Adding a New Test Component

Create your first Component. This wiki page serves as a helpful reference and source for creating a New Test Component and Register it with CR. We will take the example of Raspberry Pi device as reference to demonstrate the involved steps.

- System Requirements
- Data Model
    - Test Component API List
- Build RDK-B Image for Raspberry Pi device
- Build ccsp-testcomponent for Raspberry Pi device
- Run the Test Component
    - Copy the supported files to Raspberry Pi device
    - Execute the TestComponent in the Raspberry Pi device
- Validate if component is registered with CR
- Include Newly Added Component in Package Group

System Requirements

- **Raspberry Pi device**
  Exercise will be executed on the Raspberry Pi device. A Raspberry Pi 2 or 3 device can function as a basic router through which a connected CPE device can access the admin portal and surf the Internet.

- **Source Code**
  Sample component named "CcspTestComponent" is provided as part of this exercise

Data Model

In RDK-B, TR-181 data model is used for interacting with the CPE (Customer Premises Equipment). These data models contain objects and parameters that describe functions and capabilities available to devices, that are manageable via CWMP (Eg: SNMP, TR069). Each component has its own data model.

Data Model - Code Snippet

```
<dataModelInfo>
  <objects>
    <object>
      <name>TestComponent</name>
      <objectType>object</objectType>
      <functions>
        <func_GetParamUlongValue>TestComponent_GetParamUlongValue</func_GetParamUlongValue>
        <func_SetParamUlongValue>TestComponent_SetParamUlongValue</func_SetParamUlongValue>
      </functions>
      <parameters>
        <parameter>
          <name>TestSampleParamUlong</name>
          <type>unsignedInt</type>
          <syntax>uint32</syntax>
          <writable>true</writable>
        </parameter>
      </parameters>
    </object>
 </objects>
</dataModelInfo>
```

- objects – Data Model can have multiple objects
  name - Name of the object (Here, TestComponent)
  functions - Callback APIs to Get/Set Parameters of TestComponent object
    - TestComponent_GetParamUlongValue
    - TestComponent_SetParamUlongValue
- parameters - Attributes of Parameter
    - name – Name of the Parameter (TestSampleParamUlong)
    - type – Type of the Parameter (unsignedInt)
    - syntax – Syntax representation of the parameter type (uint32)
    - writable – Read/Write access of the parameter (true - rw access)

Test Component API List

API Name	Description
• TestComponent_GetParamUlongValue	API is used to retrieve the parameter value of type “unsignedInt”
• TestComponent_SetParamUlongValue	API is used to set the parameter value of type “unsignedInt”
• TestComponent_GetParamStringValue	API is used to retrieve the parameter value of type “string”
• TestComponent_SetParamStringValue	API is used to set the parameter value of type “string”
• TestComponent_Commit	To Commit all the update to the data model

Build RDK-B Image for Raspberry Pi device

1. *mkdir <workspace dir>*
   cd <workspace dir>
2. *repo init -u https://code.rdkcentral.com/r/manifests -m rdkb-raspberrypi.xml -b morty*
3. *repo sync*
4. *source meta-cmf-raspberrypi/setup-environment*
5. Select option *raspberrypi-rdk-broadband.conf*
6. *bitbake rdk-generic-broadband-image*

Build ccsp-testcomponent for Raspberry Pi device

1. Put the ccsp-testcomponent.bb recipe file inside meta-rdk-broadband/recipes-ccsp/ccsp location.
2. Create folder names "files" under "meta-rdk-broadband/recipes-ccsp/ccsp" path
3. Put CcspTestComponent.tar.gz (source code of TestComponent) file inside meta-rdk-broadband/recipes-ccsp/ccsp/files location.
4. Build the test component inside build directory i.e. build-raspberrypi-rdk-broadband directory.
   bitbake ccsp-testcomponent
5. Binary file of test component will be present inside build-raspberrypi-rdk-broadband/tmp/work/cortexa7hf-neon-vfpv4-rdk-linux-gnueabi/ccsp-testcomponent/1.0-r0/image/usr/bin/ location
6. XML file of test component will be present inside build-raspberrypi-rdk-broadband/tmp/work/cortexa7t2hf-neon-vfpv4-rdk-linux-gnueabi/ccsp-testcomponent/1.0-r0/image/usr/ccsp/testcomponent/ location.

NOTE : For rpi4 , yocto 3.1 use ccsp_testcomponent.zip to avoid build errors due to cfg directory missing

Run the Test Component

Copy the supported files to Raspberry Pi device

Copy test_component (Binary of ccsp-testcomponent, generated in step 4 of Build) and TestComponent.xml (present at location mentioned in step 5 of Build) to /tmp/ directory of Raspberry Pi device.

Execute the TestComponent in the Raspberry Pi device

Go to /tmp/ directory of Raspberry Pi 3 and start the ccsp-testcomponent
./test_component -subsys eRT.

You can check the status of the process by doing *ps* in the RPi device.

Validate if component is registered with CR

DMCLI (Database Manager Command Line Interface) provides interface used to send and receive command/messages via CLI (Command Line Interface) over Telnet and SSH protocols.

List all the test component parameters using dmcli:

dmcli eRT getv Device.TestComponent.

```
Expected Output:
CR component name is: eRT.com.cisco.spvtg.ccsp.CR
subsystem_prefix eRT.
getv from/to component(eRT.com.cisco.spvtg.ccsp.testcomponent): Device.TestComponent.
Execution succeed.
Parameter    1 name: Device.TestComponent.TestSampleParamUlong
              type:       uint,    value: 1
```

Change the "TestSampleParamUlong" parameter value by using below command:

dmcli eRT setv Device.TestComponent.TestSampleParamUlong uint 5

Now verify the value by this command

dmcli eRT getv Device.TestComponent.TestSampleParamUlong

Include Newly Added Component in Package Group

Once the validation is successfully done and we can now include the new component (ccsp-testcomponent) in the package groups.

1. Open "packagegroup-rdk-ccsp-broadband.bb" file present at "meta-rdk/recipes-core/packagegroups/packagegroup-rdk-ccsp-broadband.bb" location.
2. In the "RDEPENDS_packagegroup-rdk-ccsp-broadband" flag add the name of new component.