Tizen Telephony Stack

DongHoo Park

Samsung Linux Platform Group

Contents

Introduction

- **Why Tizen Telephony stack?**

  - Verified Open source telephony stack
    - It is a proven qualified stack with dominant modem chip vendor in industry
    - Applications of Tizen are already implemented on Tizen Telephony stack.
    - It already supports well-defined interface with Connman.

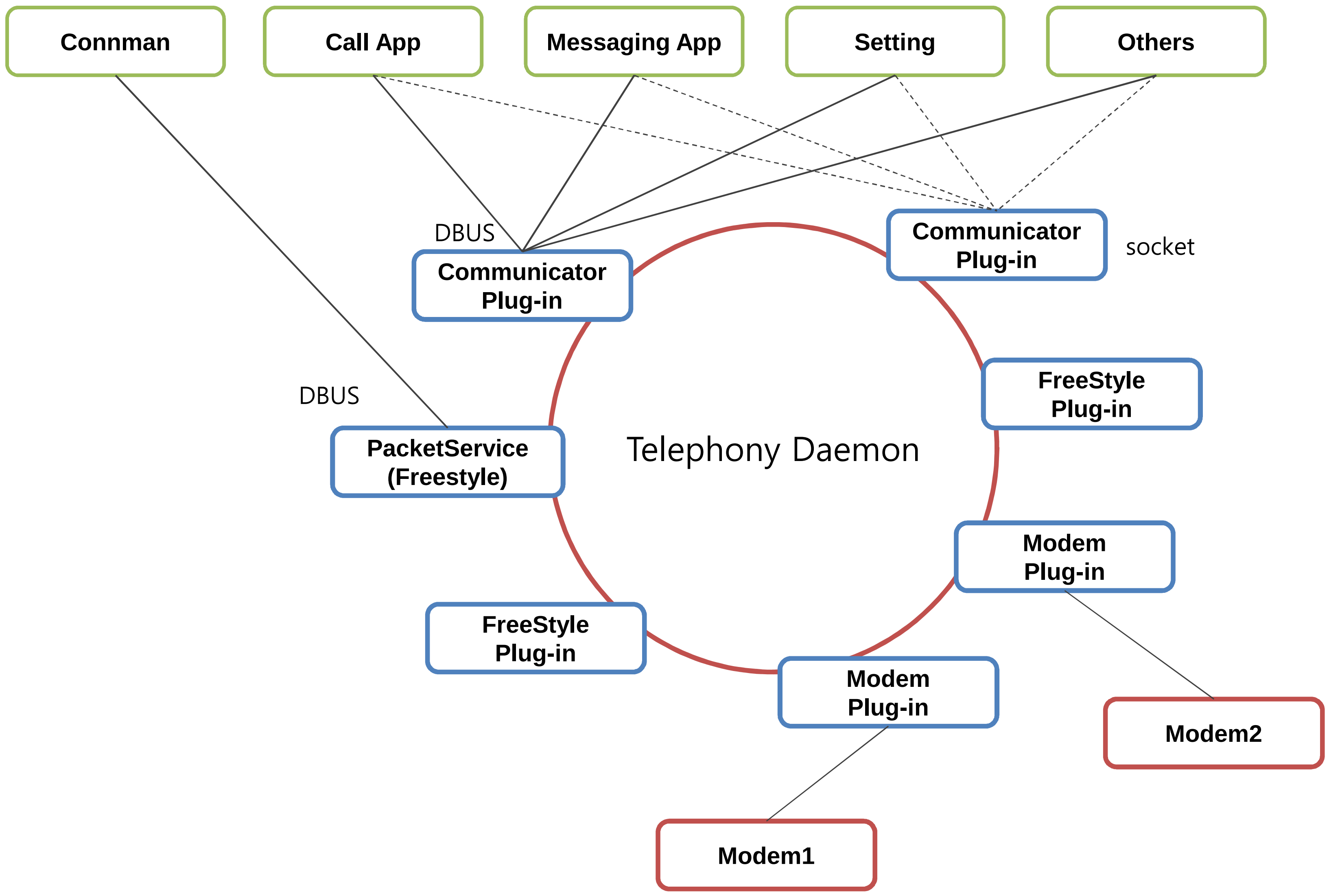
  - The benefits when commercialized
    - It supports flexible plug-in architecture so that manufacturer can customize from top to bottom.
      - Interface of application
      - Interface of modem
    - It has been updating so that it can be actually ready for commercialization start.
      - GCF, PTCRB certification
    - Manufacturer can make commercial product without license burden.
      - Various carrier requirements can be easily accommodated with plug-in and plug-in license can be managed by manufacturer decision.
      - Tizen Telephony stack has modular architecture that can be customized for any business area which needs telephony stack.

*GCF : Global Certificate Forum
*PTCRB : PSC Type Certification Review Board

Introduction

- **What makes it special!**
  - Rich Telecommunication functionalities
    - SIM, SIM Phonebook, SIM Application Toolkit
    - Network Registration, Voice/Video Call Service, Managing SMS
    - Packet Service
  - Tiny
    - Minimal API
    - Tiny Tizen Telephony core
  - Flexible for expanding and customizing
    - Modem Venders' modem interface
    - The differentiated services of Service Providers
    - The competitive functionalities of Manufactures
  - Easy to use
    - Do Not require the telephony background
    - Only focusing on the functionalities what application want to implement
  - License
    - Apache License Version 2.0

Architecture

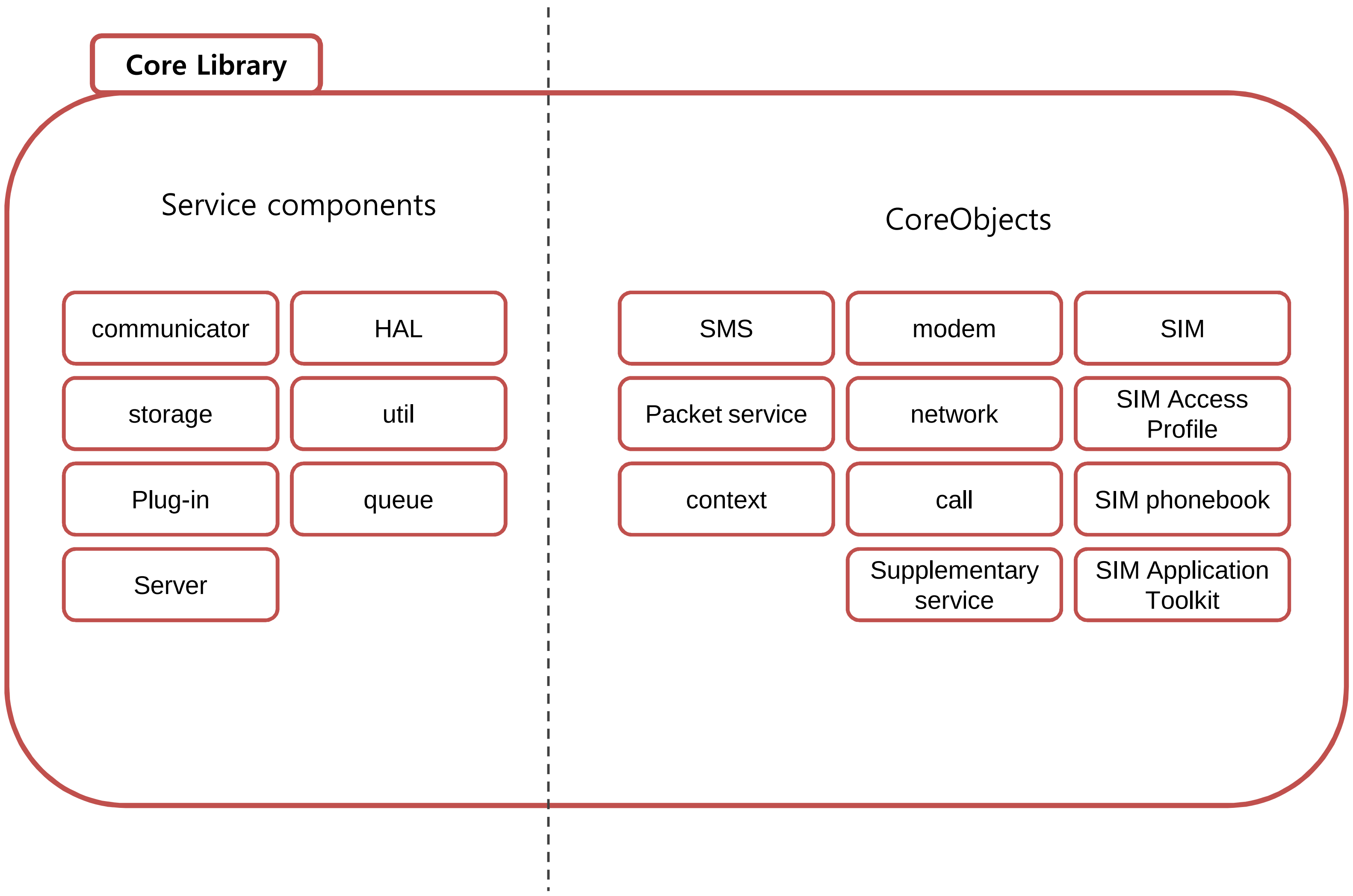


Tizen Telephony Components

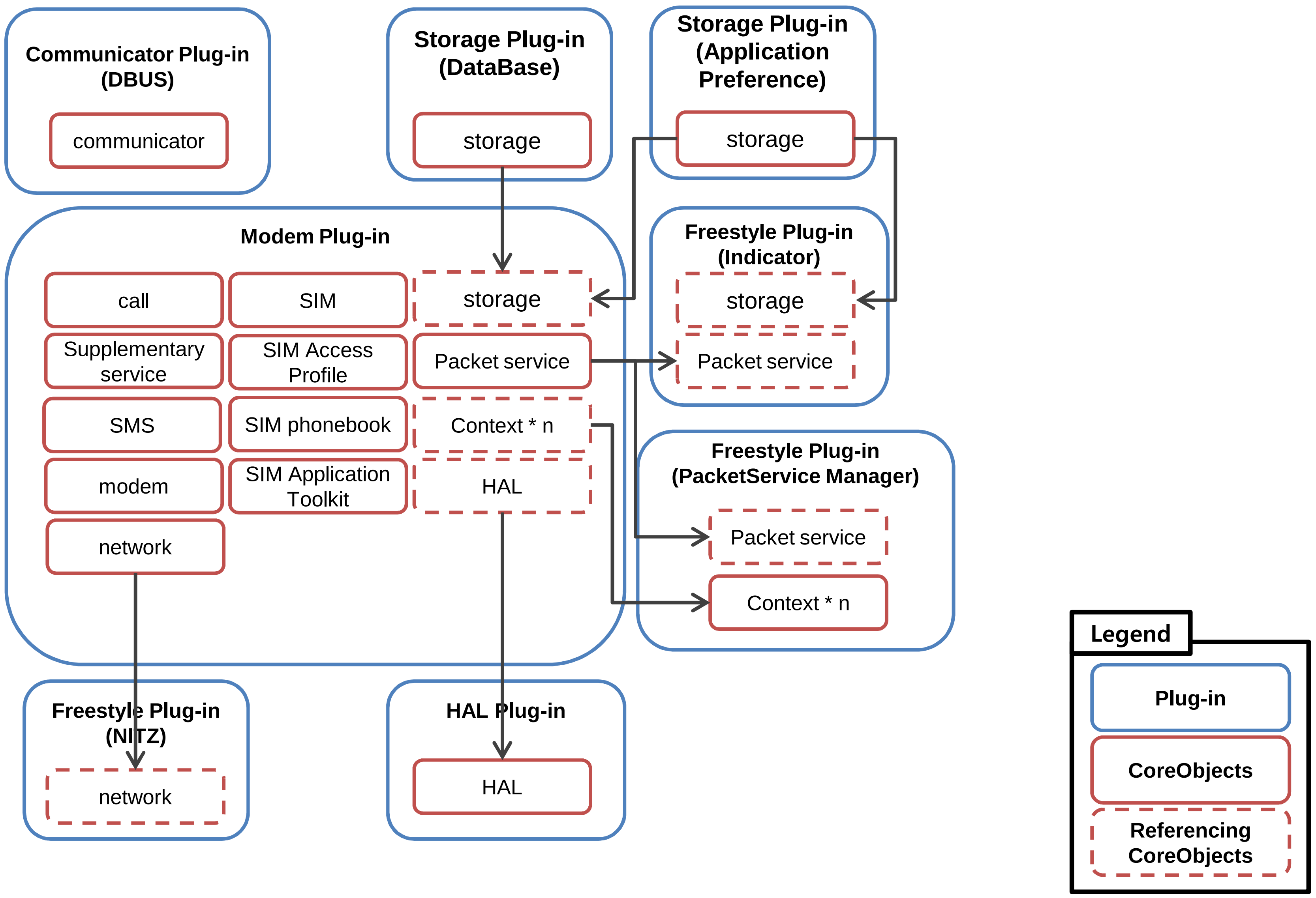
- **Core Library**
  - The base library for consisting Tizen Telephony
  - Service Components
    - Server, Plugin, Queue, HAL, Communicator, Storage, Util
  - Core Objects
    - The functional object
      - Modem, Network, Call, SS, SMS, PS, Context, SIM, SAP, SAT, SIM Phonebook
    - operation table
      - The functions of object are defined by operation table
    - private object
      - The data of objects are stored, and get/set APIs are provided
- **Plug-in**
  - Integrated service module
    - Communicator plug-in
      - Interaction between applications and Tizen Telephony stack
    - Modem plug-in
      - Processing requests/responses/notifications between AP and CP
    - Freestyle plug-in
      - Independently processing the tasks by a certain trigger
- **Daemon**
  - Dispatcher
    - Sending the requests/responses/notifications to a proper plug-in

*AP : Application Processor
*CP : Communication Processor

Core Library

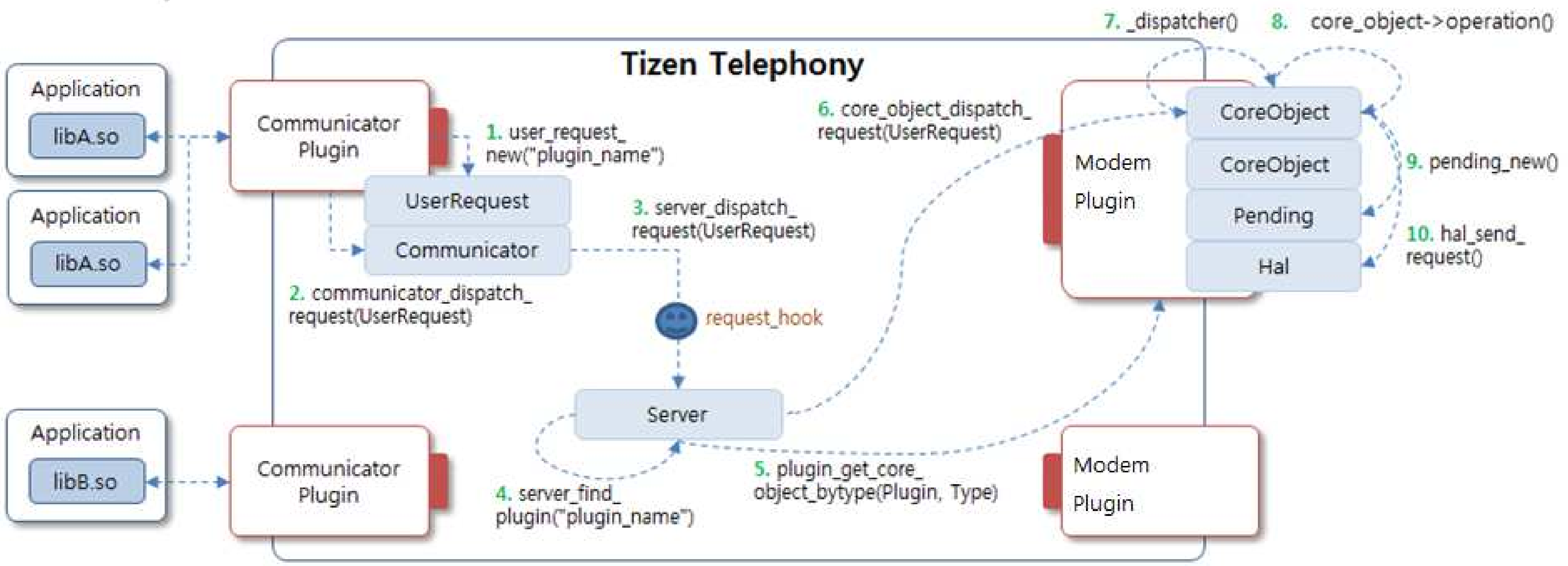


Plug-in



Communicator Plug-in (DBUS)
communicator
Storage Plug-in (DataBase)
storage
Storage Plug-in (Application Preference)
storage
Modem Plug-in
call
SIM
storage
Supplementary service
SIM Access Profile
Packet service
SMS
SIM phonebook
Context * n
modem
SIM Application Toolkit
HAL
network
Freestyle Plug-in (Indicator)
storage
Packet service
Freestyle Plug-in (PacketService Manager)
Packet service
Context * n
Freestyle Plug-in (NITZ)
network
HAL Plug-in
HAL
Legend
Plug-in
CoreObjects
Referencing CoreObjects

Work Flow



7. _dispatcher()
8. core_object->operation()
Tizen Telephony
Application
libA.so
Application
libA.so
Application
libB.so
Communicator Plugin
Communicator Plugin
1. user_request_new("plugin_name")
UserRequest
Communicator
2. communicator_dispatch_request(UserRequest)
3. server_dispatch_request(UserRequest)
request_hook
Server
4. server_find_plugin("plugin_name")
5. plugin_get_core_object_bytype(Plugin, Type)
6. core_object_dispatch_request(UserRequest)
Modem Plugin
CoreObject
CoreObject
Pending
Hal
9. pending_new()
10. hal_send_request()
Modem Plugin

Developing plug-in

- **Set the plug-in description**
  - It should be in any plug-in
  - The symbol table for dynamic loading
  - Defines the name, priority, version and load, init, unload action

Plugin description

```
struct plugin_define_desc_t  {
    gchar *name;
    enum plugin_priority_e priority;
    int version;
    gboolean (*load)();
    gboolean (*init)(TcorePlugin *);
    void (*unload)(TcorePlugin *);
};
enum plugin_priority_e {
    PLUGIN_PRIORITY_HIGH = -100,
    PLUGIN_PRIORITY_MID = 0,
    PLUGIN_PRIORITY_LOW = +100
};
```

- **Communicator Plug-in**
  - Set the operation table
    - Response from modem plug-in
    - Notification from modem plug-in
  - Create the communicator object.
  - It can create own data structure.

communicator plugin

```
struct communitor_operations_t ops = {
    .send_response = send_response,
    .send_notification = send_notification,
};

static gboolean on_init(TcorePlugin *p)
{
    Communicator *comm;
    comm = communicator_new(p, &ops);
    …
    return TRUE;
}
```

Developing plug-in

- **HAL Plug-in**
  - Create the data channel to modem
  - Naming the certain modem for other plugins
- **Modem Plug-in**
  - Find the HAL for interacting physical modem
  - Initialize the core objects
    - Core objects' operation table has to be set
- **Free-Style Plug-in**
  - Just make the code what you want

HAL plugin

```
static struct hal_operations_t hops = {
    .power = hal_power,
    .send = hal_send,
};

static gboolean on_init(TcorePlugin *p) {
     TelephonyHal *h;

    /* Create MODEM TX/RX Channel */
    h = hal_new(p, "dpram", &hops);

    return TRUE;
}
```

modem plugin

```
static gboolean on_init(TcorePlugin *p)
{
    TelephonyHal *h;
    h = tcore_server_find_hal(p, "dpram");
    initialize the core objects which will be using
    …
    return TRUE;
}
```

freestyle plugin

```
static gboolean on_init(TcorePlugin *p)
{
    …
    return TRUE;
}
```

Further work

- **Provides various communicator**
  - Developing the communicators for supporting various application interface
    - DBUS, Socket and others
- **Support Feature**
  - Concept
    - Dual SIM/Dual Stand by
  - Packet Service
    - LTE
    - IPv6
  - SIM Application Toolkit
    - BIP (Bearer Independent Protocol)

Summary

- **Tizen official telephony stack**
  - Will be included in Tizen 1.0
- **Telecommunication functionality are fully supported.**
- **Tizen Telephony stack is designed for accommodating customization.**
  - Working with modem vendors' specific interface
    - Modem plug-in should be added.
  - Adding the carrier specific features without public
    - It can be any plug-in such as freestyle, communicator, modem and other plug-ins
  - Customizing any plug-ins for applying manufacturers' know-how
    - All plug-ins can be intentionally modified or replaced.
- **It will be fully kept compatibility.**
- **Apache license**

Thank You.