ONAP DDF, 11-14 June 2019
Modularity

Stephen Terrill
• Aligned working assumption on terminology:
  - **Module**: Implements a business capability accessed through a defined set of APIs
    • E.g. A DCAE Data Collector microservice, A&AI data repository
  - **Component**: A collection of modules that are related in some form
    • E.g. SO, Controllers, A&AI, etc
  - **ONAP**: A collection of ONAP Components
  - **Microservice**: Small, single-capability focused, standalone services
    • E.g. IP address assignment, Tosca parser
  - **Cloud-Native**: Container-packaged, dynamically managed, microservices-oriented applications
    • E.g. Containerized microservices managed by Kubernetes
  - **Service Mesh**: Connective tissue between microservices
    • E.g. traffic control, resiliency, security, observability
    • Control plane (Istio, linkerd) and Data plane (Envoy, linkerd)
    • Note: This is different from service chaining

• Aligned working assumption on approach
  - Evolutionary approach
  - One component at a time
  - Start with SO and Controllers
- SO Decomposition working assumption
  - API handler
  - Request DB
  - BPMN Infra
  - SDC controller
  - Catalog Adapter
  - Adapters for the controllers (SDNC/VFC/…)
  - Cloud Adapter
- Controller decomposition working assumption:
  - Extract IP assignment from the controllers as a common microservice
  - Extract Tosca Parser from SO and make a common microservice
- Feedback welcome to mature to working assumption for Dublin.
  - Will discuss in Project meetings
  - Will share with PTLs in PTL meeting
• Shared DBMS
  - Having the database as a dependency of the component (not in the component).
  - Some work ongoing for this.
    • Driven by footprint reduction and centralized DBMS activities
  - There is a comment that characteristics requirements must be considered.
  - There was a comment that the deployment scenarios may influence the data layer.
  - There was a comment that how to externalize access to data should be plugable.
  - There could be the recommendation to have a list of "best in class" technologies. It was counted that we should instead explain the "requirements" instead.
  - Casablanca has 27 database instances, 10 different technologies.
  - Recommendation:
    • Capture the current state.
    • Provide recommendations on how to use Data Access based on current best practices.
    • Create architectural guidelines for publication. Take to the TSC
Modularity Brainstorm

• Further Internal component modularization

• Replacability
  - Principles around APIs, contract testing etc.
  - Granularity of the contracts.
  - ...... Take as a discussion point in ArchCom