RM Automation Content

	Item	Assigned to	Due by Date	Status
1	Create PR for TOC and Structure for 9.5 in RM Ch09	Pankaj Goyal	2020-12-14 1400UTC	Done
2	Approvals of #1	Walter Kozlowski Tomas Fredberg Karine Sevilla	2020-12-14 1500 UTC	Done
3	Mege/Squash #1	Walter Kozlowski	2020-12-14 1500 UTC	Done
4	9.5.2.1 PR	Walter Kozlowski	2020-12-15	Done
5	9.5.2.2 PR	Pankaj Goyal	2020-12-15	Done
6	9.5.1.1 and 9.5.1.2 Requirements	Tomas Fredberg		Tomas Fredberg to introduce the concepts and then team can work on requirements
7	9.5.1.3 Content + Requirements	Saad Ullah Sheikh		
8	CI/CD pipeline design and job integration	Cedric Ollivier		PR #2179 created ans should be reviewed

README

- This TOC page has been created with Ahmed El Sawaf Karine Sevilla Petar Torre Saad Sheikh Tomas Fredberg Walter Kozlowski and Pankaj Goyal as editors/owners

Action Items

++

9.5 Automation

9.5.1 Cloud Infrastructure LCM Automation (including Platform Software and CI/CD)

End-2-End understanding of what type of automation ought to be done on what layers and includes

9.5.1.1. hardware configuration CI/CD

- Example, BIOS settings, Reset, Power Modes,
- HW component discovery and status/health supervision

- Composability of physical hardware resources (components)
- Hardware Accelerator discovery, loading and assignment
- Firmware Update and supervision (measurement) of Signed FW on known Authentic HW
- Security related certificates, keys and roles

9.5.1.2. networking automation

- Physical network cabling detection and supervision
- L2, L3 and QoS Automation of Server NIC Network Assignment and Switch Fabric Network Assignment based on HW Provisioning in the HW Infrastructure Layer
- Partitioning, provisioning, enforcement of Overlay switching shared resources e.g. VLAN and VxLAN

9.5.1.3. software development CI/CD for infrastructure (not workloads) – Saad

what are the components of "infrastructure software": provisioning and configuration of the infrastructure (servers, network and storage; operations and management software)

Telco DevOps is still not mature within context of Telco infrastructure and service primarily due to reason that software in Telecom is primarily provided by vendors with operators almost no clear visibility about the software constructs itself , it means only Infrastructure and Networking Automation can not automate complete service .

End to End specification of how to deliver automation in a hybrid Infrastructure is required with following features

- CI (Continuous integration) for Telco service

    a) When vendor will update new feature or new code files (Packages , Helm charts etc) how it will be merged with existing software

    b) Track and fix bugs of incremental software running on Infrastructure

- CD (Continuous Delivery) for Telco service

    a) Testing of new code base e.g Test Bench , vSPerf etc

    b) Automation of Code base with Infrastructure e.g Python , GNPy , Heat etc

    c) Produce the software artifacts which will be deployed on Infrastructure , example is Sol4 packages , Sol1 templates

    d) Testing of packages including Infrastructure DevOps

    e) Output and benchmark the packages in staging Environment , key example here is Telco PaaS capability

- CD (Continuous Deployment) for Telco service

    a) Software onboarding in the Infrastructure it involves all deployments in both CNTT RM infrastructure and related VNF/CNF codes /configurations including simulation and Test execution environment

    b) CI/CD pipeline , when a new VNF/CNF code need onboard or just a simple capacity expansion is required , It is not just about Infrastructure and Networking Automation but E2E automation including VNF/CNF part

    c) Here the Key requirement is once a new Release is output all the process from build/test/validate and onboard is done automatically

    d) Handover points between Infrastructure and Orchestration need to be validated and verifiable in the form of SLA/KPI's

I think if all team agree then we can output DevOps Reference Architecture after team consensus covering Infra

Lessons Learnt from ETSI NOC and CNCF conformance:

Following two domains we must catch for E2E definition and service

    CI explaining pipeline view between Lab , Staging and Production

    CT , the testing of service

    Pipeline architecture e.g Jira , Jenkins and Necessary SDK's in Orchestration through which we can automate E2E including Infrastructure

CI/CD requirements

Ref	Description	Comments /Notes
auto.devops.cicd.001	The CI/CD pipeline must be model driven i.e characterised by service intent without mapping each component manually	
auto.devops.cicd.002	The CI/CD pipeline must be declarative and not imperative	
auto.devops.cicd.003	Must support necessary SDK's needed for complete E2E automation and service validation	

~~**9.5.1.4. software deployment CI/CD (operator environment)**~~ **– covered in 9.5.2.2 (inserted 2020-12-07)**

~~**9.5.1.5.**~~ **Identify "Closed Loop Automation" as a Gap in Ch10 – Done (PR #2123 -need reviews)**

9.5.2 Software onboarding automation (including CI/CD). Owner: Walter Kozlowski

9.5.2.1. Software onboarding automation - the scope for RM is to describe support only but leave the details to RA/RI; Walter Kozlowski to open PR

The Cloud Infrastructure workload onboarding process describes activities needed for the integration of tenants' workloads into the Cloud Infrastructure environment. Typically, this business process consists of the following key phases:

1. Tenant Engagement and ~~Workload~~ Evaluation:
   a. In this phase the request from the tenant to host a workload on the Cloud Infrastructure platform is assessed and a decision made on whether to proceed with the hosting request.
   b. This phase may also involve the tenant accessing a pre-staging environment to perform their own evaluation and/or pre-staging activities in preparation for later onboarding phases.
2. Workload Packaging:
   a. The main outcome of this phase is to produce the workload deployable image and the deployment manifests (such as TOSCA blueprints or HEAT templates or Helm charts) that will define the Cloud Infrastructure service attributes for the workload.
   b. The workload packaging can be performed by the tenant, through self-service capabilities or by the Cloud Infrastructure Operations team.
3. Workload Validation and Certification:
   a. In this phase the workload is deployed and tested to validate it against the service design and other Operator specific acceptance criteria, as required.
   b. Workload validation and certification should be automated using CI/CD toolsets / pipelines and Test as a Service (TaaS) capabilities.
4. Publish Workload:
   a. After the workload is certified the final onboarding process phase is for it to be published to the Cloud Infrastructure production catalogue from where it can be instantiated on the Cloud Infrastructure platform by the tenant.

All phases described above can be automated using technology specific toolsets and procedures. Hence, details of such automation are left for the technology specific Reference Architecture and Reference Implementation specifications.
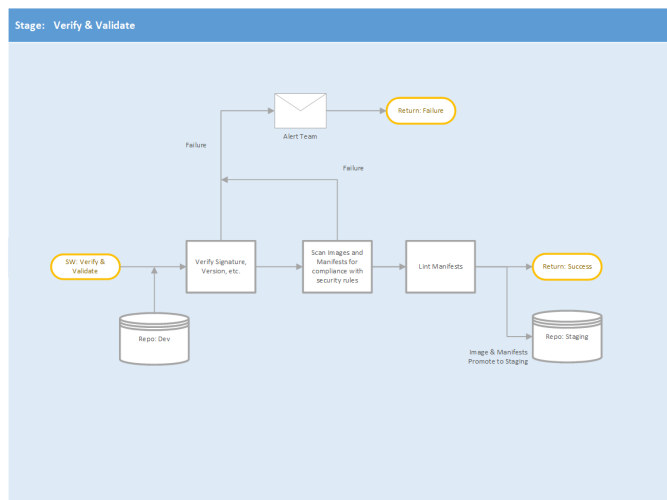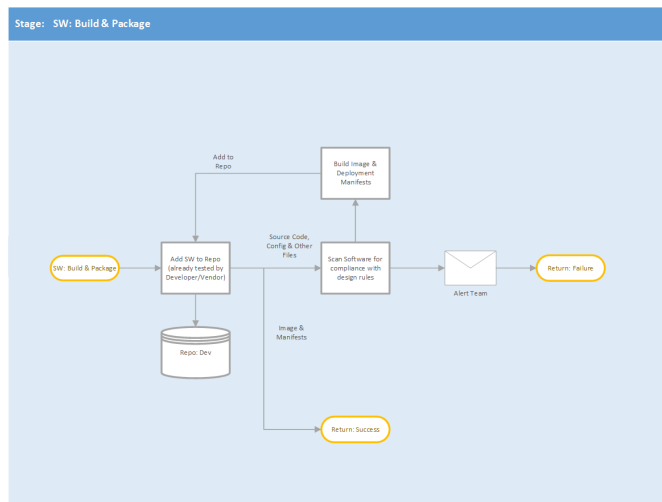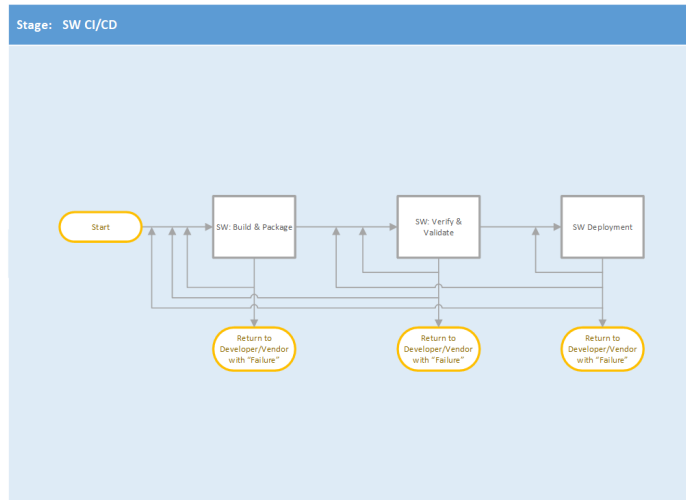
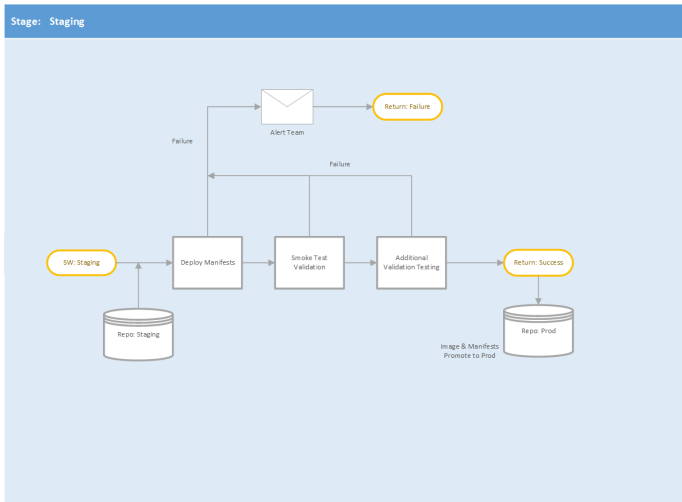### 9.5.2.2. Software CI/CD Requirements Pankaj Goyal to open PR

The requirements including for CI/CD for ensuring software security scans, image integrity checks, OS version checks, etc. prior to deployment, are listed in the Table XX.XX (below). Please note that the tenant processes for application LCM (such as updates) are out of scope. For the purpose of these requirements, CI includes Continuous Delivery, and CD refers to Continuous Deployment.

Ref	Description	Comments/Notes
auto.cicd.001	The CI/CD pipeline **must** support deployment on any cloud and cloud infrastructures including different hardware accelerators.	CI/CD pipelines automate CI/CD best practices into repeatable workflows for integrating code and configurations into builds, testing builds including validation against design and operator specific criteria, and delivery of the product onto a runtime environment. Example of an open-source cloud native CI/CD framework is the Tekton project (https://tekton.dev/)
auto.cicd.002	The CI/CD pipelines must use event-driven task automation	
auto.cicd.003	The CI/CD pipelines should avoid scheduling tasks	
auto.cicd.004	The CI/CD pipeline is triggered by a new or updated software release being loaded into a repository	The software release cane be source code files, configuration files, images, manifests Operators may support a single or multiple repositories and may, thus, specify which repository is to be used for these release. An example, of an open source repository is the CNCF Harbor (https://goharbor.io/)
auto.cicd.005	The CI pipeline must scan source code and manifests to validate for compliance with design and coding best practices.	
auto.cicd.006	The CI pipeline must support build and packaging of images and deployment manifests from source code and configuration files.	
auto.cicd.007	The CI pipeline must scan images and manifests to validate for compliance with security requirements.	Refer to RM Chapter 07 (https://github.com/cntt-n/CNTT/blob/master/doc/ref_model/chapters/chapter07.md#79-consolidated-security-requirements) Examples of such security requirements include only ingesting images, source code, configuration files, etc. only form trusted sources.
auto.cicd.008	The CI pipeline must validate images and manifests	Example, different tests
auto.cicd.009	The CI pipeline must validate with all hardware offload permutations and without hardware offload	
auto.cicd.010	The CI pipeline must promote validated images and manifests to be deployable.	Example, promote from a development repository to a production repository
auto.cicd.011	The CD pipeline must verify and validate the tenant request	Example, RBAC, request is within quota limits, affinity/anti-affinity,
auto.cicd.012	The CD pipeline after all validations must turn over control to orchestration of the software	
auto.cicd.013	The CD pipeline must be able to deploy into Development, Test and Production environments	
auto.cicd.014	The CD pipeline must be able to automatically promote software from Development to Test and Production environments	

Diagrams

Stage: SW CI/CD

```
Start → SW: Build & Package → SW: Verify & Validate → SW Deployment

Return to                  Return to                  Return to
Developer/Vendor           Developer/Vendor           Developer/Vendor
with "Failure"             with "Failure"             with "Failure"
```

Stage: SW: Build & Package

```
                              Add to        Build Image &
                              Repo          Deployment
                                            Manifests

SW: Build & Package →  Add SW to Repo  →  Scan Software for  →  Alert Team  →  Return: Failure
                       (already tested by    compliance with
                       Developer/Vendor)     design rules
                              Source Code,
                              Config & Other
                              Files

                          Repo: Dev       Image &
                                          Manifests

                                       Return: Success
```

Stage: Verify & Validate

```
                                    Alert Team  →  Return: Failure

                        Failure              Failure

SW: Verify &  →  Verify Signature,  →  Scan Images and  →  Lint Manifests  →  Return: Success
Validate         Version, etc.          Manifests for
                                        compliance with
                                        security rules

              Repo: Dev                                            Repo: Staging

                                          Image & Manifests
                                          Promote to Staging
```

9.5.3 Tenant creation automation

9.5.3.1. requirements for enterprise processes prior to tenant creation on the platform

Here is a starting set – Maybe too low level; requirements can be created once we agree on the correct set:

- Validate that the Capacity can satisfy the tenant requested quota for vCPU, RAM, Disk, Network Bandwidth
- Validate that the Cloud Infrastructure can meet tenant's performance requirements (e.g. I/O, latency, jitter, etc)
- Validate that the Cloud Infrastructure can meet tenant's resilience requirements
- ~~Validate any requested private flavours~~
- For VM-based environments:
    - Verify that any requested private flavours have been created
    - Verify that the metadata for these private flavours have been created
    - Verify that the tenant has permissions to use the requested private flavours
    - Validate that host aggregates are available for specified flavors (public and private)
    - Verify that the metadata matches for the requested new flavours and host aggregates
- Verify that the networks requested by the tenant exist
- Verify the metadata: 1. Keypairs must be higher than default 2. Networks must be higher than default
- Add all Tenant Members and configure their assigned roles in the Enterprise Identity and Access management system (e.g., LDAP)
- Create Tenant
- Using a proto- or Tenant provided HEAT-template/Helm-chart for a NF and perform sanity test (e.g., using scripts test creation of VM/container, ping test, etc.)
- Verify and Validate Tenant Images: virus scan, correct OS version and patch, etc.

9.5.3.2. tenant networking automation

++++++++++++++++++++++++++++++++++++

Cedric to develop content on CNTT RI and RC toolchains

++++++++++++++++++++++++++++++++++++