• Any - Assignment Rule will apply if any of the conditions are met.• All - Assignment rule will apply if all of the conditions are met.
Order	A number to determine priority over conflicting assignment rules. If there are conflicting assignment rules, rules with lower Order values will take precedence over rules with higher Order values. If the Order values are all set to the same number the assignment rule with the first matching condition is run with precedence over the others. Only one assignment rule will run against a record, the first with a matching condition.

Workflow Assignments

An alternative to creating data lookup or assignment rules is to create one or more workflow tasks that assign a task record as part of a workflow. Consider using a workflow for assignment if your process includes multiple steps or conditions such as requiring a particular group approve a request.

When using a workflow to manage task assignments, add a brief timer activity to the start of the workflow. Without this timer activity, the workflow runs before the parent record, the `current` record, is inserted into the database. After the timer activity completes, the workflow resumes using the parent record information from the database instead of the original `current`. Note that pausing a workflow in this way does not change a default workflow to a deferred workflow. For more information on how the workflow engine interacts with the database, see workflow engine operation order.

Examples

These examples illustrate creating assignment rules using the Data Lookup Plugin, the Condition Editor, and with a script.

Assignment Lookup Rules

In the following example, the Data Lookup Plugin assignment lookup rule automatically assigns any incident with the **Category** of Request and **Subcategory** of Password Reset to Fred Luddy.

Assignment Data Lookup

Category:Request

Subcategory:Password Reset

Configuration Item:

Location:

Assignment Group:

Assigned To:Fred Luddy

Active:☒

Order:100

UpdateDelete

Assignment data lookup example

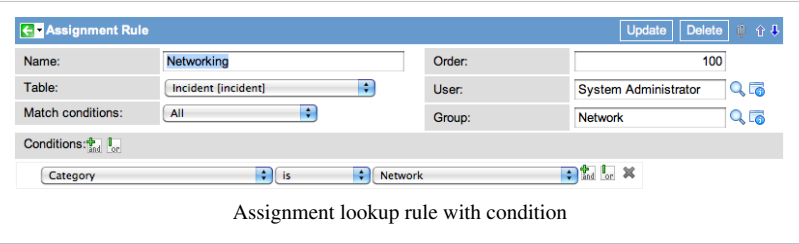
Baseline Assignment Rules

A baseline instance contains the following assignment rules.

Name	Table	Conditions	Assigned to User, Assigned to Group, or Script
Networking	incident	Category is Network	User: System Administrator Group: Network
Database or Software	incident	Category is one of Request, Inquiry / Help, Software, Hardware	User: System Administrator Group: Software
SC Item fulfillment - Field Services	Ticket	Parent.Task type is Request item	Group: Field Services
Release Planning	release_phase	Name is Plan	Script: <div>current.release.product.service.assigned_to;</div>
IT Hardware	sc_req_item	Approval is Approved and Item.Category is Request Computers and Hardware	User: System Administrator Group: Hardware
Service Desk	incident	Active is true	Group: Service Desk

Condition Editor

In the following example, the assignment rule uses a condition statement to automatically assign any incident opened in the **Network** category to the System Administrator in the **Network** assignment group.



Script

In the following example, the assignment rules use JavaScript code to assign incidents to a particular assignment group based on the incident category:

This Incident Category	Is Assigned to this Assignment Group
Hardware	Hardware
Software	Software
Malware	Security

Assignment rule with script

The following script requires personalizing the instance to add the **Malware** category and the **Security** assignment group.

```
if (current.category == "Hardware") {
    current.assignment_group.setDisplayValue("Hardware");
} else if (current.category == "Software") {
    current.assignment_group.setDisplayValue("Software");
} else if (current.category == "Malware") {
    current.assignment_group.setDisplayValue("Security");
}
```

References

[1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/administer/task-table/concept/c_DefineAssignmentRules.html

Configuring Group Types for Assignment Groups



Note: This article applies to Fuji. For more current information, see *Configure Group Types for Assignment Groups*^[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 **Type** field can be used to define categories of Groups, which can be used to filter out assignment groups based on the group type using a reference qualifier on **task.assignment_group**.

For example, when selecting an assignment group from the **Incident** form, **Type** can be used to filter groups based on whether they are typically involved in the Incident Management process. Groups such as **Network** or **Help Desk** which typically are involved would be displayed, while groups such as **HR** or **New York** would be omitted.

In the base system:

- The type **catalog** is provided for use. The type **survey** is also provided, starting with the Eureka release.
- The reference qualifier on **task.assignment_group** filters on **Type=null**.
- A reference qualifier named **GetGroupFilter** is available to filter for group types.



Note: With the introduction of Dictionary Overrides, administrators can filter for a group type on an extended table with a simple reference qualifier override.

Defining Group Types

To define a group's type, navigate to **User Administration > Groups** and select the desired group. Enter one or more types in the **Type** field, separated by commas (no spaces). For example, to define a group's type as **incident** and **problem**, enter: **incident,problem**.



Note: Because the default behavior of **task.assignment_group** is to filter out groups with group types defined, adding a type to a group will filter it out of the **Assignment Group** field on task. To change the behavior, see below.

Setting Up the Reference Qualifier

The reference qualifier on **task.assignment_group** filters which groups are available to be used as an assignment group.

Note: changing the reference qualifier on **task.assignment_group** affects *every* table which extends the Task Table. Since Spring 2010 Stable 2, Dictionary Overrides can be used to define reference qualifiers on tables that extend Task without affecting other task tables.

There are two ways to use reference qualifiers to filter assignment group types:

- Using a Simple Reference Qualifier
- Using Advanced Reference Qualifiers

Using a Simple Reference Qualifier

In the base system, the simple reference qualifier **type=null** is used to allow groups with no defined type to be selected.

In the same way, if there is one specific type which should be available from the Assignment Group, specify it in the same format. For example, if there is a group type **assignment** which all assignment groups have, use the reference qualifier **type=assignment** to return only those groups.

Using Advanced Reference Qualifiers

If a simple reference qualifier doesn't provide enough control over filtering, an advanced filtering function can be defined in a business rule and called using advanced reference qualifiers.

Using GetGroupFilter

The business rule **GetGroupFilter** (available in the base system) can be called by an advanced reference qualifier to filter group types based on arguments.

For example, the following reference qualifier would restrict the choice list to groups with type **database** or **network**:

```
javascript:GetGroupFilter('database,network')
```

The **GetGroupFilter** function can also be called from reference qualifier functions which you create.

Creating an Advanced Reference Qualifier

For more advanced filtering, create an Advanced Reference Qualifier for the **Assignment Group** field on the **Task** table. Note that this reference qualifier will apply to all tables based on **Task**, unless Dictionary Overrides are used.

The following example reference qualifier assumes that you have created a group type named 'database'. If the task is an incident record and the category is 'database', then the reference qualifier will only select database groups. In all other circumstances it will select any group which is not a catalog group.

Business Rule to define the Advanced Reference Qualifier

The screenshot shows the 'Business Rule' configuration window. The 'Name' field is 'Task Assignment Filter'. The 'Table' is 'Global [global]'. The 'Order' is '100'. The 'Active' checkbox is checked. The 'When' dropdown is set to 'after'. The 'Insert', 'Update', 'Delete', and 'Query' checkboxes are all unchecked. The 'Condition' field is empty. The 'Script' field contains the following JavaScript code:

```
function TaskAssignmentFilter() {
    var classname = current.getRecordClassName();
    var filter = "type=null";
    if (classname == "incident" && current.category == "database") {
        filter = GetGroupFilter("database");
    }
    else {
        // append 'catalog' group exclusion to the filter
        var cat = new GlideRecord("sys_user_group_type");
        cat.addQuery("name", "catalog");
        cat.query();
        if (cat.next()) {
            filter += "^ORtype!=" + cat.sys_id;
        }
    }
}
```

On the right side, there is a 'Select variables:' panel with a tree view showing 'Fields', 'GlideRecord', 'GlideElement', 'System', and 'GlideAggregate'.

```
function TaskAssignmentFilter() {
    var classname = current.getRecordClassName();
    var filter = "type=null";
    if (classname == "incident" && current.category == "database") {
        filter = GetGroupFilter("database");
    }
    else {
        // append exclusion for 'catalog' to the filter
        var cat = new GlideRecord("sys_user_group_type");
        cat.addQuery("name", "catalog");
        cat.query();
        if (cat.next()) {
            filter += "^ORtype!=" + cat.sys_id;
        }
    }
}
```

```
}  
gs.log("TaskAssignmentFilter: " + filter);  
return filter;  
}
```

Once the business rule above is defined, it can be called by the following reference qualifier:

```
javascript:TaskAssignmentFilter()
```

Caution Regarding Tree Picker

Caution must be exercised if the Tree Picker is used in conjunction with group type filtering. The reference qualifier function must select groups at all levels of the tree since a group is only selectable if its parent is selectable.

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/administer/users-and-groups/concept/c_ConfigGroupTypesForAssignGroups.html

Approvals

Approvals



Note: This article applies to Fuji. For more current information, see Approvals ^[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

Approvals can be defined for all tasks and allow users or groups to be associated with a task, for the purpose of either approving or rejecting that task. Approvals are defined by navigating to **System Policy > Approvals**.

The following information defines an approval:

Field	Input Value
Approver	A reference to the user who is responsible for approving the related record.
State	Choices are: <ul style="list-style-type: none"> • Not Yet Requested (This state indicates that you are not yet asking your approvers to approve this request. Until you set the status to Requested they will receive no email notifications about the request.) • Requested • Approved • Rejected
Approving	A document_id reference field to the record being approved, on any table.
Comments	A journal field for storing comments regarding the approval.
Approval Summarizer	A formatter that displays key fields relevant to the approval from the referenced document. This summarizer will not display if there is no record referenced.

Approval Status

The approval status of a change request is determined by looking at the current status of all the approvers. If any approver has rejected the change, the approval status will be Rejected. If all approvers have approved the change, the approval status will be Approved. If all approvers are in the Not Requested status or if there are no approvers, the change status will be Not Requested, otherwise the status will be Requested.

For added flexibility when creating approvals, including the ability to set up an "one of" approval where only one person of a group of approvers needs to approve, consider using Workflows.

Generating Approvals

Generating Approvals using Workflows

Workflows are a powerful and flexible method of generating approvals. Use workflows to create group approvals and user approvals. A variety of variables are available to fine tune the approval process, including the actions that occur when approval or rejection take place. When a workflow activity generates an approval record, ServiceNow automatically populates the **Workflow activity** field on the approval record with a reference to the activity. Do not use this field when creating business logic. For more information, see Approval and Rollback Activities.

Generating an Approval using Approval Rules

The system can automatically generate an approval request to individuals or groups when specific criteria are met. The automatic generation of approval requests is driven using the System Policy feature.

In the sample below, a change opened in the category **network** is assigned to the System Administrator:

Approval Rules

Update

Name:Network Approvals

User:System Administrator

Table:Change

Group:

Match conditions:All

Execution Order:100

Conditions: Add

Add Condition

Category

contains

network

When an approver is automatically added based on approval rules, the status of the approval automatically defaults to "Requested".

Generating Approvals using the Approvers Related List

It is also possible to manually add approvers to a request. Additional approvers can be added by clicking the "Edit" button in the Approvers section near the bottom of a request.

When an approver is added manually, the status for that approver defaults to "Not Requested". When the status of the approver is changed to "Requested", the approver will be sent an email requesting approval action.



Note: You may need to enable the edit button for the related list.

Multiple Approvers

With multiple approvers, all approvers must authorize the request before the status will change to "Approved". Should any approver reject the request, the status will immediately be set to "Rejected"

Receiving Notifications

Approval notifications will be sent at the following times:

- When an individual is assigned as an approver either automatically or manually. If a group is chosen, then all members of the group will be sent an email. By default, the email an approver receives will contain a "mailto" link that will allow the approver to either approve or reject the request directly from their email system.
- When the request reaches approved status, the person assigned to the request will receive an email indicating it has been approved.

The details contained in the emails and the points at which they are sent can be tailored using **System Definition > Business Rules** and **System Policy**.

Note for Blackberry users: In order to see the "mailto" links mentioned above to approve or reject a request (i.e. 'Click here to approve CHG55555' or 'Click here to reject CHG55555'), your Blackberry device must be using version 4.5 of their software which supports HTML emails. If your Blackberry device is using an earlier version, you will not be able to view or use the "mailto" links. However, as a workaround, users can reply to the email and simply add the statements **state:approved** or **state:rejected** within the body of the email before sending it to force the automatic approval/rejection functionality.

If you create an appropriate Inbound Email Action, you can let approvers respond to approval email notifications with a simple "yes" or "no" answer.

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/administer/service-administration/reference/r_Approvals.html

Approval Summaries



Note: This article applies to Fuji. For more current information, see *Approval Summarizer Formatter*^[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 summary at the bottom of an approval form is created by the approval summarizer formatter. The approval summarizer displays different information depending on what's being approved (such as a change request or a service catalog request, for example).

Example 1: Summary of a Change Request

Approval

Update Approve Reject Delete

Approver: David Loo Approval for: CHG00001

State: Requested

Comments:

Approval summarizer formatter

Summary of Change being approved:

Short Description:	Rollback Oracle Version
Category:	Software
Requested By	David Loo
Planned Start:	2006-04-22
Planned End:	2006-04-23
Description:	Performance of the Siebel SFA software has been severely degraded since the upgrade performed this weekend. We moved to an unsupported Oracle DB version. Need to rollback the Oracle Instance to a supported version.

Example 2: Summary of a Catalog Request

Approval

Update Delete

Approver: System Administrator

State: Not Yet Requested

Approval for: REQ0010001

Comments: ABC

Approval summarizer formatter

Summary of Request being approved:

Opened by: System Administrator

Requested for: System Administrator

Total: \$500.00

Status	Number	Description	Price	Quantity	Total
Deny	RITM0010001	A Blackberry Wireless Device	\$500.00	1	\$500.00

Activity >>

2014-04-04 06:29:37 System Administrator - Changed: Approver, State

Approver: System Administrator

State: Not Yet Requested

Update Delete

The **Deny** button allows the approver to deny one or more requested items in a multi-item request, before approving the overall request. If a requested item is denied, the workflow for that item never starts. The approver can then choose to **Accept** the item, if required.



Note: When the overall request is approved, you must ensure this **Deny** button is hidden. If this button is used after request approval, this cancels the requested item workflow, leaving the stage in an inconsistent state. Similarly, the **Accept** button on requested items should only appear before the overall request is approved or rejected.

Summarizers

Approval summarizers are stored in the Macro [sys_ui_macro] table. From the left navigation pane, select **System UI > UI Macros**. Summarizers use a naming convention of *approval_summarizer_ + '<table_name>'* (for example, **approval_summarizer_change_request** is the summarizer for change requests, while **approval_summarizer_sc_request** is the summarizer for service catalog requests).

Each summarizer is written in Jelly script, which is used by ServiceNow to define internal forms. The script is stored in the large XML field at the bottom of the UI Macro form.

Changing a Summarizer

You can modify existing approval summaries to include additional information. These are advanced customizations that might not be appropriate for all implementations and require creating a custom form.

1. Navigate to **System UI > UI Macros**.
2. Open the summarizer you want to change.
3. Copy the script to another location before editing, in case you need to revert it.
4. Modify the script.
5. Click **Update**.

Creating a New Custom Summarizer

If you added to the instance a new table that has approvals, you can add a custom summarizer by creating a new UI macro.

1. Navigate to **System UI > UI Macros**.
2. Click **New**.
3. Give the macro a name that follows the summarizer naming convention:
approval_summarizer_<tablename>
4. Complete the rest of the form and click **Submit**.
5. Add the custom summarizer to the appropriate form.

References

- [1] <https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/administer/service-administration/reference/approval-summarizer.html>

Execution Order of Scripts and Engines



Note: This article applies to Fuji and earlier releases. For more current information, see *Execution Order of Scripts and Engines*^[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

Scripts, assignment rules, escalations, and engines all take effect in relation to a database operation, such as *insert* or *update*. In many cases, the order of these events is important.



Note: Client-based code that executes in the browser, using Ajax or running as Javascript, always executes before the form submission to the server.

Order

The order of execution is as follows:

1. *Before* business rules: Scripts configured to execute before the database operation with an order less than 1000.
2. *Before* engines. The following are not executed in any specific order:
 - Approval engine (for task and sys_approval_approver tables)
 - Assignment rules engine (for task tables)
 - Data policy engine
 - Escalation engine
 - Field normalization engine
 - Role engine - keeps role changes in sync with sys_user_has_role table (for sys_user, sys_user_group, sys_user_grmember, and sys_user_role tables)
 - Execution plan engine (for task tables)
 - Update version engine - creates version entry when sys_update_xml entry is written (for sys_update_xml table)
 - Workflow engine (for default workflows)
3. *Before* business rules: Scripts configured to execute before the database operation with an order greater than or equal to 1000.
4. The data base operation (insert, update, delete).
5. *After* business rules: Scripts configured to execute after the database operation with an order less than 1000.
6. *After* engines. The following are not executed in any specific order:
 - Label engine
 - Listener engine
 - Table notifications engine
 - Role engine - keeps role changes in sync with sys_user_has_role table (for sys_user, sys_user_group, sys_user_grmember and sys_user_role tables)
 - Text indexing engine
 - Update sync engine
 - Data lookup engine inserts or updates
 - Workflow engine (for deferred workflows)

7. Email notifications. The following are executed based on the weight of the notification record:
 - Notifications sent on an insert, update, or delete
 - Event-based notifications
8. *After* business rules. Scripts configured to execute after the database operation with an order greater than or equal to 1000.

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/script/general-scripting/reference/r_ExecutionOrderScriptsAndEngines.html


Approval Engines

Overview

The differences in the way that companies handle their approvals as well as the differences between approvals for the various applications (such as Service Catalog Requests and Change Management) calls for supporting flexibility in setting up approvals within the ServiceNow applications.

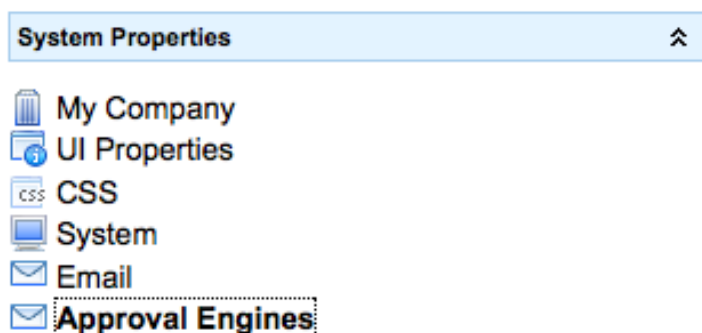
This flexibility is provided through the selection of an **Approval Engine** that is used to manage the approvals for each of the Task tables (that is, all tables that extend the *Task* table). There are three different options available for each Task table:

- **Approval Rules** - a simple set of Rules that are evaluated until one matches for the Task Table. The matching Approval Rule is used to create the user(s) that are to Approve the task. Set up Approval Rules using **System Policy -> Approvals**
- **Process Guides** - a sequence of approval steps over which you may control how approvals and rejections are handled. Set up Process Guides using **System Policy -> Process Guides**
- **Turn off Engines** - turn off both Approval Engines for this Task table. This option should be selected when **Workflow** is used to manage the Approval process for the Task table.

	Warning: Not turning off the approval engines might have a performance or behavioral impact on your instance.
---	--

Setup

1. Navigate to **System Properties > Approval Engines**.



The following page appears with the Approval Engine option for each Task table in the system.

Select the Approval Engine to be used for each of the Task tables. The valid options are:

- **Approval Rules** - Use Approval Rules to create approvals
- **Process Guides** - Use Process Guides to create approvals
- **Turn engines off** - Turn the approval engines off for this table (use this when Workflow is being used to manage approvals)

Table	Name	Approval Engine	Notes
Transfer Order	ast_transfer_order	Approval Rules	
Change Phase	change_phase	Approval Rules	
Change Request	change_request	Turn engines off	
IMAC	change_request_imac	Approval Rules	
Change Task	change_task	Process Guides	
Incident	incident	Approval Rules	
Issue	issue	Approval Rules	
KB Submission	kb_submission	Approval Rules	
Planned task	planned_task	Approval Rules	
Project	pm_project	Approval Rules	
Project Task	pm_project_task	Approval Rules	
Problem	problem	Approval Rules	
Problem Task	problem_task	Approval Rules	
Release Phase	release_phase	Approval Rules	
Feature Task	release_task	Approval Rules	
Requested Item	sc_req_item	Approval Rules	The Approval engine is only used when the Request Item has a Delivery Plan associated with it. If the Request Item is being managed by a Workflow, then the approval engines are automatically turned off.
Request	sc_request	Turn engines off	
Catalog Task	sc_task	Process Guides	
Group approval	sysapproval_group	Turn engines off	
Task	task	Approval Rules	
Ticket	ticket	Approval Rules	

Save

2. Select the Approval Engine for each Task table from the choice list.
3. Click **Save** to set the Approval Engine preferences.

These preferences are saved as System Properties that are named **glide.approval_engine.<table_name>**

Approval Rules



Note: This article applies to Fuji. For more current information, see [Approval Rules](http://docs.servicenow.com)^[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

Many organizations rely on an approval process to ensure that requests are reasonable and fit an organization's budget. The service catalog can use these classes of approvals:

- **Gating approvals:** must occur before a request can be initiated. For example, allow a manager to reject an employee's request for a company car.
- **Process approvals:** take place *within* an execution plan process that has been initiated. For example, allow the security group to reject a request for access to SSN even though the employee's manager approved it.



Note: To enable approval processes to operate smoothly, make sure that the appropriate users have the correct role, and that the role grants access to the necessary tables for users in all the relevant departments and domains.

Gating Approvals

A gating approval acts as a *gate* through which a request must pass before it can start. Until all gating approvals are met, no notifications go out, no tasks get sent to technicians, and nobody starts working on the request in question.

Generate gating approvals with:


- **Approval rules:** can apply to the service catalog as well as any other task table.
- **Item-based approvals:** flag specific catalog items as requiring specific approvals. Any requests for these items automatically require these approvals.

Setting up a Gating Approval via an Approval Rule

To set up a gating approval, navigate to **System Policy > Rules > Approval**, and create or edit a record (see table for field descriptions).

Set these to auto start the approval process

Gating approval that requires approval from Fred Luddy if the total price of a request exceeds \$1,000.00

Field	Description
Name	Name of this rule
Table	Task table to which this rule applies. For most service catalog approvals, select Request .
	 <p>Note: The list shows only tables and database views that are in the same scope as the approval rule (starting with the Fuji release).</p>
Active	Indicator of whether the rule is active (defaults to true).
Run Rule Before	Indicator of whether the rule runs before or after the request record is saved. For most approvals, select this check box.
User	User who must approve this request (can be empty).
Group	Group that must approve this request (can be empty).
Set State	Value of the approval field on the task in after this rule runs. In most cases, select Requested .
Condition	Condition under which the rule applies.
Script	An optional server script to programmatically specify who the approver should be. For example, for the one-line script <code>current.requested_for.manager</code> , ServiceNow checks the <code>requested_for</code> reference field on the current record. It then locates the <code>manager</code> field on the referenced record and assigns that person as the approver. For other examples, see the Script field on approval rules provided by ServiceNow.

Notes/Limitations

1. You can have as many rules as you want on a given table. If more than one rule apply, you'll get more than one approver.
2. You can't get duplicate approvers e.g. if two rules both want Fred Luddy to approve a particular request, the system will only create one approval entry for him.
3. By default all requests start out in a "not yet requested" approval state. Approval notifications *will not* go out until the request's approval state is set to "Requested". You can do that manually, or you can do it in script, but by far the easiest way to do it is to use the Set State field to automatically set the request to "Requested".
4. Approval rules only run when no approvals already exist for the record.

Setting up a Gating Approval Based on the Item Being Ordered

In addition to adding approvals via approval rules, you can also add approvals based on what kind of item is being ordered. We can, for example, specify that all BlackBerries need to be approved by David Loo.

To do so, simply navigate to the item in question and scroll down to the bottom where the list of required approvers are. There are two lists:

- Approved By Group -- A list of *groups* that have to approve requests for this item
- Approved By -- A list of *users* who have to approve requests for this item

Approved By **New** **Edit...** ▶ [Sc cat item = Blackberry](#) > << 1 to 1 of 1 >>

Approval group

☐ Hardware

☐ Actions on selected rows...

Approved By **New** **Edit...** ▶ [Sc cat item = Blackberry](#) > << 1 to 1 of 1 >>

Approver

☐ David Loo

☐ Actions on selected rows...

In the example above, this request must be approved by all members of the Hardware group and David Loo.

Notes/Limitations

1. As with approval rules, you are protected against duplicate entries. Thus if David Loo is a member of the hardware group, as well as being a standalone approver, he'll only get one approval request.
2. Item based approved work *in addition to* rather than *instead of* approval rules so you can (and probably will) use both.

Process Approvals

Once a request has passed its gating approvals, any relevant Execution Plans are initiated. Those plan, in turn, create a sequence of required tasks. You can add an *approval step* to an execution plan, which is configured to occur at the appropriate point.

From the left navigation pane, select **Service Catalog > Execution Plans**, and then select the plan to which to add an approval step. Then click the **New Approval** button.

Delivery Plan

Name: Blackberry Delivery Plan Total del

Short description: Delivery plan for outfitting an employee with a Blackberry a

Update **Delete**

To add an Approval

Delivery Plan Tasks **New** **New Approval** **Edit...** ▶ [Delivery plan = Blac](#)

Name **Short description**

You'll find yourself on the Approval Task screen. Just like a regular Service Catalog execution task, an approval execution task has:

- Name -- The name of this task
- Order -- Sequence of this task within the plan
- SLA -- SLA to which this task applies
- Delivery Time -- Time allowed for the completion of this task

After you create the task, right click the title bar and select Save. Two new, related lists appear at the bottom of the screen:

- Approved By Group -- A list of *groups* that must approve the request before this task is complete
- Approved By -- A list of *users* who must approve the request before this task is complete

Delivery Plan Approval Task Update

Name:	Security Approval	Delivery plan:	Blackberry Delivery Pla
SLA:		Order:	50
Auto close:	<input checked="" type="checkbox"/>	Delivery time:	Days 00 Hours 00 :00 :00
Short description:			
Instructions:			

Update Delete

Approved By Group New Edit... [Sc cat item dt approval = Security Approval](#) >

Approval group

Approved By New Edit... [Sc cat item dt approval = Security Approval](#) >

Approver	
<input type="checkbox"/>	Fred Luddy

☐ Actions on selected rows...

In our example above this security approval task must be approved by Fred Luddy.

Notes/Limitations

1. When an in-process approval is rejected, that particular line item is canceled as well, but the request itself isn't necessarily canceled. Thus if one ordered a blackberry and a laptop, and the blackberry was rejected, the laptop would continue on with its processing.

Process Guide Approvals

The default version of approval tasks allows you to specify that the approval in question be approved by:

1. One or more specific people
2. One or more groups of people

You can optionally use Process Guides instead of approval tasks. Process guides are more flexible in that they allow for:

1. "Any of" or "All of" approvals
2. Sequenced approvals

Linking a process guide to a execution task

Process guides work similarly to approval rules in that their execution is controlled via a condition.

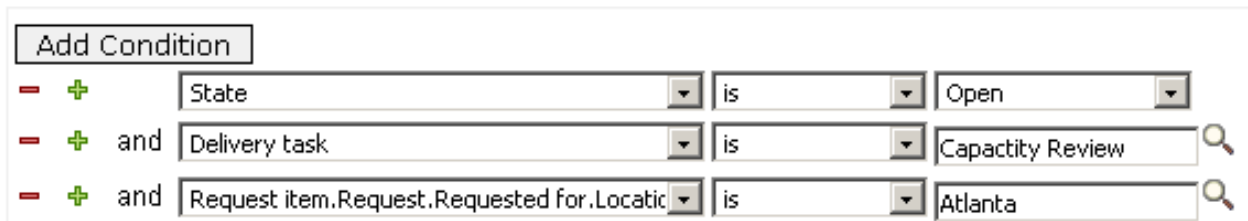
1. From the left navigation pane, select **System Policy > Process Guides**
2. Create a new guide
3. Set the table to **Catalog task**
4. Fill in a condition under which this guide should attach.

Example #1: Apply to all "Capacity Review" tasks



The screenshot shows the 'Add Condition' dialog with two conditions. The first condition is 'State is Open'. The second condition is 'Delivery task is Capacity Review'. The conditions are separated by an 'and' operator. There are red minus and green plus icons for each condition, and a magnifying glass icon at the end of the second condition.

Example #2: Apply to all "Capacity Review" tasks where there requester is in Atlanta



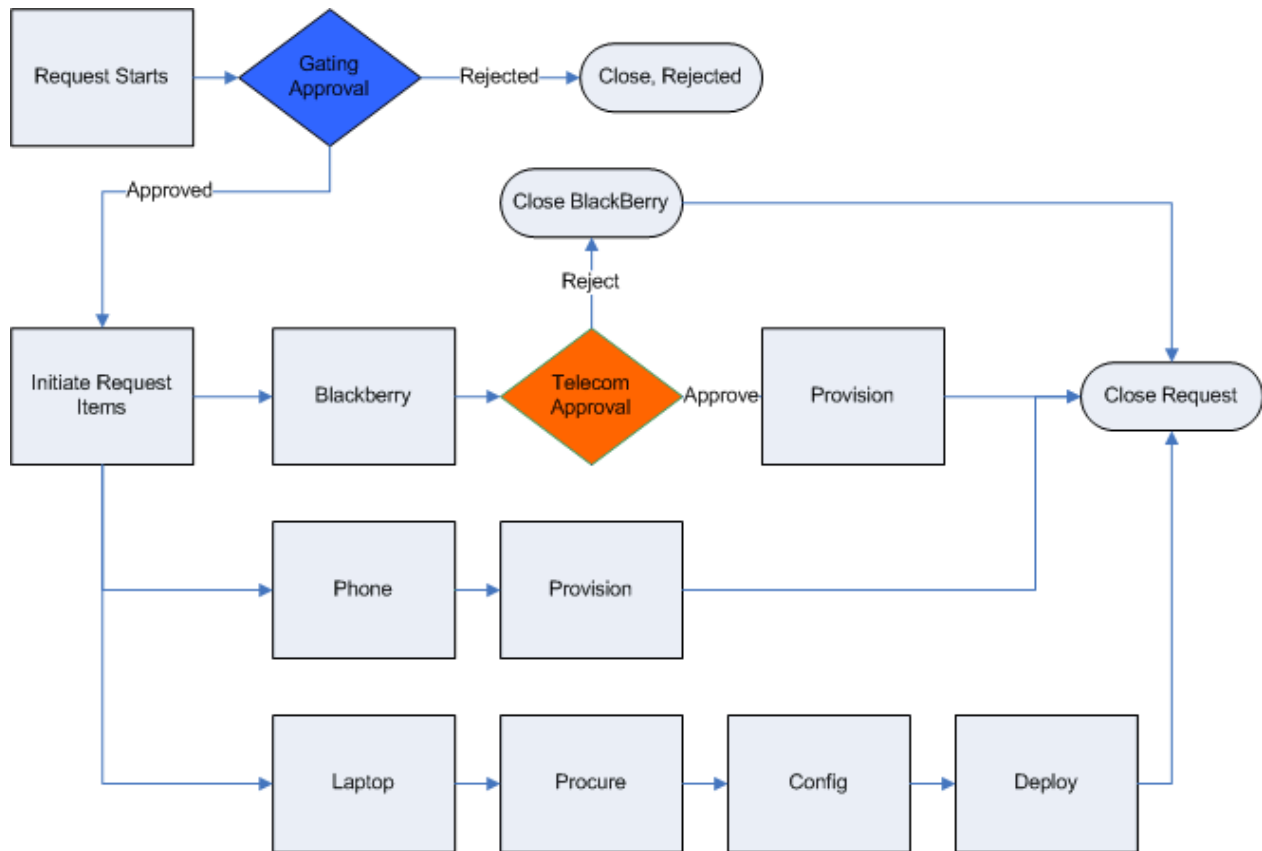
The screenshot shows the 'Add Condition' dialog with three conditions. The first condition is 'State is Open'. The second condition is 'Delivery task is Capacity Review'. The third condition is 'Request item.Request.Requested for.Locatic is Atlanta'. The conditions are separated by 'and' operators. There are red minus and green plus icons for each condition, and magnifying glass icons at the end of the second and third conditions.

Process Guide Tips and Tricks

1. All catalog tasks are generated when a request is first submitted, but tasks which aren't active yet have a state of "pending". So if you don't want to send out approval requests until a task has actually started, you'll want to add "state=open" as part of your condition.
2. There's a "Default" process guide in the system for catalog tasks with a sequence number of 10,000. It behaves exactly the same way the old, pre process guide code did in regards to approvals e.g. approvals are based on the execution task related lists.

Schematic of a Hypothetical Approval Process

In the diagram below of a hypothetical approval process, we've color coded the gating approval blue and an in-process approval orange.



References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/administer/service-administration/concept/c_ApprovalRules.html

Setting Automatic Approval Rules



Note: This article applies to Fuji. For more current information, see *Approval Rules* ^[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

Many organizations rely on an approval process to ensure that requests are reasonable and fit an organization's budget. The service catalog can use these classes of approvals:

- **Gating approvals:** must occur before a request can be initiated. For example, allow a manager to reject an employee's request for a company car.
- **Process approvals:** take place *within* an execution plan process that has been initiated. For example, allow the security group to reject a request for access to SSN even though the employee's manager approved it.



Note: To enable approval processes to operate smoothly, make sure that the appropriate users have the correct role, and that the role grants access to the necessary tables for users in all the relevant departments and domains.

Gating Approvals

A gating approval acts as a *gate* through which a request must pass before it can start. Until all gating approvals are met, no notifications go out, no tasks get sent to technicians, and nobody starts working on the request in question.

Generate gating approvals with:


- **Approval rules:** can apply to the service catalog as well as any other task table.
- **Item-based approvals:** flag specific catalog items as requiring specific approvals. Any requests for these items automatically require these approvals.

Setting up a Gating Approval via an Approval Rule

To set up a gating approval, navigate to **System Policy > Rules > Approval**, and create or edit a record (see table for field descriptions).

Set these to auto start the approval process

Gating approval that requires approval from Fred Luddy if the total price of a request exceeds \$1,000.00

Field	Description
Name	Name of this rule
Table	Task table to which this rule applies. For most service catalog approvals, select Request .
	 <p>Note: The list shows only tables and database views that are in the same scope as the approval rule (starting with the Fuji release).</p>
Active	Indicator of whether the rule is active (defaults to true).
Run Rule Before	Indicator of whether the rule runs before or after the request record is saved. For most approvals, select this check box.
User	User who must approve this request (can be empty).
Group	Group that must approve this request (can be empty).
Set State	Value of the approval field on the task in after this rule runs. In most cases, select Requested .
Condition	Condition under which the rule applies.
Script	An optional server script to programmatically specify who the approver should be. For example, for the one-line script <code>current.requested_for.manager</code> , ServiceNow checks the <code>requested_for</code> reference field on the current record. It then locates the <code>manager</code> field on the referenced record and assigns that person as the approver. For other examples, see the Script field on approval rules provided by ServiceNow.

Notes/Limitations

1. You can have as many rules as you want on a given table. If more than one rule apply, you'll get more than one approver.
2. You can't get duplicate approvers e.g. if two rules both want Fred Luddy to approve a particular request, the system will only create one approval entry for him.
3. By default all requests start out in a "not yet requested" approval state. Approval notifications *will not* go out until the request's approval state is set to "Requested". You can do that manually, or you can do it in script, but by far the easiest way to do it is to use the Set State field to automatically set the request to "Requested".
4. Approval rules only run when no approvals already exist for the record.

Setting up a Gating Approval Based on the Item Being Ordered

In addition to adding approvals via approval rules, you can also add approvals based on what kind of item is being ordered. We can, for example, specify that all BlackBerries need to be approved by David Loo.

To do so, simply navigate to the item in question and scroll down to the bottom where the list of required approvers are. There are two lists:

- Approved By Group -- A list of *groups* that have to approve requests for this item
- Approved By -- A list of *users* who have to approve requests for this item

Approved By **New** **Edit...** [Sc cat item = Blackberry >](#) 1 to 1 of 1

Approval group

☐ ☐ Hardware

☐ Actions on selected rows...

Approved By **New** **Edit...** [Sc cat item = Blackberry >](#) 1 to 1 of 1

Approver

☐ ☐ David Loo

☐ Actions on selected rows...

In the example above, this request must be approved by all members of the Hardware group and David Loo.

Notes/Limitations

1. As with approval rules, you are protected against duplicate entries. Thus if David Loo is a member of the hardware group, as well as being a standalone approver, he'll only get one approval request.
2. Item based approved work *in addition to* rather than *instead of* approval rules so you can (and probably will) use both.

Process Approvals

Once a request has passed its gating approvals, any relevant Execution Plans are initiated. Those plan, in turn, create a sequence of required tasks. You can add an *approval step* to an execution plan, which is configured to occur at the appropriate point.

From the left navigation pane, select **Service Catalog > Execution Plans**, and then select the plan to which to add an approval step. Then click the **New Approval** button.

Delivery Plan

Name: Blackberry Delivery Plan Total del

Short description: Delivery plan for outfitting an employee with a Blackberry a

Update **Delete**

To add an Approval

Delivery Plan Tasks **New** **New Approval** **Edit...** [Delivery plan = Blac](#)

Name **Short description**

You'll find yourself on the Approval Task screen. Just like a regular Service Catalog execution task, an approval execution task has:

- Name -- The name of this task
- Order -- Sequence of this task within the plan
- SLA -- SLA to which this task applies
- Delivery Time -- Time allowed for the completion of this task

After you create the task, right click the title bar and select Save. Two new, related lists appear at the bottom of the screen:

- Approved By Group -- A list of *groups* that must approve the request before this task is complete
- Approved By -- A list of *users* who must approve the request before this task is complete

Delivery Plan Approval Task Update

Name:	Security Approval	Delivery plan:	Blackberry Delivery Pla
SLA:		Order:	50
Auto close:	<input checked="" type="checkbox"/>	Delivery time:	Days 00 Hours 00 :00 :00
Short description:			
Instructions:			

Update Delete

Approved By Group New Edit... [Sc cat item dt approval = Security Approval](#) >

Approval group

Approved By New Edit... [Sc cat item dt approval = Security Approval](#) >

Approver	
<input type="checkbox"/>	Fred Luddy

☐ Actions on selected rows...

In our example above this security approval task must be approved by Fred Luddy.

Notes/Limitations

1. When an in-process approval is rejected, that particular line item is canceled as well, but the request itself isn't necessarily canceled. Thus if one ordered a blackberry and a laptop, and the blackberry was rejected, the laptop would continue on with its processing.

Process Guide Approvals

The default version of approval tasks allows you to specify that the approval in question be approved by:

1. One or more specific people
2. One or more groups of people

You can optionally use Process Guides instead of approval tasks. Process guides are more flexible in that they allow for:

1. "Any of" or "All of" approvals
2. Sequenced approvals

Linking a process guide to a execution task

Process guides work similarly to approval rules in that their execution is controlled via a condition.

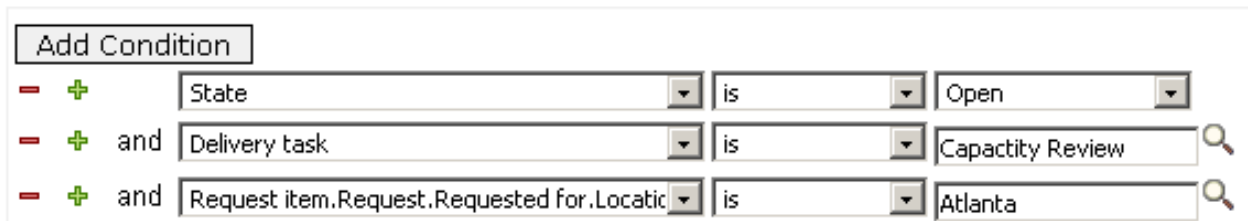
1. From the left navigation pane, select **System Policy > Process Guides**
2. Create a new guide
3. Set the table to **Catalog task**
4. Fill in a condition under which this guide should attach.

Example #1: Apply to all "Capacity Review" tasks



The screenshot shows the 'Add Condition' dialog with two conditions. The first condition is 'State is Open'. The second condition is 'Delivery task is Capacity Review'. The 'and' operator is used between the two conditions.

Example #2: Apply to all "Capacity Review" tasks where there requester is in Atlanta



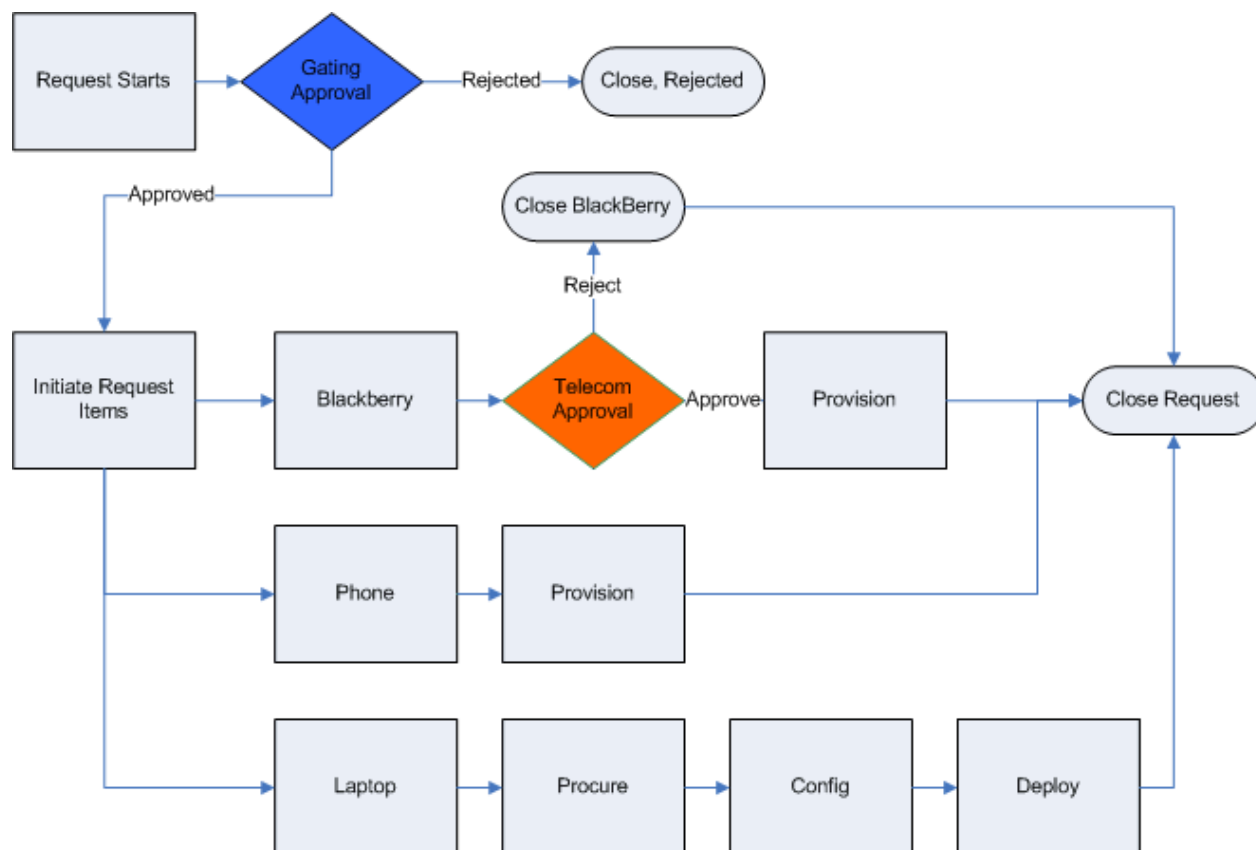
The screenshot shows the 'Add Condition' dialog with three conditions. The first condition is 'State is Open'. The second condition is 'Delivery task is Capacity Review'. The third condition is 'Request item.Request.Requested for Locatic is Atlanta'. The 'and' operator is used between the three conditions.

Process Guide Tips and Tricks

1. All catalog tasks are generated when a request is first submitted, but tasks which aren't active yet have a state of "pending". So if you don't want to send out approval requests until a task has actually started, you'll want to add "state=open" as part of your condition.
2. There's a "Default" process guide in the system for catalog tasks with a sequence number of 10,000. It behaves exactly the same way the old, pre process guide code did in regards to approvals e.g. approvals are based on the execution task related lists.

Schematic of a Hypothetical Approval Process

In the diagram below of a hypothetical approval process, we've color coded the gating approval blue and an in-process approval orange.



Process Guides



Functionality described here is obsolete since the Spring 2009 release and has been replaced by Graphical Workflow Support Plugin.

Overview

This functionality has been available since Summer, 2007. Process guides are an advanced approval engine that is available in lieu of the standard approval rules based system. By default, the system still uses approval rules to generate and manage approvals, but you have the option of turning on process guides on top of four parts of the system:

- Catalog Requests
- Change Request
- Catalog Tasks
- Change Tasks

If turned on, the process guide engine takes over control of the approval process for these areas of the system, meaning any existing approval rules you may have will stop functioning. So if you're looking to implement process guides, bear in mind that you may have to revisit your existing approval rules.

Advantages of Process Guides

Process guides offer additional functionality that isn't present in the standard approval rules based system. Specifically:

1. "Any of" approvals e.g. "either Tom, Dick, or Harry has to approve this, but all three of them do not"
2. Sequenced approvals e.g. "first ask mom. If she says yes, approve the request. If she says no, ask dad"

Anatomy of a Process Guide

A process guide is made up of two things.

1. The guide itself, which has a name and a condition under which the guide applies
 2. One or more approval steps and the relationships transitions between them
-

Where to find them

From the left navigation pane, select **System Policy > Process Guides**.

Process Guide

The screenshot shows the 'Process Guides' configuration window. At the top, there's a header bar with a back arrow, the title 'Process Guides', and buttons for 'Update', 'Delete', and a list icon. Below this, the configuration fields are arranged in a grid-like fashion. The 'Name' field contains 'Sample Approval'. The 'Active' checkbox is checked. The 'Order' field is set to '100'. The 'Start' field contains 'Bow or Fred Approve' with a search icon. The 'Table' dropdown is set to 'Request'. Below these fields is a 'Conditions' section with an 'Add' button. Underneath, there's an 'Add Condition' button and a condition entry: a red minus sign, a green plus sign, a dropdown with 'Requested for', the word 'is', another dropdown, and a text field with 'Bow Ruggeri' and a search icon. At the bottom of this section are 'Update' and 'Delete' buttons.

The screenshot shows the 'Process Steps' section. At the top, there's a 'New' button, a plus icon, a checkbox, and a link 'Process guide = Sample Approval >'. To the right, there are navigation arrows and a page indicator '1 to 2 of 2'. Below this is a table with a header row 'Name'. The table contains two rows: 'Bow or Fred Approve' and 'David Loo Approves'. Each row has a checkbox on the left. Below the table, there's a dropdown menu labeled 'Actions on selected rows...'.

The important fields here are:

- Name -- the name of this process guide (useful so you can tell them apart)
- Order -- order in which this process guide is applied
- Table -- the table to which this process guide applies
- Start -- the first approval step in the guide (see below)

Process guides are applied just like approval rules or SLAs.

1. All guides which might apply to a particular table are queried out according to their order (low numbers first)
2. The condition is tested for each guide in turn
3. When a matching condition is found, that guide is applied to the current record
4. No subsequent conditions are evaluated

Approval Step

Approval Step

Update Delete

Name:

Bow or Fred Approve

Process guide:

Sample Approval

Control:

Any Of

User list:

Bow Ruggeri

Fred Luddy

Group list:

Approver script:

Select variables:

Upon approval

Update Delete

On approval:

Go to Step

Approval goto:

David Loo Approves

Upon rejection

Update Delete

On rejection:

Reject Entity



Note: In order to enable Any Of Rejects on instances provisioned before **Winter 2009**, ensure that the system property `glide.pg.any_rejection_rejects` is present and true.

The important fields here are:

- Control -- Either **Any Of** or **All Of** the identified approvers
- User List -- List of specific approvers e.g. "Bow and Fred"
- Group List -- List of specific groups e.g. "the members of Database New York"
- Approver Script -- Freeform script where you can dynamically return any number of approvers based on criteria that make sense to your organization

Beneath that, you see two transition sections which dictate what happens after this step is either approved (Upon Approval) or rejected (Upon Rejection). Your options are:

1. Approve entity -- the current record is approved and the guide terminates
2. Reject entity -- the current record is rejected and the guide terminates
3. Go to step -- another approval step starts (and potentially adds a new set of approvers)
4. Run script -- a script executes and performs actions relevant to your organization. The script may return the `sys_id` of an approval step to allow dynamic branching and/or take other important actions (like throw an event, update some data, etc)

How to turn them on

Catalog Requests

1. From the left navigation pane, select **System Properties > Service Catalog**.
2. Locate the **Service Catalog Requests approval engine** option.
3. Select **Process Guides**.

Catalog Tasks

1. From the left navigation pane, select **System Properties > Service Catalog**.
2. Locate the **Service Catalog Tasks approval engine** option.
3. Select **Process Guides**.

Change Requests

1. From the left navigation pane, select **System Properties > Change Management**.
2. Check the **Should the change management module use the advanced approval engine?** option.

Change Tasks

1. From the left navigation pane, select **System Properties > Change Management**.
2. Check the **Should the change management module use the advanced approval engine for tasks** option.

How and When do Process Guides Bind to a Task

Once you have activated the property to specify that you would like to run a Process Guide for approvals in lieu of Approval rules and have set up your Process Guide for that table with appropriate Process Steps, the Approval Engine will try to run the Process Guides with the following behavior:

1. A Task, such as a Change Request or Catalog Request, is created and will be bound to all Process Guides that it meets conditions for.
2. Once the Task has been bound to one or many Process Guides it will not bind again.

How do I know if my Process Guide(s) have bound

To know if your Process Guide(s) have bound to the Task you expect, you can personalize the Approvers list to show the State binding field. If the Approvers related list is not on the form, you can personalize the form's related lists to add it.

How do I know what Process Step my Process Guide is currently running

To know what Process Step your Process Guide is currently running you can personalize the Approvers list to show the Process Step field. If the Approvers related list is not on the form, you can personalize the form's related lists to add it.

FAQs

Q: Before Process Guides, I was able to specify approvers and approval groups directly on a catalog approval task. How will they interact with the new process guide code? In other words, did my existing approval tasks just break?

A: There's a default process guide on `sc_task` with an order of 10000 (called, unsurprisingly **default**). It mimics the functionality of the old, less functional, code that used to be in place. So if no other process guide applies, this one will kick in and things will work the same way they always have.

Q: I've got a lot of process already built out as approval rules. What happens if I decide I want to use process guides instead?

A: Once you turn on the process guide engine for a particular table, your old approval rules stop being processed (for that particular table) so the previous logic won't apply anymore. The good news is that process guide functionality is generally a superset of the old approval rule functionality so anything you could do with an approval rule could be re-done as a process guide.

Q: I like approval rules and they work for me. Do I have to change to process guides?

A: No, you don't, but the long term product goal is to supplant the approval rules engine entirely with the newer (and more powerful) process guide code. In the intermediate term, this means we probably won't be appreciably enhancing the approval rules engine. Most new approval related features will probably show up in the process guide engine.

Q: I've got a green field implementation. Which should I use, process guides, or approval rules?

A: We'd recommend process guides for the enhanced functionality they offer.

Approval with E-Signature Plugin



Note: This article applies to Fuji. For more current information, see *Approval with Electronic Signature* ^[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

Approval with E-Signature allows users to approve requests by re-entering their login credentials. Approval with E-Signature supports the following authentication credentials:

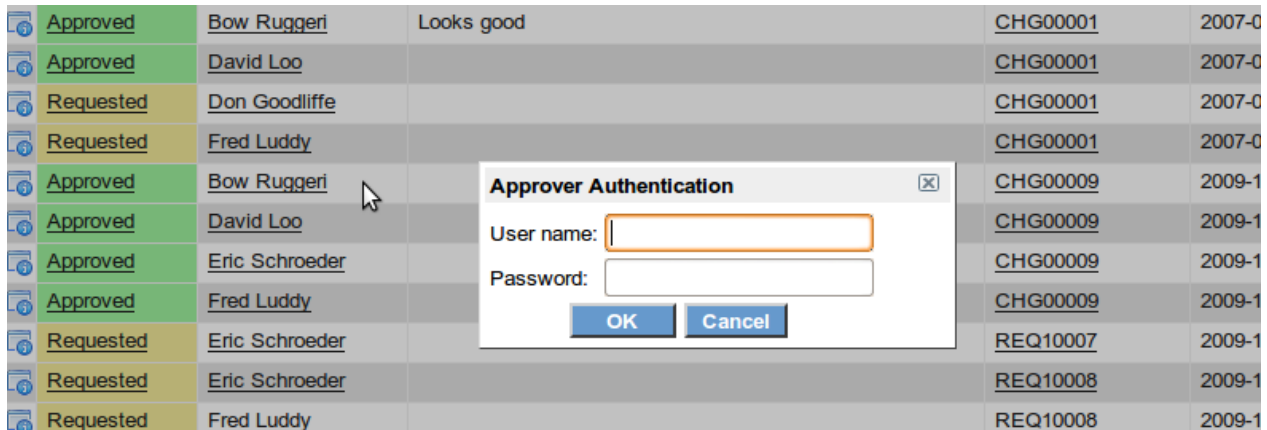
- User name and password matching a user in the local database.
- User name and password matching a user authorized by an external identity provider as part of a SAML 2.0 single sign-on integration.

Using E-Signature Approvals

Users can approve requests with an e-signature by:

- Selecting the **Approve** or **Reject** option on a list context menu.
- Clicking the **Approve** or **Reject** button on a form.
- Changing a request record's **State** to **Approved** in either the list-editor or form.

Selecting any of these options present the user with an **Approver Authentication** window, which requests the user's credentials again:



Setting Up E-Signature Approvals

The setup required depends on where user credentials are stored. See the section that matches the source of your user credentials. Customers on pre-Dublin instances should see the previous version information.



Note: ADFS 2.0 does not support re-authentication requests that E-Signature requires.

Local Database

To set up Approvals with e-signature using credentials from a local database:

1. Activate the Approval with E-Signature plugin.
2. Create user records for approval users.

SAML 2.0

To set up Approval with e-signature using credentials from a SAML 2.0 authentication:

1. Activate or upgrade to a SAML 2.0 Update 1 integration.
2. Activate the Approval with E-Signature plugin.
3. Verify the SAML Identity Provider (IdP) supports and honors the **forceAuthn** attribute in SAML Assertion Requests.
4. Enter the e-signature SAML properties.
5. Regenerate the service provider metadata and update it on the IdP.
6. Create user records for approval users.



Note: Customers on SAML 2.0 Update 1 no longer have to create custom UI pages for logging the user out or deleting session cookies (starting with the Dublin release).

E-Signature SAML Properties

Configure the following properties for E-Signature with SAML 2.0 update 1 (starting with the Dublin release).

AuthnRequest URL for eSignature Authentication

Enter the URL that points to the SAML 2.0 Identity Provider AuthnRequest Consumer for eSignature Authentication. In most cases, this will be the same as the AuthnRequest URL used in general authentication. Leave this setting blank if you intend to use the same AuthnRequest Consumer URL that is used for general SAML 2.0 authentication in your instance.

The SAML 2.0 Assertion Consumer URL for eSignature authentication

In most cases, this URL will be https://YOURINSTANCE.service-now.com/consumer.do. However, if you employ a customized method of handling the SAML authentication for eSignatures, you can set up your own consumer URL.

https://YOURINSTANCE.service-now.com/consumer.do

The SAML 2.0 Assertion Consumer Index for eSignature authentication

If your Service Provider has more than one URL set for the AssertionConsumerURL, then you can set the index of which one to use for eSignature here starting with index 0.

1

Authentication Pop-up Dialog Width

When a user approves a request using eSignature, a dialog will pop up that allows the user to enter their credentials. This setting controls the width of that dialog box.

500

Authentication Pop-up Dialog Height

When a user approves a request using eSignature, a dialog will pop up that allows the user to enter their credentials. This setting controls the height of that dialog box.

300

Save

E-Signature with SAML properties

Property	Description
AuthnRequest URL for eSignature Authentication. <code>com.snc.integration.saml_esig.idp_authnrequest_url</code>	Enter the URL that points to the SAML 2.0 Identity Provider AuthnRequest Consumer for eSignature Authentication. In most cases, this will be the same as the AuthnRequest URL used in general authentication. Leave this setting blank if you intend to use the same AuthnRequest Consumer URL that is used for general SAML 2.0 authentication in your instance.
The SAML 2.0 Assertion Consumer URL for eSignature authentication. <code>com.snc.integration.saml_esig.approval_consumer_url</code>	In most cases, this URL will be: [2] However, if you employ a customized method of handling the SAML authentication for eSignature, you can set up your own consumer URL. If you are using Multi-Provider SSO, you do not need to use this property. Configure the Identity Provider form, add the Assertion Consumer URL for eSignature authentication , field, and set the URL in that field.
The SAML 2.0 Assertion Consumer Index for eSignature authentication <code>com.snc.integration.saml_esig.assertion_consumer_service_index</code>	If your Service Provider has more than one URL set for the AssertionConsumerURL, you can set the index to use for eSignature, starting with index 1 or more.
Authentication Pop-up Dialog Width <code>com.snc.integration.saml_esig.popup_dlg_width</code>	When a user approves a request using eSignature, a dialog allows the user to enter their credentials. This setting controls the width of that dialog box.
Authentication Pop-up Dialog Height <code>com.snc.integration.saml_esig.popup_dlg_height</code>	When a user approves a request using eSignature, a dialog allows the user to enter their credentials. This setting controls the height of that dialog box.

Selecting Approval Tables

By default, activating the Approval with E-signature plugin enables e-signature for all tables for which a previous approval exists. E-signature approvals can also be enabled on a table-by-table basis.

To enable e-signatures for a table:

1. Navigate to **System Definition > E-Signature Registry**.
2. In **Table name**, use the drop-down list to select a specific table.

Field	Input Value
Table	A table drop-down to select the table that requires e-signatures.
Enabled	If selected, e-signature is required. Clear this option to remove the e-signature requirement.

De-Activating E-Signatures

Although plugins cannot be removed, E-signatures can be disabled by navigating to **System Dictionary > E-Signature Registry**, and setting **Enabled** to **False** on any tables where e-signatures are no longer required.

Pre-Dublin Setup

Instances on pre-Dublin versions require manual customization to support Approval with E-Signature.

Click the plus to view the pre-Dublin procedure

1. Activate the Approval with E-Signature plugin.
2. Create custom UI pages to logout the approval user, store information about the record being approved, allow approver to re-enter their user credentials, and act on the original record being approved.
3. Create user records for approval users.

Customizations Required

Approval with E-Signature requires users to re-enter a name and password a second time. To force a second sign in, the user and identity provider session need to be terminated so the user must log in again. The challenge is how to maintain the state of the user (capturing what the user was doing before logout) so that after the user logs back in successfully the system takes the correct action. In the case of e-signature, maintaining the state of the user involves updating the approval record correctly based on the decision that the user had made for a specific approval record.

At a high level, your custom UI pages must perform the following actions:

1. Log the user out.
2. Redirect the user to a page.
3. Add parameters to the URL.
4. Parse the parameters out.
5. Use the parameters to determine the correct action to perform on the original approval record.



Logout the User

A UI Action redirects the approval user to your custom logout UI page. The main objective is to ensure that both the ServiceNow session and the SSO session are terminated. This can be done using two different methods:

- Leverage `logout.do`
- Delete the session cookies

The following scripts log the user out of the ServiceNow session and the SSO session, then redirect to another UI Page. However, since the user no longer has a session, they are redirected to the SSO login with the RelayState set to the second UI Page in the ServiceNow instance.



Note: For the scripts to work properly, ensure that your IdP supports deep linking.

Example: Logout with `logout.do`

```
<script>

<?xml version="1.0" encoding="utf-8" ?>

<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null" xmlns:g2="null">

<g2:evaluate jelly="true">

    var appSys_id = jelly.sysparm_approvalsids;

    var button_name = jelly.sysparm_button_name;

    var instanceName = gs.getProperty('instance_name');

</g2:evaluate>

    function readyToLogin() {

        window.location =

"https://[${instanceName}].service-now.com/nav_to.do?uri=esig_decision.do?sysparm_approval_sys_id=${appSys_id}%26sysparm_button_name=${button_name}

    }

</script>

<iframe name="close" src="https://[${instanceName}].service-now.com/logout.do" onload="readyToLogin()" width="1" height="1"></iframe>

</j:jelly>

</script>
```

Example: Logout with Cookies

```
<script>

<?xml version="1.0" encoding="utf-8" ?>

<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null" xmlns:g2="null">

<g2:evaluate jelly="true">

    var appSys_id = jelly.sysparm_approvalsids;

    var button_name = jelly.sysparm_button_name;

    var instanceName = gs.getProperty('instance_name');

</g2:evaluate>

<script type="text/javascript">

    function readyToLogin() {
```

```

window.location =

"https://${instanceName}.service-now.com/nav_to.do?uri=esig_decision.do?sysparm_approval_sys_id=${appSys_id}%26sysparm_button_name=${button_name}

    }

function Delete_Cookie( name, path, domain ) {

    document.cookie = name + "=0;expires=Thu, 01-Jan-1970 00:00:01

GMT";

}

var domain = "${instanceName}.service-now.com";

var path = "/";

Delete_Cookie("JSESSIONID" , path, domain);

Delete_Cookie("glide_user" , path, domain);

Delete_Cookie("glide_user_session" , path, domain);

</script>

<iframe name="close" src="SSO logout url" onload="readyToLogin()" width="1" height="1"></iframe>

</j:jelly>

</script>

```

Update the Approval Record

A second UI page needs to update the approval record. After the user has been re-authenticated, the SSO will redirect the user back to the ServiceNow instance based on the RelayState that was initially passed to the SSO. The RelayState will be the UI Page that parses out the parameters passed in the URL and uses them to update the correct approval record in sysapproval_approver table to be either approved or rejected.

For example, this script takes the parameters in the URL and uses them to update the approval record. The script also uses a base instance function, isApprovalMine(), to ensure that the current user is an appropriate user to make decisions on the approval record in the system. Finally, the script redirects to the original approval record.

```

<script>
<?xml version="1.0" encoding="utf-8" ?>
<j:jelly trim="false" xmlns:j="jelly:core" xmlns:g="glide" xmlns:j2="null" xmlns:g2="null">
  <g2:evaluate jelly="true">
    var appSys_id = jelly.sysparm_approval_sys_id; //<< sys_id of approval record
    var button_name = jelly.sysparm_button_name; //<< name of button pushed
    var session_id = jelly.sysparm_session_id; //<< session id from session before log out

    //Look up approval record
    var decision;
    var grApp = new GlideRecord('sysapproval_approver');
    grApp.addQuery('sys_id', appSys_id);
    grApp.addQuery('u_session_id', session_id);
    grApp.query();
    grApp.next();

    //Check if logged in user is valid approver of record

```

```

var appMine = isApprovalMine(grApp);

//Decide if the decision was approve or not
if(button_name.indexOf('approve') >= 0)
    decision = 'approved';
else
    decision = 'rejected';

//If user is a valid approver, update the decision.
if(appMine) {
    grApp.state = decision;
    grApp.update();
}

//Redirect to the original approval record.
<script>
    window.location = "sysapproval_approver.do?sys_id=${appSys_id}";
</script>

</j:jelly>
</script>

```

UI Action Modifications

The base instance UI actions make updates to the approval record, either setting the state to ‘approved’ or ‘rejected’. These will need to be replaced or somehow circumvented so they do not work. One way to do this is an onSubmit client script that looks for button pushes of approval decision that need an E-Signature. However it is done, the end result needs to be a redirect to the first UI Page explained earlier that logs users out of the ServiceNow and SSO sessions. That redirection URL should contain the system ID of the approval record, as well as the button that was pressed, as parameters.

Installed with the Plugin

Installing the plugin installs the following:

- **Module** - E-Signature Registry
- **UI Action** - Approve (on table sysapproval_approver, with no action name)
- **UI Action** - Approve (on table sysapproval_approver, with no action name)
- **UI Action** - Approve (on table sysapproval_approver, with the action name authenticated_list_approval)
- **UI Page** - form_login_validate_dialog
- **UI Page** - login_validate_dialog
- **UI page:** saml2_esignature_login, the re-authentication page that appears when an approver tries to approve a request.
- **Properties:** see E-Signature SAML Properties.
- **Client Script** - Authenticate Approver
- **Script Include** - User
- **Script Include** - UserAuthentication
- **Processor:** eSigSaml2AssertionConsumer.

Installing the plugin also disables the two out-of-box **Approve** UI Actions on the sysapproval_approver table.

Activating Approval with E-Signature

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**.

Enhancements

Dublin

- Provides UI actions to approve or reject requests.
- Provides improved support for SAML 2.0 integrations.
- Provides E-Signature properties for SAML 2.0.
- Requires SAML 2.0 integrations to activate or upgrade to SAML 2.0 Update 1.

References

- [1] https://docs.servicenow.com/bundle/jakarta-servicenow-platform/page/administer/service-administration/reference/r_ApprovalWithESignature.html
- [2] <https://YOURINSTANCE.service-now.com/consumer.do>

Approvals with Workflows

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**.

New Workflow

* Name: Change - Approval

* Table: Change Request [change_request]

If condition matches: Run if no other workl

Order: 100

Checked out: 2014-08-26 16:08:44

Checked out by: System Administrator

Condition: -- choose field -- -- oper -- -- value --

Delivery based on: User-specified duration

Expected time: Days 00 Hours 16 : 00 : 00

Schedule: 8-5 weekdays excluding

Timezone: US/Pacific

Published: ☐

Max activity count: 100

Stage field: State

Stage rendering: Workflow-driven

Stage order: Computed

Description: This workflow determines what kind of change request has been made and generates the appropriate approvals.

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**.

Activity Properties: If

Name: If Change Is Routine

Stage: Pending

Condition: [Type] is [Routine]

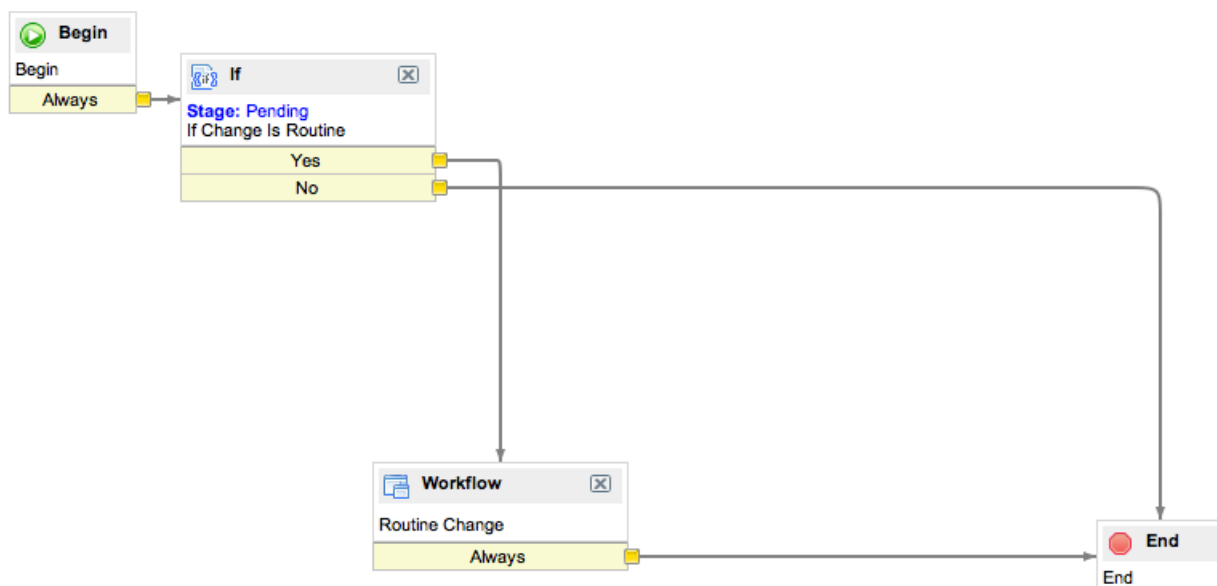
Advanced:

Update

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**.

New Activity: If

Name: If Risk Is High

Stage: Pending

Condition: Risk is High OR Risk is Very High

Advanced: ☐

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**.

Activity Properties: Approval - Group

Name: CAB Approval

Stage: Pending

Condition: -- choose field -- -- oper -- -- value --

Groups: CAB

Wait for: An approval from each group

When anyone rejects: Reject the approval

Advanced: ☐

Update

Due date based on: A user specified duration

Duration: Days 00 Hours 00:00:00

Schedule based on: (no schedule)

Time zone based on: (no time zone)

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.

- 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**:

- Drag the **Set Values** activity into the area below the new **CAB Approval** activity.
- 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*.
- Click **Submit**.
- From the **CAB Approval** activity, click the **Rejected** tab and drag to the new **Rejected** activity.
- From the **Rejected** activity, click the **Always** tab and click to the **End** activity.

Activity Properties: Set Values

Name:	Rejected
Stage:	Closed Incomplete

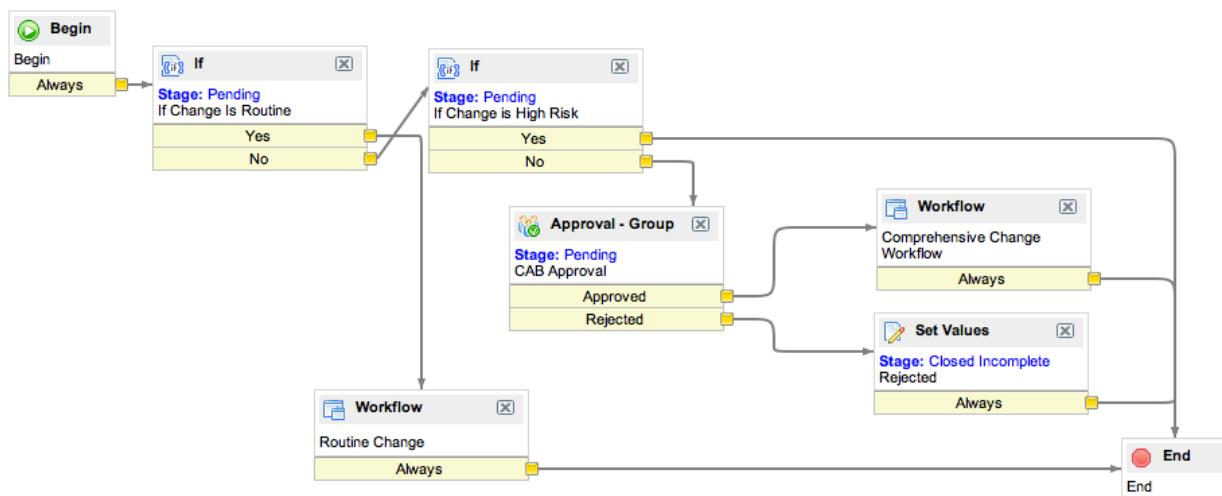
Set these values:

Approval	Rejected
-- choose field --	-- value --

Update

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 - Users** 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.

Activity Properties: Approval Coordinator

Name:	Involved Parties Approval		
Stage:	Pending		
Wait for:	All child activities to be approved		
When a rejection occurs:	Reject the approval		

Update

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());
}
```

Activity Properties: Approval - User

Name: Infrastructure at Location

Stage: Pending

Condition: ☐ and ☐ or

-- choose field -- -- oper -- -- value --

Users:

Groups:

Wait for: Anyone to approve

When anyone rejects: Reject the approval

Advanced: ☒

Additional approvers script:

```

answer = [];
var objUser = new GlideRecord('sys_user');
objUser.addQuery('location', current.cmdb_ci.location);
objUser.addQuery('department', 'd7b1abb00a0a3c1e009aab792c705ed7');
objUser.query();

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

```

Select variables:

- Fields
- GlideRecord
- GlideElement
- System
- GlideAggregate

Due date based on: A user specified duration


Duration: Days 00 Hours 00 : 00 : 00

Schedule based on: (no schedule)

Time zone based on: (no time zone)

Update

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

New Activity: Approval - Group

Name: CAB Approval

Stage: Pending

Condition: ☐ and ☐ or

-- choose field -- -- oper -- -- value --

Groups: CAB

Wait for: An approval from each group

When anyone rejects: Reject the approval

Advanced: ☐

Submit

Due date based on: A user specified duration

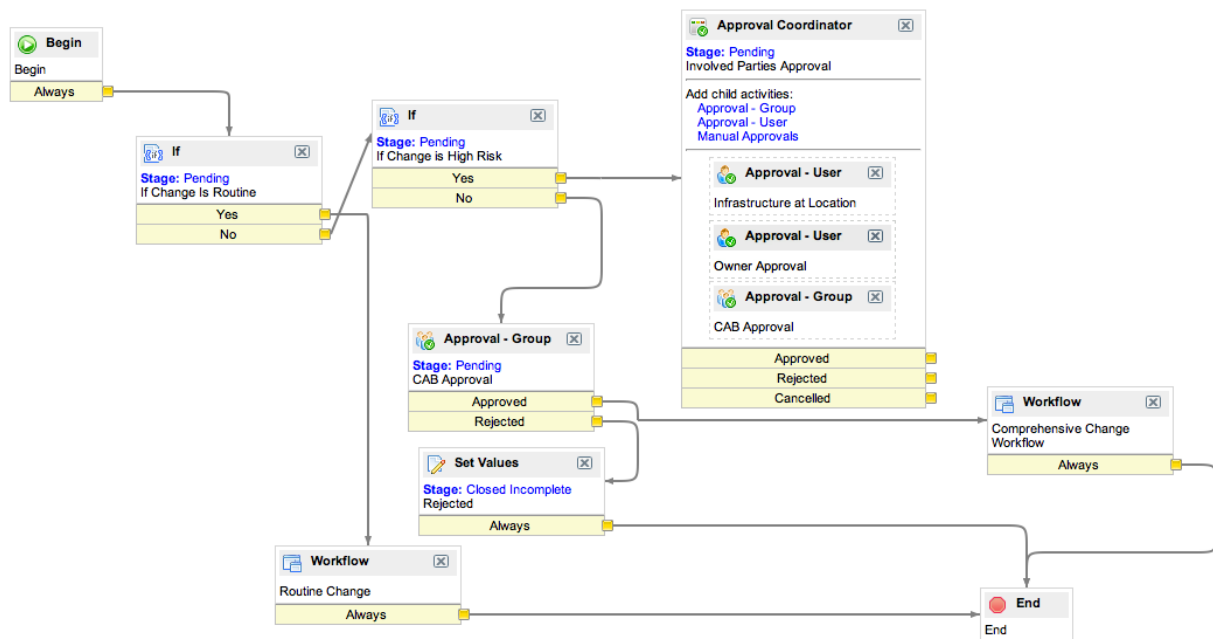
Duration: Days 00 Hours 00 : 00 : 00

Schedule based on: (no schedule)

Time zone based on: (no time zone)

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

Activity Properties: Approval - User

Name:

Stage:

Condition:

-- choose field -- -- value --

Users:

Groups:

Wait for:

When anyone rejects:

Advanced: ☐

Due date based on:

Duration: Days Hours Minutes Seconds

Schedule based on:

Time zone based on:

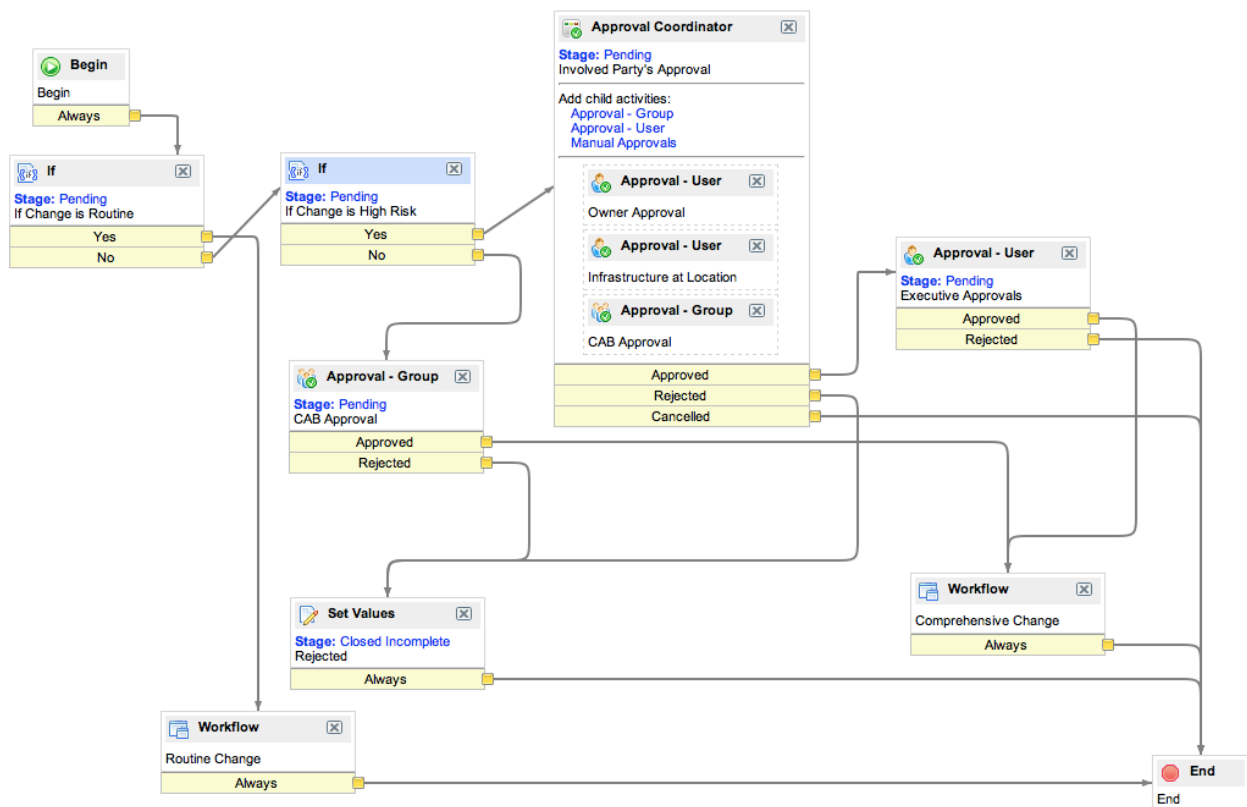
Tying the Approvals Together

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:



<h1>Article Sources and Contributors</h1>	
Task Table	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=250945 <i>Contributors:</i> Davida.hughes, Emily.partridge, G.yedwab, Guy.yedwab, John.ramos, Joseph.messerschmidt, Rachel.sienko, Suzanne.smith
Many to Many Task Relations Plugin	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=82827 <i>Contributors:</i> Davida.hughes, Emily.partridge, Fuji.publishing.user, G.yedwab, Guy.yedwab, John.ramos, Joseph.messerschmidt, Mark.stanger, Peter.smith, Rachel.sienko, Stacey.schwingle, Steven.wood
Process Flow Formatter Plugin	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=82825 <i>Contributors:</i> Cheryl.dolan, Davida.hughes, Emily.partridge, Fuji.publishing.user, G.yedwab, Guy.yedwab, Joe.zucker, John.ramos, Joseph.messerschmidt, Phillip.salzman, Rachel.sienko, Steven.wood, Suzanne.smith
Task Parent Breadcrumbs Formatter	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=241708 <i>Contributors:</i> Cheryl.dolan, Emily.partridge, Fuji.publishing.user, Guy.yedwab, John.roberts, Joseph.messerschmidt, Rachel.sienko, Steven.wood
Task Outage Plugin	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=82792 <i>Contributors:</i> Davida.hughes, Emily.partridge, Fuji.publishing.user, Guy.yedwab, Joseph.messerschmidt, Neola, Rachel.sienko, Steven.wood, Wallymarx
Time Cards	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=250381 <i>Contributors:</i> Davida.hughes, Emily.partridge, Fuji.publishing.user, G.yedwab, Guy.yedwab, Jerrod.bennett, Joe.Westrich, John.ramos, John.roberts, Joseph.messerschmidt, Phillip.salzman, Rachel.sienko, Steven.wood
Planned Task Plugin	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=83117 <i>Contributors:</i> David.Bailey, Davida.hughes, Emily.partridge, G.yedwab, Guy.yedwab, John.roberts, Joseph.messerschmidt, Rachel.sienko
Planned Task Hierarchy	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=250781 <i>Contributors:</i> Davida.hughes, Guy.yedwab, John.ramos, Joseph.messerschmidt, Rachel.sienko
Gantt Chart	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=250588 <i>Contributors:</i> Cheryl.dolan, Emily.partridge, Fuji.publishing.user, G.yedwab, Guy.yedwab, John.ramos, Joseph.messerschmidt, Phillip.salzman, Steven.wood
Task Resources	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=250944 <i>Contributors:</i> G.yedwab, Guy.yedwab, John.ramos, John.roberts, Joseph.messerschmidt, Phillip.salzman, Publishing.user, Steven.wood
Managing Planned Tasks in Project	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=150369 <i>Contributors:</i> Cheryl.dolan, Fuji.publishing.user, John.ramos, Phillip.salzman, Prakash.chandra, Rachel.sienko
Assignments	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=250266 <i>Contributors:</i> Davida.hughes, Emily.partridge, Fuji.publishing.user, G.yedwab, John.ramos, Joseph.messerschmidt, Norris.merritt, Phillip.salzman, Rachel.sienko, Vaughn.romero
Configuring Group Types for Assignment Groups	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=250176 <i>Contributors:</i> Davida.hughes, Emily.partridge, G.yedwab, Guy.yedwab, Ishrath.razvi, John.ramos, Joseph.messerschmidt, Phillip.salzman, Vaughn.romero
Approvals	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=250126 <i>Contributors:</i> CapaJC, Davida.hughes, Emily.partridge, G.yedwab, Guy.yedwab, Jessi.graves, John.ramos, Joseph.messerschmidt, Publishing.user, Suzanne.smith, Vhearne, Virginia.kelley, Wallymarx
Approval Summaries	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=92223 <i>Contributors:</i> CapaJC, Cheryl.dolan, David.Bailey, Davida.hughes, Eric.jacobson, G.yedwab, Guy.yedwab, John.ramos, Joseph.messerschmidt, Pat.Casey, Steven.wood, Vhearne
Execution Order of Scripts and Engines	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=250572 <i>Contributors:</i> Andrew.martin, Davida.hughes, Emily.partridge, Eric.jacobson, Fuji.publishing.user, G.yedwab, Guy.yedwab, John.ramos, Neola, Orcarattini, Steven.wood, Suzanne.smith, Vaughn.romero
Approval Engines	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=248926 <i>Contributors:</i> Davida.hughes, G.yedwab, Guy.yedwab, Jay.berlin, Joseph.messerschmidt, Phillip.salzman, Steven.wood, Vhearne
Approval Rules	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=250122 <i>Contributors:</i> Amy.bowman, CapaJC, Cheryl.dolan, David.Bailey, Davida.hughes, Emily.partridge, Fuji.publishing.user, G.yedwab, Guy.yedwab, John.ramos, Joseph.messerschmidt, Pat.Casey, Rachel.sienko, Rob.woodbyrne, Steven.wood, Vaughn.romero, Vhearne
Setting Automatic Approval Rules	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=237778 <i>Contributors:</i> Amy.bowman, CapaJC, Cheryl.dolan, David.Bailey, Davida.hughes, Emily.partridge, Fuji.publishing.user, G.yedwab, Guy.yedwab, John.ramos, Joseph.messerschmidt, Pat.Casey, Rachel.sienko, Rob.woodbyrne, Steven.wood, Vaughn.romero, Vhearne
Process Guides	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=55036 <i>Contributors:</i> CapaJC, Christen.mitchell, Davida.hughes, G.yedwab, Gadi.yedwab, Guy.yedwab, Joseph.messerschmidt, Pat.Casey, Scott.marshall, Steven.wood, Valor, Vhearne
Approval with E-Signature Plugin	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=82775 <i>Contributors:</i> Cheryl.dolan, Chuck.tomasi, David Loo, David.Bailey, Davida.hughes, Fuji.publishing.user, G.yedwab, Guy.yedwab, John.ramos, Joseph.messerschmidt, Julie.phaviset, Peter.smith, Phillip.salzman, Rachel.sienko, Steven.wood, Suzanne.smith, Vaughn.romero
Part One	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=244818 <i>Contributors:</i> Davida.hughes, Fuji.publishing.user, G.yedwab, Guy.yedwab, Ishrath.razvi, Jennifer.ball, Joe.Westrich, Joseph.messerschmidt, Julie.phaviset, Peter.smith, Publishing.user, Rachel.sienko, Russ.sarbora, Steven.wood, Vaughn.romero
Part Two	<i>Source:</i> http://wiki.servicenow.com/index.php?oldid=239954 <i>Contributors:</i> Davida.hughes, Fuji.publishing.user, G.yedwab, Guy.yedwab, Ishrath.razvi, Joe.Westrich, Joseph.messerschmidt, Rachel.sienko, Steven.wood

Image Sources, Licenses and Contributors

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

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

Image:TaskInterceptor2.png Source: <http://wiki.servicenow.com/index.php?title=File:TaskInterceptor2.png> License: unknown Contributors: G.yedwab, Suzanne.smith

Image:ActivityFormatter.png Source: <http://wiki.servicenow.com/index.php?title=File:ActivityFormatter.png> License: unknown Contributors: CapaJC, Kenny.gabel

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:Workflow field.png Source: http://wiki.servicenow.com/index.php?title=File:Workflow_field.png License: unknown Contributors: Fuji.publishing.user, Guy.yedwab

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

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

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

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

Image:RelItems2.png Source: <http://wiki.servicenow.com/index.php?title=File:RelItems2.png> License: unknown Contributors: G.yedwab

Image:Caution-diamond.png Source: <http://wiki.servicenow.com/index.php?title=File:Caution-diamond.png> License: unknown Contributors: John.roberts, Suzanne.smith

Image:Workflow Display UI15.jpg Source: http://wiki.servicenow.com/index.php?title=File:Workflow_Display_UI15.jpg License: unknown Contributors: Suzanne.smith

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

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

Image:Adding_Process.png Source: http://wiki.servicenow.com/index.php?title=File:Adding_Process.png License: unknown Contributors: Emily.partridge, Guy.yedwab

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

File:ParentBreadcrumbs Customized.png Source: http://wiki.servicenow.com/index.php?title=File:ParentBreadcrumbs_Customized.png License: unknown Contributors: Cheryl.dolan

Image:TOP-menu.png Source: <http://wiki.servicenow.com/index.php?title=File:TOP-menu.png> License: unknown Contributors: Guy.yedwab

Image:TOP-outage.png Source: <http://wiki.servicenow.com/index.php?title=File:TOP-outage.png> License: unknown Contributors: Guy.yedwab

Image:TOP-task.png Source: <http://wiki.servicenow.com/index.php?title=File:TOP-task.png> License: unknown Contributors: Guy.yedwab

Image:TOP-outage2.png Source: <http://wiki.servicenow.com/index.php?title=File:TOP-outage2.png> License: unknown Contributors: Guy.yedwab

Image:Time_Worked_Started.png Source: http://wiki.servicenow.com/index.php?title=File:Time_Worked_Started.png License: unknown Contributors: Steven.wood

Image:Time_Worked_Stopped.png Source: http://wiki.servicenow.com/index.php?title=File:Time_Worked_Stopped.png License: unknown Contributors: Steven.wood

File:Time_worked_notice.jpg Source: http://wiki.servicenow.com/index.php?title=File:Time_worked_notice.jpg License: unknown Contributors: John.roberts

Image:User_Resource.png Source: http://wiki.servicenow.com/index.php?title=File:User_Resource.png License: unknown Contributors: Steven.wood

Image:User_Resource2.png Source: http://wiki.servicenow.com/index.php?title=File:User_Resource2.png License: unknown Contributors: Steven.wood

Image:Time_Card.png Source: http://wiki.servicenow.com/index.php?title=File:Time_Card.png License: unknown Contributors: Steven.wood

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

Image:PTaskInterceptor2.png Source: <http://wiki.servicenow.com/index.php?title=File:PTaskInterceptor2.png> License: unknown Contributors: G.yedwab

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

Image:PTBaseline2.png Source: <http://wiki.servicenow.com/index.php?title=File:PTBaseline2.png> License: unknown Contributors: Guy.yedwab

Image:PTBaseline3.png Source: <http://wiki.servicenow.com/index.php?title=File:PTBaseline3.png> License: unknown Contributors: Guy.yedwab

Image:Rm2-heirarchy.png Source: <http://wiki.servicenow.com/index.php?title=File:Rm2-heirarchy.png> License: unknown Contributors: G.yedwab

image:Gantt_chart_example.png Source: http://wiki.servicenow.com/index.php?title=File:Gantt_chart_example.png License: unknown Contributors: Phillip.salzman

Image:Gannt_chart_tooltip_summary.png Source: http://wiki.servicenow.com/index.php?title=File:Gannt_chart_tooltip_summary.png License: unknown Contributors: Phillip.salzman

Image:gantt_chart_drag.png Source: http://wiki.servicenow.com/index.php?title=File:Gantt_chart_drag.png License: unknown Contributors: Phillip.salzman

Image:drag_new_task.png Source: http://wiki.servicenow.com/index.php?title=File:Drag_new_task.png License: unknown Contributors: Phillip.salzman

Image:gantt_chart_critical_path.png Source: http://wiki.servicenow.com/index.php?title=File:Gantt_chart_critical_path.png License: unknown Contributors: Phillip.salzman

Image:User_Resource_New.png Source: http://wiki.servicenow.com/index.php?title=File:User_Resource_New.png License: unknown Contributors: Steven.wood

Image:User_Resource_Config.png Source: http://wiki.servicenow.com/index.php?title=File:User_Resource_Config.png License: unknown Contributors: Steven.wood

Image:Group_Resources_List2.png Source: http://wiki.servicenow.com/index.php?title=File:Group_Resources_List2.png License: unknown Contributors: Steven.wood

Image:Group_Resources_List3.png Source: http://wiki.servicenow.com/index.php?title=File:Group_Resources_List3.png License: unknown Contributors: Steven.wood

Image:Timeline_Start.png Source: http://wiki.servicenow.com/index.php?title=File:Timeline_Start.png License: unknown Contributors: Steven.wood

Image:Timeline_Assign_to_User.png Source: http://wiki.servicenow.com/index.php?title=File:Timeline_Assign_to_User.png License: unknown Contributors: Steven.wood

Image:Timeline_Assigned.png Source: http://wiki.servicenow.com/index.php?title=File:Timeline_Assigned.png License: unknown Contributors: Steven.wood

Image:Timeline_Detail.png Source: http://wiki.servicenow.com/index.php?title=File:Timeline_Detail.png License: unknown Contributors: Steven.wood

Image:Finish_to_start_dependency.png Source: http://wiki.servicenow.com/index.php?title=File:Finish_to_start_dependency.png License: unknown Contributors: Phillip.salzman

Image:successor_pending_WIP.png Source: http://wiki.servicenow.com/index.php?title=File:Successor_pending_WIP.png License: unknown Contributors: Phillip.salzman

Image:delete_task_relationship.png Source: http://wiki.servicenow.com/index.php?title=File>Delete_task_relationship.png License: unknown Contributors: Phillip.salzman

Image:Parent.png Source: <http://wiki.servicenow.com/index.php?title=File:Parent.png> License: unknown Contributors: Phillip.salzman

Image:remove_parent-child_relationship.png Source: http://wiki.servicenow.com/index.php?title=File:Remove_parent-child_relationship.png License: unknown Contributors: Phillip.salzman

image:rollup_popup.png Source: http://wiki.servicenow.com/index.php?title=File:Rollup_popup.png License: unknown Contributors: Phillip.salzman

Image:assignment_lookup_rule.png Source: http://wiki.servicenow.com/index.php?title=File:Assignment_lookup_rule.png License: unknown Contributors: Vaughn.romero

Image:Assignment Rule Eureka.png Source: http://wiki.servicenow.com/index.php?title=File:Assignment_Rule_Eureka.png License: unknown Contributors: Maintenance script

Image:Assignment_data_lookup.png Source: http://wiki.servicenow.com/index.php?title=File:Assignment_data_lookup.png License: unknown Contributors: Vaughn.romero

Image:Assignment.png Source: <http://wiki.servicenow.com/index.php?title=File:Assignment.png> License: unknown Contributors: CapaJC, Vaughn.romero

Image:assignment_by_script.png Source: http://wiki.servicenow.com/index.php?title=File:Assignment_by_script.png License: unknown Contributors: Vaughn.romero

Image:Task assignment business rule2.png Source: http://wiki.servicenow.com/index.php?title=File:Task_assignment_business_rule2.png License: unknown Contributors: Gflewis

Image:Approvals1.png Source: <http://wiki.servicenow.com/index.php?title=File:Approvals1.png> License: unknown Contributors: CapaJC

Image:Approval summary.png Source: http://wiki.servicenow.com/index.php?title=File:Approval_summary.png License: unknown Contributors: Cheryl.dolan, Pat.Casey

Image:Approval summary catalog.png Source: http://wiki.servicenow.com/index.php?title=File:Approval_summary_catalog.png License: unknown Contributors: Cheryl.dolan, David.Bailey, Pat.Casey

Image:ApprovalEngineModule.png Source: <http://wiki.servicenow.com/index.php?title=File:ApprovalEngineModule.png> License: unknown Contributors: Jay.berlin

Image:ApprovalEnginePage.png Source: <http://wiki.servicenow.com/index.php?title=File:ApprovalEnginePage.png> License: unknown Contributors: Jay.berlin, Steven.wood

Image:Approval rule.png Source: http://wiki.servicenow.com/index.php?title=File:Approval_rule.png License: unknown Contributors: Pat.Casey

Image:Approve list.png Source: http://wiki.servicenow.com/index.php?title=File:Approve_list.png License: unknown Contributors: Pat.Casey

Image:Add approval.png Source: http://wiki.servicenow.com/index.php?title=File:Add_approval.png License: unknown Contributors: Pat.Casey

Image:Approve task.png Source: http://wiki.servicenow.com/index.php?title=File:Approve_task.png License: unknown Contributors: Pat.Casey

Image:Capacity.png Source: <http://wiki.servicenow.com/index.php?title=File:Capacity.png> License: unknown Contributors: Pat.Casey

Image:Capacity atlanta.png Source: http://wiki.servicenow.com/index.php?title=File:Capacity_atlanta.png License: unknown Contributors: Pat.Casey

Image:Approval chart.png Source: http://wiki.servicenow.com/index.php?title=File:Approval_chart.png License: unknown Contributors: Pat.Casey

Image:Obsolete.gif Source: <http://wiki.servicenow.com/index.php?title=File:Obsolete.gif> License: unknown Contributors: CapaJC, Joseph.messerschmidt

Image:Process guide.png Source: http://wiki.servicenow.com/index.php?title=File:Process_guide.png License: unknown Contributors: Pat.Casey

Image:Approval step.png Source: http://wiki.servicenow.com/index.php?title=File:Approval_step.png License: unknown Contributors: Pat.Casey

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

image:Esig_SAML_properties.png Source: http://wiki.servicenow.com/index.php?title=File:Esig_SAML_properties.png License: unknown Contributors: Phillip.salzman

Image:E-Signature_Flow_Diagram.png Source: http://wiki.servicenow.com/index.php?title=File:E-Signature_Flow_Diagram.png License: unknown Contributors: Suzanne.smith

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