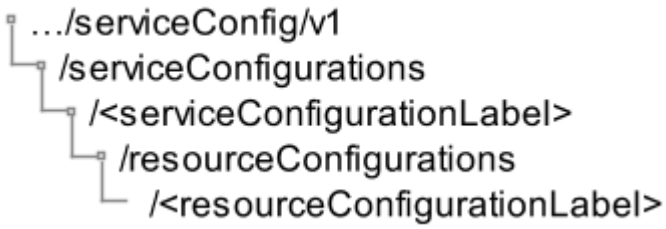
"D" - <<SMO>> ServiceConfiguration

SMO - ServiceConfiguration

The ServiceConfiguration provides a collection point for future or deployed configurations. Only Services which are available can have a configuration created. A configuration for a "Deployed" instance cannot be deleted. The Service Resource Configuration join a runtime location with a service instance at that location.

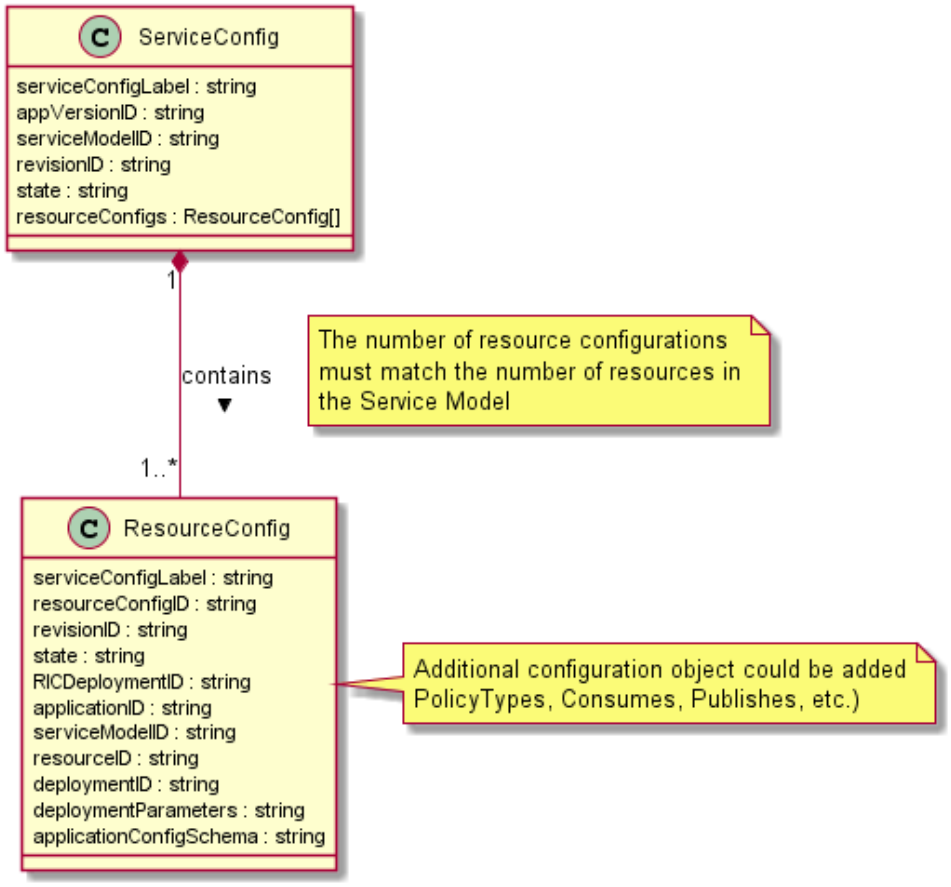
For the purposes of this implementation we will assume the following:

- The Controller will be RESTful from the service endpoint of {apiRoot}/serviceConfig/v1. The initial REST resource tree is show below.




```
@startsalt
{
scale 1.5
{T
+…/serviceConfig/v1
++/serviceConfigurations
+++/<serviceConfigurationLabel>
++++/resourceConfigurations
+++++/<resourceConfigurationLabel>
}
}
@endsalt
```

- The Service Configuration Model is represented in the class diagram below

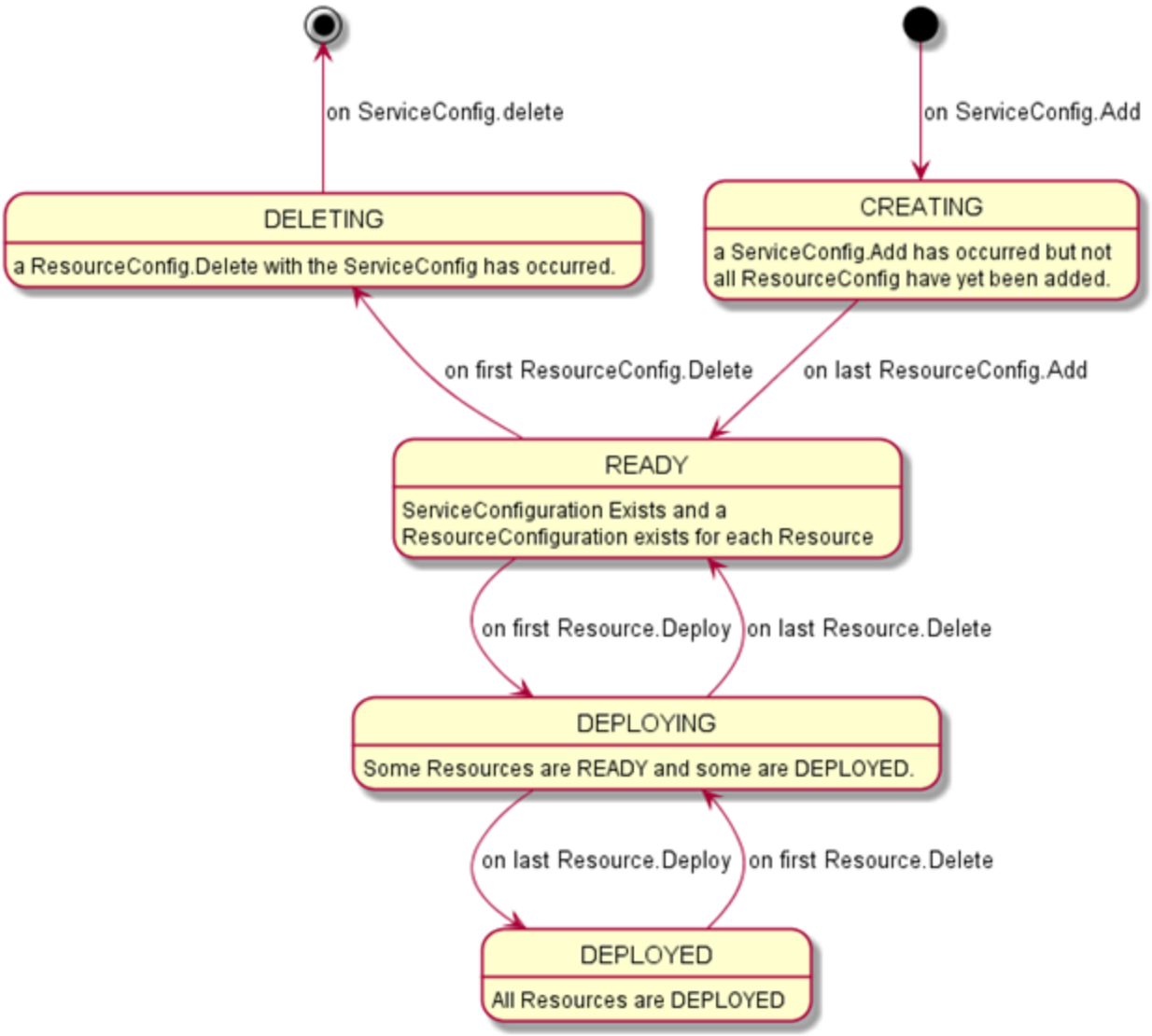



```
@startuml
Class ServiceConfig {
  serviceConfigLabel : string
  appVersionID : string
  serviceModelID : string
  revisionID : string
  state : string
  resourceConfigs : ResourceConfig[]
}

Class ResourceConfig {
  serviceConfigLabel : string
  resourceConfigID : string
  revisionID : string
  state : string
  RICDeploymentID : string
  appVersionID : string
  serviceModelID : string
  resourceID : string
  deploymentID : string
  deploymentParameters : string
  applicationConfigSchema : string
  applicationConfig : string
}
Note right: Additional configuration object could be added\nPolicyTypes, Consumes, Publishes, etc.)

ServiceConfig "1" *-- "1..*" ResourceConfig : contains >
Note right on link
  The number of resource configurations
  must match the number of resources in
  the Service Model
End note

@enduml
```

- The View will be JSON.

The ServiceConfiguration record in the Model Catalog follows a Stateful lifecycle. The ResourceConfigurations follow a sub-state model. The State can be updated with a partial update (PUT) as long as the current revisionID is supplied as a query parameter. Upon a successful update the revisionID will be changed by the system to a newly generated value. Deletes can only occur if the object has returned to its initial state.

The valid values for ServiceState are "READY", "INPROGESS", and "DEPLOYED". The state transitions allowed are:



@startuml

[*] -down-> CREATING : on ServiceConfig.Add
CREATING -down-> READY : on last ResourceConfig.Add
READY -down-> DEPLOYING : on first Resource.Deploy
DEPLOYING -up-> READY : on last Resource.Delete
DEPLOYING -down-> DEPLOYED : on last Resource.Deploy
DEPLOYED -up-> DEPLOYING: on first Resource.Delete
READY -up-> DELETING : on first ResourceConfig.Delete
DELETING -up-> [*] : on ServiceConfig.delete

CREATING : a ServiceConfig.Add has occurred but not
CREATING : all ResourceConfig have yet been added.

DELETING : a ResourceConfig.Delete with the ServiceConfig has occurred.

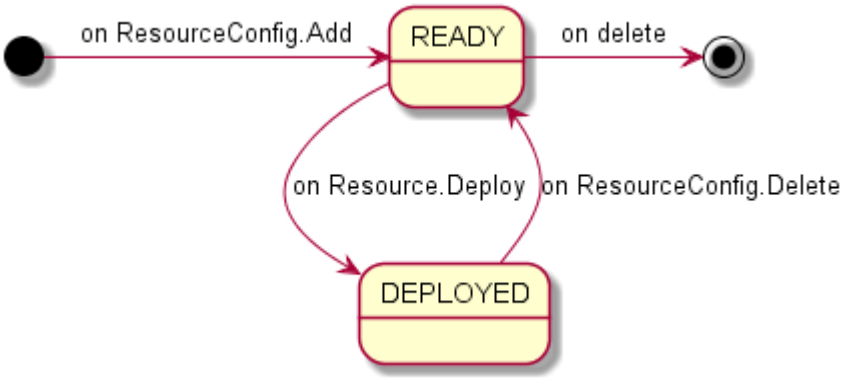
READY : ServiceConfiguration Exists and a
READY : ResourceConfiguration exists for each Resource

DEPLOYING: Some Resources are READY and some are DEPLOYED.

DEPLOYED: All Resources are DEPLOYED

@enduml

The valid values for ResourceState are "READY" "DEPLOYED". The state transitions allowed are:




```
@startuml

[*] -right-> READY : on ResourceConfig.Add
READY -right-> [*] : on delete

READY -down-> DEPLOYED : on Resource.Deploy
DEPLOYED -up-> READY : on ResourceConfig.Delete

@enduml
```