Revised Requested Item Form

CHUBB

Continued Development Instructions

Allison Aprile

TABLE OF CONTENTS

BACKGROUND

Problem Overview

The IT Infrastructure Operations team (Manager: Gerard deBeus) fulfills general IT requests through the ServiceNow platform. They use a structure consisting of Approvals, Requests, Requested Items, and Tasks to complete the requests.

The current workflow is concrete until the Requested Item/Service Catalog Task transition. After generating the Requested Item form, it requires separate tasks to be created for each respective fulfillment team, e.g. a server build would require a task for the IT Infrastructure Operations Team, a second for the Server Build Team, and more depending on Backup, Installation, etc.

Several issues, generally **complicating the fulfillment and status reporting processes**, follow:

1.1 Redundant data input

Several data fields do not transfer when creating new tasks. Basic information (Number, Requested for, Opened, etc.) from the original Request form pre-populates into the respective fields. Customizable fields (Work notes, Short description, Description, etc.) **DO NOT**. As these are to be the same between Service Catalog Tasks under the parent Requested Item, this results in **redundant data input**.

1.2 Requestor and implementer separated

The requestor will typically look to the Requested Item form for detailed status on their Request. There is a list of one or more Tasks in the Catalog Tasks section of the form, where the corresponding status can be found. As various groups, assignees, and processes can be affiliated with the Requested Item, the requestor may be unable to see/understand specific status based on a) their ServiceNow role and b) their familiarity with the platform. Therefore, the IT Infrastructure assignee acts as a liaison for the requests.

The service process is not necessarily transparent to the requestor without navigating to several fields on the Requested Item form. Additionally, the form specifies the Item as an 'IT Infrastructure Request', indicating to requestors that the IT Infrastructure team is the primary contact. In most instances, this is not an issue; however, as fulfillments become increasingly complicated (i.e. additional assignment groups or specifications), miscommunications are probable.

1.3 Dot-walking incompatible with 'State' field

Status reports are generated through the ServiceNow platforms on a weekly and/or need basis. They are intended to summarize the team's Open, Work in Progress, and Pending Tasks in the queue. Whether using the Catalog Task [sc_task] or Requested Item [sc_req_item] data table, it is not possible to access **ALL** associated Task 'State' fields in a singular report entry. **The current workflow confines the reporting to a general overview of the entire Request/Requested Item 'State'.**

Dot-walking could typically be implemented in this reporting to access both 'State' fields. However, the Tasks are **NOT UNIQUELY IDENTIFIED** as two **SEPARATE FIELDS** – rather, they are **RELATED** configuration items. From the Requested Item scope, Tasks are recognized as a **GROUP** rather than two

entities, or related forms. Thus, dot-walking is not compatible with Task fields such as 'State', which are necessary for status reporting.

Due to such, any Requested Item-/Task-based status reporting generated through the ServiceNow platform is constrained to a general scope for the team. Status reporting regarding the 'State' of ALL Tasks contributing to Request completion must be done manually by downloading a static CSV spreadsheet and visiting each Task of the Requested Item. **This process is tedious and inefficient.**

Proposed Solution

To counter these issues, a new Requested Item (referred to throughout the remainder of the document as RITM) form was proposed. It would consolidate the RITM/Task portion of the fulfillment process with the following:

2.1 Eliminate Task entity

Both Task 'State' fields are not accessible in a singular report entry (**Section 1.3**). **By eliminating the Task item/entity altogether, and further incorporating its necessary fields on the Requested Item form, the workflow is reduced and data entry is no longer redundant.**

2.2 Transfer Task fields to RITM form

Most of the data fields in the Task form parallel those in the RITM form. To reduce redundant data input while maintaining the pre-imposed Assignment protocol, it would be necessary to transfer/implement the following fields from/on the Task to RITM form: 'Assignment group', 'Assigned to', 'State' and 'Work Notes'. Additional fields can be transferred/added at the designer's discretion.

2.3 Reconfigure Assignment Group fields/vision

The original RITM form did not allow for multiple 'Assignment group', etc. fields and dot-walking access from the Tasks (**Section 1.3**). Therefore, the new form is designed to treat the fulfillment assignments as **separate but parallel fields**.

For example, two sections will appear on the default form: 'Primary Assignment' and 'Secondary Assignment'. Both sections contain the 'Assignment group', 'Assigned to', 'State', and 'Work Notes' fields, but are preceded by 'Primary' or 'Secondary'. By implementing this, the UI is similar to the original RITM form regarding Tasks and their fulfillment processes. However, on the server-side, **it treats each field as a unique entity within the form and can thus be directly accessed when generating reports.** This also consolidates the workflow, and decreases fulfillment time and miscommunication.

Upon creation of a new form, the layout will default to contain 'Primary Assignment' and 'Secondary Assignment' abilities. For the IT Infrastructure's use, the 'Primary Assignment' concerns their processes while the 'Secondary Assignment' and so forth concerns the fulfiller's processes. The form is configured to display a Tertiary, etc. Assignment ONLY IF the user specifies in the dialog. However, the current form can easily be modified to automatically display these Assignments and default the fields to 'empty' if not needed.

The revised form outline originally included client-side scripting to view the 'Primary Assignment' fields depending on the ServiceNow Instance used. For example, a Chubb technician would see the 'Primary

Assignment' as IT Infrastructure and the 'Secondary Assignment' as, say, Server Build, while an IBM technician would see the reverse. *Upon transfer to GitHub (see Section), these scripts were lost and can only be viewed on the Personal Developer Instance (PDI).* **These scripts are not necessary to the general function of the form, rather, to the roles and corresponding visions of such.**

2.4 Organize tabular layout

In the Form Designer (**Section 3.3**), the Assignments are sectioned. As discussed previously, the Assignment sections contain the same amount and abilities of fields, yet are two separate entities. The new RITM form layout is optimized when tabular vision is enabled on the ServiceNow instance.

To enable tabular vision, navigate to the main ServiceNow window > System Settings (gear icon in top right corner of the UI) > Forms > Enable Tabbed Forms.

If tabular vision is enabled, the form will tabulate each Assignment towards the bottom of the display. The end user can then view all of the Assignments directly on the form while still comprehending it as separate entities, as with the original RITM form.

If tabular vision is NOT enabled, the function is not hindered – only the UI display. Instead of tabular organization, the Assignments will be listed at the bottom of the form, separated with large headers.

2.5 Communicate between Assignment groups

Proposed but not entirely implemented

A large issue with the current workflow is that only certain events trigger notification. For example, a technician will only receive an email if a task is opened, closed, or recently assigned. If the fulfillment team closes their task (meaning the entire RITM is complete), the technician has no indication unless a) the fulfiller notifies them or b) they manually check the status**. Completed tickets then clutter the queue and inhibit other completion.**

Two approaches can be taken to eliminate this:

2.5.1 Automate 'Closed Complete' status

Depending on the type of service (**Section 2.6**), it is possible to write a server-side script that updates the 'State' field of both the RITM form and the 'Primary State' depending on the 'Secondary State'. More specifically, if the fulfillment team updates their 'Secondary State' field to 'Closed Complete', the script can automatically update the 'Primary State' field and/or close the RITM ticket. This automates the process of de-cluttering completed tickets from the queue.

2.5.2 Enable notifications

In some instances (such as server builds or decommissions), requestor confirmation is required before closing the tickets. Therefore, automating the 'Closed Complete' status would not necessary be helpful. Email notifications could be enabled as an alternative.

Notifications are enabled with a combination of scheduled/conditional scripting and the creation of a Notification Application File within the Studio Application. It could be useful to create notifications to the Requestor and Assignees 'State' change in the revised RITM form. For example, notifying the 'Primary Assignee' when the 'Secondary Status' is updated to 'Closed Complete' would prompt them to close the ticket sooner than if manually checking the status.

2.6 Other automation

Proposed but not entirely implemented

At times, the assignment of a fulfillment team can be ambiguous. To reduce incorrect assignment, the revised RITM form introduces a new field for general service category. The 'Category' field contains a drop-down of choices, allowing the requestor to select from the most popular services: Server Build, Server Decommission, Space Increase, Memory Increase, and Other. With the exception of the Other category, the 'Secondary Assignment' field will be pre-populated with the recommended fulfillment team upon 'Category' selection.

It is recommended to further specify choices (i.e. Disk Increase, CPU Increase) to further improve the automation of the fulfillment team.

Additional automation can be implemented by editing the RITM workflow (**Section 4.2**).

Interfaces, Applications and Resources

***Required**

3.1 ServiceNow Instance

3.2 *ServiceNow Application Studio

Accessed from main ServiceNow instance by navigating to **System Applications > Studio**

3.3 *ServiceNow Form Designer

IF CREATING NEW FORM: Accessed from Application Studio by navigating to **[Application] > + Create Application File > Forms & UI > Form > Create**

IF ACCESSING EXISTING FORM: Accessed from Application Studio by navigating to **[Application] > Application Explorer > Form & UI > Form > [Form]**

3.4 ServiceNow Personal Developer Instance

Visit https://developer.servicenow.com/ (Developer Homepage) and log in with the following credentials:

Email: Allison.Aprile@Chubb.com
Password: WHSritm01!

Navigate to **Manage > Instance > dev69743**.

[NOTE: The instance will be automatically released if unassessed for 10 days. 'Remaining inactivity' specifies how much longer the technician has to a) Wake the instance from hibernation (as prompted by the system) or b) select 'Extend instance'. With the completion of either open, the 'Remaining inactivity' will default back to 10 days.**]**

Click the URL or navigate to https://dev69743.service-now.com/ and login with the following credentials:

Username: admin
Password: WHSritm01!

Instance should mimic main ServiceNow UI. Proceed to the Application Studio/Form Designer (**Sections 3.2/3.3**).

3.5 ServiceNow Developers Training & Documentation

From the ServiceNow Developers homepage, navigate to the **Learn** tab.

3.6 GitHub

Complications

The revised RITM form design was completed primarily on the ServiceNow PDI. This does NOT have access to the Chubb instance and corresponding Chubb-specific ServiceNow applications. *For that reason, continued development through the PDI is not preferable.* However, security constraints may prevent further development in the Chubb Instance.

To use the Application Studio and test applications on the Chubb instance (or any instance), the user MUST have the 'admin' role, specifically for the following:

4.1 View/Edit Business Rules

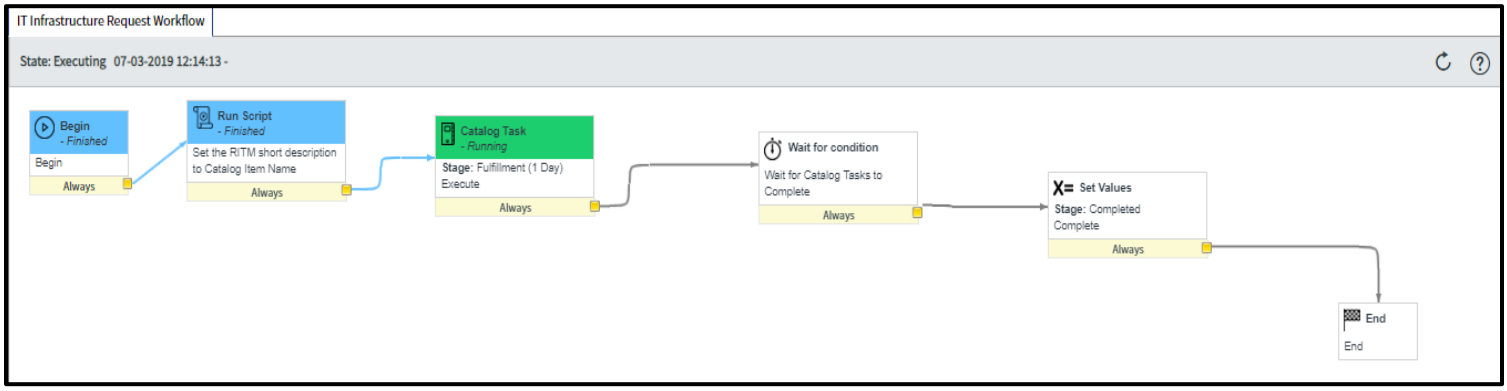
Business rules are generally server-side scripts that control record/table functions within ServiceNow. They play a large role in the automation of form fields and notifications, as well as role vision.

Administrator access is required to view and/or edit a business rule. **Before continuing development, it is necessary to thoroughly study and implement business rules similar to those in the original form.** The pre-existing business rules, particularly those concerning role vision, should be referenced and essentially mimicked in the new form, otherwise, security may not allow it to be put into production.

4.2 Revising workflow

Workflow automates a sequence of activities, such as pushing notifications, approvals and scripts, for records. It is essential to the automation process because of its cause-and-effect event trigger.

Administrator access is **NOT** required to view a workflow. To view the workflow for the original RITM form, navigate to a RITM Record and select 'Show Workflow' under the Related Links header. The workflow should open in a new window and appear similar to the following:



The original workflow is contingent on the Catalog Task structure, as highlighted in green in the above figure. Because the Task entity ideally would be eliminated, it is necessary to restructure the workflow.

A workflow should be implemented and tested thoroughly before attempting production.

Completion

Estimated time of completion: 3 - 6 months

- **Understanding ServiceNow Developer platform (if applicable):** 2 weeks
- **Reviewing original RITM business rules, workflow, etc. before developing**: 1 month
- **General form layout:** 1- 2 days
- **Revising and implementing business rules, workflow, etc. in development:** 1 - 2 months
- **Testing:** 1 – 2 months

Estimated expense: None

DEVELOPMENT

Access 'Revised RITM' Repository from GitHub (Continued Development)

Repository URL: https://github.com/aaprile2/RevisedRITMForm-Chubb-2019.git

Cloning the repository:

Navigate to **Studio > Select Application > Import from Source Control**

Proceed by entering the above URL and appropriate GitHub Credentials.

Development Procedure:

Once the above URL is cloned, edits will only be made to the GitHub repository corresponding to the entered credentials. Edits WILL NOT and CANNOT be made to the original repository. **The original repository acts only as a template.**

Importing from Source Control instantiates the template application into the Application Studio. To continue development, navigate to the **main ServiceNow window > System Applications > Studio**. Then, open the RevisedRITMForm-Chubb-2019 application and proceed, making sure to save any work.

Saving the work through Studio will NOT automatically save to the repository.*

To save to the repository, in the Application Studio navigate to **Source Control > Commit Changes**. Make sure to review the noted changes before committing as a commit CANNOT be reversed.

Access 'Requested Item' Application from PDI (Continued Development - expires 8/26/2019)

See Section for detailed instructions to access and retain the PDI. **Remember that the instance will expire after ten days of inactivity.**

Continue development using the Application Studio. **Save any work through Studio and/or a GitHub repository.***

Build from Scratch (New Development)

Navigate to the main **ServiceNow page > System Applications > Studio**, then select **Create Application > Start from Scratch** (select Create).

The documentation linked in Section is recommended for familiarizing the Studio interface and capabilities.

Save any work through Studio and/or a GitHub repository.*

*It is recommended to save both to Studio and a GitHub repository. The repository acts as a backup for development and can be accessed regardless of permissions/PDI state. See Section for instructions to save to a repository or view the **documentation**.

Support

For any questions regarding continued development, reach out to Allison Aprile at the following:

Email (preferred): allisonnicoleaprile@gmail.com

Mobile Phone: (908) 442-2108