CcspWifiAgent

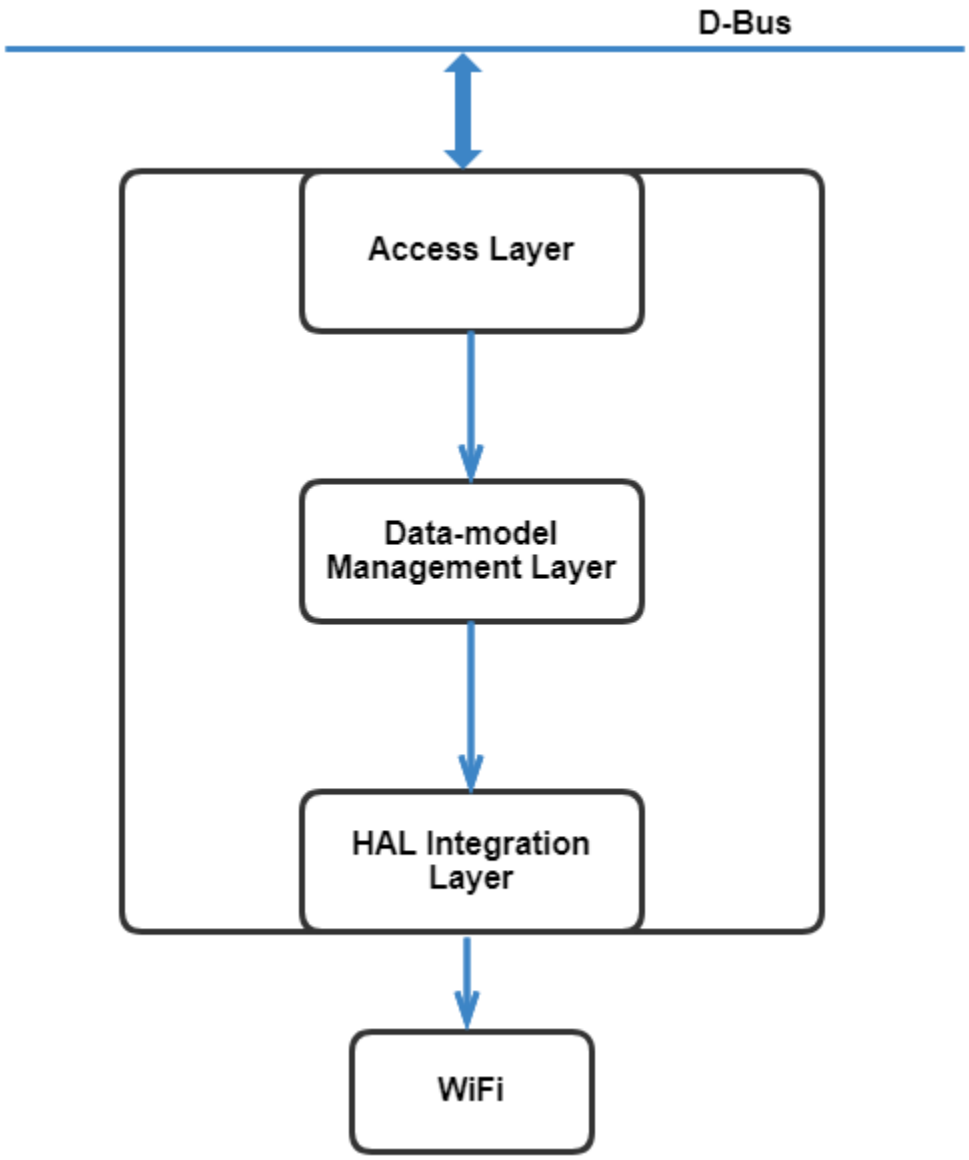
Introduction

RDK-B software stack provides networking interfaces such as Wi-Fi. WiFi Agent provides support for Home Wi-Fi, Hotspot and HomeSecurity AP functions.

Architecture

Wi-Fi component implements Wi-Fi functionality of the device, it supports all the parameters which are defined in the TR-181 data-model and CCSP specific Wi-Fi extensions.
Figure 1 shows the architecture diagram of Wi-Fi module in the gateway stack. Here, Access Layer takes care of call to/from D-Bus. It provides all API's for GET/SET parameters based on request from various other modules (components/PAs) in the gateway.

Data-model Management Layer (DML) loads all data model access APIs through a pre-defined XML file TR181-Wi-Fi-USGV2.XML. An XML description of data model objects and parameters is given for Data Model Management Layer to load. Subsequently, Data Model Management Layer loads the shared library (libWi-Fi.so) which contains data model implementation, specified in the XML file.
The data model implementation in shared library interacts with HAL Integration Layer by calling component specific HAL APIs. These calls covert the user space calls to system level calls and takes the necessary action at driver level.



Code Flow

Initialization sequence of Wi-Fi

Once the Wi-Fi driver comes up at platform layer there are a sequence of operations carried at different layers of Wi-Fi subsystem of the gateway. Wi-Fi initialization is carried out by different layers.

During gateway boot-up, the XML file which has all information of the objects, parameters and the GET/SET handlers will be registered using common component library. TR181-Wi-Fi-USGV2.XML has the information related to Wi-Fi objects and parameters.

Once the component specific library is loaded (libWi-Fi.so), the APIs are registered and then the component specific initialisation happens.

During the process of component specific data-model initialisation when Wi-Fi gets the component specific plugin initialisation command from CCSP data-model library, corresponding back end manager is called to initialise the Wi-Fi. Procedure flow of Wi-Fi initialisation is depicted in Figure-3 below.

Wi-Fi driver initialisation is carried by the component specific API layer, which will try to fetch the parameters from PSM and apply them to the driver if needed and reinitialise the Access Points.

Wi-Fi follows the layered approach for interaction. Middle layer provides the necessary APIs to the common component so as to interact with the component specific layer.

Middle layer then interact with the Integration layer whose responsibility is to mainly deal with the data-model APIs, driver specific APIs or dbus APIs to manage Wi-Fi specific parameters.

Integration layer will act as a gluing layer between the driver/dbus/PSM databases and the upper layers.

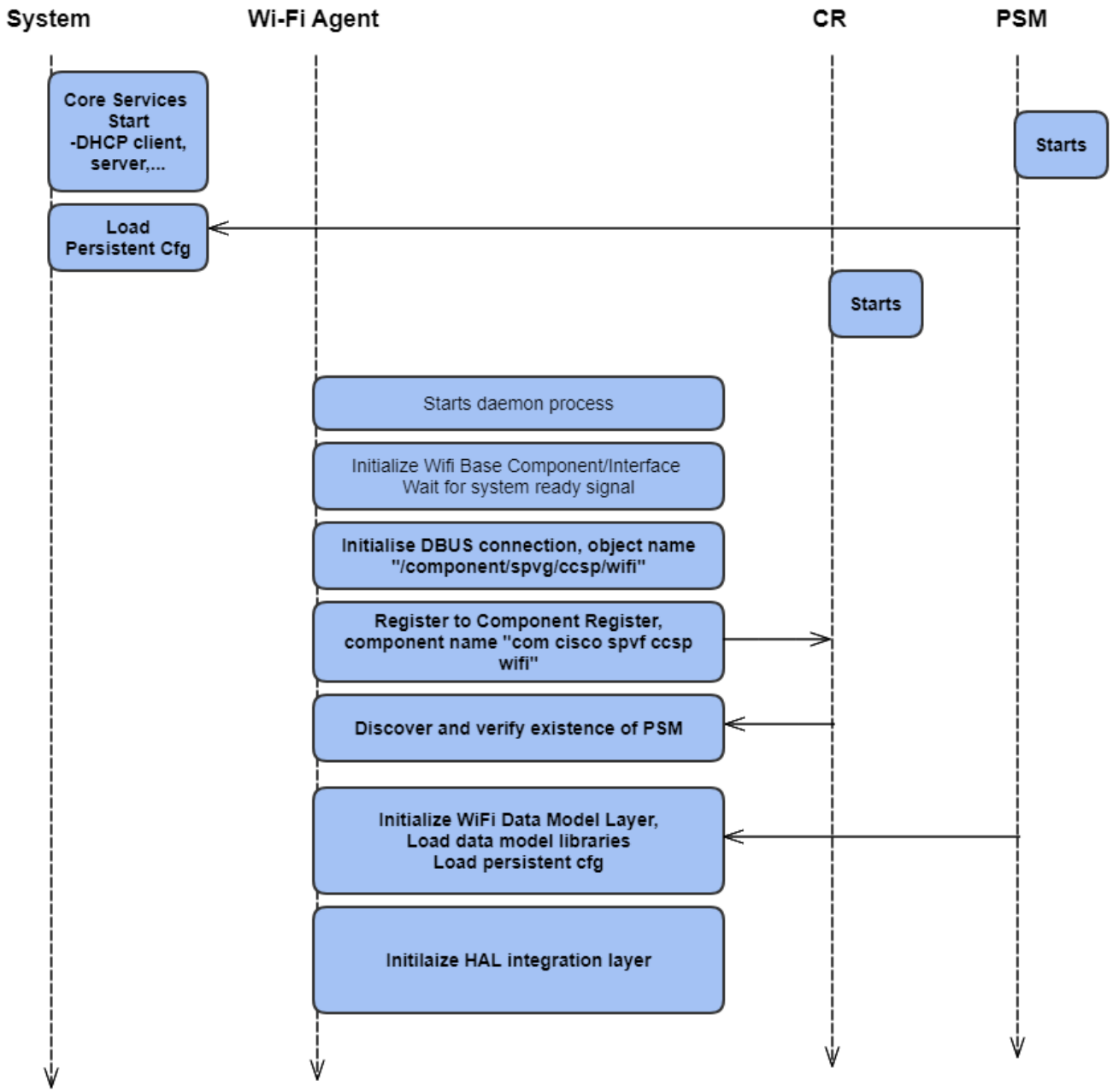


Figure 2: Wi-Fi Agent Boot-up Flow

During startup, Wi-Fi Agent initializes the parameter tree and retrieves the existing persistent configuration and related static or dynamic info from HAL integration layer API.

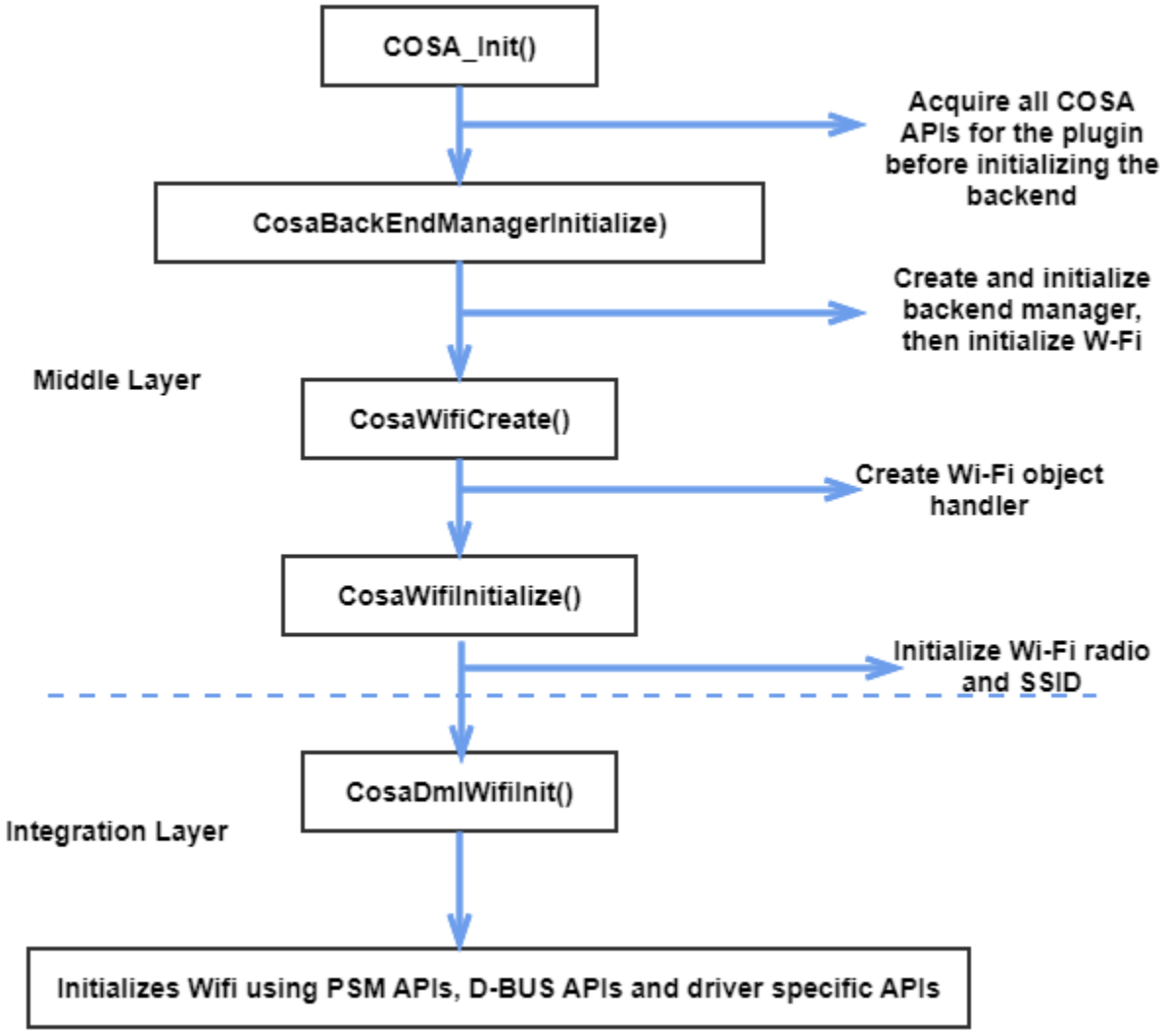


Figure 3: Wi-Fi Initialization

Code flow for getting a Wi-Fi parameter

During initialisation phase of gateway, all the necessary APIs related to an object mentioned in the data-model USG file will be registered with the data-model management system. So if a query comes for a particular parameter of an object, the registered API for that particular parameter type will be called so that the information can be retrieved from the Wi-Fi module.

Once the query comes from UI/SNMP/TR69, the PA accesses corresponding D-BUS API. Later the object is identified from the registry and the corresponding API call specific to the parameter being queried is accessed. API to be accessed is determined by the data-model XML which is used for the library. Then the parameter retrieval process end up in retrieving the value for the parameter either from a centralized database maintained by Persistent Storage Manager (PSM) or from a Wi-Fi HAL.

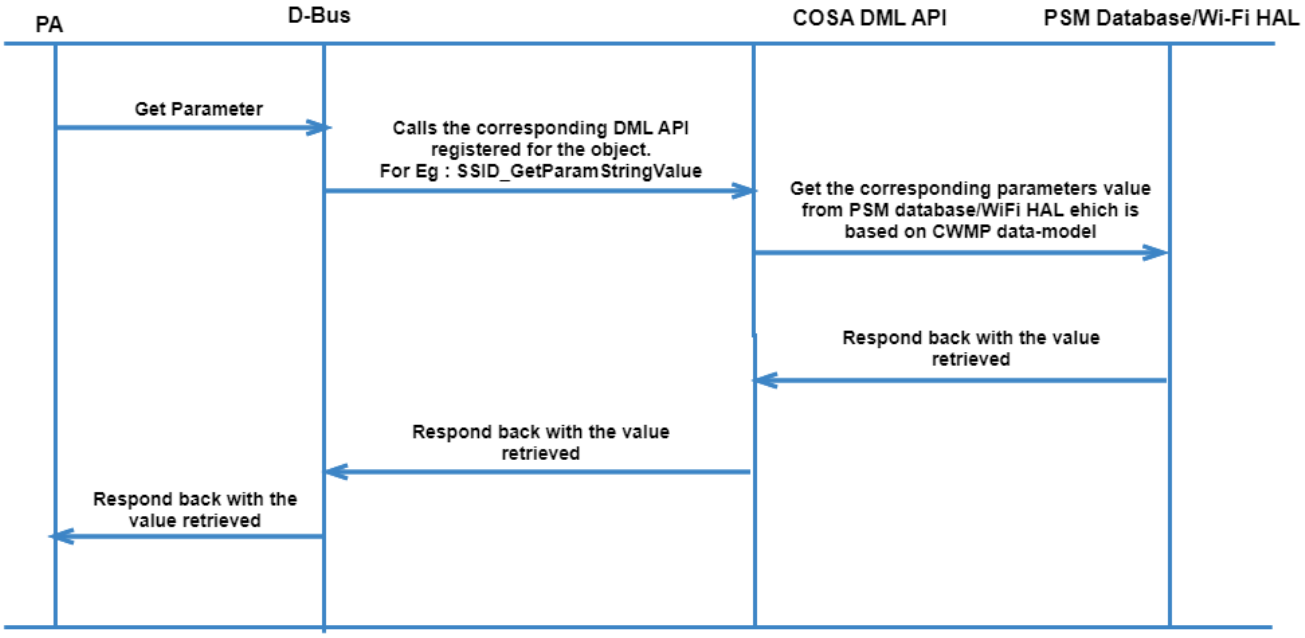


Figure 4: Code flow of Wi-Fi subsystem

Code details of RDK-B implementations

Wi-Fi subsystem is made as part of a shared library libWi-Fi.so

Interactions of Wi-Fi libraries are as below:

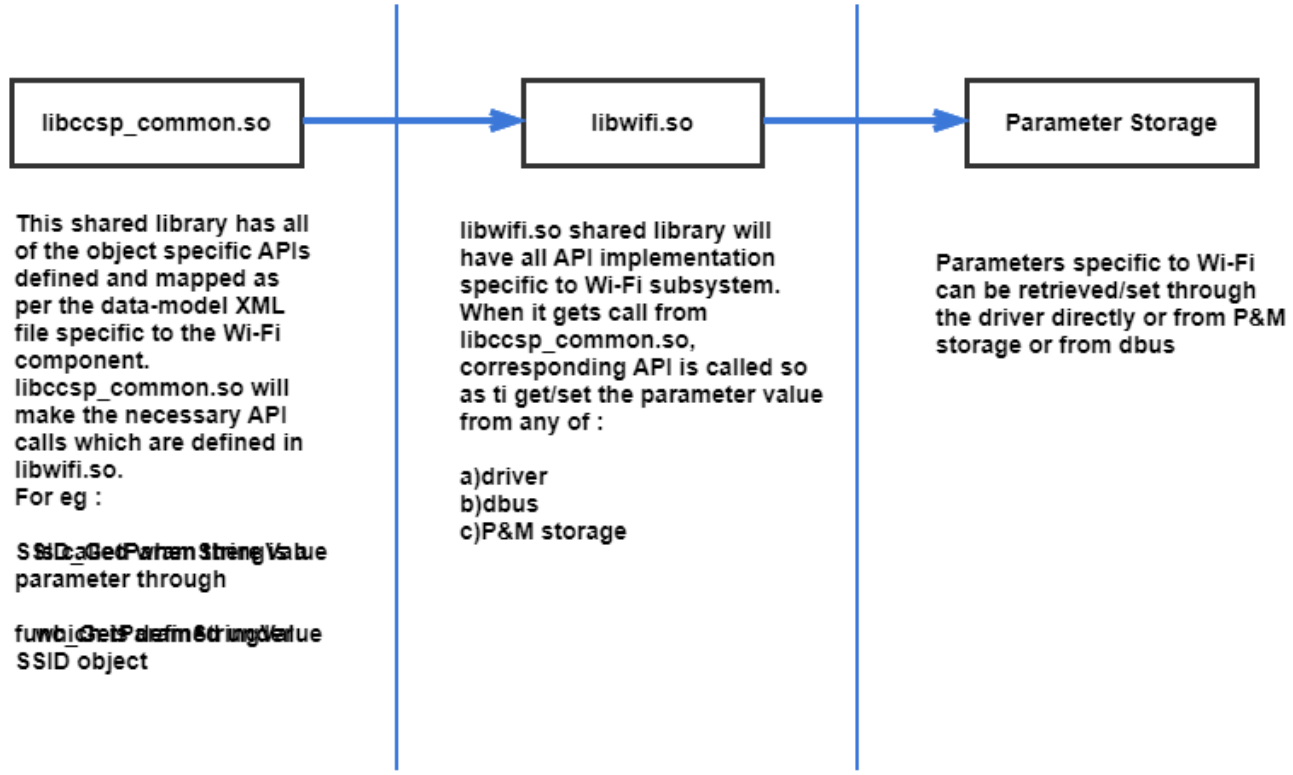


Figure 5: Library Interactions Involved

Objects

Objects supported by WiFi:

Note: The RDK data model naming convention prefix was changed in March 2020 to “X_RDK_”. We request you use this new prefix going forward.

Device.WiFi.Radio.
Device.WiFi.SSID.
Device.WiFi.AccessPoint.

Device.WiFi.AccesPoint.{i}.AssociatedDevice

Device.WiFi.X_RDKCENTRAL-COM_BandSteering.
Device.WiFi.X_RDKCENTRAL-COM_ATM.

$ dmcli eRT getv Device.WiFi.AccessPoint.
CR component name is: eRT.com.cisco.spvtg.ccsp.CR
subsystem_prefix eRT.
getv from/to component(eRT.com.cisco.spvtg.ccsp.wifi): Device.WiFi.AccessPoint.
Execution succeed.
Parameter 1 name: Device.WiFi.AccessPoint.1.Enable
type: bool, value: true
Parameter 2 name: Device.WiFi.AccessPoint.1.Status
type: string, value: Enabled
Parameter 3 name: Device.WiFi.AccessPoint.1.Alias
type: string, value: AccessPoint1
Parameter 4 name: Device.WiFi.AccessPoint.1.SSIDReference
type: string, value: Device.WiFi.SSID.1.
Parameter 5 name: Device.WiFi.AccessPoint.1.SSIDAdvertisementEnabled
type: bool, value: true
Parameter 6 name: Device.WiFi.AccessPoint.1.RetryLimit
type: uint, value: 16
Parameter 7 name: Device.WiFi.AccessPoint.1.X_CISCO_COM_LongRetryLimit
type: uint, value: 16
Parameter 8 name: Device.WiFi.AccessPoint.1.WMMCapability
type: bool, value: true
Parameter 9 name: Device.WiFi.AccessPoint.1.UAPSDCapability
type: bool, value: true
Parameter 10 name: Device.WiFi.AccessPoint.1.WMMEnable
type: bool, value: true
Parameter 11 name: Device.WiFi.AccessPoint.1.UAPSDEnable
type: bool, value: true
Parameter 12 name: Device.WiFi.AccessPoint.1.AssociatedDeviceNumberOfEntries
type: uint, value: 0
Parameter 13 name: Device.WiFi.AccessPoint.1.X_CISCO_COM_WmmNoAck
type: int, value: 0
Parameter 14 name: Device.WiFi.AccessPoint.1.X_CISCO_COM_MulticastRate
type: int, value: 123
Parameter 15 name: Device.WiFi.AccessPoint.1.IsolationEnable
type: bool, value: false
Parameter 16 name: Device.WiFi.AccessPoint.1.X_CISCO_COM_BssMaxNumSta
type: int, value: 30
Parameter 17 name: Device.WiFi.AccessPoint.1.X_CISCO_COM_BssCountStaAsCpe
type: bool, value: true
Parameter 18 name: Device.WiFi.AccessPoint.1.X_CISCO_COM_BssUserStatus
type: int, value: 1
Parameter 19 name: Device.WiFi.AccessPoint.1.X_CISCO_COM_BssHotSpot
type: bool, value: false
Parameter 20 name: Device.WiFi.AccessPoint.1.X_RDKCENTRAL-COM_BeaconRate
type: string, value:
Parameter 21 name: Device.WiFi.AccessPoint.1.X_CISCO_COM_KickAssocDevices
type: bool, value: false
Parameter 22 name: Device.WiFi.AccessPoint.1.MaxAssociatedDevices
type: uint, value: 5
Parameter 23 name: Device.WiFi.AccessPoint.1.X_COMCAST-COM_AssociatedDevicesHighWatermarkThreshold
type: uint, value: 50
Parameter 24 name: Device.WiFi.AccessPoint.1.X_COMCAST-COM_AssociatedDevicesHighWatermarkThresholdReached
type: uint, value: 3
Parameter 25 name: Device.WiFi.AccessPoint.1.X_COMCAST-COM_AssociatedDevicesHighWatermark
type: uint, value: 3
Parameter 26 name: Device.WiFi.AccessPoint.1.X_COMCAST-COM_AssociatedDevicesHighWatermarkDate
type: uint, value: 1578311709
Parameter 27 name: Device.WiFi.AccessPoint.1.X_COMCAST-COM_InterworkingServiceCapability
type: bool, value: false

```
Parameter 28 name: Device.WiFi.AccessPoint.1.X_COMCAST-COM_InterworkingServiceEnable
type: bool, value: false
Parameter 29 name: Device.WiFi.AccessPoint.1.X_COMCAST-COM_MAC_FilteringMode
type: string, value: Allow-ALL
Parameter 30 name: Device.WiFi.AccessPoint.1.X_RDKCENTRAL-COM_ManagementFramePowerControl
type: int, value: 0
Parameter 31 name: Device.WiFi.AccessPoint.1.X_RDKCENTRAL-COM_rapidReconnectCountEnable
type: bool, value: true
Parameter 32 name: Device.WiFi.AccessPoint.1.X_RDKCENTRAL-COM_rapidReconnectMaxTime
type: int, value: 180
Parameter 33 name: Device.WiFi.AccessPoint.1.X_COMCAST-COM_TXOverflow
type: uint, value: 0
Parameter 34 name: Device.WiFi.AccessPoint.1.X_RDKCENTRAL-COM_StatsEnable
type: bool, value: false
Parameter 35 name: Device.WiFi.AccessPoint.1.X_RDKCENTRAL-COM_WirelessManagementImplemented
type: bool, value: false
Parameter 36 name: Device.WiFi.AccessPoint.1.X_RDKCENTRAL-COM_BSSTransitionImplemented
type: bool, value: false
Parameter 37 name: Device.WiFi.AccessPoint.1.X_RDKCENTRAL-COM_BSSTransitionActivated
type: bool, value: false
Parameter 38 name: Device.WiFi.AccessPoint.1.X_RDKCENTRAL-COM_NeighborReportActivated
type: bool, value: false
Parameter 39 name: Device.WiFi.AccessPoint.1.X_CISCO_COM_MacFilterTableNumberOfEntries
type: uint, value: 0
```

HAL APIs

wifi_hal.h provides the function prototypes and structure definitions used for the RDK-Broadband Wifi HAL

Git repo: https://code.rdkcentral.com/r/plugins/gitiles/rdkb/components/opensource/ccsp/halinterface/+/rdk-next/wifi_hal.h

Some example APIs are listed below:

APIs
wifi_init
wifi_initRadio
wifi_down
wifi_reset
wifi_factoryReset
wifi_factoryResetRadio
wifi_setApEnable
wifi_getApEnable
wifi_getApStatus
wifi_getSSIDName
wifi_getWifiChannelStats
wifi_getRadioChannelStats
wifi_getApAssociatedDeviceRxStatsResult
wifi_getHalVersion
wifi_factoryResetRadio
wifi_setLED
wifi_setRadioCountryCode
wifi_pushCountryCode
wifi_getATMCapable
wifi_setATMEnable
wifi_getATMEnable

wifi_setApATMAirTimePercent
wifi_getApATMAirTimePercent
wifi_getApATMSta
wifi_setApATMSta
wifi_getRadioNumberOfEntries
wifi_getSSIDNumberOfEntries
wifi_getRadioIfName
wifi_getRadioMaxBitRate
wifi_getRadioSupportedFrequencyBands
wifi_getRadioOperatingFrequencyBand
wifi_getRadioSupportedStandards
wifi_getRadioMode
wifi_setRadioChannelMode
wifi_setRadioMode
wifi_getRadioPossibleChannels
wifi_getRadioChannelsInUse
wifi_setRadioAutoChannelEnable
wifi_getRadioAutoChannelSupported
wifi_getRadioDCSSupported
wifi_getRadioDCSEnable
wifi_getRadioDCSChannelPool
wifi_getRadioDCSScanTime
wifi_getRadioDcsDwelltime
wifi_getRadioDfsSupport
wifi_getRadioAutoChannelRefreshPeriodSupported
wifi_getRadioAutoChannelRefreshPeriod
wifi_setRadioAutoChannelRefreshPeriod
wifi_setRadioOperatingChannelBandwidth
wifi_getRadioExtChannel
wifi_setRadioExtChannel
wifi_getRadioGuardInterval
wifi_setRadioGuardInterval
wifi_getRadioMCS
wifi_setRadioMCS
wifi_getRadioTransmitPowerSupported
wifi_getRadioTransmitPower

To see the API specfification. Please refer Wifi HAL API Secifications