Communication Framework:

1. UI 🡪 Workflow
   1. UI to call an API which will invoke the Workflow
      1. UI to Pass forward the following parameters
         1. Service Name
         2. Service Options 🡪 Single Server, Cluster, Failover Cluster, Master-Slave etc etc.
         3. Service Map between the instances being provisioned.
         4. Service Instance Properties
            1. Instance\_Hostname(s), Availability Zones, Setup, Network details, Instance OS, Service App\_Version
            2. Service Config 🡪

User Defined Parameters,

Service Binding Port,

Service User Credentials, Username, Password

Other relevant Parameters.

Best Practice / Formula Based parameters ( Fixed by the system)

Other Default Parameters.

* + - 1. Service Instance Config.
      2. Service Term
  1. Common API to Parse the Parameters sent by UI as mentioned above and forward the same to the Workflow as a XML.

1. Workflow Initialization
   1. Parse the Input for Infrastructure details to initiate Provisioning of the required Infrastructure.
   2. Create a Map of the Roles being provisioned.
   3. Create an Object Structure to be passed onto the API’s embedding the corresponding Rundeck jobs.
   4. Define the Response expected from the RunDeck Jobs to be passed on by the called API’s.
   5. Call the Provisioning requests per device.
   6. Keep a track of the received Object\_Id as a response from the Provisioning requests by the corresponding Rundeck Jobs.
   7. Based on the instances getting provisioned, get the Object\_id and the Instance role mapping verified.
   8. Install the required software during the provisioning phase or later post provisioning. Today this is done along with the provisioning request, however in the future would like to see this being controlled via the Rundeck jobs, rather than being done from the Provisioning request itself. This would be helpful to have a Rundeck Job which will install the Software on the fly on any infrastructure as an independent instruction.
   9. Identify the TCP Finetuned Parameters per Service. Define a XML / Define them in a DB for every service so that they can be changed on the fly if required anytime.
   10. Then define the API specs & the corresponding Rundeck Job, for calling the Basic Config function per Instance provisioned. Include the above mentioned TCP Finetuned Parameters (OS) also as a part of the Basic Configuration.
   11. Execute the Basic Config API which will call the underlying RunDeck Job.
   12. Define the API Specs and parameters list for the Group / Cluster Config for the service.
   13. Post Installation and config of all the Infrastructure, Please call the Group Config to configure the Service.
   14. Define the Basic test Use cases for:
       1. Functional Checks.
       2. Failover checks.
   15. Execute “tests” on the Overall Cluster for its proper functioning.
   16. Send the Commencement email for the Setup which should contain:
       1. Service Specific Infrastructure Provisioned.
          1. Infrastructure Config & Role with Availability Zone, Setup & Network details.
          2. Access Credentials Per Infrastructure provisioned.

Note: No credentials to be sent for Std Appliances being provisioned like Firewall, Load balancer etc.

* + - 1. Service – Administrative User credentials.
      2. Any other specific Information which is important, that has to go as a part of the Email body.
      3. Send Attachment-1 which gives the Service Parameters Configured.
         1. These need to be grouped in the same Format as mentioned on the UI side. (Fixed etc etc ).
      4. If tests are performed, then kindly send the test reports for all the tests performed either for Functional tests / Failover tests or both.

Note: Critical Points to remember as a part of the design:

1. Core Workflow structure not to change
2. Add more steps or remove the individual functional steps which may not be relevant for that particular Service.
3. Or Modify the underlying procedures to suit to the needs of the Service.
4. Keep the core Concepts and constructs of CREATE, BASIC\_CONFIG, CLUSTER CONFIG etc similar.
5. Keep a single API Structure for the frequently / multiple times used API’s, which will adapt itself to the new requirements. What is meant here is the Structure should not change, however the individual line items like Parameters can change.

Guidelines to Design a New Service:

1. Define a Service Name. 🡪Should have the App / Platform-name as a set of Characters in it for people to clearly identify.
2. Define the App Platform Version. 🡪 Should contain the Major+Minor Revision number.
3. Define the Parameters List per Service in advance and Categorize the same in various categories as defined above in the UI 🡪 Workflow section.
4. Define the Various API’s and their corresponding Rundeck Jobs.
5. Define the Parameters List to be passed on to each of the API’s and their Rundeck Jobs.
6. No Service Config parameter to be hard-coded anywhere. Everything should come as a part of the Inputs coming from the Ui / API invoking the Workflow. Reason being, we need to provide the user with the functionality to custom define all the parameters and override the same as per their requirement.
7. User should be able to define their parameters list and their required default settings for each of these parameters. Also to note is the fact that the user should also be able to keep different templates for individual environments etc as and when required. Idea here is to define our Std parameters which would be the default used by the customer, However if he needs he will override them if he wishes to.
8. Ability to define the Service Parameters in YAML so that these parameters can be easily imported as per the customer requirement. This would also help the Customer Customize the Service Parameters List and also create multiple profiles.

The Customer should be able to Create a Profile by any of the following means:

* 1. Import from the Default Service Config as defined by Netmagic Best Practices.
  2. Create a YAML File and import it as a Profile Online.
  3. Inherit an existing Custom Profile and change the required Parameters and store it as a new Profile.
  4. Edit an Existing Service Config Profile for desired Changes.