CcspCr

Overview

- The Component registry is a centralised database of registered components and services, which supports multiple management protocols and data models, such as SNMP and TR-069 with TR-181.
- The Component Registrar(CR) in RDK-B Architecture section gives an overview of CCSP Component Registrar, what it contains and illustrates how it is used by other internal components and client applications.
- The "Component Registrar" works like an internal name server.
- Allows other components to register the internal data model objects that they support and the DBuspath used to reach these objects
- Component Registrar provides this information , when other components need to determine what features are available in a particular software build and discover how to access these features
- Through these capabilities, this mechanism provides a flexible way to build a software system that can use any number of software components, allows components to discover what features are available and how to access these features
- Maintains the status and health of all registered components

How it Works?

- When a component needs to interact with an object it first asks Component Registrar for a bus address for the object.
- Component Registrar responds with the object address (if the object exists in this software build), hence the object can be accessed.
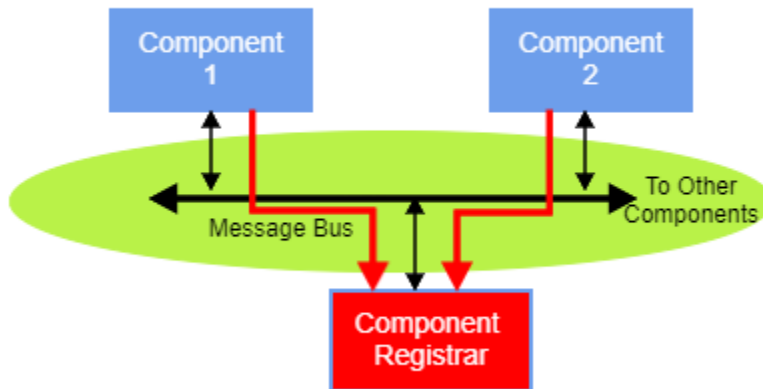
APIs

Some of the APIs supported by Component Registrar:

API	Description
registerCapabilities	Components to use this API to register supported parameters and objects with Cr
unregistername_space	API to unregister a specific parameter/object
unregisterComponent	API to deregister a specific component
discComponentSupportingNamespace	API to discover the component supporting a parameter or object
checkNamespaceDataType	API to discover the data type of the parameters supported
getRegisteredComponents	API to discover all the registered components at that time

discNamespaceSupportedByComponent	API to discover all the parameters registered by a component

Sequence of API calls to be used to register component

Example of registering PAM component

```
          ┌─────────────────────────────┐
          │      Create and Init        │
          │        Component            │
          │      ssp_engage_pnm()       │
          └──────────────┬──────────────┘
                         │
          ┌──────────────▼──────────────┐
          │     Register Component      │
          │   RegisterCcspDataModel()   │
          └──────────────┬──────────────┘
                         │
          ┌──────────────▼──────────────┐
          │    Callback function from   │
          │      CcspCommonLibrary      │
          │ DslhCpecoRegisterDataModel()│
          └──────────────┬──────────────┘
                         │
          ┌──────────────▼──────────────┐
          │ CcspBaseIf_registerCapabilities │
          └──────────────┬──────────────┘
                         │
          ┌──────────────▼──────────────┐
          │  New Dbus msg to invoke method │
          │    "registerCapabilities"   │
          └──────────────┬──────────────┘
                         │
          ┌──────────────▼──────────────┐
          │  Dbus request Processing from CR │
          │          component          │
          │   CcspCrProcessDbusRequest()│
          └──────────────┬──────────────┘
                         │
          ┌──────────────▼──────────────┐
          │     registerCapabilities()  │
          └──────────────┬──────────────┘
                         │
          ┌──────────────▼──────────────┐
          │   CcspCrRegisterCapabilities│
          └──────────────┬──────────────┘
                         │
          ┌──────────────▼──────────────┐
          │ Register the component in namespace │
          │           Object            │
          │      RegisterNamespaces()   │
          │                             │
          │ and return component registeration status │
          └─────────────────────────────┘
```