Week 1: ServiceNow Fundamentals

Submitted by: Ananya Herle: Dayananda Sagar College of Engineering
Superset ID: 5098517
Registered mail id: ananyaherle2003@gmail.com

What is ServiceNow:

- **Cloud based IT service** providing platform that runs on "**Application platform as a Service**" model. Enables organizations to easily report incidents and raise tickets, eliminating excessive dependencies on the IT wing.
- Established in the year **2003** by **Fred Luddy** (formerly called GlideSoft). Currently headed by **Bill McDermott** (CEO)
- Provides clients with **Infrastructure** (Security, Computational Resources, Security), **Applications** (Prebuilt ServiceNow apps or 3rd party app integrations, Inbuilt workflows), and **Platform** (for customisations, configurations and building custom applications/ workflows).
- Easily accessible via **mobile devices** or **desktops**.
- Spans services to various parts of the World.
- Consolidated Definition: Servicenow is a software company introduced to solve, IT issues faced by large organizations, with the provision of a robust, user friendly cloud based application platform, allowing employees to independently deal with business problems

ServiceNow Platform Overview

- Hosted on enterprise cloud infrastructure, has a **single, common data model and database**. Users can connect to a ServiceNow instance, which follows single tenant architecture to leverage its functionalities.
- ServiceNow classifies its applications or workflows in 4 major categories:
    a) **IT Workflows**: to support enterprise wide IT activities (ITSM, business management, security management)
    b) **Creator Workflows:** for application developers; helps in creation of custom solutions and applications (GAC, App Engine, Studio, Integration Hub)
    c) **Customer Workflows:** workflows to solve customer issues and queries; (CSM, Telecommunication service management)
    d) **Employee Workflows:** targeted at needs of employees (HR, Procurement, Workplace services)

- Upon requesting for an instance, it's servicenow's responsibility to provide infrastructure, platform and necessary computing devices for the client.
- Unlike most cloud models, Servicenow is **not multi-tenant**, Implying each organization handles its own platform, database which does not intermingle with other organizations.
- Follows Multi-Instance Architecture; Multiple instances each having its own platform and database can be launched by different authorized users.
- **Redundant copies of the instance are stored in different Servicenow data centers, to promote availability. Redundancy applicable to all layers**.
- Backups and Security provided using certified security measures provided by 3rd party organizations
- Instance data, processes and administrative tasks can be grouped into logical grouping called Domains. All users can see records under "global domain", but only authorized users can see domain specific records
- UI:
    a) **Now Platform UI**: desktops/ Laptops
    b) **Servicenow Mobile App**s (Agent (for requesters) /Now Mobile (Employees) /Servicenow Onboarding (New Hire employees))
    c) **Service Portal:** Meant for self service ESS users to levy servicenow functionalities; user friendly; uses special URL (/sp)
- Access to servicenow elements (tables, records, modules) provided using r**ole based access**
- Servicenow has:
    a) **Users**: people who own an instance; can have no role or multiple roles. Access granted based on the role of the user. User with no role called self-service user. Self service users can login,view dashboard, service catalog, self service portal, public knowledge base articles
    b) **Groups**: groups users with same vision and access levels
    c) **Roles**: specify the strength of the user or the group. Best practice is to assign roles to groups than individual users
- Authentication methods:
    a) LDAP
    b) Local database
    c) OAuth 2
    d) MFA
    e) SSO
    f) Digest Token

Servicenow User Interface Overview

- UI platform : Application navigator / content frame / Banner frame
- Banner Frame:
    a) Logo: customisable
    b) User menu: impersonate/ logout/ elevate role/ profile / user preferences
    c) Settings: General settings/ Theme settings/ Accessibility/ list settings / form setting/ notification/ developer settings
    d) Tools : global search/ connect chat(real time messaging tool) / help
- Application navigator:
    a) Filter
    b) Favorites
    c) History'
    d) All applications(menus> modules)

ServiceNow Branding Overview:

- Branding: refers to customisation of Servicenow instance, so as to represent organizations identity and retain client trust. Involves changing instance logo, font, color scheme
- Guided step up: step by step guidelines for system admin, to customize and configure instance applications/modules
- Accessible by application navigator> **ITSM/ ITOM guided setup**
- ITSM guided setup> IT services like company details, data, task, problem mgmt, change mgmt, incident mgmt, service catalog, knowledge base
- ITOM> IT operations like MID server, event mgmt, cloud provisioning..
- UI development: Service Portal(widget based user friendly) and UI builder(helps build functional UI pages with visual elements like buttons for client interaction)
- Steps to configure instance:
    a) Application navigator> ITSM guided setup> Company> System Configuration
    b) Alter features like logo, header caption, instance time zone, data format, time format, background color, themes..
    c) Application navigator> ITSM guided setup> Company> Welcome page
    d) Text field> change text to change text on welcome page

ServiceNow Lists and Platforms:

- Data in servicenow stored in the format of tables(relational format). Table records showcased to user in format of **LISTS**
- Lists comprise of records, columns and unique column values for each list record
- Access to list> Application navigator> **table name.list / table name.LIST(open in new tab)/ Table name module + ALL**
- Sys_db_object.list if you want to know about all tables in instance
- Context menus options
    A) Group by(can be done by any list column even if not currently on list)
    B) Filters
    C) Views
    D) Show (depicts the number of records per page.. order (show from last or first))
    E) Refresh
    F) Create favorite(unique to your own instance)
- above options alter the list for all users of the instance, unlike personalize list option
- Header of listview also includes:
    a) **New button**> to create new record to a concerned table
    b) **Search option**> to search for a specific field value: select the field name and a desired value; **The operator between them can be '=' indicating the field must have the listed value or '%' indicating the field must contain the value and '*' indicating search anything**
    c) **Activity stream option**> available for tables that track changes or activity (incident, change tables, **available only for task records**)> contains information about all changes made to the table with details like who made the change, timestamps..etc
    d) **List scroll**> move to next or previous list pages
- Other important list feature
    a) **Personalize list**> can edit the list view; allows users to add or remove list columns; **personal to own instance**, changes made on your instance not visible by other users
    b) **Condition builder**> includes 3 major fields: field name, operator, and field value; used to query list records based on condition built; **EX: short description contains issue (here short description is the field,**

contains is the operator and issue is the value : this condition queries only those records that contains the value 'issue in short description field)

c) **Breadcrumbs**> consolidates and lists all filters applied on list in the order of implication; users can copy this condition for building business rules or can rollback to the previous list states by clicking on the bread crumbs
d) Column context menu allows features like editing list layout, creating reports for columns, adding to visual task board(only for task records), import, export
e) **Column label> user friendly names**; clicking on them sorts values of column in ascending or descending order
f) **Field context menu**> filter out a specific value, inline editing, assigning tag, show matching value, copy sys_id, adding to visual task board, copy URL
g) Information icon> quick overview of each record without opening the actual form

ServiceNow Forms:

- User interface to seek necessary information for creation of a record to any servicenow table which user has create access to
- Opening of a form
    a) table_name.form/ table_name.FORM
    b) Click on the number field link of list
    c) New option on list view to create new form
    d) Module of a table menu (create new to open new form)
- Every table has a default list and form
- Form UI entities
    a) Form header
        - Form context menu to access form layout/ form designer
        - Name of table to which form adds record
        - Form view name
        - Form number(unique identifier sys_id of record)
        - submit/ update/ resolve/ delete buttons
    b) **fields**(can be mandatory or optional/ readonly)
    c) Sections
    d) Activity stream
    e) Work notes
    f) Related links
    g) Related lists(showcases dependencies of current record)

- Form field types: string, boolean, number, choice, reference, **List field(similar to reference but allows populating field with multiple list values of the referenced list/table**), **journal fields**(Additional comment(visible to customers/any user), Work notes(visible to only back end servicenow users working on the particular record))
- Saving options
    a) Users must explicitly save their forms
    b) **Save** option from context menu, simply saves current changes and stays on the form to display additional options; the record is not added to database
    c) **Insert** option inserts and saves the record and returns to list of records(similar to submit and update)
    d) **Insert and stay** option saves current record and returns a new record form UI
- Form **sections are logical organizations of relevant form fields**
- Form sections can be altered to be represented as tabs by changing user preference
- Related lists showcases all the lists with which the current table has one to many relationships with ex. **An incident table has one to many relationships with cmdb_ci, sys_user**
- **Formatter** is a special form field that lists the history of activities performed on the record(only for task based records)
- Form view(available in form context menu)> **various versions of the form meant to showcase necessary fields for specific users**(filtered list showcasing only high priority cases for user with VP role)
- The form view which user is currently on is shown in form header(if not default view)
- Personalize form allows users to personalize their own form views by altering the fields shown on the form(does not affect other users of the instance)
- **Form attachmen**t option allows attaching media files to the record
- **Templates** used to auto populate certain field values (toggle template bar to enable or disable)
- Templates can to set to fields regardless their visibility; checkmark against field indicates it was populated by a template
- Field values can be static or dynamic values
- **Template bars** can be used to apply, create, edit templates for current record
- Saving template with same name as table ensures to apply template by default when a form is opened

Hands on Demo

- On list > select multiple record simultaneously to perform group actions
- Knowledge base> categories> articles; guarded by user criterias
- Article functions: rate, mark as helpful, flag articles, leave feedback
- System database > single enterprise wide database; system definition> tables

Importing data to servicenow

- Through integrations to the servicenow instance, users can import data from external sources mainly in the format of **excel, csv, json, LDAP, REST, JDBC**.
- The importing of data is bought by the **System import sets> Load DATA**. Data from the external resource is directly not included in servicenow tables, but rather are initially kept in a **staging area called an import set**. Data in stored in the import set table
- The final objective is to move this data from the import set table into an existing servicenow table also called the target table
- **SOURCE> STAGING> TRANSFORM>TARGET**
- The movement from the import set table to target table is bought using transform maps, that transform staged data into suitable format
- Transformation of data can be automatically done using **auto mapping utility** or by the creation of transform maps(**mapping assist**).

Creating a Data Source in ServiceNow

- Data source table> **sys_data_source.list or System import sets> administration > data sources**
- Data source can be created on servicenow or imported from external files
- To create a data source> sys_data_source.list> new
    a) Add name
    b) Select the import set table for the data source
    c) Select type of source
    d) Source format
    e) Check ZIPPED option for zip files
- If type> jdbc form includes fields for MID server(used to connect servicenow instance server to servers/networks of organization guarded by firewalls), type of database (mySQL, oracle, SQL server), database name, port number, authentication details of database, query to be passed(all rows of a table/ specific SQL), table name/ sql statement, server
- For type> file; list file format(csv, json, xml, custom, excel), file retrieval type(HTTP, attachment) (add file attachment to the record in case of choosing attachment option)

Import Sets:

- While defining data source, we mention an import set table; the instance checks if the import set table mentioned exists, if it doesn't a new table is created
- To import data from a defined data source> click on the **load all records** related link. This initiates the import process and pulls all records from the file attached by creating a new import set table, if the specified table does not exists
- To check the imported data> go to import_set_name.list
- View the import set table columns> those marked with a red cross are custom fields introduced by the data source> the column heading is the document was used to label the fields
- Import set table lists all the imports run on the instance
- **Each entry in the import set table> actually is an import action on the data source, each importing some 'x' records**

Transform maps & field maps:

- Transforms maps help transform data in the staging table to the target table by mapping fields
- **Field map> sys_transform_entry> all field level mappings**
- **Transform maps> collection of field mappings for the entire import > sys_transform_maps**
- Create a new transform map> source: staging table, target table: final destination(existing servicenow table)
- Then create field maps; **If field names in table and data source are the same> then use auto mapping utility**, if **not use mapping assist**(this creates field maps!)
- **Coalesce fields**> **prevents redundancy by opting for a particular field or multiple fields as key**, if the key matches in the target table> record is updated, else new record created
- Enable Coalesce option on a field map to make the field a coalesce field

Incident management and Task Administration

- **Task is a base and parent table extended by incident, problem, change tables, all included in the baseline implementation of the instance**
- When creating records> records added to child tables and not the task table directly
- Can be bound with approvals, assignment, workflows etc
- **Task assignment> assigned to(user)/ assignment group(group users); each task can be assigned to one user/one group**
- Assignment> ALL>System Policy>Rules>Assignment
- table> sysrule_assignment
- Assignment rule creation> name it, assigned table conditions> assigned to(user/group)
- **Assignment rules applicable to only Task records**
- Collaboration on task records> shown with user presence
- **User presence allows multiple stakeholders to view/ update records simultaneously**
- A pulse icon is set against the field edited by another real time user
- Activity stream entries created by system automatically when a change is made to the task record
- **Visual task boards: visual to do list for a team or group; includes cards(representing tasks) and lanes(grouping parameters to group tasks)**
- Card includes task short description; lane includes name of parameter it considers for grouping and the number of records under the lane
- Cards can be dragged and dropped to different lanes to change their values
- Tools> manage filters, users, labels and configuration options
- Quick panel allows the quick filtering of VTB by user, group of users or labels
- Hands on >
    a) Self service> visual task board
    b) Assign users to tasks by dragging and dropping username pills in the quick panel
    c) Assign priority labels to tasks(labels and tags can be customized)
    d) Click on the user pill to select and filter only tasks assigned to specific user
    e) Info button shows detailed info about the VTB, provides link to share VTB to other users
    f) User icon helps manage users of the VTB; activity steam lists all changes made to the VTB; Configuration icon allows to make configuration changes to the VTB

- VTB forms
    a) **Guided**: created from list column context menu, where the distinct column values are made as VTB lanes; list has a predefined set of values as lanes, changing cards between lanes(ex. State attribute has a predefined list of values that form lanes of VTB), changes their values
    b) **Flexible**: list does not contain predefined set of lanes; task values cannot be changed by moving cards between lanes(lanes will be by default set as ToDo, Doing and Done> names can be altered)
    c) **Freeform**: not created from a list, used for personalized work mgmt

Reporting Fundamentals

- Reports are stored as records of the **sys_report table**
- Sources of reports are stored in the **sys_report_source**, used to store **saved queries for retrieving data from source table and populating a report**
- Reports once created can be
    a) Shared to users/groups: **sysy_report_users_groups**
    b) Put to dashboards: **pa_dashboards**
    c) Scheduled Email of report: **sysauto_report**
- Report table extends Application File
- Report fields:
    a) Sys ID> string: unique identifier for records of list
    b) Title> string: report title
    c) Source type> string(table/data source)
    d) Table> reference: tables of servicenow
    e) Field name> string: field upon which report is created
    f) Filter> condition: filter to be applied on the report data
    g) Type> string: type of report
- Report creation steps:
    a) Reports > create now
    b) **Data section**> report name, source type(data source/table), table
    c) **Type section**> select the report type
    d) **Configure section**> select report field, group by attributes, aggregations
    e) **Style section**> style the report with themes, chart size, precision
- Report creation> application navigator/ studio/ list
- Scheduled run and email fields:
    a) Sys_id
    b) Report to be run recurring and mailed to user
    c) Users to whom report is sent via email

    d) Groups to whom report is to be sent
    e) Email addresses to which report is to be sent
    f) Run: frequency of scheduled report execution
    g) Time at which report is to run
    h) Subject of mail
    i) Introductory message of mail in HTML format
    j) Condition for triggering scheduled execution and mailing
    k) Type of attachment
- reports> **Scheduled report; reports/view and run; or studio> scheduled reports**
- Report views:
    a) My reports: created by currently logged in user
    b) Group reports: created by the group user belong to
    c) Global: reports accessible by all users as it is in global scope
    d) All reports
- Report sharing options
    a) Share to user/ group/ globally
    b) Add to dashboard
    c) Publish
    d) Schedule
    e) Export as PDF

Low code No code development:

- Traditional software development→ business person/ stakeholder writes his requirements for implementing a business solution
- Requirements sent to an IT professional to develop the solution and implement the same
- There can be disagreements between the IT professional and the stakeholder in the solution's implementation leading to an endless cycle of changes and feedbacks thus increasing time for development
- Traditional software development replaced with agile software development, paired programming where stakeholder always considered during development process, but still not satisfactory
- Then came, low code no code; format of development that allows business personnel to build the solution on their own without dependency on IT professionals.
- This approach prevent hard core coding thus allowing stakeholders with less technical knowledge to build solutions with easy drag and drop type interfaces
- Servicenow low code no code tools:
    a) **App engine studio:** delegate development

    b) **Guided application creator**: step by step intuitive tour for initial custom app setup
    c) **UI builder**: for developing User interfaces via drag and drop
    d) **Flow designer**: help in building automatic workflows
- Pros of low code no code:
    a) Empowers business personnel to develop applications, reducing dependency on IT
    b) Agility
    c) Lower costs and faster development
    d) Increased automation
- Cons:
    a) Reduced flexibility
    b) Lesser technical implementation(due to low code)