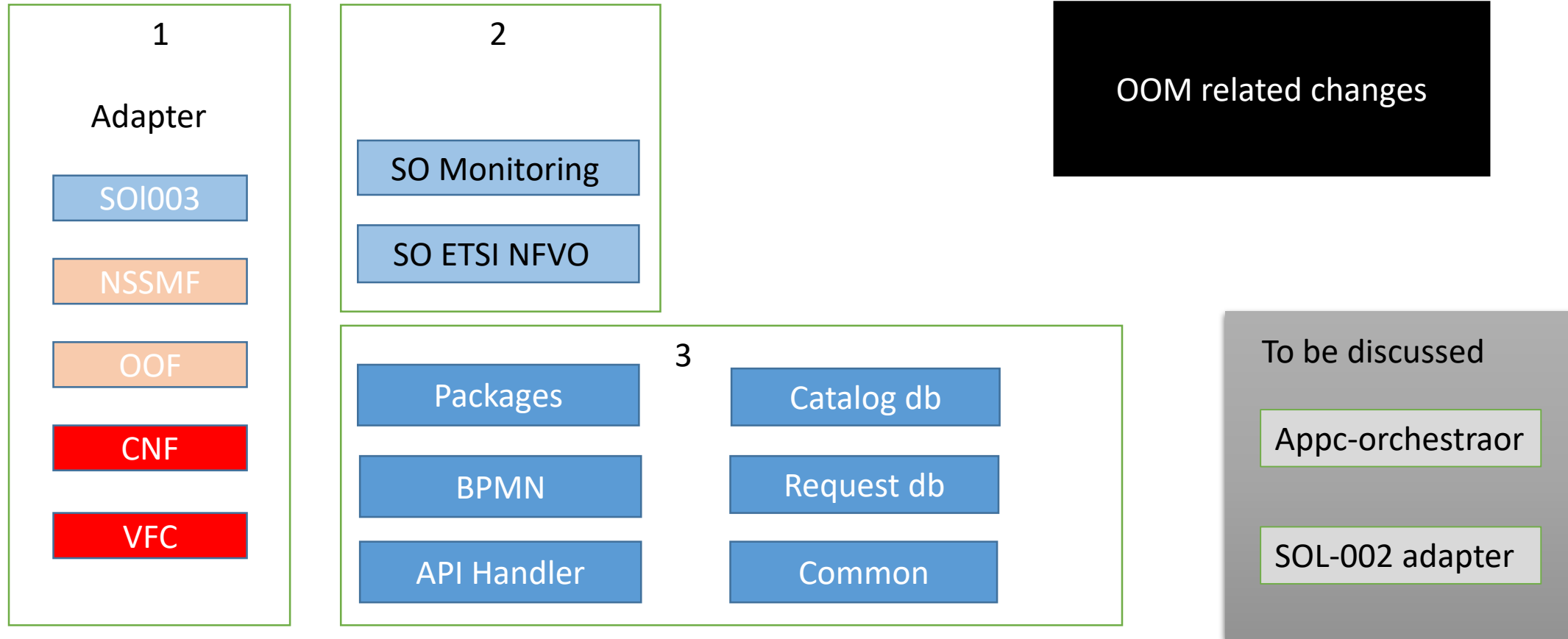




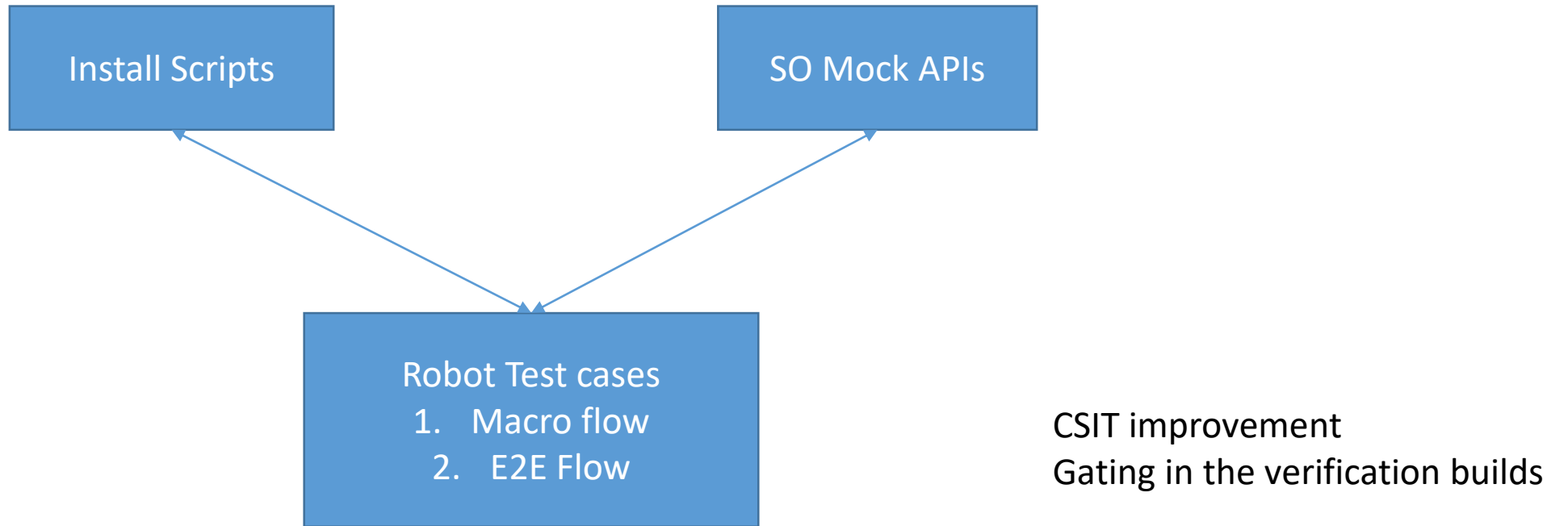
ONAP SO Honolulu Plan – High Priority Items

January 7th, 2021

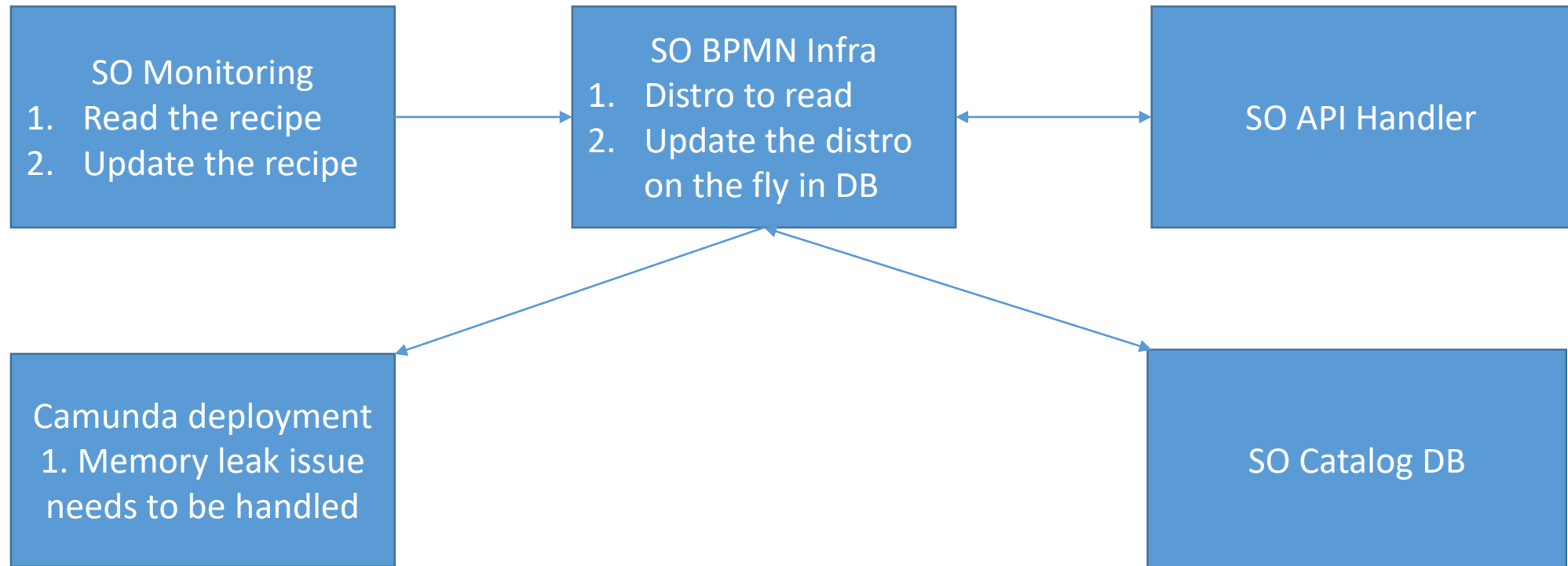
Refactoring – Make SO smaller to deploy



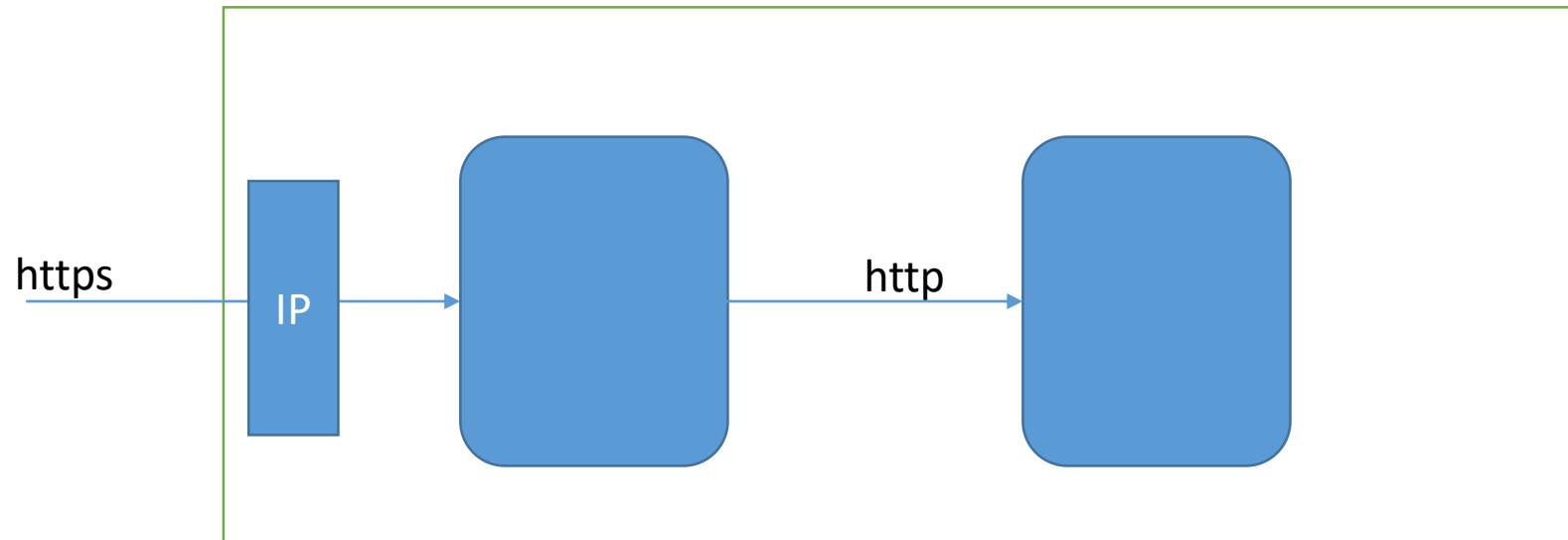
E2E Integration Test



Dynamic Orchestration



Communication Medium– No change



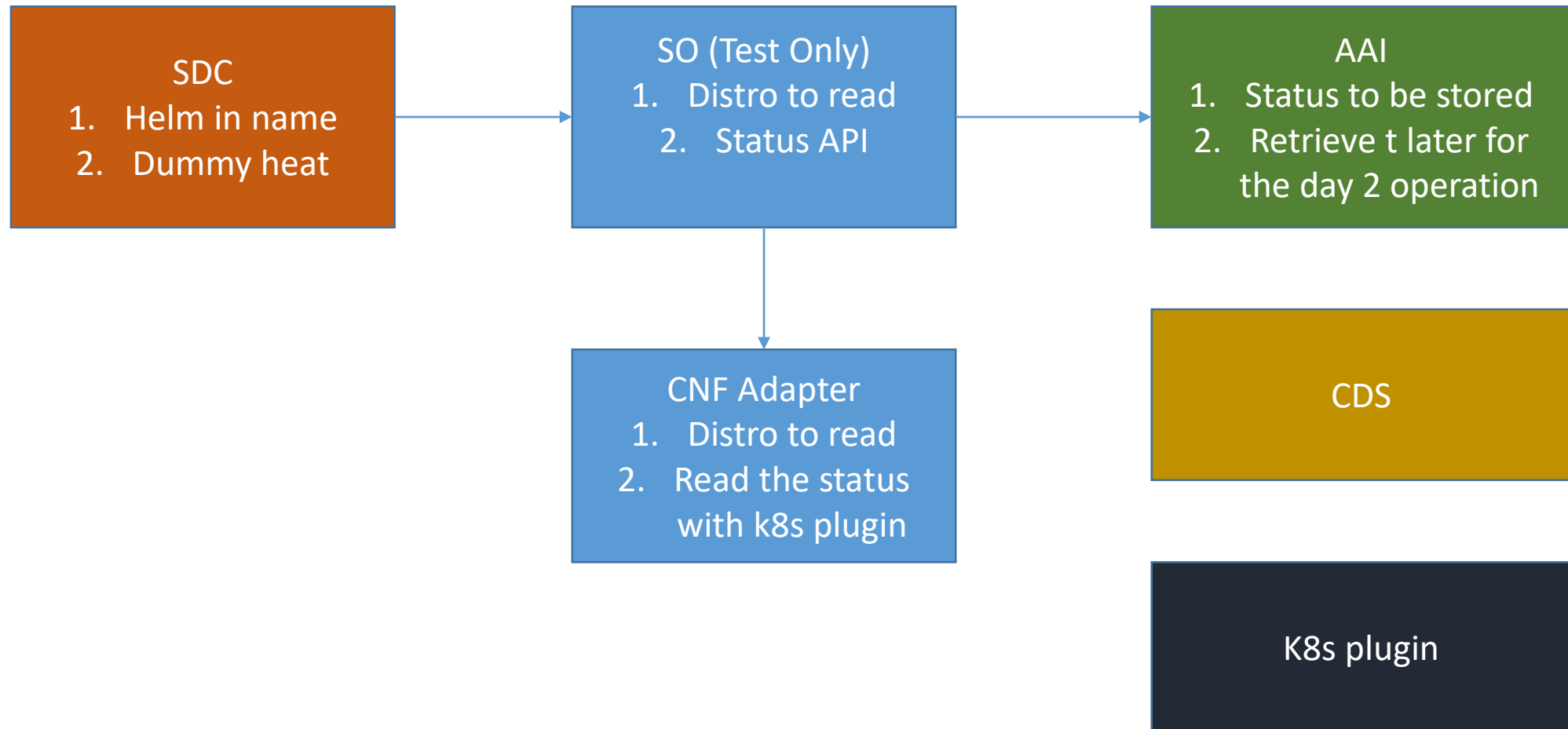
Ingress proxy service for all the https communication to current http ports

Guilin - Maintenance

Will be defects on top of Guilin Maintenance release.

- Dec – 1.7.11
 - 24th Dec
- Feb – 1.7.12
- March - 1.7.13

Cnfo requirement



ONAP SO Honolulu Plan – High Priority item #1

SO Modules Refactoring

– Make SO sub-modules independent modules from the building, packaging and deployment perspective; separate their projects, repos, Jenkins, releases, yet under the SO group umbrella

Priority 1: Adapter Module Group

- SOL003 Adapter
- NSSMF
- OOF
- CNF
- VFC

Priority 2: SO UI & ETSI Group

- SO Monitoring
- SO ETSI NFVO

Priority 3: Base Group

- Packages
- BPMN
- SO-Optimization-Clients
- Common

OOM related changes

Note: remove obsolete sub-components:

- Appc-orchestrator
- SOL 002 Adapter

Problem Statements:

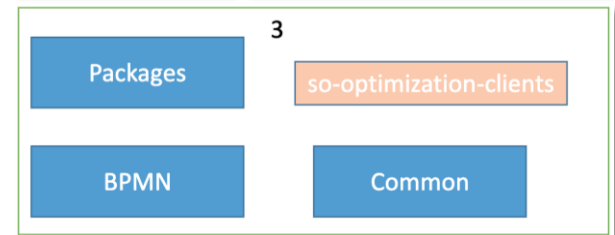
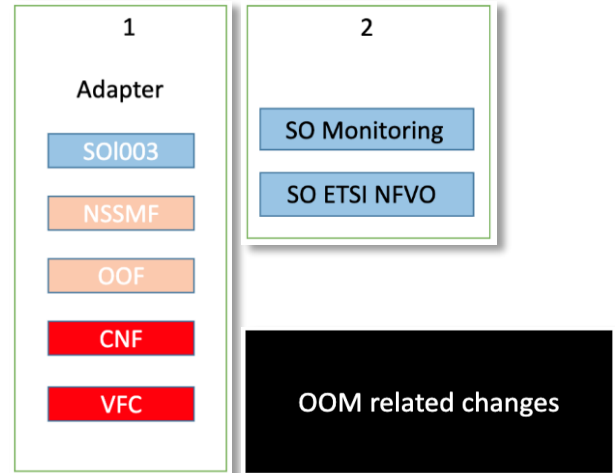
- SO itself and sub-components together tend to be monolithic and too large, without leveraging Microservice Architecture fully – its build takes a long time
- One sub-module build failure could cause the entire SO build to fail
 - This caused SO build crisis during Guilin
- SO needs be a platform, which supports additional sub-component Microservice plug-ins and flexible sub-function aggregation

Pros:

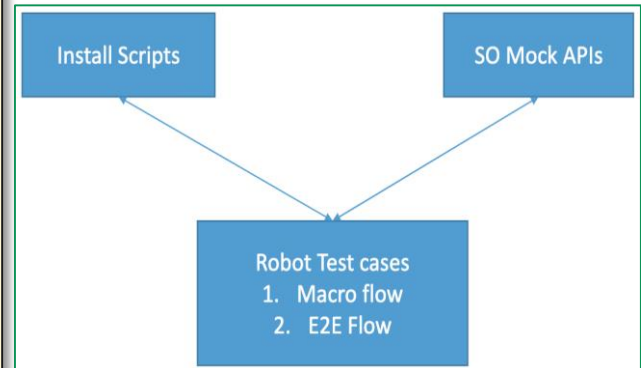
- Each SO sub-component has its own project, repo, build script, deployment and release lifecycle
- A module build does not impact other SO sub-components, and build time is quicker
- Isolating a module failure from other sub-components, and scaling independently
- True Microservice architecture approaches

Challenges:

- Inherit Microservice complications (interfaces, service discovery)
- Need to design module inter-connections and dependencies well
- Common/shared function Microservices need to be carefully organized to make each sub-component have all necessary items
- OOM Impact needs to be analyzed carefully
- Sub-component CSIT for E2E integration testing is a must
- ONAP SO feature enhancements (e.g., ETSI-Alignment) could be deprioritized for resource issues



E2E Integration Test



ONAP SO Dynamic Orchestration – high Priority #2

ONAP SO Dynamic Orchestration Capability without bouncing ONAP SO Supports

- Hot deployment of BPMN Workflows and business logic (both base and custom)
- Decoupling ONAP SO code from custom Workflows and business logic.
 - Camunda engine is not coupled with ONAP SO code
- Allowing vendors to build their custom BPMN Workflows and business logic as WAR file(s)
- Allowing operators to deploy custom Workflow package(s) to ONAP SO (stand-alone/clustered Camunda) at anytime
- ONAP SO will be able to manage custom orchestration without interruption
 - It is a must for production-quality orchestration
- Current SO Monitoring UI scope will be extended to manage Workflow package onboarding
 - “SO Monitoring” name will be changed to include the onboarding

– Problem Statements:

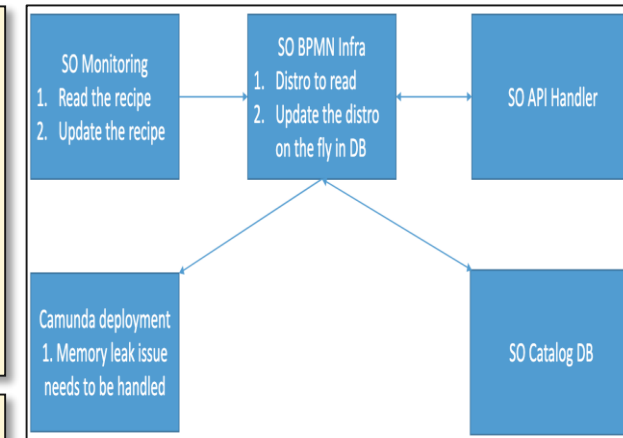
- To incorporate custom BPMN workflows and business logic into ONAP SO, the custom code needs to be part of ONAP SO code
 - It is a maintenance nightmare
 - Customized ONAP SO ends up becoming non-generic
- It requires rebuilding ONAP SO itself, stopping ONAP SO, redeploying a new ONAP SO and running ONAP SO
 - This does not meet production-quality SLA requirements

– Pros:

- Will enable vendors to build their custom BPMN workflow packages independently
- Will allow operators to deploy vendor BPMN workflow packages anytime without bouncing ONAP SO
- ONAP SO code will be generic, and custom BPMN workflow packages will be plugged in on top of the ONAP SO base

– Challenges:

- Decouple BPMN workflows and business logic from ONAP SO code
 - The current BPMN Infra will be packaged separately and deployed as the default workflow package
- Extend SO Monitoring UI and ONAP SO to manage custom package onboarding and SO Catalog DB sync-up
- SO needs to support standalone / clustered Camunda engine, instead of using embedded Camunda engine
 - Load-balancing plan needs to be addressed.
 - Get the latest Camunda engine without memory leak



Ericsson will participate in its reviews and discussions, but not code contribution

ONAP SO Component Security – High Priority #3

ONAP SO and its sub-components will be protected by Ingress Proxy service

- ONAP SO Ingress Proxy service protects ONAP SO by supporting secure communication via HTTPS and transforming its southbound communication protocol from HTTPS to HTTP for ONAP SO sub-components
- No direct incoming communication exposure from ONAP SO sub-components
- Communications among ONAP SO sub-components will be treated as internal ones thru HTTP

– Problem Statements:

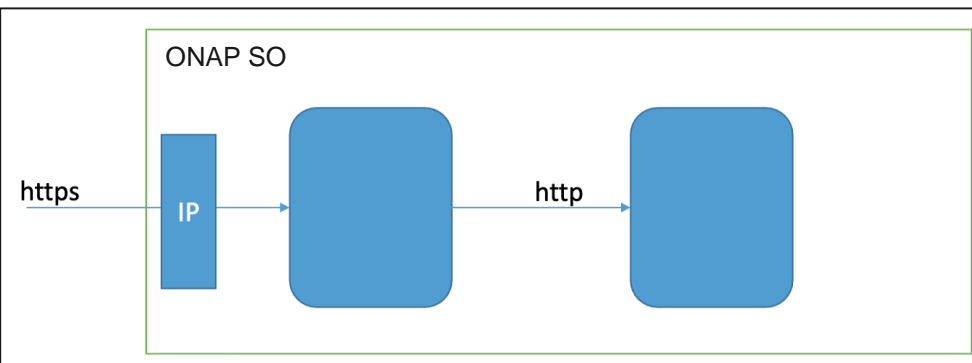
- Each ONAP SO sub-component needs to handle its own secure communications; in many cases, without a uniform way
- It has been a significant burden to sub-component builders to conform to the security standards / best practices

– Pros:

- ONAP SO Ingress Proxy service protects the entire ONAP SO including its sub-components
- The sub-component builders will be able to focus on their business logic
- Uniform way of security standards / best practices will be applied centrally

– Challenges:

- Define and Configure Ingress Proxy services for ONAP SO
- Sub-component outgoing communication security needs to be addressed



Ericsson will participate in its reviews and discussions, but not code contribution

ONAP SO Honolulu Plan – High Priority item #4

E2E Integration Test need to cover the entire ONAP SO operations

- Support E2E Integration Test enhancements for ONAP SO
- Validate ONAP SO sub-components

– Problem Statements:

- Currently, ONAP SO E2E Integration Test is not comprehensive.
- Some of E2E test flows are broken and need to be enhanced
- There is not comprehensive component validation when additional sub-components / functions are added to ONAP SO

– Pros:

- ONAP SO will become more modular, and we need to validate SO changes beforehand
- SO itself and its sub-component CSIT will help

– Challenges:

- Revise / enhance Robot Test and add each sub-component CSIT
- Set testing best practices for adding / enhancing sub-components

Ericsson will provide CSIT contribution for SO NFVO and SOL003 Adapter; so those CSITs can be used for E2E Integration testing

E2E Integration Test

