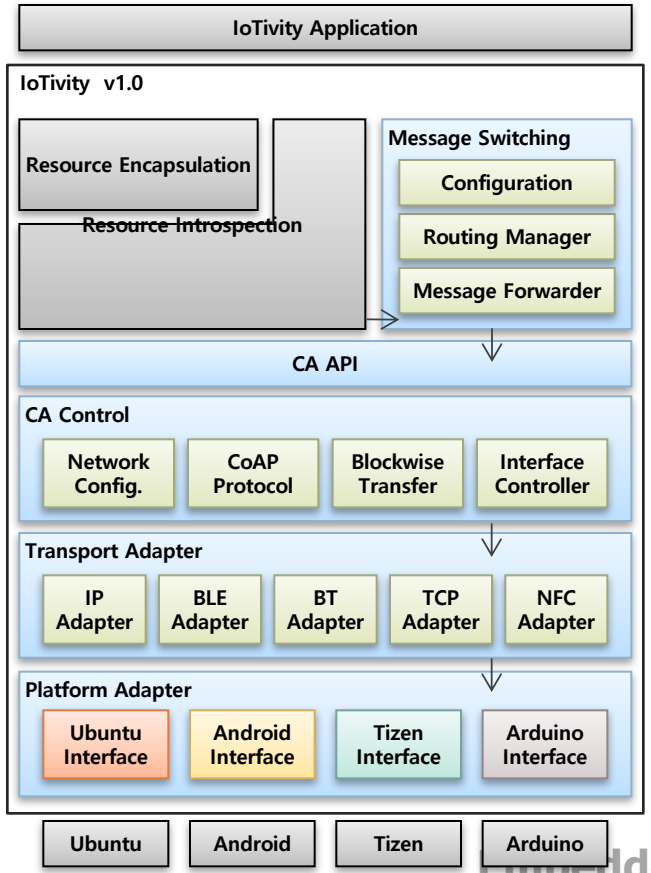
Tizen/Artik IoT Lecture Chapter 7. IoTivity Connectivity Abstraction

Sungkyunkwan University

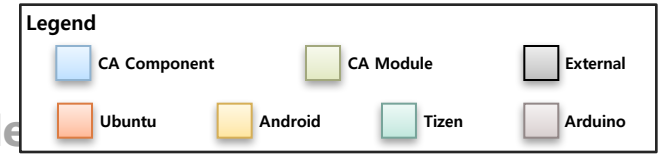
Contents

- **Architecture**
- **Routing Through Heterogeneous Connectivity**
- **Blockwise Transfer**
- **Call Path**
  - Sending Data
  - Receiving Data
- **CA APIs**

Architecture

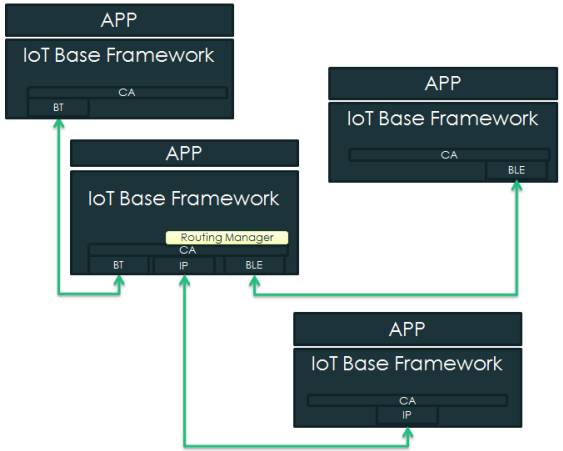


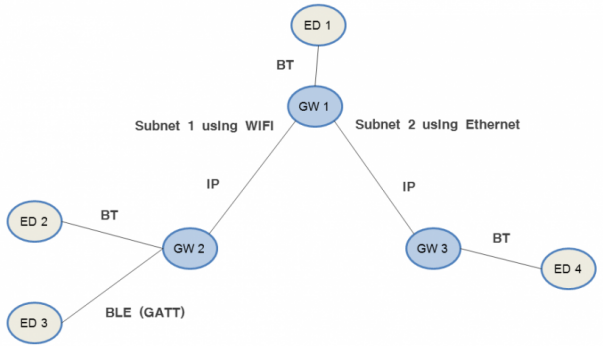
- **Message Switching**
  - Gateway resource discovery
  - Routing table update and managing
  - Message routing(forwarding)
- **CA Control Component**
  - Target network selection and interface control and monitoring
  - CoAP message serialization and parsing
  - Blockwise messaging flow control
- **Transport Adapter Component**
  - Data transmission over UDP, TCP, BLE(GATT), BT(SPP) and NFC
  - Secure data exchanging using DTLS
- **Platform Adapter Component**
  - Ubuntu Wifi, Ethernet and BLE
  - Android Wifi, BLE and BT
  - Tizen Wifi, BLE and BT
  - Arduino Wifi, Ethernet and BLE

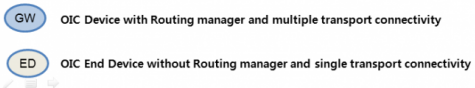


Routing Through Heterogeneous Connectivity

- **If any intermediary gateway device has multiple connectivity (ex. IP and BT), it can forward requests or responses so that resources on different transports can be discovered and communicated.**





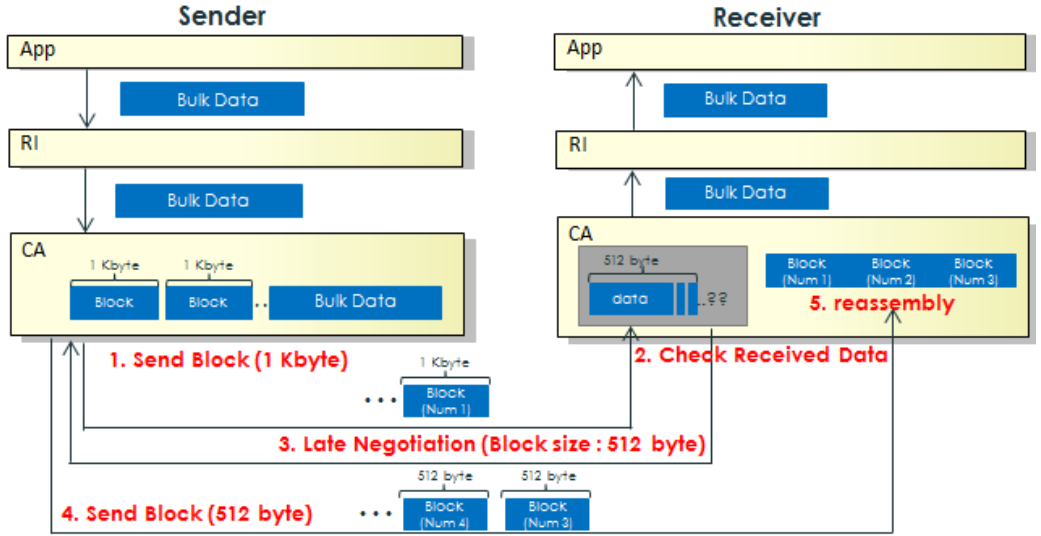


ROUTING=GW compile option

ROUTING=EP compile option

Blockwise Transfer

- **OIC client and OIC server can send/receive the large size messages by means of transferring of small block unit**



Start from Basic API Example

- **OCPlatform::findResource (const std::string& host**
  const std::string& resourceName
  OCConnectivityType connectivityType
  FindCallback resourceHandler)

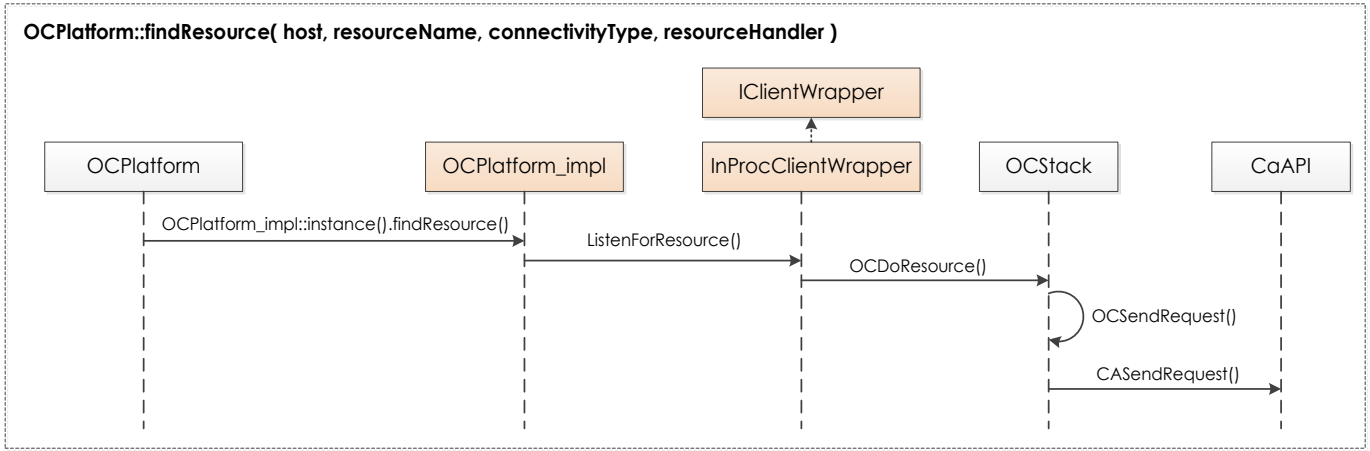
```
Example: simpleClient.cpp
...                                  "/oic/res" = all OIC devices that are discoverable

requestURI << OC_RSRVD_WELL_KNOWN_URI;

OCPlatform::findResource("", requestURI.str(),
             CT_DEFAULT, &foundResource);

...
```

→ API for Service and Resource discovery (client side only)

host == null: multicast resource discovery query

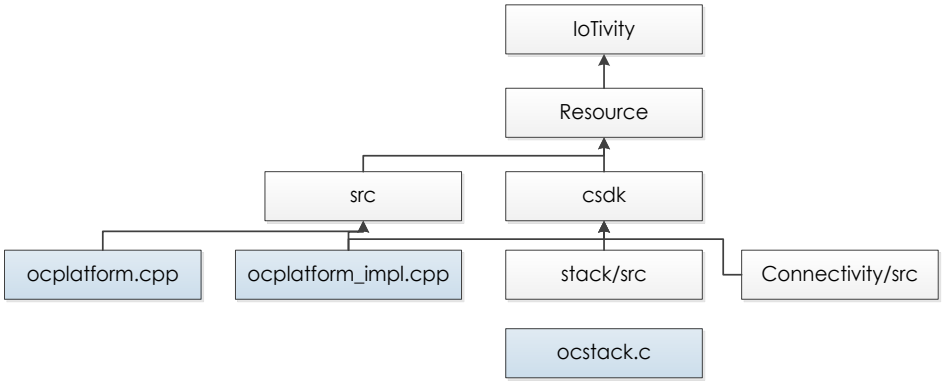
Call Path

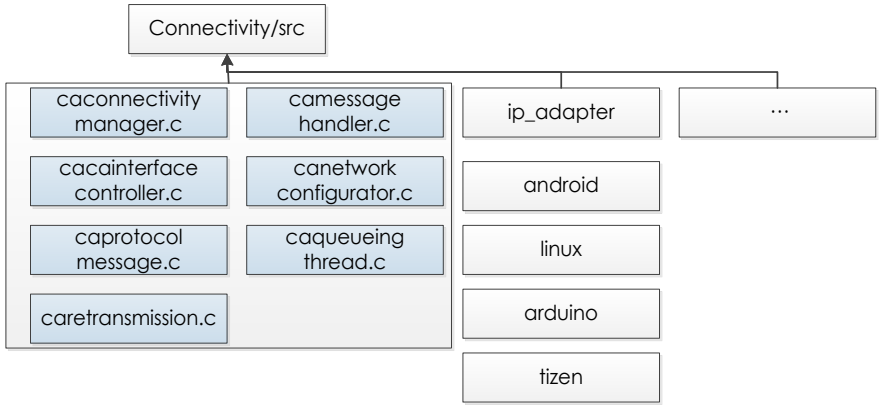
- **OCPlatform → OCPlatform_impl → InProcClientWrapper → OCStack → CaAPI**
  - Same as other Restful APIs' call path; POST, GET, PUT, OBSERVE



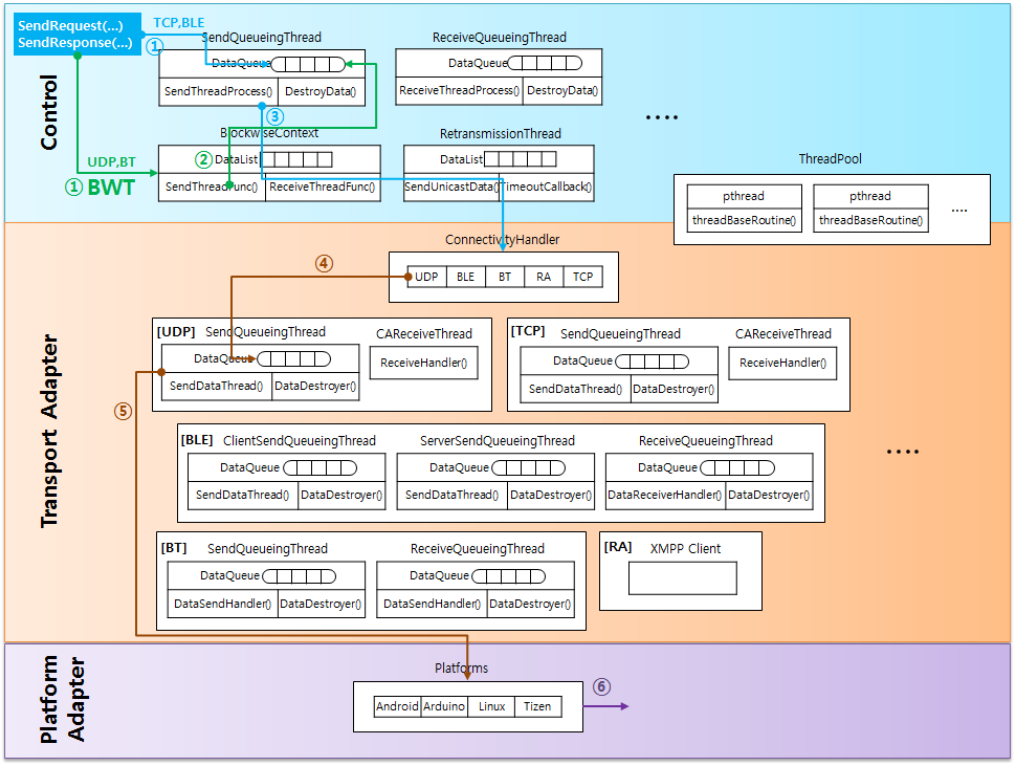
Source Code Hierarchy

- iotivity/resource/csdk/connectivity/src
- Routing manager source code is under the iotivity/resource/csdk/routing





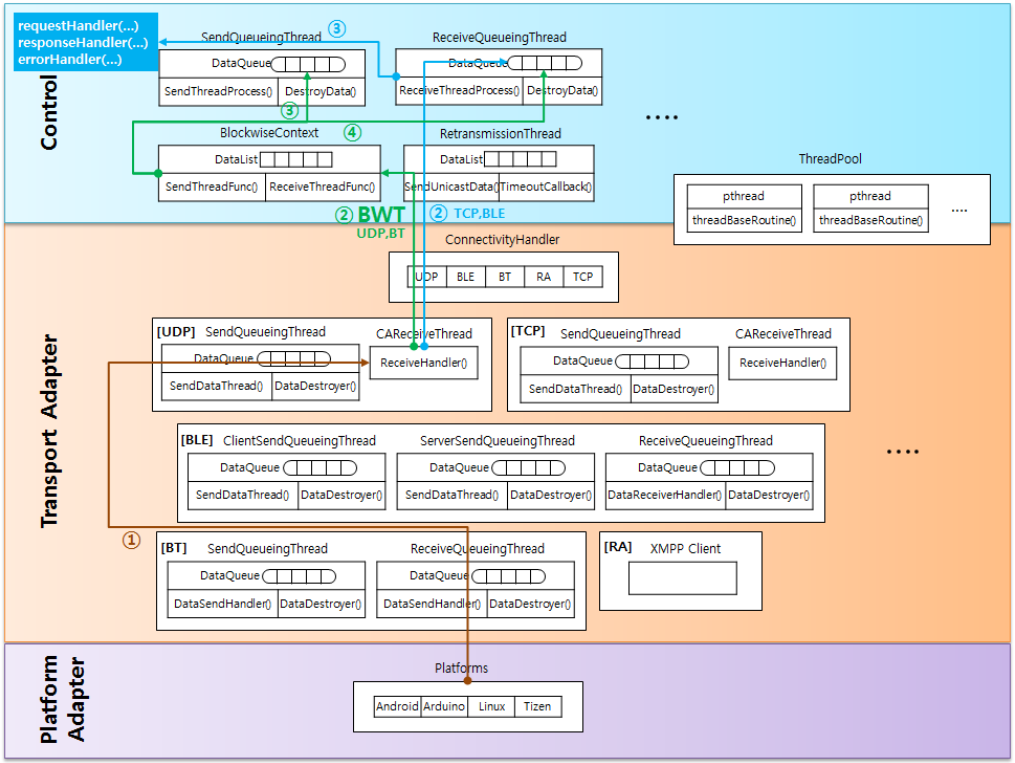
Call Path: Sending Data



1. Send requests are processed by Block-wise transfer(BWT) basically in case of UDP and BT
2. BWT prepares atomic data with default size (1KB) of block data and sends it to SendQueue thread
3. SendQueueThread sends data to handler of interested transport
4. ~ 6. In case of UDP, SendQueueThread for UDP sends data to endpoint

2 Types of SendQueue: Common | Adapter

Call Path: Receiving Data



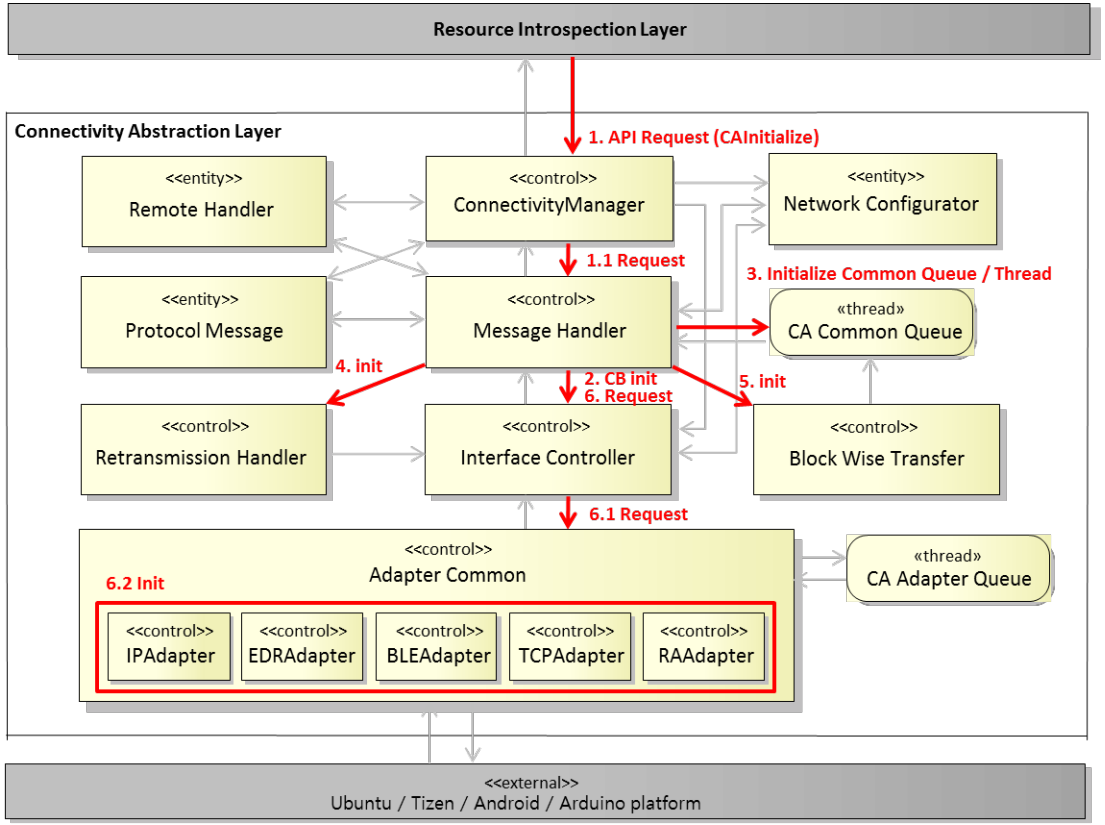
1. In case of UDP, ReceiveThread for UDP receives data from endpoint
2. In case that interested transport is UDP or BT, BWT prepares next atomic data of block data
3. It is sent to SendQueueThread again
4. If received data is the last data for block in BWT, it is sent to ReceiveQueueThread and finally sent to upper layer

CA APIs


```
CAResult_t    CAInitialize();
void          CATerminate();
CAResult_t    CAStartListeningServer();
CAResult_t    CAStopListeningServer();
CAResult_t    CAStartDiscoveryServer();
void          CARegisterHandler( CARequestCallback       ReqHandler,
                                 CAResponseCallback      RespHandler,
                                 CAErrorCallback         ErrorHandler );
CAResult_t    CACreateEndpoint(  CATransportFlags_t      flags,
                                 CATransportAdapter_t    adapter,
                                 const char              *addr,
                                 uint16_t                port,
                                 CAEndpoint_t            **object );
void          CADestroyEndpoint(CAEndpoint_t *object);
CAResult_t    CAGenerateToken(CAToken_t *token, uint8_t tokenLength);
void          CADestroyToken(CAToken_t token);
CAResult_t    CASendRequest(const CAEndpoint_t *object, const CARequestInfo_t *requestInfo);
CAResult_t    CASendResponse(const CAEndpoint_t *object, const CAResponseInfo_t *responseInfo);
CAResult_t    CASelectNetwork(CATransportAdapter_t interestedNetwork);
CAResult_t    CAUnSelectNetwork(CATransportAdapter_t nonInterestedNetwork);
CAResult_t    CAGetNetworkInformation(CAEndpoint_t **info, uint32_t *size);
CAResult_t    CAHandleRequestResponse();
CAResult_t    CASetRAInfo(const CARAInfo_t *caraInfo);
```

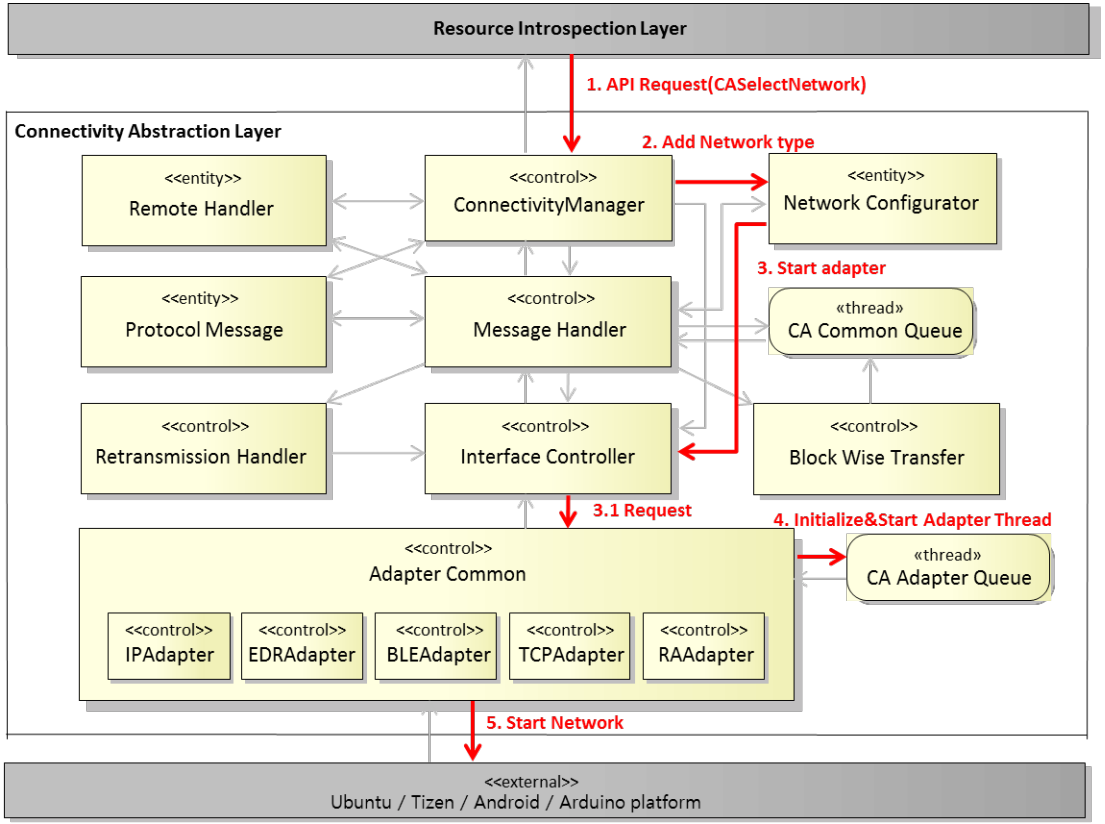
*input parameter
*output parameter

CA API – Initialize



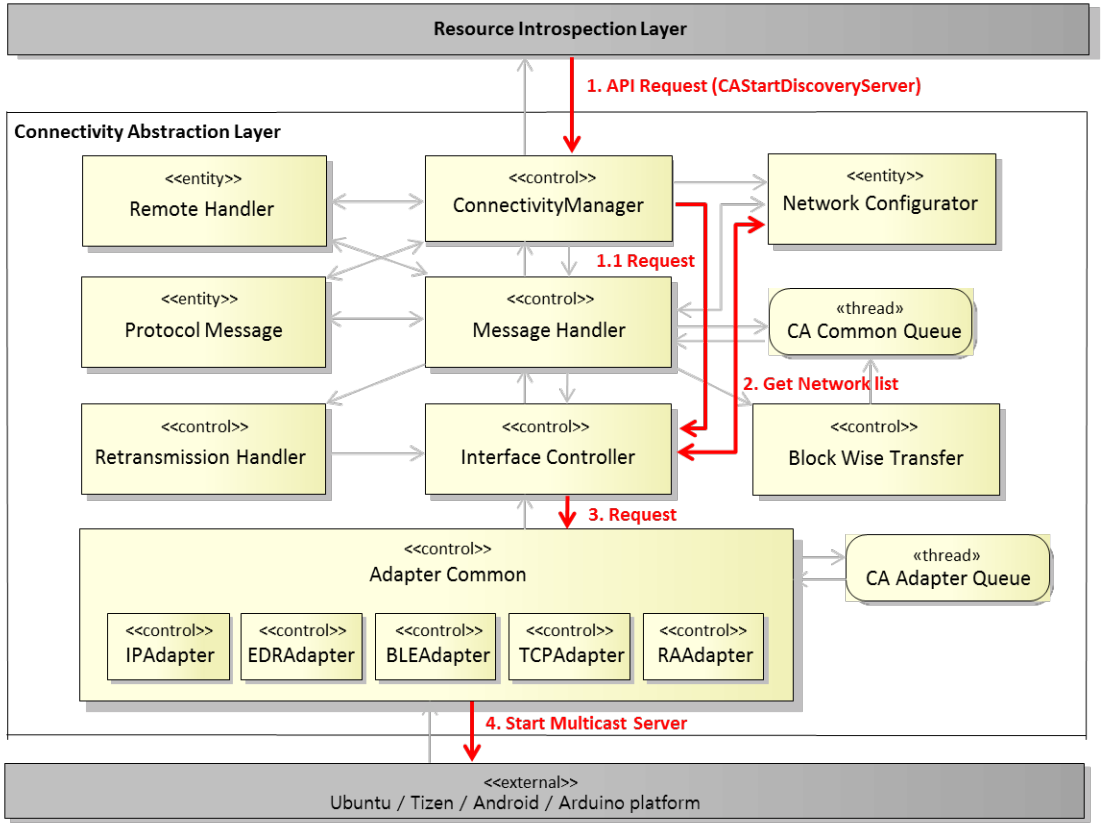
- **CAInitialize()**
  - Initialize the connectivity abstraction module
  - Initialize
    - adapters
    - common thread pool
    - other modules

CA API – Select Network



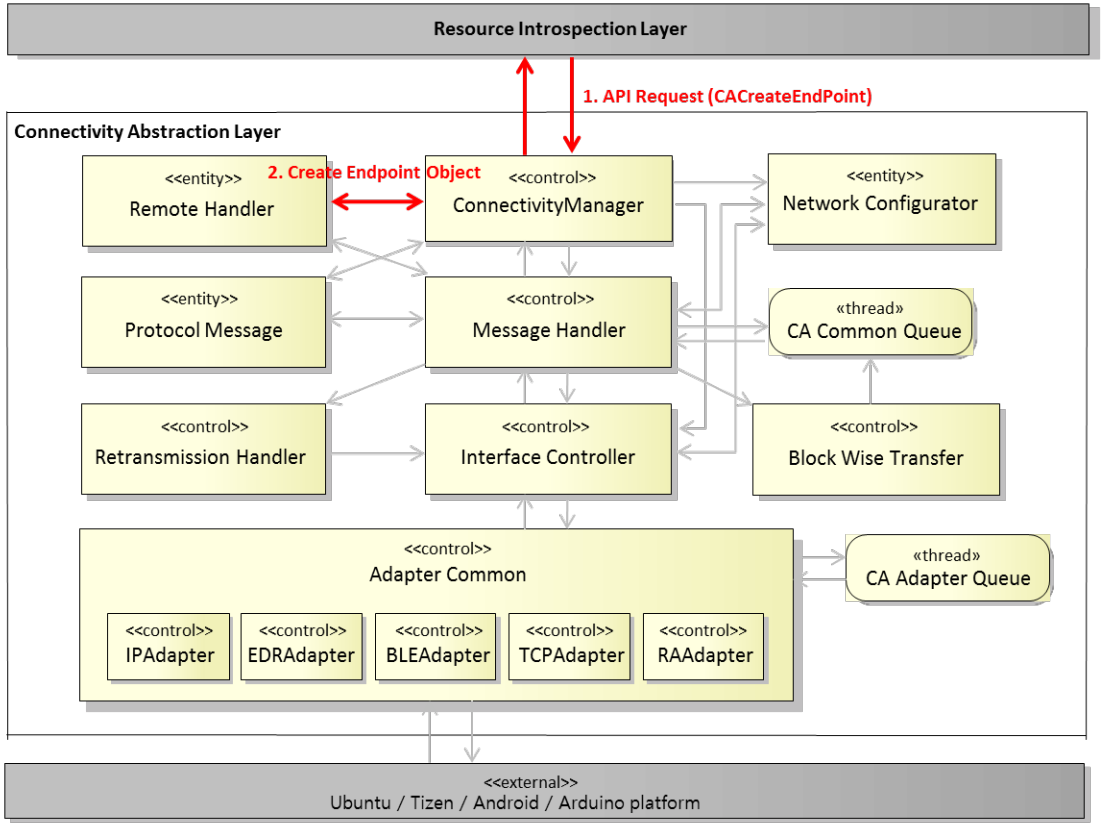
- **CASelectNetwork ()**
  - Select network to use

CA API – Start Server



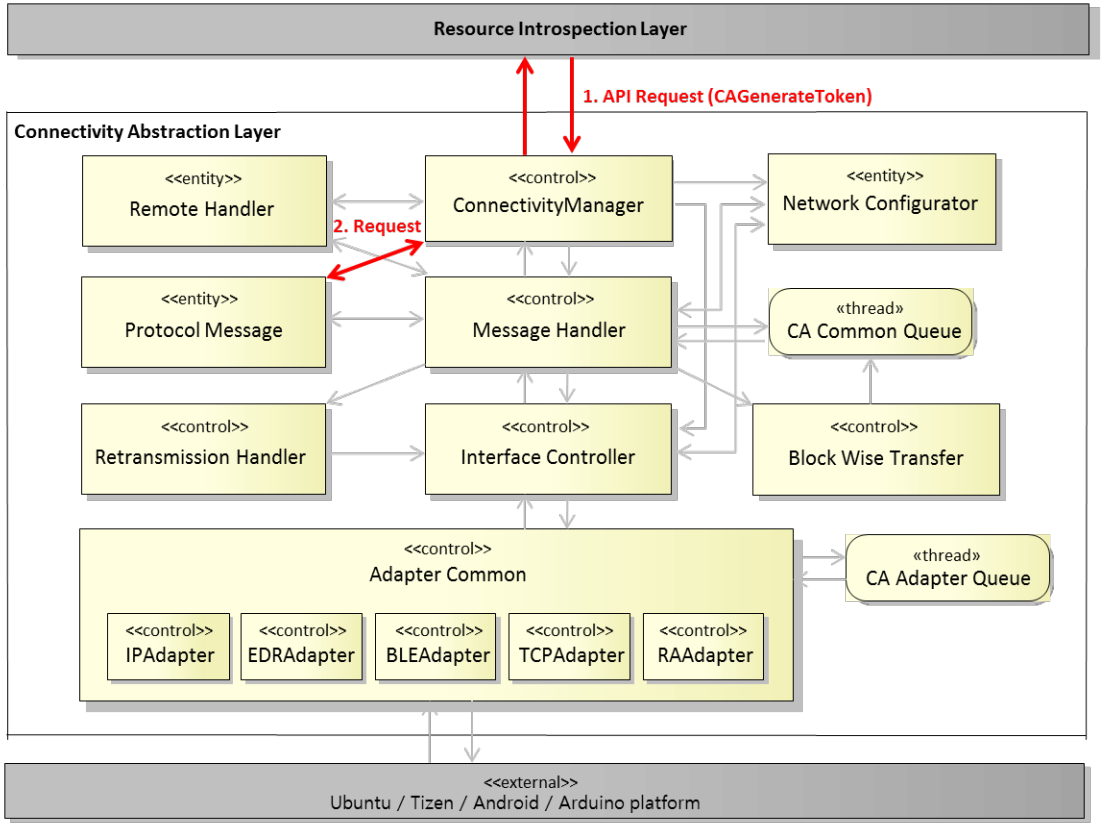
- **CAStartDiscoveryServer()**
  - Used by resource required clients for listening multicast requests
  - Based on the adapters configurations, different kinds of servers are started

CA API – Create Endpoint



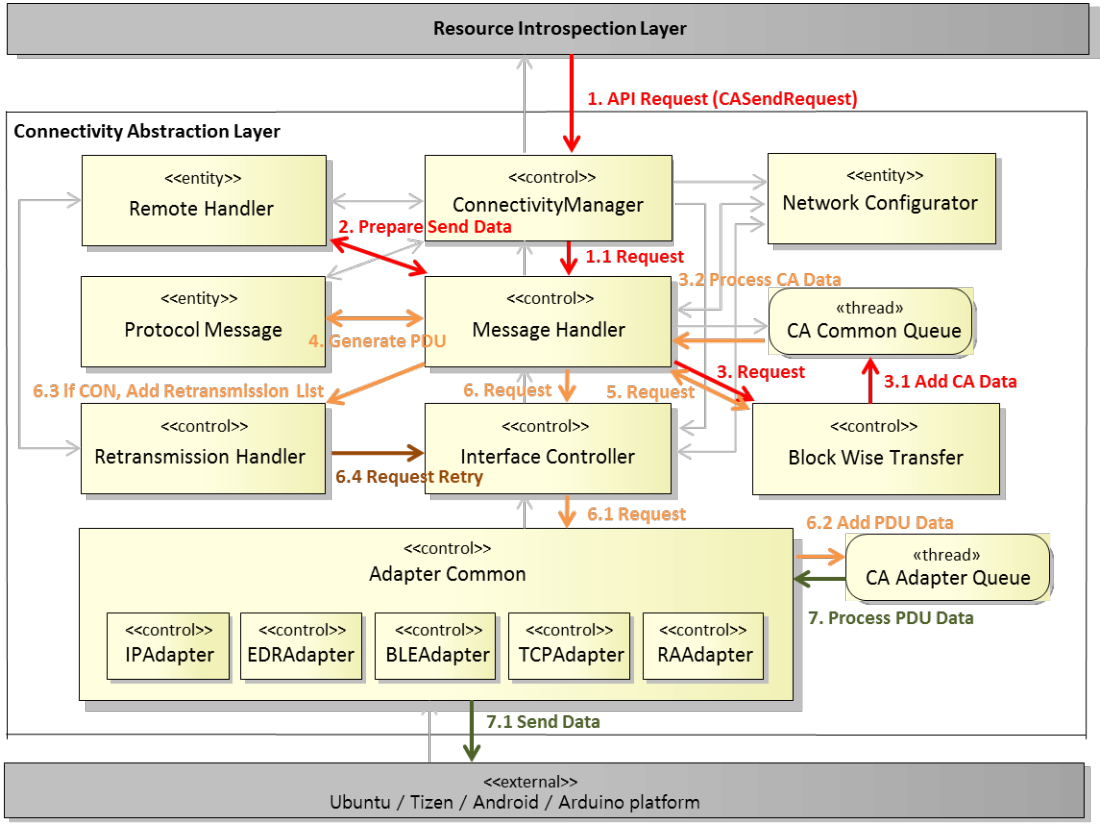
- **CACreateEndpoint()**
  - Create an endpoint description
  - Freed using CADestroyEndpoint()

CA API – Generate Token



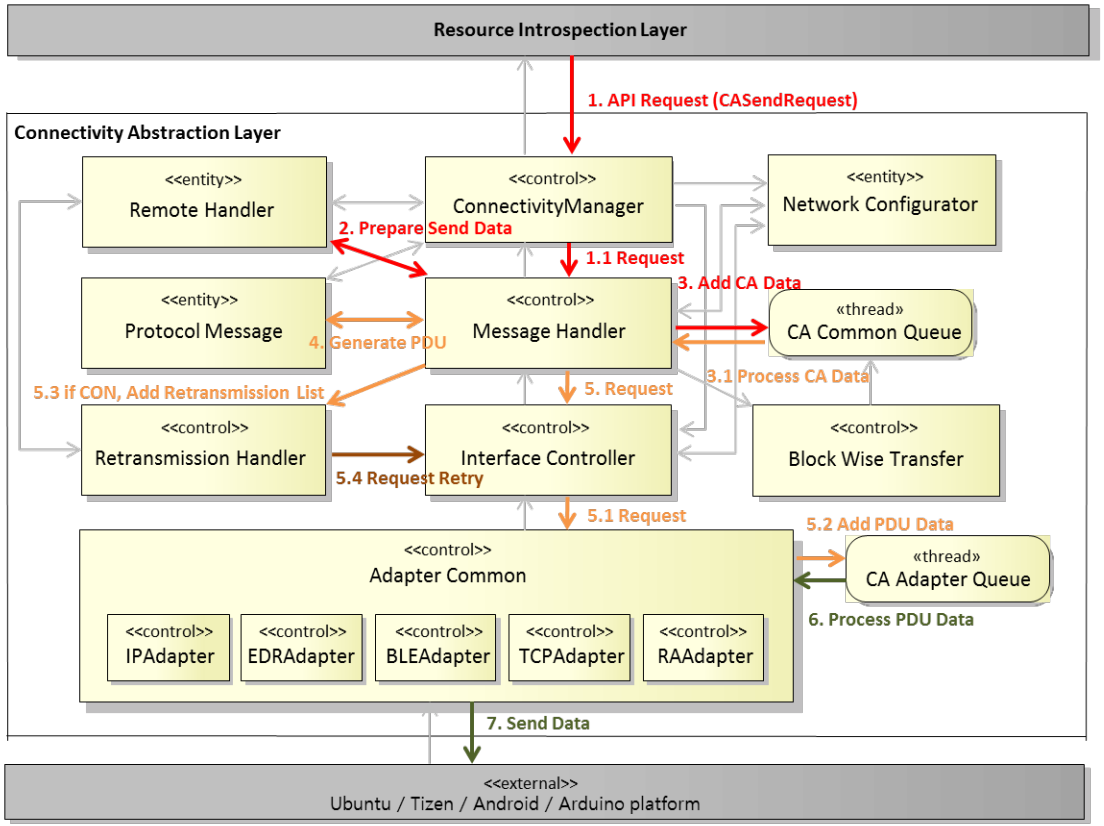
- **CAGenerateToken()**
  - Generating the token for matching the request and response

CA API – Send Request/Response



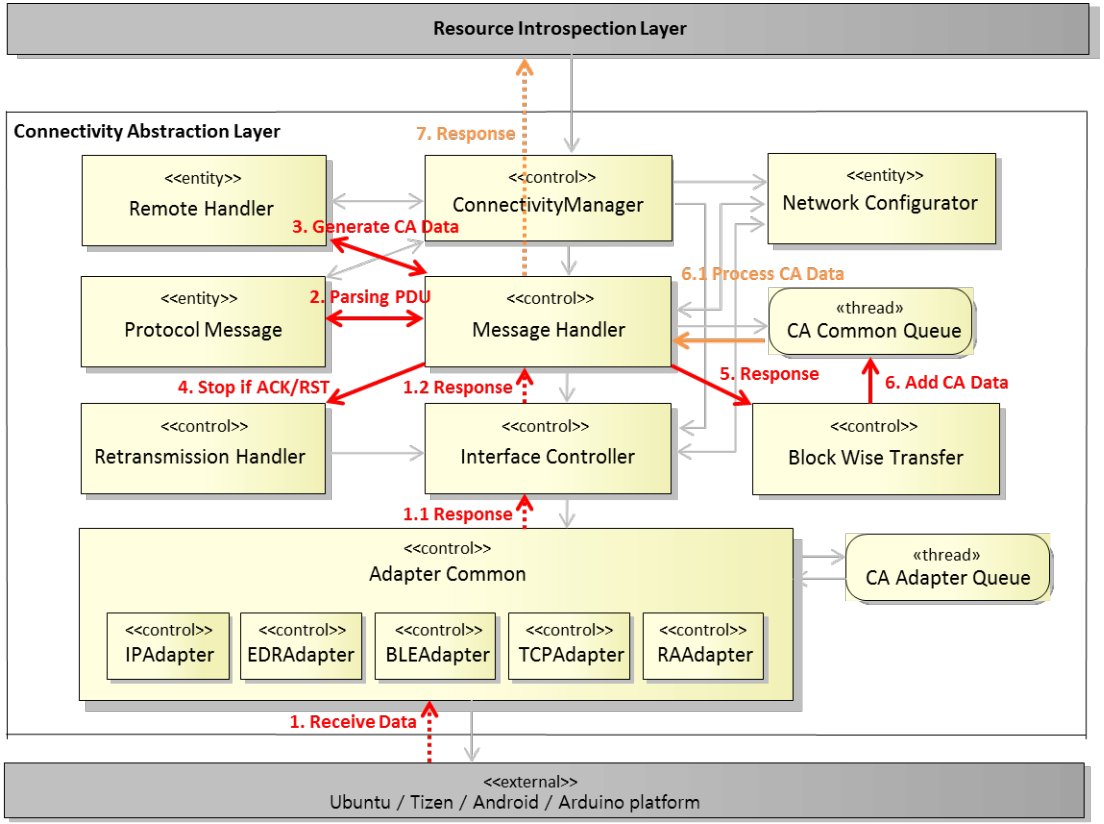
- **CASendRequest()**
  - Send control request on a resource
    - CA_GET/CA_POST/CA_PUT/CA_DELETE
2. CADetachSendMessage() for the interested transport(s)
3. BWT if IP, NFC, DEFAULT CASendThreadProcess() in common queue thread
4. CAGeneratePDU()
5. Check block options
6. CASendUnicastData() Retransmission if IP/RFCOMM/GATT
7. CASendIPUnicastData()

CA API – Send Request/Response if BLE, TCP



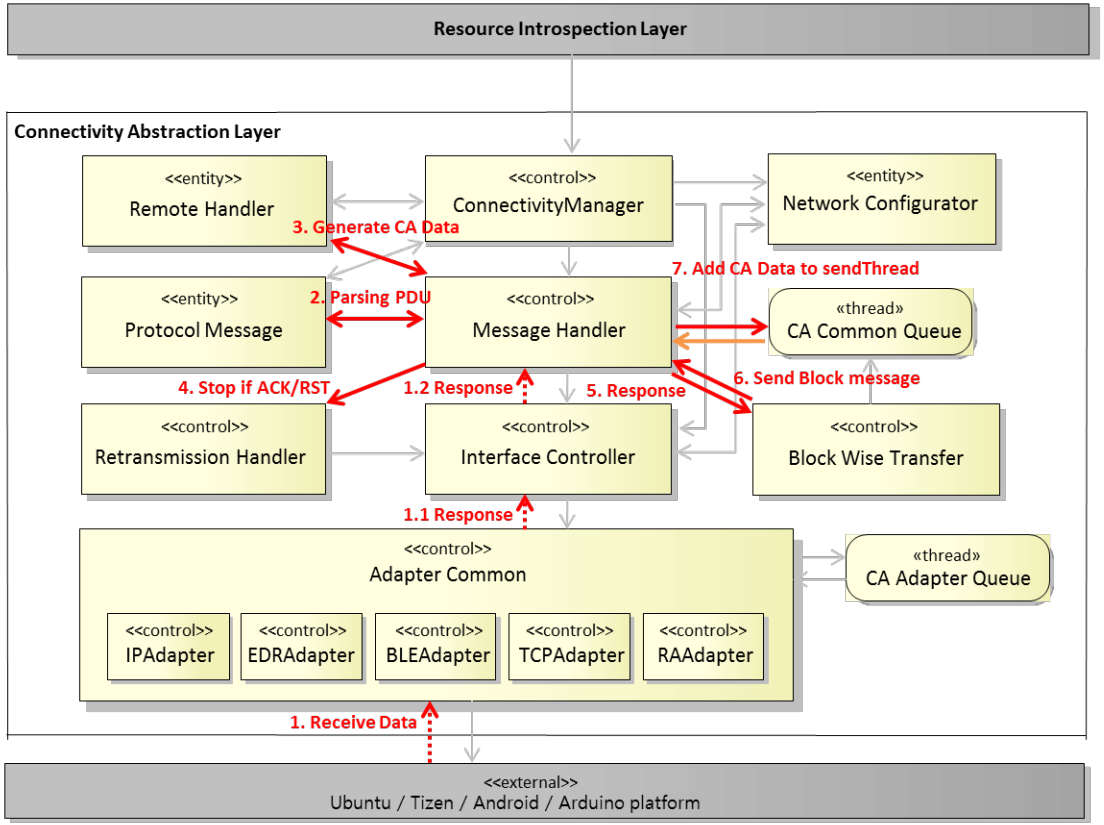
- **Not through BWT**

CA API - Receive Request/Response



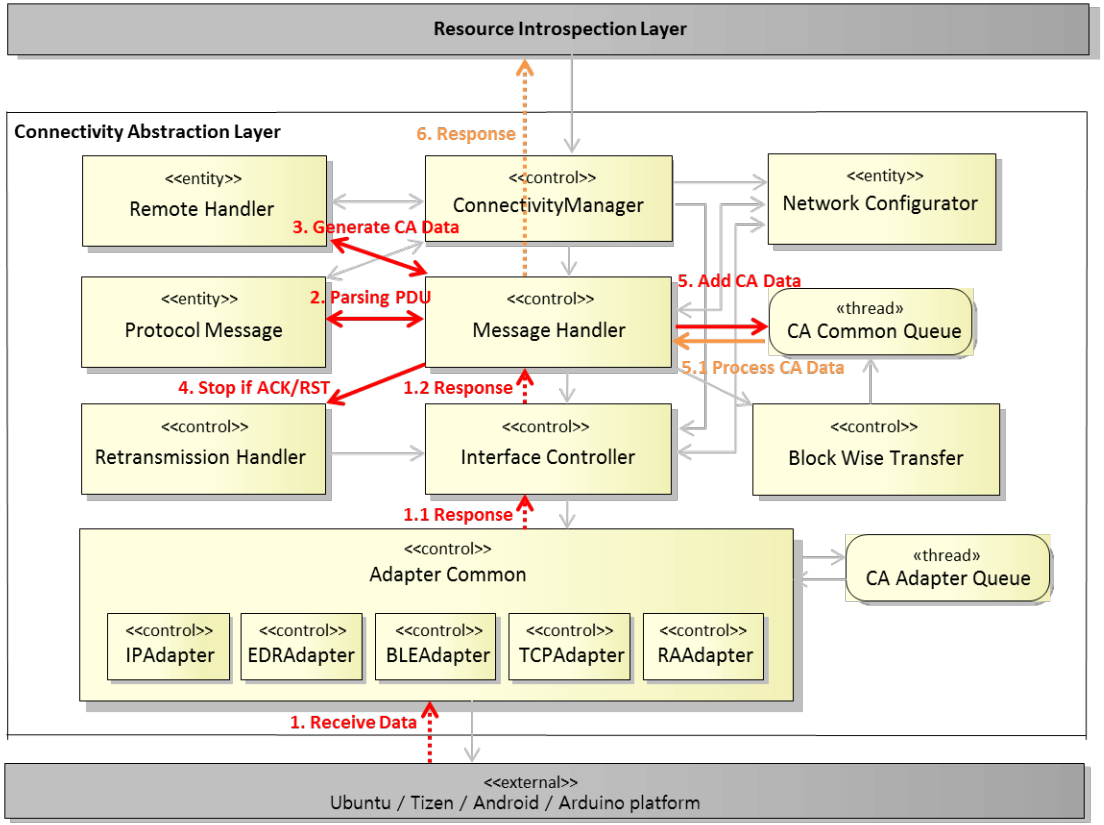
- **CAReceivePacketCallback()**
  2. CAParsePDU()
  3. CAGenerateHandlerData() Generate CA data to enqueue
  4. Stop if ACK/RST message has been received
  5. BWT
  6. Add CA data to queueing thread
     - CAReceiveThreadProcess()
  7. HandleCARequest() in OCStack

CA API – Receive Request/Response



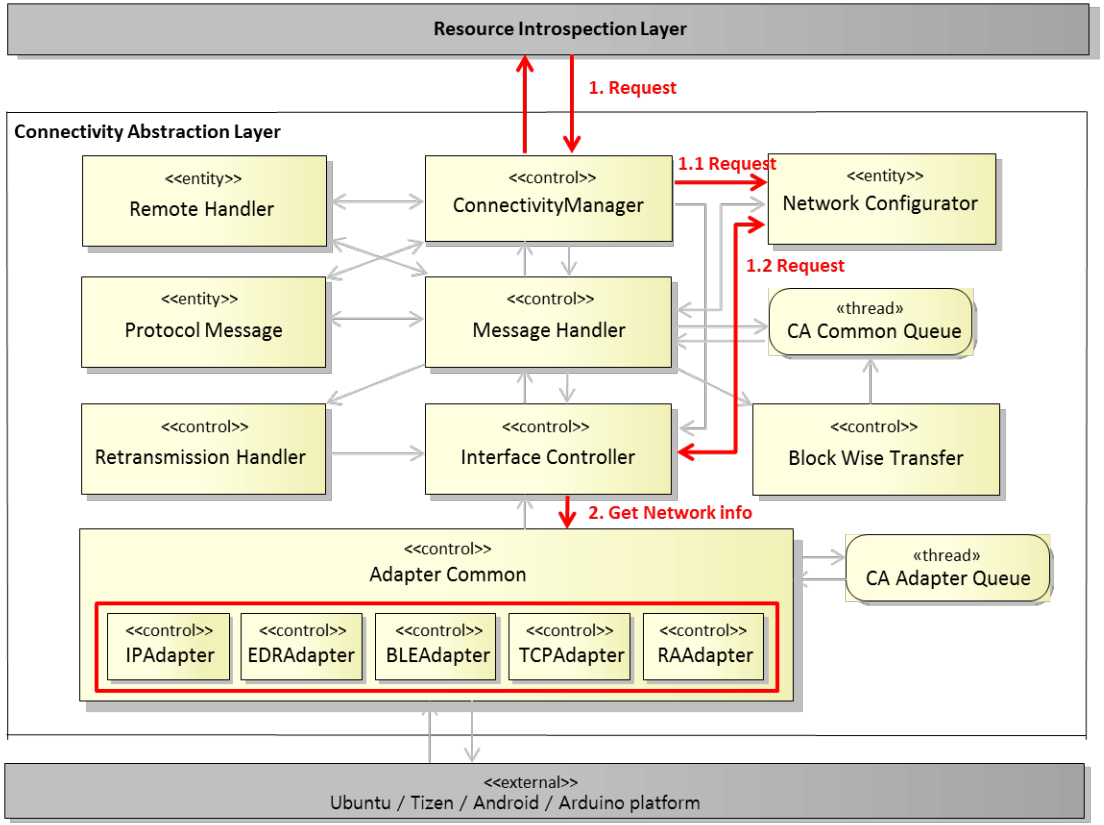
- **Not passing to upper layer, but just keep it in send queueing thread**

CA API – Receive Response if BLE, TCP



- **No BWT**

CA API – Get Network Information



- **CAGetNetworkInformation()**
  - Get network information