Service Orchestrator Enhancements

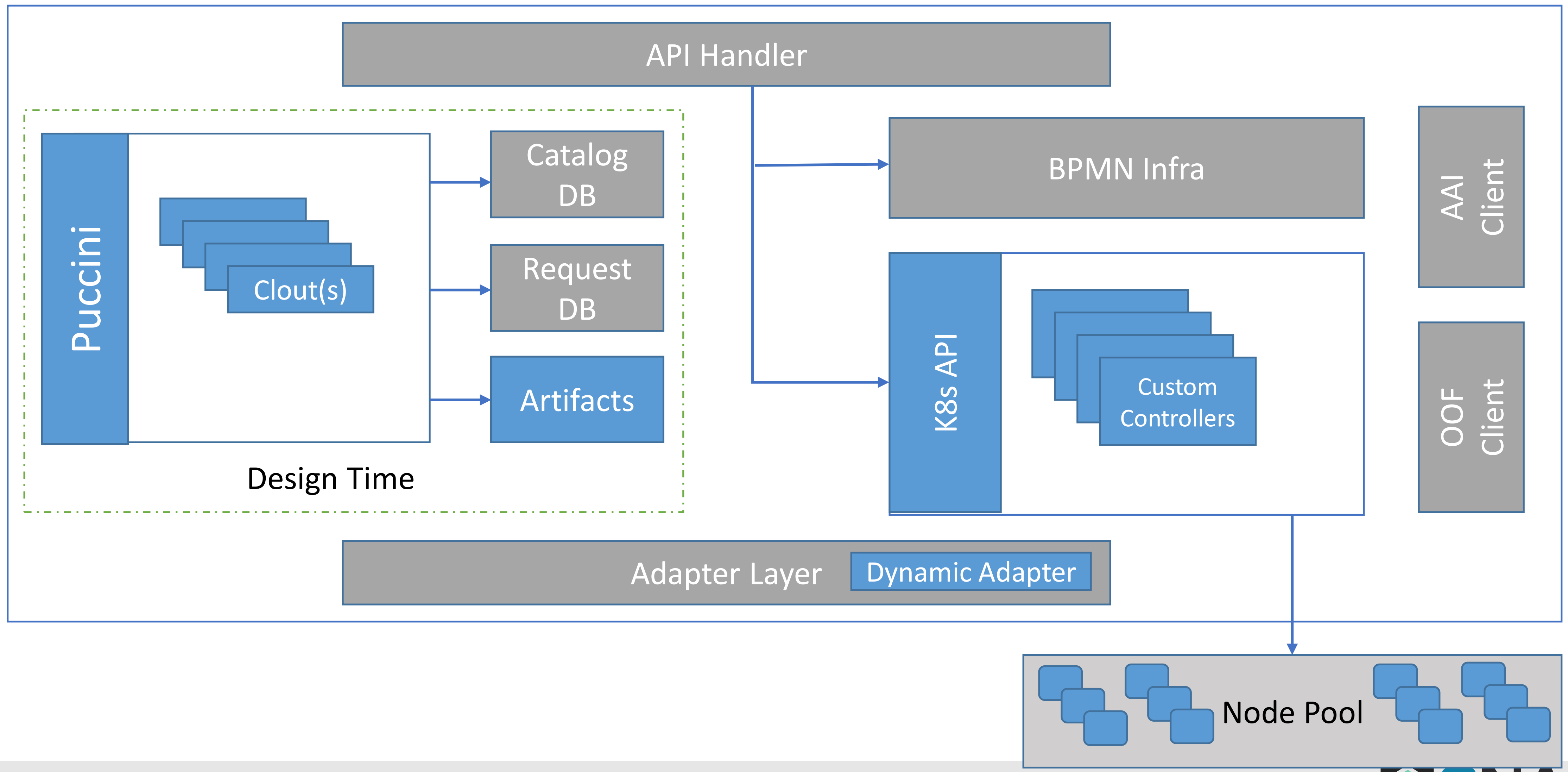
- Support Containerized Network Functions

Seshu Kumar M

So… What's Next SO…

- Dynamism
  - Customized Orchestration
- Orchestration next steps
  - **CNF support**
  - TOSCA
- Plug and Play
  - Both brown and Green Field adoption

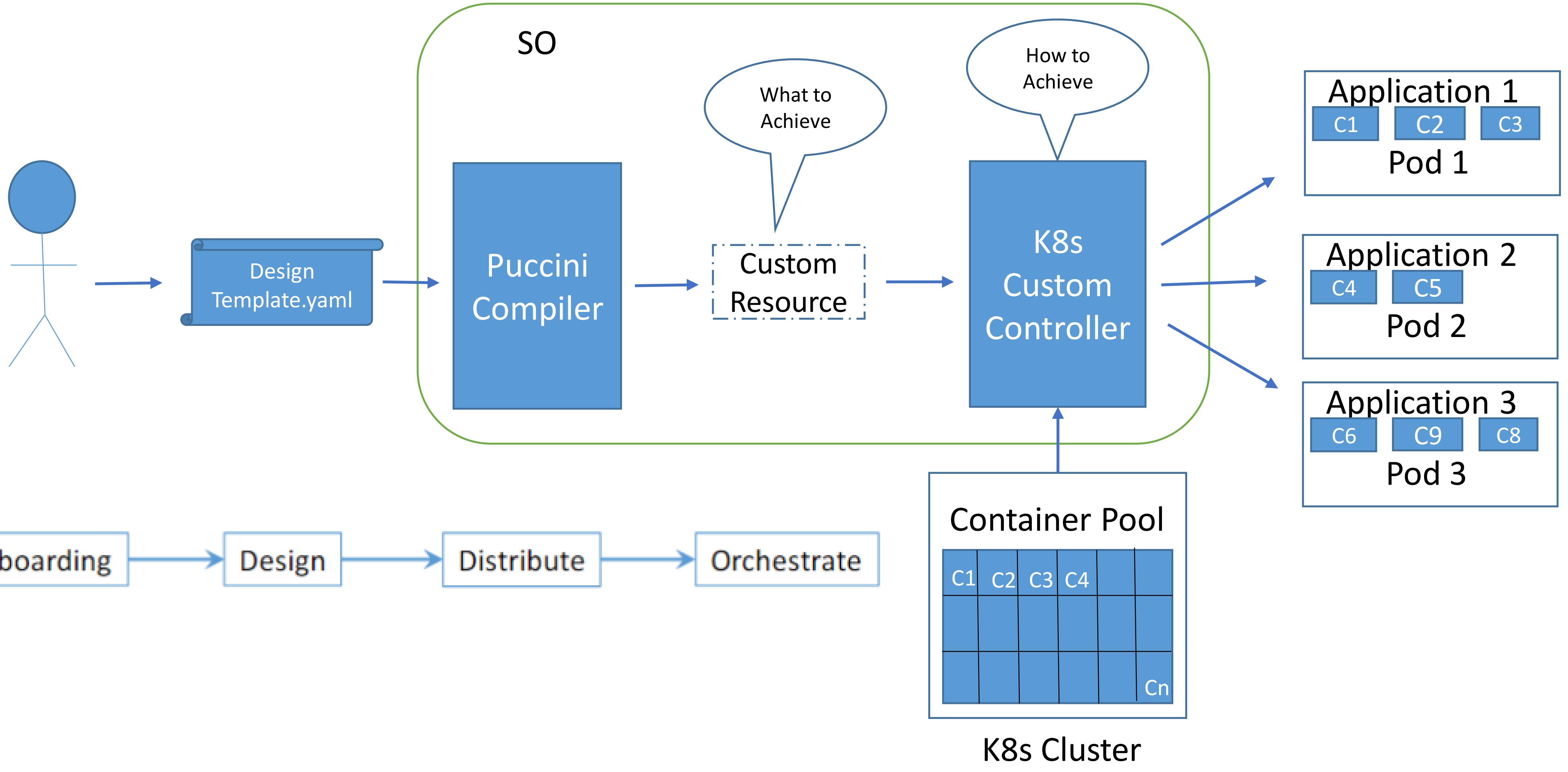
Desired SO Architecture

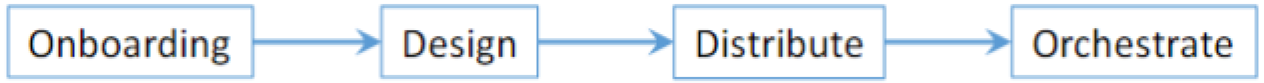


Key additions to the Architecture

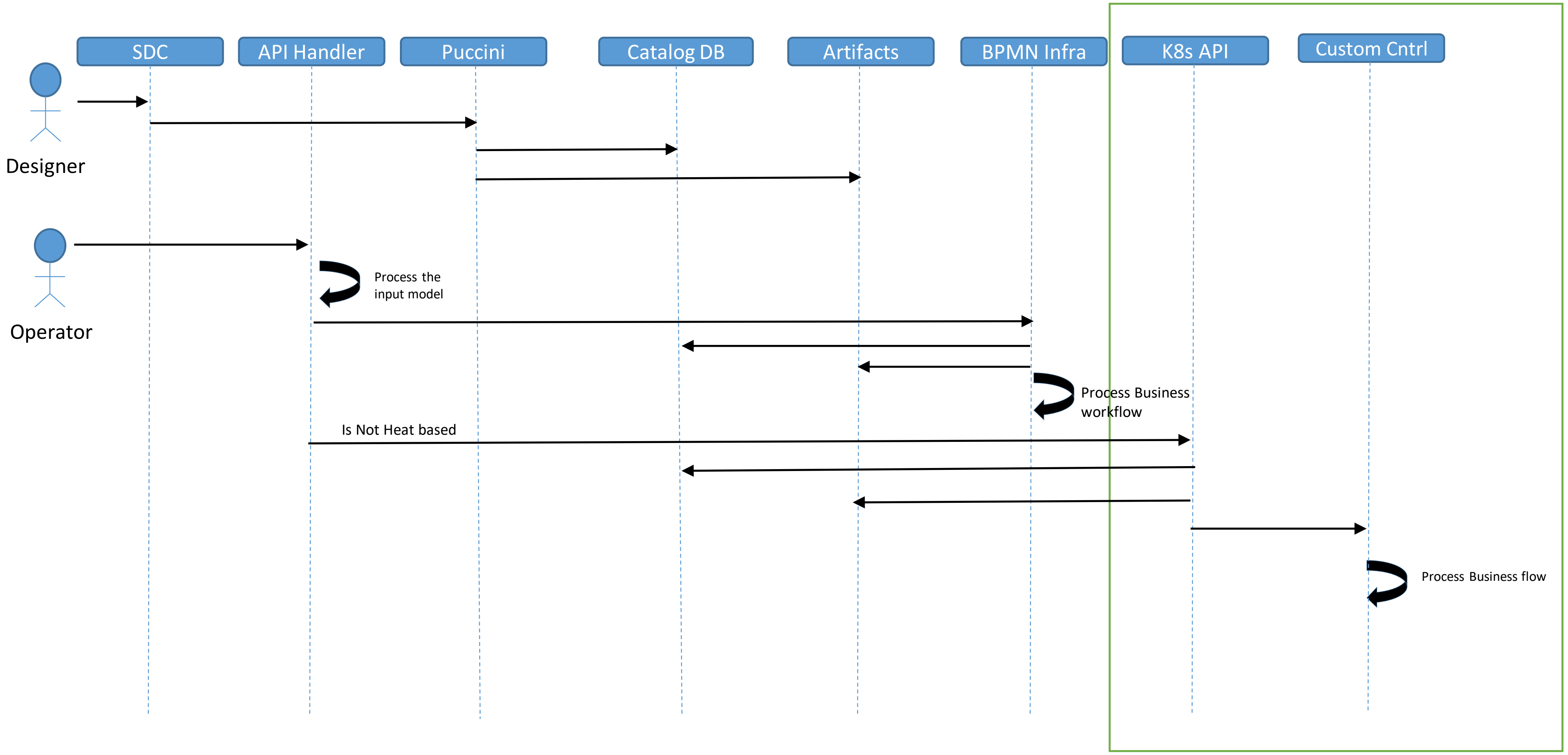
- Puccini
  - A TOSCA compiler that parses a given TOSCA service template and compiles it to Clout (Cloud Topology)
  - Puccini-tosca comes with TOSCA profiles for the Kubernetes and OpenStack cloud infrastructures, as well as BPMN processes.
  - Profiles include node, capability, relationship, policy, and other types that would work with any TOSCA-compliant product.
- Artifacts
  - Constitutes the design time entities that are onboarded to SDC and distributed to SO.
  - These could include the configurations, custom workflows designed, custom resource definitions, etc…
- K8s API
  - This constitutes of 2 key components Custom resources and Custom controllers.
- Custom resources
  - A *resource* is an endpoint in the K8s that stores a collection of API objects of a certain kind.
  - Custom resource is an extension of the Kubernetes API that is not necessarily available in a default Kubernetes installation.
  - It represents a customization of a particular Kubernetes installation and hence help in making Kubernetes more modular.
- Custom controllers
  - Custom resources let you store and retrieve structured data. When you combine a custom resource with a custom controller, custom resources provide a true declarative API.
  - Custom controllers interprets the structured data as a record of the user's desired state, and continually maintains this state.
  - CCs can work with any kind of resource, but they are especially effective when combined with custom resources.
- Dynamic Adapter
  - Built over the ONAP OComP (Open Command Platform) provides the plug and play functionality for the adaptation.
  - Enhances the codeless integration of the required external modules to SO.

Typical Functional Flow – CNF

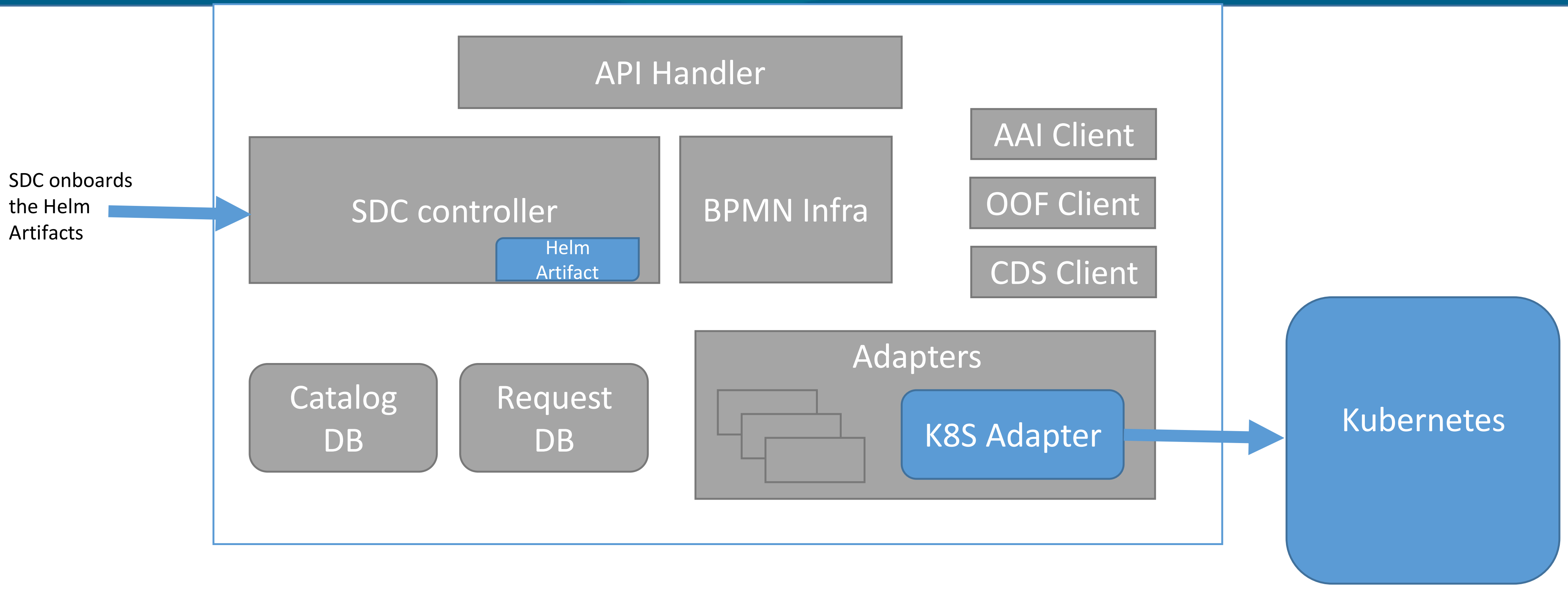




Typical Functinal Flow Diagram

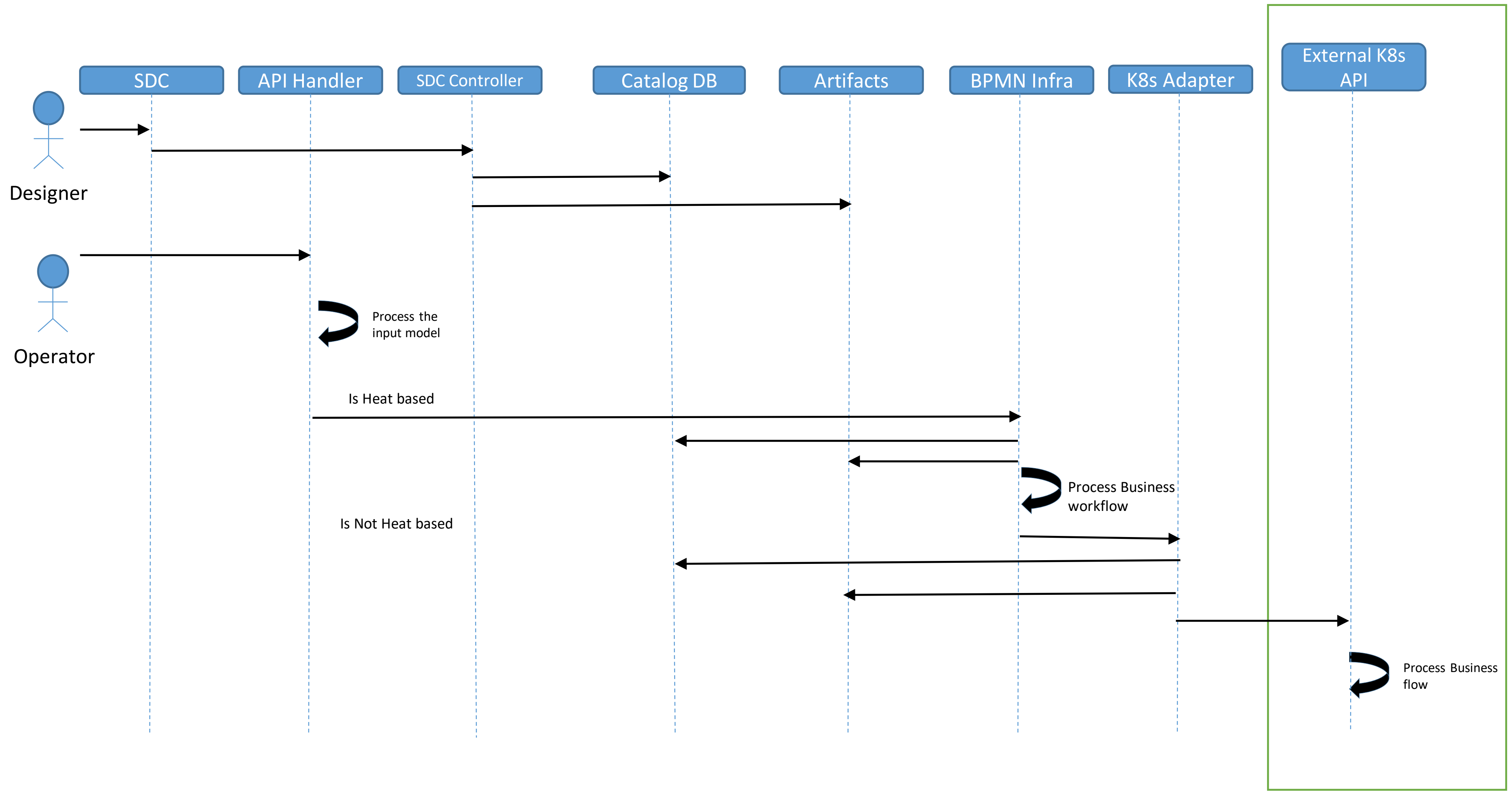


CNF support G Release - Proposal



- Leverage the K8s API adaptation in SO
  - Currently K8s Integration done in Multi Cloud is more a hacky way of passing the helm info embedded in a heat package.
  - In G release we intend to correct this by integrating this functionality in SO.
- Participating Companies
  - Huawei, Intel, Orange (Poland), Samsung

Flow Diagram for G Release



Summary for the requirement subcommittee

Executive Summary - Provide CNF orchestration support through integration of K8s adapter in ONAP SO

- Support for provisioning CNFs using an external K8s Manager
- Support the Helm based orchestration
- leverage the existing functionality of Multi cloud in SO
- Bring in the advantages of the K8s orchestrator and
- Set stage for the Cloud Native scenarios

Business Impact - Enables operators and service providers to orchestrate CNFs based services along with the VNFs and PNFs

Business Markets - All operators and service providers that are intended to use the CNFs along with PNFs / VNFs

Funding/Financial Impacts - Reduction in the footprint of the ONAP for CNF support.

Organization Mgmt, Sales Strategies -*There is no additional organizational management or sales strategies for this requirement outside of a service providers "normal" ONAP deployment and its attendant organizational resources from a service provider.*

Advantages

- The new architecture would leverage the existing ONAP SO functionality to even orchestrate the CNFs
- It brings in the advantages of a customization of resources aka Network functions and provides the bundles of advantages of K8s included with it.
- Support for provisioning CNFs using an external K8s Manager
- Support the Helm based orchestration
- leverage the existing functionality of Multi cloud in SO
- Bring in the advantages of the K8s orchestrator and
- Set stage for the Cloud Native scenarios