

Problem Management

Functional Requirement Specification

| Authors | Austin Magno  James Hardy  Jimmy Merrick  Malykai Mejia  Zach Funderburk |
| --- | --- |
| Version | 4.0 |
| Approved By | Prakash Agrawal |
| Approved (Y/N) | Yes |
| Date Approved | 12/10/2021 |

Table of Contents

[**1 Introduction**](#_uanewl2ni465) **3**

[1.1 Background](#_1t8sy3lw5bla) 3

[1.2 Purpose](#_j35g6fek9lz2) 3

[1.3 Scope](#_jvrt62ossmq) 3

[**2 Form Design**](#_js0nkg5jv2ls) **4**

[2.1 Field Label Update](#_1u3cwdxeriip) 4

[2.1.1 Problem Form](#_we3gabom2ez1) 4

[2.1.2 Problem Task Form](#_60q06wvldz76) 4

[2.2 Category Choice List Fields](#_qj0g5um3dexg) 5

[2.3 Categories & Subcategories](#_7yzvofmv6ya5) 5

[2.4 Problem States & Substates](#_bqrc32figj0b) 5

[2.4.1 Problem Lifecycle](#_ost1zfvk0cxb) 6

[2.4.2 State Model](#_u5z3ni374vdr) 7

[2.4.3 Substates](#_mar69osm5n58) 7

[2.5 Related Lists](#_hkk4bjyefr1o) 8

[2.6 Form Layout](#_6589ihtapr43) 8

[2.6.1 Problem Form](#_ahbiwoaoirlo) 8

[2.6.2 Problem Task Form](#_n9tpaemawbup) 10

[2.7 Form View Options](#_z10nulyvwqno) 11

[**3 Form Functionality**](#_etcv5ouse9j1) **12**

[3.1 Business Rules](#_rtx3kk4m32vf) 12

[3.2 Assignment Rules](#_fvtlr3s3q346) 12

[3.3 Data Policies](#_ldlc1fowx5o9) 13

[3.3.1 Problem Table](#_vws5blqk8wrn) 13

[3.3.2 Problem Task Table](#_9p2e8os8kccj) 14

[3.4 UI Policies](#_k9ljii2uu7y3) 14

[3.4.1 Problem Form](#_bw11u17qakap) 14

[3.4.1.1 ‘New’ State](#_lpihtvwg4jkb) 14

[3.4.1.2 ‘In Progress’ State](#_rnv3bltwnfr6) 15

[3.4.1.3 ‘Resolved’ State](#_fsyt2c0jub1) 16

[3.4.1.4 ‘Closed’ State](#_o1n4cze0wj3j) 17

[3.4.2 Problem Task Form](#_jxm025a9iwxz) 18

[3.4.2.1 ‘New’ State](#_ma8bmeeouh78) 18

[3.4.2.2 ‘In Progress’ State](#_tf373d4bkufj) 19

[3.4.2.3 ‘Complete’ State](#_ybo0wrakcey3) 19

[**4 UI Actions**](#_2z94oxcp1w05) **20**

[4.1 UI Action Definitions](#_yfdtq1bdum95) 20

[4.1.1 Problem Form](#_w6mkfxa3to1v) 20

[4.1.2 Problem Task Form](#_yabxa5x39kjj) 21

[4.2 UI Action-State Matrix](#_r9hej1ehkz3n) 22

[4.2.1 Problem Form](#_qucpn968v8aw) 22

[4.2.2 Problem Task Form](#_xu1iypz714r0) 22

[4.3 Change Record](#_e1av04zdsm3f) 23

[4.3.1 Problem Data for Change Record](#_884gl19zdvt5) 23

[**5 Notifications**](#_qivucm8dklze) **23**

[5.1 Problem Notification](#_jqembeufw8nb) 24

[5.1.1 Problem Notification Workflow](#_nsjvpwyugksj) 24

[5.1.2 Problem Notification Contents](#_b6mcgtobcp73) 24

[**6 Application and Modules**](#_oy07m9nu786h) **25**

[**7 Conclusion**](#_2n4byyyyejr2) **26**

# 1 Introduction

## 1.1 Background

The ITIL (Information Technology Infrastructure Library) was created to align the needs of the customer or business with the IT services produced by the ITSM (Information Technology Service Management) provider. The ITSM platform utilizes ITIL practices to resolve customer concerns and issues in a timely manner; while at the same time improving supplied IT services and reducing overall costs. An ITSM platform can exist in numerous forms. The platform can go through several different executions and states; as long as the purposes and services have been delivered per the customer’s requirements. Through the years, experts have created many ITSM templates to improve this process of data intake and management; as well as modifying the processes of IT service delivery as needed. Of course, these templates are complemented by the practices of the Information technology infrastructure library (including functions from Problem Management).

## 1.2 Purpose

In ServiceNow, the Problem table is used to track application and system problems. These can be raised either internally, or through repeated Incident tickets that relate to the same core issues. Like incidents, problems will need to be researched and have corrections implemented, though the timeframe for problems can be extended to account for a more permanent solution to be found. To this end, having an easy and friendly Problem Application and table will make the creation and progression of Problem records a more streamlined process.

## 1.3 Scope

The scope of this document will be focused on the problem table and related forms / fields / tasks. Fields and functionalities will be customized to meet the needs of the customer. This customization process will include adding hidden/mandatory fields at certain conditions; as well as stopping the user from editing a given field during certain states of the problem lifecycle. This project will also develop new UI actions and field connections via backend scripts, as described in the respective sections

# 2 Form Design

## 2.1 Field Label Update

Problem field names will be updated to match with created Problem Tasks. A new field of “Substate” will also be added. **Dictionary override will be implemented**; so as not to interfere with parent table fields.

### 2.1.1 Problem Form

| **Current Name** | **New Name** |
| --- | --- |
| Number | Problem ID |
| First Reported By | Reported By |
| Configuration Item | CI |
| Problem Statement | Problem Subject |
| Description | Problem Details |
| Work notes list | Subscribers List |

### 2.1.2 Problem Task Form

| **Current Name** | **New Name** |
| --- | --- |
| Problem | Problem ID |
| Number | Problem Task ID |
| Configuration Item | CI |
| Short Description | Problem Task Subject |
| Description | Problem Task Details |
| Work notes list | Subscribers List |

## 

## 2.2 Category Choice List Fields

| **Categories** |
| --- |
| Account Access |
| Hardware |
| Software |
| Other |

## 2.3 Categories & Subcategories

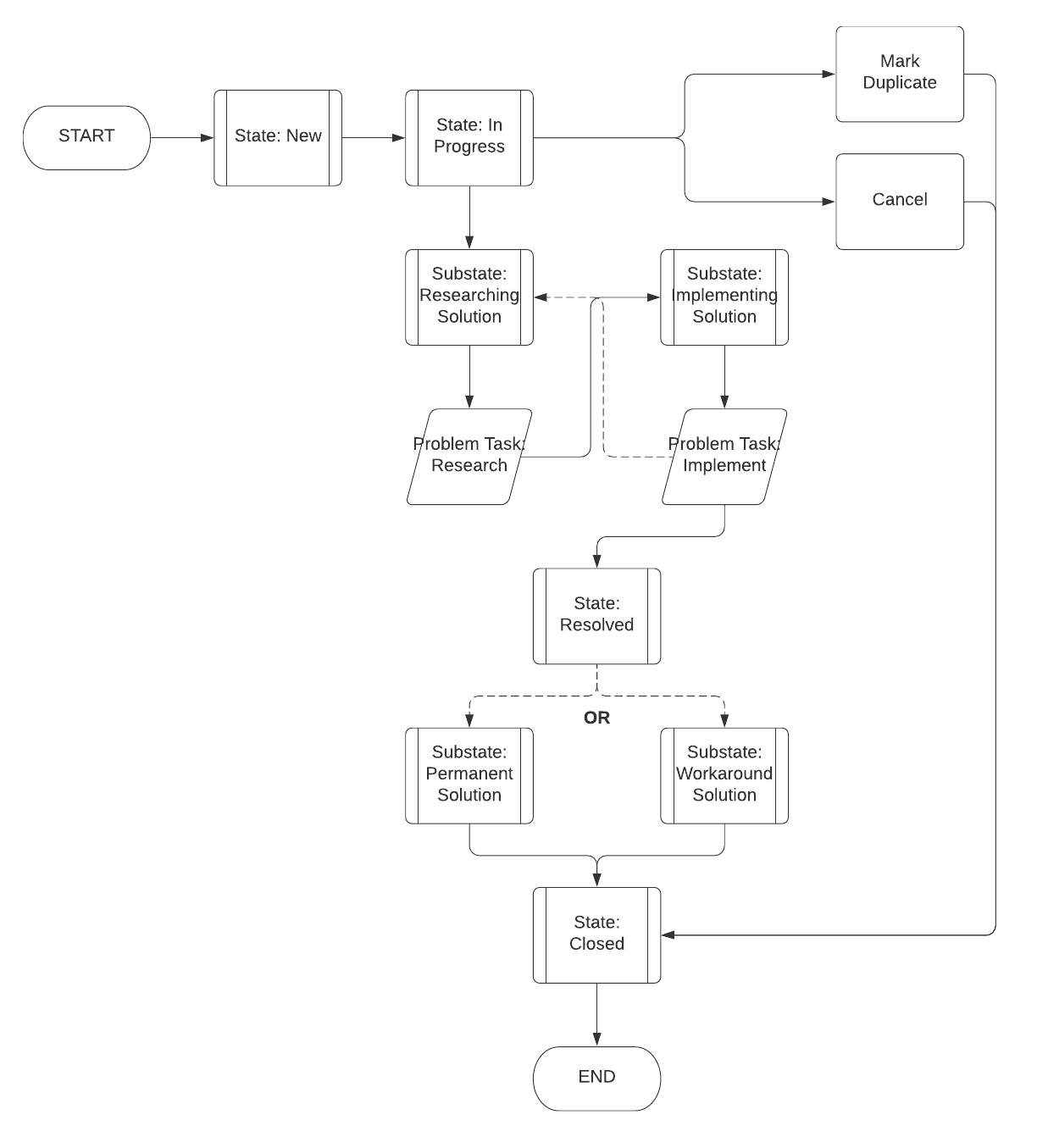
New Subcategories will be attached to custom Categories.

| **Category** | **Subcategories** |
| --- | --- |
| Account Access | Email, ServiceNow, Other |
| Hardware | Computer, Router, Peripherals, Other |
| Software | Messenger, Pay Portal, OS, Other |
| Other | Other |

## 2.4 Problem States & Substates

The Problem Records will transition between 4 primary states, with substates for In Progress and Resolved. The Substate choices will be dependent on which State the problem is currently set to. The Problem Form Process Flow will represent the primary four states.

### 2.4.1 Problem Lifecycle

[](https://lucid.app/documents/edit/695887ec-5820-41dd-b19f-5d87d0954de1/0?callback=close&name=docs&callback_type=back&v=1098&s=612)

### 

### 2.4.2 State Model

| **State** | **Enter Condition** | **Remarks** |
| --- | --- | --- |
| New | N/A | Assignment group is required before moving to “In Progress” |
| In Progress | State is ‘New’ | Contains 2 Substates: Researching and Implementing Fix |
| Resolved | State is ‘In Progress’ | Contains 2 Substates: Workaround and Permanent |
| Closed | State is ‘In Progress’ or ‘Resolved’ | Contains 4 Substates: Canceled, Duplicate, Workaround and Permanent (Dependent on the Resolved substate) |

### 2.4.3 Substates

| **Substates** | **State** | **Remarks** |
| --- | --- | --- |
| Researching Solution | In Progress | Selectable when State is “In Progress” |
| Implementing Solution | In Progress | Selectable when State is “In Progress” and Substate is “Researching Solution” |
| Workaround Solution | Resolved Closed | Selectable when State in “Resolved”; Stays visible and in read-only after moving to “Closed” |
| Permanent Solution | Resolved Closed | Selectable when State in “Resolved”; Stays visible and in read-only after moving to “Closed” |
| Duplicate Problem | Closed | Marks the form as duplicate |
| Problem Canceled | Closed | Marks the form as canceled |

## 

## 2.5 Related Lists

The Related List section will show additional records that connect to the created problem. All Related Lists will appear for the problem throughout its lifecycle

| **Related List** | **Description** | **Remarks** |
| --- | --- | --- |
| Problem Tasks | Lists the connected Problem Tasks | Hidden when list is empty |
| Incidents | Lists the connected Incident record(s) | Hidden when list is empty |
| Affected CIs | Lists the connected Affected CIs |  |
| Change Request | Lists the connected Change Request records | Hidden when list is empty |
| Attached Knowledge | Lists the connected Attached Knowledge records | Hidden when list is empty |

## 2.6 Form Layout

Each stage of the Problem form will display specified fields, depending on what will be needed for that state. Certain fields will be viewable on all state forms, while some will only be set once the problem is moved to a specific state.

### 2.6.1 Problem Form

| **Main Section** | | | | |
| --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Reference** | **Max Length** | **Remarks** |
| **Problem ID** | String |  | 40 | Auto generated |
| **Reported by** | Reference | sys\_user | 32 |  |
| **Category** | Choice List |  |  |  |
| **Subcategory** | Choice List |  |  |  |
| **CI** | Reference | cmdb\_ci | 32 |  |
| **Problem Subject** | String |  | 160 |  |
| **Problem Details** | String |  | 4000 |  |
| **State** | Choice List |  | 40 | Changed based on UI Action |
| **Substate** | Choice List |  |  | Only available when State is not “New” |
| **Impact** | Integer |  | 40 |  |
| **Urgency** | Integer |  | 40 |  |
| **Priority** | Integer |  | 40 | Calculated based on impact and urgency |
| **Assignment group** | Reference Field | sys\_user\_group | 32 |  |
| **Assigned to** | Reference Field | sys\_user | 32 |  |

| **Notes Section** | | | | |
| --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Reference** | **Max Length** | **Remarks** |
| **Subscribers List** | List | sys\_user | 4000 |  |
| **Work Notes** | Journal Input |  | 4000 |  |
| **Activities** | *Auto generated from record updates and Work notes* | | | |

| **Analysis Information Section** | | | | |
| --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Reference** | **Max Length** | **Remarks** |
| **Workaround** | HTML |  | 4000 |  |
| **Cause notes** | HTML |  | 4000 |  |
| *Only available when State is not “New”* | | | | |

| **Resolution Information Section** | | | | |
| --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Reference** | **Max Length** | **Remarks** |
| **Resolved by** | Reference | sys\_user | 32 | Auto populated |
| **Resolved** | Date/Time |  | 40 | Auto generated |
| **Fix notes** | HTML |  | 65,536 |  |
| *Only Available when State is not “Resolved” or “Closed”* | | | | |

| **Other Information Section** | | | | |
| --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Reference** | **Max Length** | **Remarks** |
| **Opened by** | Reference | sys\_user | 32 | Auto populated |
| **Opened** | Date/Time |  | 40 | Auto generated |
| **Closed by** | Reference | sys\_user | 32 | Auto populated |
| **Closed** | Date/Time |  | 40 | Auto generated |

### 2.6.2 Problem Task Form

| **Main Section** | | | | |
| --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Reference** | **Max Length** | **Remarks** |
| **Problem Task ID** | String |  | 40 | Auto generated |
| **Problem ID** | Reference | problem | 32 |  |
| **Priority** | Integer |  | 40 | Auto calculated |
| **Problem Task Subject** | String |  | 160 |  |
| **Problem Task Details** | String |  | 4000 |  |
| **Assignment group** | Reference | sys\_user\_group | 32 |  |
| **CI** | Reference | cmdb\_ci | 32 |  |
| **State** | Choice List |  | 40 | Changed based on UI Action |

| **Notes Section** | | | | |
| --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Reference** | **Max Length** | **Remarks** |
| **Subscribers List** | List | sys\_user | 4000 |  |
| **Work Notes** | Journal Input |  | 4000 |  |

| **Closure Information Section** | | | | |
| --- | --- | --- | --- | --- |
| **Field Name** | **Type** | **Reference** | **Max Length** | **Remarks** |
| **Closed by** | Reference | sys\_user | 32 | Auto populated |
| **Closed** | Date/Time |  | 40 | Auto generated |
| **Close notes** | HTML |  | 65,536 |  |

## 2.7 Form View Options

| **View Options** |
| --- |
| Accept Risk Dialog Form View |
| Assess Dialog Form View |
| Cancel Dialog Form View |
| Cxs\_popup |
| Default View |
| Mark Duplicate Dialog Form View |
| Mobile |
| Resolve Dialog Form VIew |
| Start Fix Dialog Form View |
| Sys\_popup |
| Workspace |

# 3 Form Functionality

## 3.1 Business Rules

Business rules are server side scripts attached to a corresponding table in ServiceNow. The business rules (BR’s) implemented will consist of notification triggers and problem task generation.

| **Name** | **When to Run** | **Condition(s)** | **Action(s)** |
| --- | --- | --- | --- |
| Problem Create Researching Task | After Update | State is ‘In Progress’ and Substate changes to ‘Researching Solution’ | Create Problem Task |
| Problem Create Implementation Task | After Update | State is ‘In Progress’ and Substate changes to Implementing Solution’ | Create Problem Task |
| Problem\_Task-Delete-Guard-JH | Before Delete | Active is ‘True’, State is not ‘Closed’; | Abort action, show error message |

## 3.2 Assignment Rules

Assignment rules are an autonomous way of setting values for the “assigned to” or “assignment group” fields when certain conditions are met. Here, they will be constructed to directly link “Assignment Groups” to “Categories” that the problems will be filed under. Once a specific Category is set, a group will be designated through a Client Script and Script Include function.

| **Category** | **Assignment Group** |
| --- | --- |
| Account Access | Data Team |
| Hardware | IT - Hardware |
| Software | IT - Software |
| Other | Help Desk |

## 3.3 Data Policies / Table Configurations

Data Policies are established in the event that information from an external source needs to be adopted into the ServiceNow instance. Controlling only the “Mandatory” and “Read Only” interactions on a table (and its related form), the incoming data will be characterized based on the applicable requirements of a new Problem or Problem Task. Generally speaking, we used ui policies over data policies to control different states.

### 3.3.1 Problem Table Configuration

| **Field Name** | **Mandatory** | **Read Only** | **Remarks** |
| --- | --- | --- | --- |
| **Problem ID** | no | yes |  |
| **Reported by** | yes | no |  |
| **Category** | yes | no |  |
| **Subcategory** | no | no |  |
| **CI** | no | no |  |
| **Problem Subject** | yes | no |  |
| **Problem Details** | yes | no | DP: Only when state is new |
| **State** | no | yes | UI actions change State |
| **Substate** | no | yes | Only available when State is not “New” |
| **Impact** | no | no |  |
| **Urgency** | no | no |  |
| **Priority** | no | yes | Calculated based on impact and urgency |
| **Assignment group** | no | no | Populated based on Category |
| **Assigned to** | Yes on condition (otherwise no); | no | DP:Only when state is in progress |

### 3.3.2 Problem Task Table Configuration

| **Field Name** | **Mandatory** | **Read Only** | **Remarks** |
| --- | --- | --- | --- |
| **Problem Task ID** | no | yes | Auto generated |
| **Problem ID** | no | yes | Values copied from parent Problem |
| **Priority** | no | yes | Values copied from parent Problem |
| **Problem Task Subject** | no | yes | Values copied from parent Problem |
| **Problem Task Details** | no | yes | Values copied from parent Problem |
| **Assignment group** | no | no |  |
| **CI** | no | yes | Values copied from parent Problem |
| **State** | no | yes | UI actions change State |

## 3.4 UI Policies

New UI policies will update the status of form fields upon State changes. These policies will set certain fields to be Mandatory or Read Only, based on other field inputs and the current state of the problem record.

### 3.4.1 Problem Form

#### 3.4.1.1 ‘New’ State

| **Field Name** | **Mandatory** | **Read Only** | **Visible** | **Remarks** |
| --- | --- | --- | --- | --- |
| **Problem ID** | no | yes | yes | Auto generated |
| **Reported by** | yes | no | yes |  |
| **Category** | yes | no | yes |  |
| **Subcategory** | no | no | no | Hidden when category is ‘None’ |
| **CI** | no | no | yes |  |
| **Problem Subject** | yes | no | yes |  |
| **Problem Details** | yes | no | yes |  |
| **State** | no | yes | yes | UI actions change State |
| **Substate** | no | no | no | Hidden when State is ‘New’ |
| **Impact** | no | no | yes | Choice List without ‘None’ option; Default value is 3 |
| **Urgency** | no | no | yes | Choice List without ‘None’ option; Default value is 3 |
| **Priority** | no | yes | yes | Calculated based on impact and urgency |
| **Assignment group** | no | yes | yes | Populated based on Category |
| **Assigned to** | no | no | yes |  |

#### 3.4.1.2 ‘In Progress’ State

| **Field Name** | **Mandatory** | **Read Only** | **Visible** | **Remarks** |
| --- | --- | --- | --- | --- |
| **Problem ID** | no | yes | yes |  |
| **Reported by** | no | yes | yes |  |
| **Category** | no | yes | yes |  |
| **Subcategory** | no | yes | no |  |
| **CI** | no | yes | yes |  |
| **Problem Subject** | no | yes | yes |  |
| **Problem Details** | no | yes | yes |  |
| **State** | no | yes | yes | UI actions change State |
| **Substate** | no | yes | yes |  |
| **Impact** | no | no | yes | Choice List without None option; Default value is 3 |
| **Urgency** | no | no | yes | Choice List without None option; Default value is 3 |
| **Priority** | no | yes | yes | Calculated based on impact and urgency |
| **Assignment group** | no | yes | yes |  |
| **Assigned to** | yes | no | yes |  |

#### 3.4.1.3 ‘Resolved’ State

| **Field Name** | **Mandatory** | **Read Only** | **Visible** | **Remarks** |
| --- | --- | --- | --- | --- |
| **Problem ID** | no | yes | yes |  |
| **Reported by** | no | yes | yes |  |
| **Category** | no | yes | yes |  |
| **Subcategory** | no | yes | no |  |
| **CI** | no | yes | yes |  |
| **Problem Subject** | no | yes | yes |  |
| **Problem Details** | no | yes | yes |  |
| **State** | no | yes | yes | UI actions change State |
| **Substate** | no | yes | yes |  |
| **Impact** | no | no | yes | Choice List without None option; Default value is 3 |
| **Urgency** | no | no | yes | Choice List without None option; Default value is 3 |
| **Priority** | no | yes | yes | Calculated based on impact and urgency |
| **Assignment group** | no | yes | yes |  |
| **Assigned to** | no | yes | yes |  |

#### 3.4.1.4 ‘Closed’ State

| **Field Name** | **Mandatory** | **Read Only** | **Visible** |
| --- | --- | --- | --- |
| **Problem ID** | no | yes | yes |
| **Reported by** | no | yes | yes |
| **CI** | no | yes | yes |
| **Category** | no | yes | yes |
| **Subcategory** | no | yes | yes |
| **State** | no | yes | yes |
| **Substate** | no | yes | yes |
| **Problem Subject** | no | yes | yes |
| **Problem Details** | no | yes | yes |
| **Impact** | no | yes | yes |
| **Urgency** | no | yes | yes |
| **Priority** | no | yes | yes |
| **Assignment group** | no | yes | yes |
| **Assigned to** | no | yes | yes |

### 3.4.2 Problem Task Form

#### 3.4.2.1 ‘New’ State

| **Field Name** | **Mandatory** | **Read Only** | **Visible** | **Remarks** |
| --- | --- | --- | --- | --- |
| **Problem Task ID** | no | yes | yes | Values copied from parent Problem |
| **Problem ID** | no | yes | yes | Values copied from parent Problem |
| **Priority** | no | yes | yes | Values copied from parent Problem |
| **Problem Task Subject** | no | yes | yes | Values copied from parent Problem |
| **Problem Task Details** | no | yes | yes | Values copied from parent Problem |
| **Assignment group** | no | yes | yes |  |
| **CI** | no | yes | yes | Values copied from parent Problem |
| **State** | no | yes | yes | UI actions change State |

#### 3.4.2.2 ‘In Progress’ State

| **Field Name** | **Mandatory** | **Read Only** | **Visible** | **Remarks** |
| --- | --- | --- | --- | --- |
| **Problem Task ID** | no | yes | yes | Values copied from parent Problem |
| **Problem ID** | no | yes | yes | Values copied from parent Problem |
| **Priority** | no | yes | yes | Values copied from parent Problem |
| **Problem Task Subject** | no | yes | yes | Values copied from parent Problem |
| **Problem Task Details** | no | yes | yes | Values copied from parent Problem |
| **Assignment group** | no | no | yes |  |
| **CI** | no | yes | yes | Values copied from parent Problem |
| **State** | no | yes | yes | UI actions change State |

#### 3.4.2.3 ‘Complete’ State

| **Field Name** | **Mandatory** | **Read Only** | **Visible** |
| --- | --- | --- | --- |
| **Problem Task ID** | no | yes | yes |
| **Problem ID** | no | yes | yes |
| **Priority** | no | yes | yes |
| **Problem Task Subject** | no | yes | yes |
| **Problem Task Details** | no | yes | yes |
| **Assignment group** | no | yes | yes |
| **CI** | no | yes | yes |
| **State** | no | yes | yes |

# 4 UI Actions

UI Actions (user interface actions) will be utilized to provide additional functionality to the Problem table and its related forms. The available UI Actions will be determined based on the state of the given problem. All out-of-the-box UI Actions will be used. Other UI Actions are created and customized based on the operation that needs to be performed. *To prevent a user from continuing the problem state outside of our pre-designed protocol, the UI actions call a ‘script-include’ that checks the number of non-closed problem tasks, and aborts the action if any are found.*

## 4.1 UI Action Definitions

### 4.1.1 Problem Form

| **UI Action** | **Type** | **Action** |
| --- | --- | --- |
| **Submit** | Button | Submits new record and returns to list |
| **Save** | Context Menu Option | Saves change(s) and remains on form |
| **Research** | Button | Changes state to ‘In Progress’ and substate to ‘Researching Solution’ |
| **Implement** | Button | Changes substate to ‘Implementing Solution’ |
| **Create Change** | Context Menu Option | Creates a Change Request Record (Normal) |
| **Resolved: Permanent** | Button | Changes state to ‘Resolved’ and substate to ‘Permanent’ |
| **Resolved: Workaround** | Button | Changes state to ‘Resolved’ and substate to ‘Workaround’ |
| **Cancel Problem** | Button | Changes state to ‘Closed’ and substate to ‘Canceled’ |
| **Close Problem** | Button | Changes state to ‘Closed’ |
| **Update** | Button | Updates the values in the form |
| **Mark Duplicate** | Buton | Marks Problem record as a duplicate of an existing record |
| **Delete** | Button | Deletes the current record |
| **Assess** | Button | Changes state to ‘In Progress’ and substate to ‘Researching Solution’ |

### 4.1.2 Problem Task Form

| **UI Action** | **Type** | **Action** |
| --- | --- | --- |
| **Cancel** | Button | Closes the task and mark it as ‘Canceled’ |
| **Complete/Close** | Button | Closes the task and mark it as ‘Complete’ |
| **Update** | Button | Updates the values in the form |
| **Work In Progress** | Button | Changes state to ‘In Progress’ |
| **Save** | Context Menu Option | Saves change(s) and remains on form |
| **Delete** | Button | Deletes the current record |

## 

## 4.2 UI Action-State Matrix

### 4.2.1 Problem Form

|  | **State** | | | |
| --- | --- | --- | --- | --- |
| **UI Action** | **New** | **In Progress** | **Resolved** | **Closed** |
| **Submit** | *Only available in new forms* | | | |
| **Save** | Y | Y | Y | Y |
| **Research** |  | Y |  |  |
| **Implement** |  | Y |  |  |
| **Create Change** |  | Y | Y |  |
| **Resolved: Permanent** |  | Y |  |  |
| **Resolved: Workaround** |  | Y |  |  |
| **Cancel Problem** |  | Y |  |  |
| **Close Problem** |  |  | Y |  |
| **Update** | Y | Y | Y |  |
| **Mark Duplicate** |  | Y |  |  |
| **Delete** | Y | Y | Y | Y |
| **Assess** | Y |  |  |  |

### 4.2.2 Problem Task Form

|  | **State** | | |
| --- | --- | --- | --- |
| **UI Action** | **New** | **In Progress** | **Closed** |
| **Cancel** |  | Y |  |
| **Complete/Close** |  | Y |  |
| **Update** | Y | Y |  |
| **Work In Progress** | Y |  |  |
| **Save** | Y | Y | Y |
| **Delete** | Y | Y | Y |

## 4.3 Change Record

A new UI Action will be created, allowing for the automated creation of a Change Record containing data from the problem. This action will only be available while the record is in the In Progress state.

### 4.3.1 Problem Data for Change Record

| **Field Name** | **Remarks** |
| --- | --- |
| Requested By | Set to Current User |
| Problem Subject |  |
| Problem Details |  |
| Category |  |
| Configuration Item |  |
| Priority |  |
| Impact |  |
| Justification | Not in the Problem Field, but set manually by the UI Action |

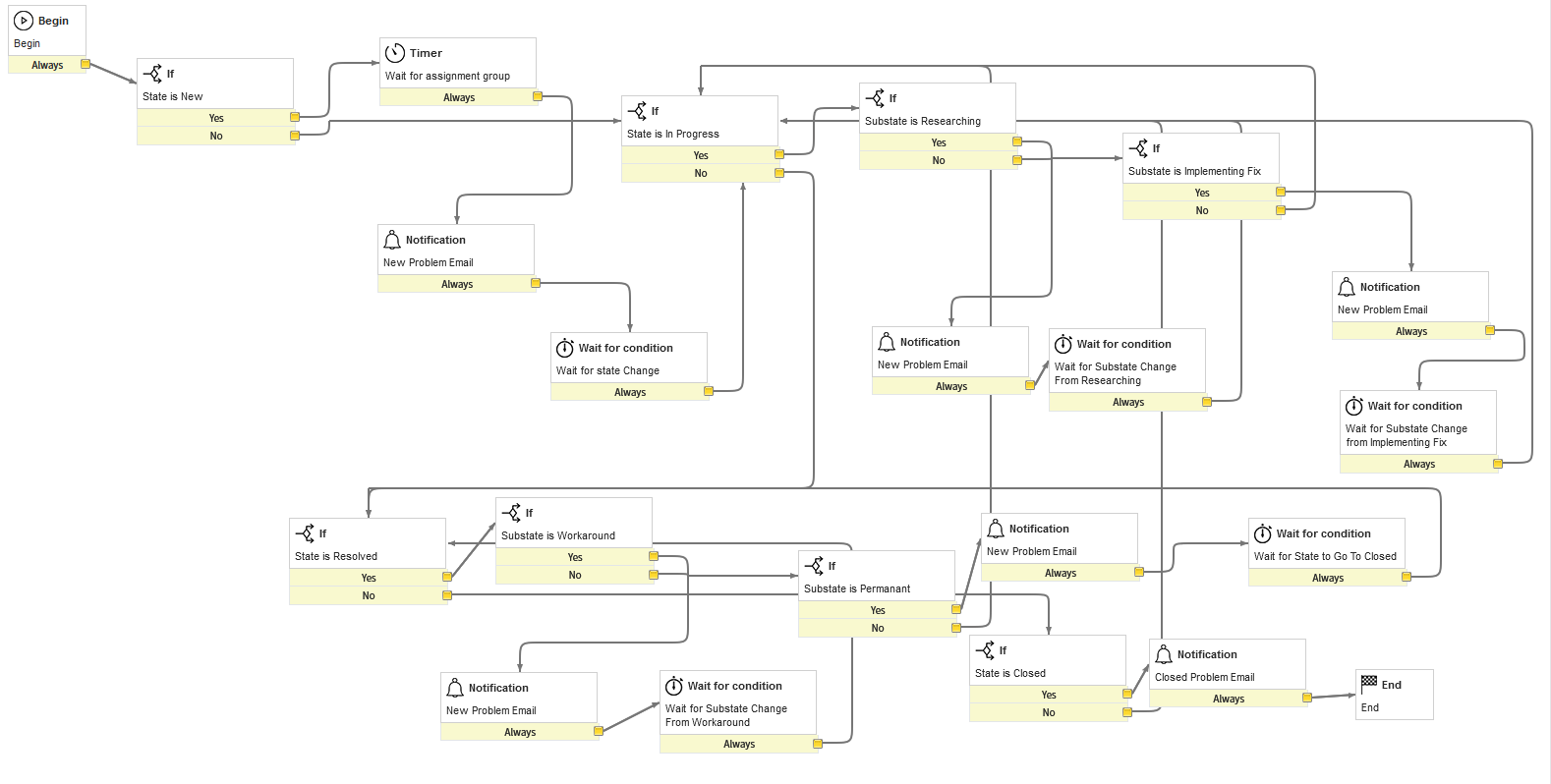
# 5 Notifications

Notifications will be generated upon creation to: assigned by, caller and assignment group. There will also be email notifications sent to the assignment group for tasks during the In Progress state for their respective sub-states, In Research and Implementing Fix. Additionally there will be email notifications sent to the assignment group manager, assigned to, and assignment group upon completion or closure of the problem. All Notifications will occur via a Workflow tied to the problem table.

## 5.1 Problem Notification

The notifications will occur via workflow with ‘if’ activities for checking state and substate. There will be notifications sent particular to state, for example, if the state is new the new problem notification will be sent to the user that initialized the problem. After the notification there is a wait for condition activity for checking when the state or substate is changed.

### 5.1.1 Problem Notification Workflow



### 5.1.2 Problem Notification Contents

| **When to Send** | **Recipient** | **Subject** | **Body** |
| --- | --- | --- | --- |
| Creation of Problem Record | Caller | “Problem ${number} has been created” | “A new problem has been created.  Problem Created: ${number}  Created on: ${sys\_created\_on}  Assignment Group: ${assignment\_group}  Assigned To: ${assigned\_to}  Please Do Not Reply to this email this is an auto-generated email.” |
| Creation of Problem Task Researching | Assigned User | “Problem ${number} is now in Researching” | “Task Notification  Assigned To: ${assigned\_to}  Assignment Group: ${assignment\_group}  Created on: ${sys\_created\_on}  Please do not reply to this Email, It is auto-generated.” |
| Creation of Problem Task Implementing Fix | Assigned User | “Problem ${number} is in Implementing Fix” | “Task Notification  Assigned To: ${assigned\_to}  Assignment Group: ${assignment\_group}  Created on: ${sys\_created\_on}  Please do not reply to this Email, It is auto-generated.” |
| Updating Problem State to Resolved, substate is Workaround | Assigned User | “${number} has been resolved with a Workaround.” | “Problem ${number} has been resolved with a Workaround.  Assigned To: ${assigned\_to}  Assignment Group: ${assignment\_group}  Created on: ${sys\_created\_on}  Resolved By: ${resolved\_by}  Resolved At: ${resolved\_at}  Please do not reply to this Email, It is auto-generated. ” |
| Updating Problem State to Resolved, substate is Permanent | Assigned User | “${number} has been resolved with a Permanent.” | “Problem ${number} has been resolved with a Permanent Fix.  Assigned To: ${assigned\_to}  Assignment Group: ${assignment\_group}  Created on: ${sys\_created\_on}  Resolved By: ${resolved\_by}  Resolved At: ${resolved\_at}  Please do not reply to this Email, It is auto-generated.” |
| Updating Problem State to Closed | Caller | “${number} has been closed.” | “Problem ${number} has been Closed.  Assigned To: ${assigned\_to}  Assignment Group: ${assignment\_group}  Created on: ${sys\_created\_on}  Please do not reply to this Email, It is auto-generated.” |

# 6 Application and Modules

The Problem Record data will be located in the Problem Application. This application will have a list of modules to access assigned problem records, create new problem records, and view resolved records, all located on the Problem table.

| **Problem Application Modules** | **Description** |
| --- | --- |
| New Problem | Create a new problem |
| Open | Shows a list of all open problems |
| In Progress | Shows a list of all “In Progress” problems |
| Resolved | Shows a list of all “Resolved” problems |
| All | Shows a list of all problems |
| Overview | Shows a page of different reports |

# 

# 

# 

# 

# 

# 7 Follow

The ***‘follow’*** functionality is used to allow teams to monitor and collaborate using a messenger system. This can be useful; as the conversation is added to work notes. We implemented this functionality to allow teams to communicate more effectively:

# 

# 

# 

# 8 Conclusion

Using the information provided in the document, developers can construct a functioning ‘Problem’ application that links to other members of the ITIL, (such as ‘Incident’ and ‘Change’). In this single tenant instance, the unique version of the problem table (and form) serves as part of a new ITSM template that takes and processes necessary information as the customer requires: Using custom business rules and assignment rules, inputting data into the form becomes autonomous and efficient. Using data policies and UI policies, the fields in new and pre-existing records can be prioritized, visible, and secured when necessary. As the problem record continues through the lifecycle, preset ‘Notifications’ will be generated to the requesting user, the assigned worker, and to the assignment group manager in order to provide updates about the problem’s status.

With the above customizations, the customer will be able to manage problems in accordance with policies and procedures.