

VoWifi Telephony Design Document

Document Number:		Document Version:	
Owner:	Lei.Huang	Date:	2015-12-18
Document Type:			
NOTE:	<p>ALL MATERIALS INCLUDED HEREIN ARE COPYRIGHTED AND CONFIDENTIAL UNLESS OTHERWISE INDICATED. The information is intended only for the person or entity to which it is addressed and may contain confidential and/or privileged material. Any review, retransmission, dissemination, or other use of or taking of any action in reliance upon this information by persons or entities other than the intended recipient is prohibited.</p> <p>This document is subject to change without notice. Please verify that your company has the most recent specification.</p> <p>Copyright © 2013 Spreadtrum Communications Inc.</p>		

Revision History

Revision	Date	Author	Description
1.0	2015-12-18	Lei.huang	draft

Table of Contents

1. Introduction	5
1.1. Request & Purpose	5
1.2. Definitions & Abbreviations	5
2. Design Overview	5
2.1. IMS Telephony Architecture	5
3. Interfaces Design	7
3.1. AOSP Interface	7
3.2. SPRD Interface	10
4. Functional Modules & Flowchart	12
4.1. IMS stack enable / disable	12
4.1.1. Feature description	12
4.1.2. IMS Object Init:	13
4.1.3. IMS stack enable/disable:	13
4.2. IMS Voice/Video Call	14
4.2.1. Feature description	14
4.2.2. MO Call	15
4.2.1. MT Call	16
4.3. Supplementary Service	16
4.3.1. Feature description	16
4.4. Switch to VoLTE (No Call)	18
4.5. Switch to VoWiFi (No Call)	19
4.6. VoWiFi to VoLTE (Calling)	20
4.7. VoLTE to VoWiFi (Calling)	21
5. AT Commands	21

Spreadtrum Confidential

1. Introduction

1.1. Request & Purpose

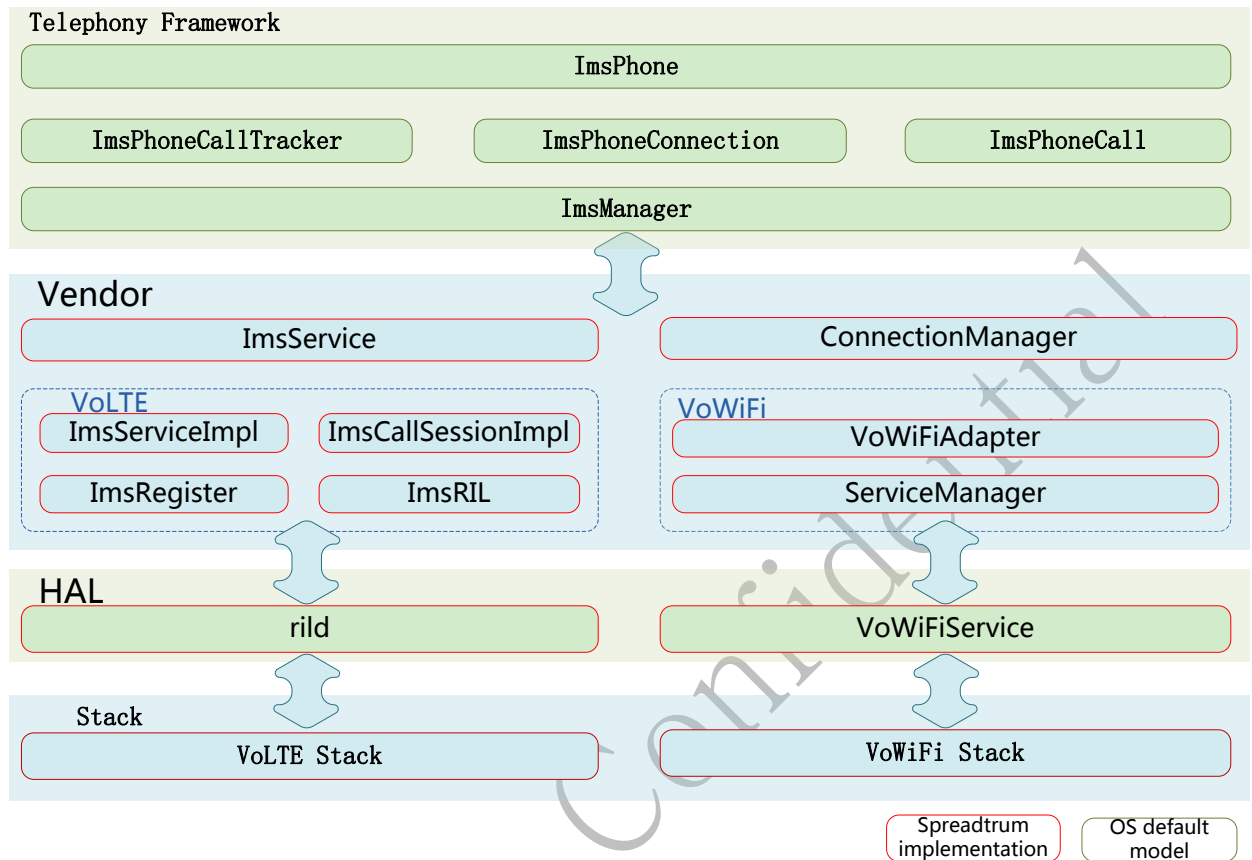
1.2. Definitions & Abbreviations

IMS	IP Multimedia Subsystem
VoWifi	Voice over WiFi
VoLTE	Voice over LTE
AP	Application Processer
VT	Video Telephony
AOSP	Android Open-Source Project

2. Design Overview

2.1. IMS Telephony Architecture

IMS Telephony Architecture :



Implementation notes :

- The IMS code of Spreadtrum implemented as an APP, use Android.UID.phone to run in the com.android.phone processes.
- ImsManager provide the interface with ImsService.
- ConnectionManager is the initiator of IMS handover, ConnectionManager will listen to the WIFI and LTE signals to decide when to initiate the handover.
- ImsService provide the interface with IMS (Both VoLTE and VoWiFi). ImsService also implement the IMS handover logic.

- ✚ lmsServiceImpl implement VoLTE core logic , such as APN configuration ,URI configuration, maintain VoLTE call session, send request to RIL, and receive response from RIL.
- ✚ VoWiFiService implement VoWiFi core logic, such as APN configuration ,URI configuration , maintain VoWiFi call session, send request to VoWiFi stack, and receive response from VoWiFi stack.

3. Interfaces Design

3.1. AOSP Interface

Class	Interface	Comments
TelephonyManager	getIccSimChallengeResponse (int subId, int appType, String data)	Returns the response of SIM Authentication through RIL
	getSimOperatorNumericForPhone (int phoneId)	Returns the MCC+MNC of the provider of the SIM
	getNetworkOperatorForPhone	Returns the numeric name

	(int phoneId)	(MCC+MNC) of current registered operator
PhoneStateListener	onSignalStrengthsChanged (SignalStrength signalStrength)	Callback invoked when network signal strengths changes
	onVoLteServiceStateChanged (VoLteServiceState stateInfo)	Callback invoked when the VoLTE service has changed
	onCallStateChanged (int state, String incomingNumber)	Callback invoked when device call state changes
Phone	getSignalStrength()	Get current signal strength.

```

/**
 * Returns the response of SIM Authentication through RIL.
 * Returns null if the Authentication hasn't been successful
 * @param subId subscription ID to be queried
 * @param appType ICC application type (@see com.android.internal.telephony.PhoneConstants#APPTYPE_XXX)
 * @param data authentication challenge data
 * @return the response of SIM Authentication, or null if not available
 * @hide
 */
public String getIccSimChallengeResponse(int subId, int appType, String data) {
    try {
        IPhoneSubInfo info = getSubscriberInfo();
        if (info == null)
            return null;
        return info.getIccSimChallengeResponse(subId, appType, data);
    } catch (RemoteException ex) {
        return null;
    } catch (NullPointerException ex) {
        // This could happen before phone starts
        return null;
    }
}

/**
 * Returns the numeric name (MCC+MNC) of current registered operator
 * for a particular subscription.
 * <p>
 * Availability: Only when user is registered to a network. Result may be
 * unreliable on CDMA networks (use {@link #getPhoneType()} to determine if
 * on a CDMA network).
 *
 * @param phoneId
 * @hide
 */
public String getNetworkOperatorForPhone(int phoneId) {
    return getTelephonyProperty(phoneId, TelephonyProperties.PROPERTY_OPERATOR_NUMERIC, "");
}

```



```

/**
 * Get current signal strength. No change notification available on this
 * interface. Use <code>PhoneStateNotifier</code> or an equivalent.
 * An ASU is 0-31 or -1 if unknown (for GSM, dBm = -113 - 2 * asu).
 * The following special values are defined:</p>
 * <ul><li>0 means "-113 dBm or less".</li>
 * <li>31 means "-51 dBm or greater".</li></ul>
 *
 * @return Current signal strength as SignalStrength
 */
SignalStrength getSignalStrength();

/**
 * Returns the MCC+MNC (mobile country code + mobile network code) of the
 * provider of the SIM for a particular subscription. 5 or 6 decimal digits.
 * <p>
 *
 * @param phoneId for which SimOperator is returned
 * @hide
 */
public String getSimOperatorNumericForPhone(int phoneId) {
    return getTelephonyProperty(phoneId,
        TelephonyProperties.PROPERTY_ICC_OPERATOR_NUMERIC, "");
}

/**
 * Callback invoked when device call state changes.
 * @param state call state
 * @param incomingNumber incoming call phone number. If application does not have
 * {@link android.Manifest.permission#READ_PHONE_STATE READ_PHONE_STATE} permission, an empty
 * string will be passed as an argument.
 *
 * @see TelephonyManager#CALL_STATE_IDLE
 * @see TelephonyManager#CALL_STATE_RINGING
 * @see TelephonyManager#CALL_STATE_OFFHOOK
 */
public void onCallStateChanged(int state, String incomingNumber) {
    // default implementation empty
}

/**
 * Callback invoked when network signal strengths changes.
 *
 * @see ServiceState#STATE_EMERGENCY_ONLY
 * @see ServiceState#STATE_IN_SERVICE
 * @see ServiceState#STATE_OUT_OF_SERVICE
 * @see ServiceState#STATE_POWER_OFF
 */
public void onSignalStrengthsChanged(SignalStrength signalStrength) {
    // default implementation empty
}

```

```

/**
 * Callback invoked when the service state of LTE network
 * related to the VoLTE service has changed.
 * @param stateInfo is the current LTE network information
 * @hide
 */
public void onVoLteServiceStateChanged(VoLteServiceState stateInfo) {
}

```

3.2. SPRD Interface

AIDL	Interface	Comments
IImServiceEx	switchImServiceFeature(int type)	Used for switch IMS feature.
	startHandover(int targetType)	Used for start IMS handover.
	notifyNetworkUnavailable()	Used for notify network unavailable
	getCurrentImServiceFeature()	Used for get IMS feature
	setImServiceListener(IImServiceListenerEx listener)	Used for set IMS service listener.
IImServiceListenerEx	operationSucceeded(int id)	Notifies the result of operation.
	operationFailed(int id)	

Get the IImServiceEx Object :

```
IBinder exBinder = ServiceManager.getService("ims_ex");
```

```
IImServiceEx IImServiceEx = IImServiceEx.Stub.asInterface(exBinder);
```

```

package com.android.ims.internal;

/**
 * {@hide}
 */
interface IImServiceListenerEx {
    /**
     * Notifies the result of operation.
     * @param id: request id
     */
    void operationSucceeded(int id);
    void operationFailed(int id);
}

```

```
interface IImServiceEx {
    /**
     * Used for switch IMS feature.
     * @param type:
     * ImsConfig.FeatureConstants.FEATURE_TYPE_VOICE_OVER_LTE = 0;
     * ImsConfig.FeatureConstants.FEATURE_TYPE_VOICE_OVER_WIFI = 2;
     * @return: request id
     */
    int switchImsFeature(int type);

    /**
     * Used for start IMS handover.
     * @param targetType:
     * ImsConfig.FeatureConstants.FEATURE_TYPE_VOICE_OVER_LTE = 0;
     * ImsConfig.FeatureConstants.FEATURE_TYPE_VOICE_OVER_WIFI = 2;
     * @return: request id
     */
    int startHandover(int targetType);

    /**
     * Used for notify network unavailable.
     */
    void notifyNetworkUnavailable();




    /**
     * Used for get IMS feature.
     * @return:
     * ImsConfig.FeatureConstants.FEATURE_TYPE_UNKNOWN = -1;
     * ImsConfig.FeatureConstants.FEATURE_TYPE_VOICE_OVER_LTE = 0;
     * ImsConfig.FeatureConstants.FEATURE_TYPE_VOICE_OVER_WIFI = 2;
     */
    int getCurrentImsFeature();

    /**
     * Used for set IMS service listener.
     */
    void setImServiceListener(in IImServiceListenerEx listener);
}
```

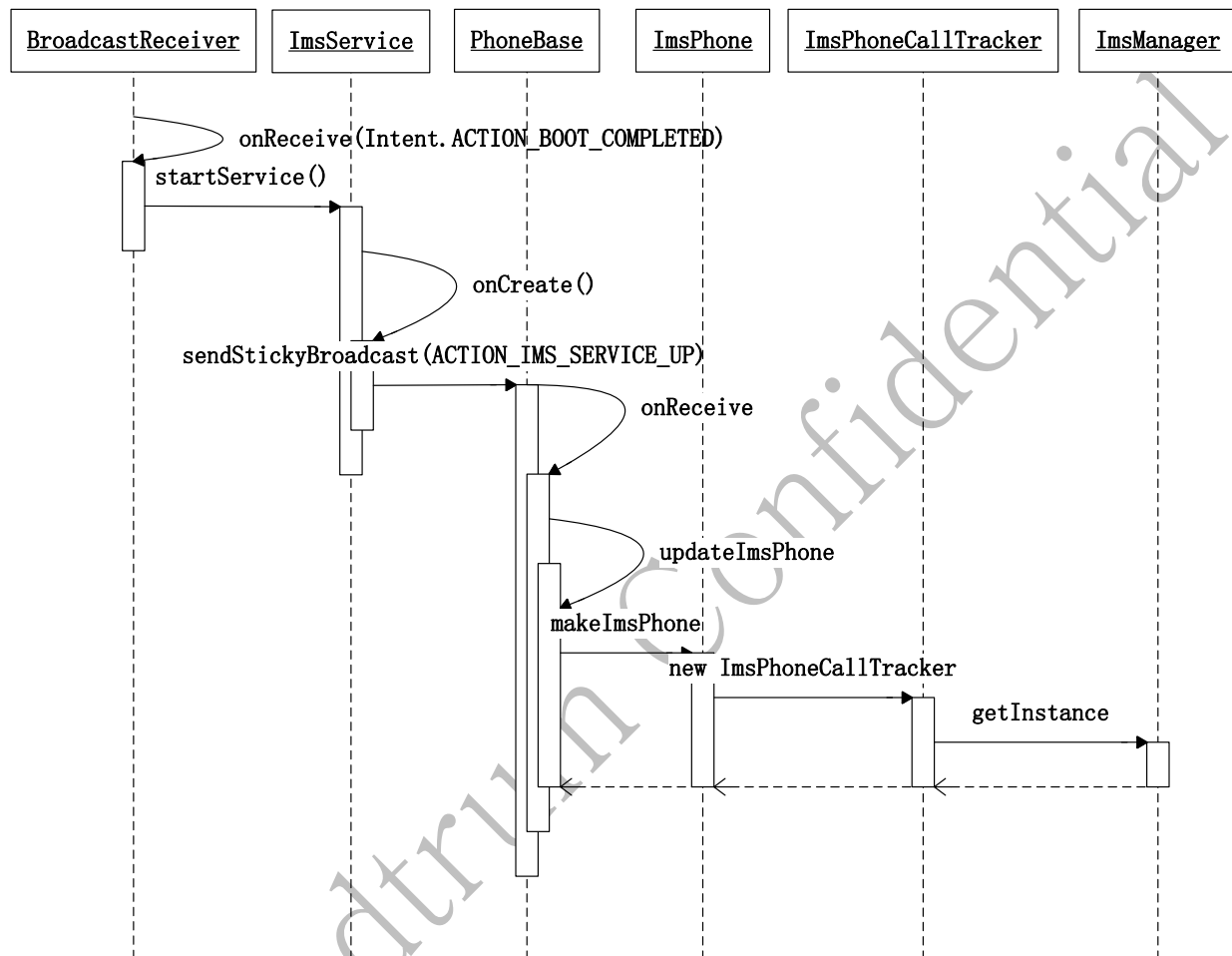
4. Functional Modules & Flowchart

4.1. IMS stack enable / disable

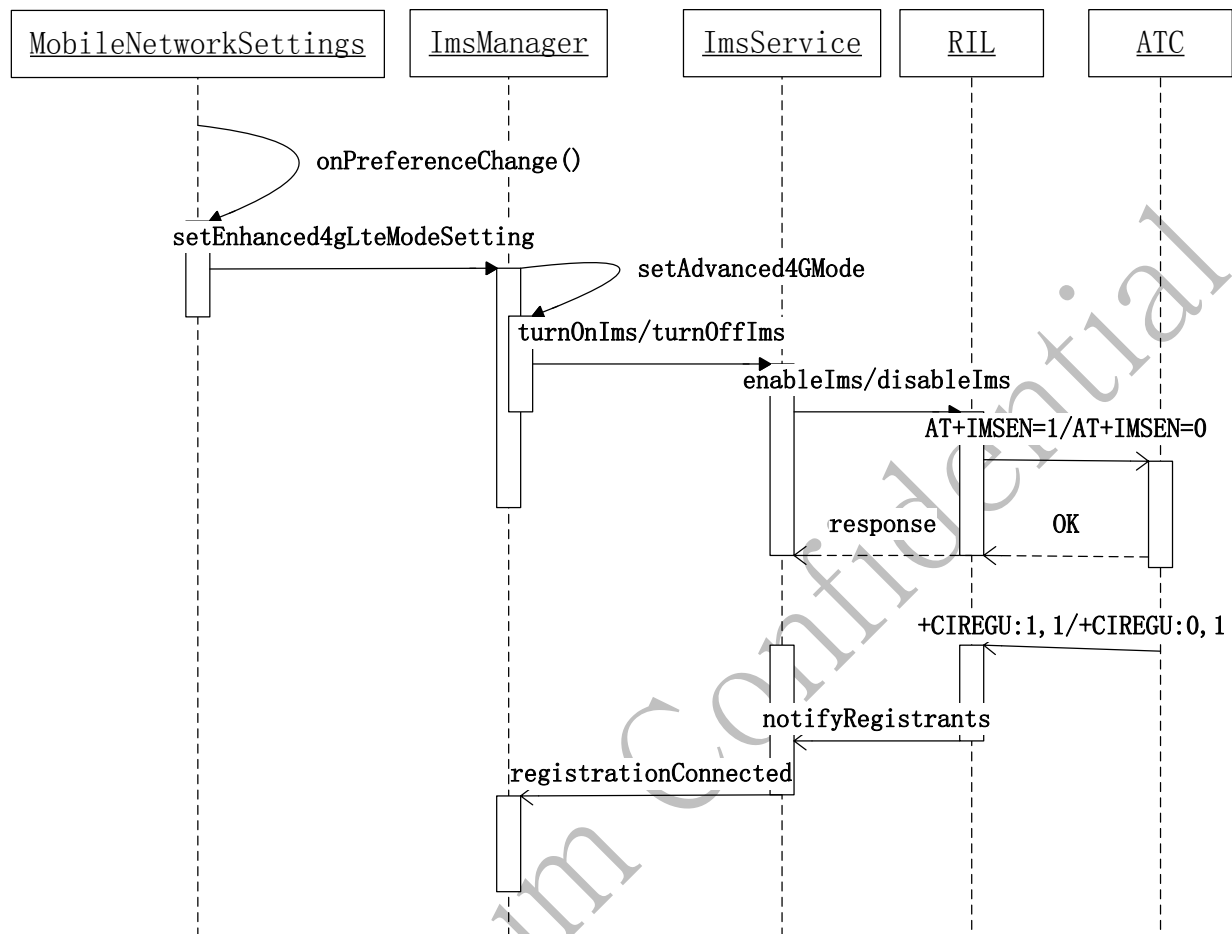
4.1.1. Feature description

-  Provides APIs for IMS stack disable/enable.
-  Opens the IMS stack for making calls and receiving generic IMS calls. The IMS service will register the device to the operator's network with the credentials(from ISIM)periodically in order to receive calls from the operator's network. When the IMS service receives a new call, it will send out an intent with the provided action string.
-  Closes the IMS stack for not to make/receive calls. All the resources that were allocated to the service are also released.

4.1.2. IMS Object Init:



4.1.3. IMS stack enable/disable:



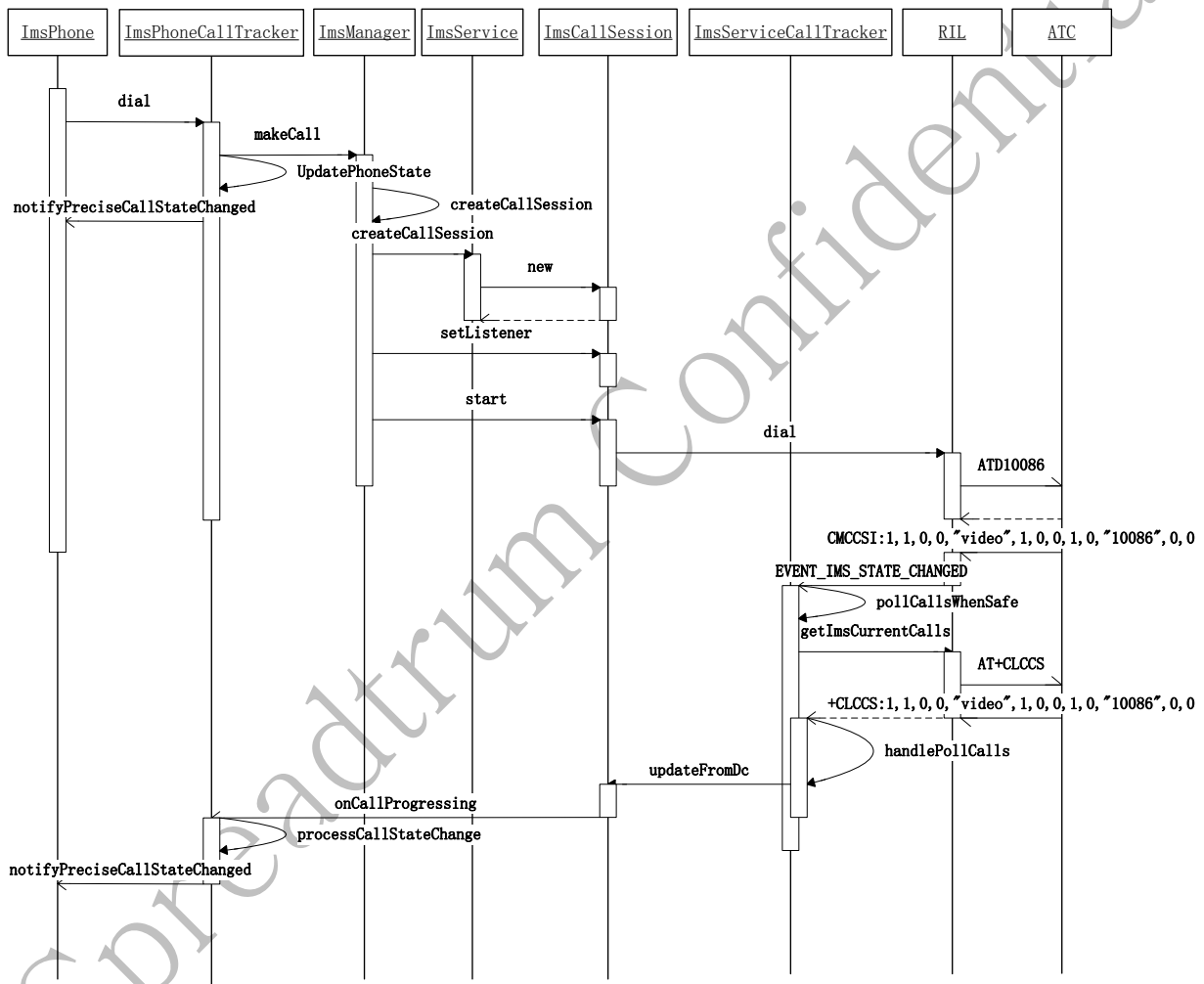
4.2. IMS Voice/Video Call

4.2.1. Feature description

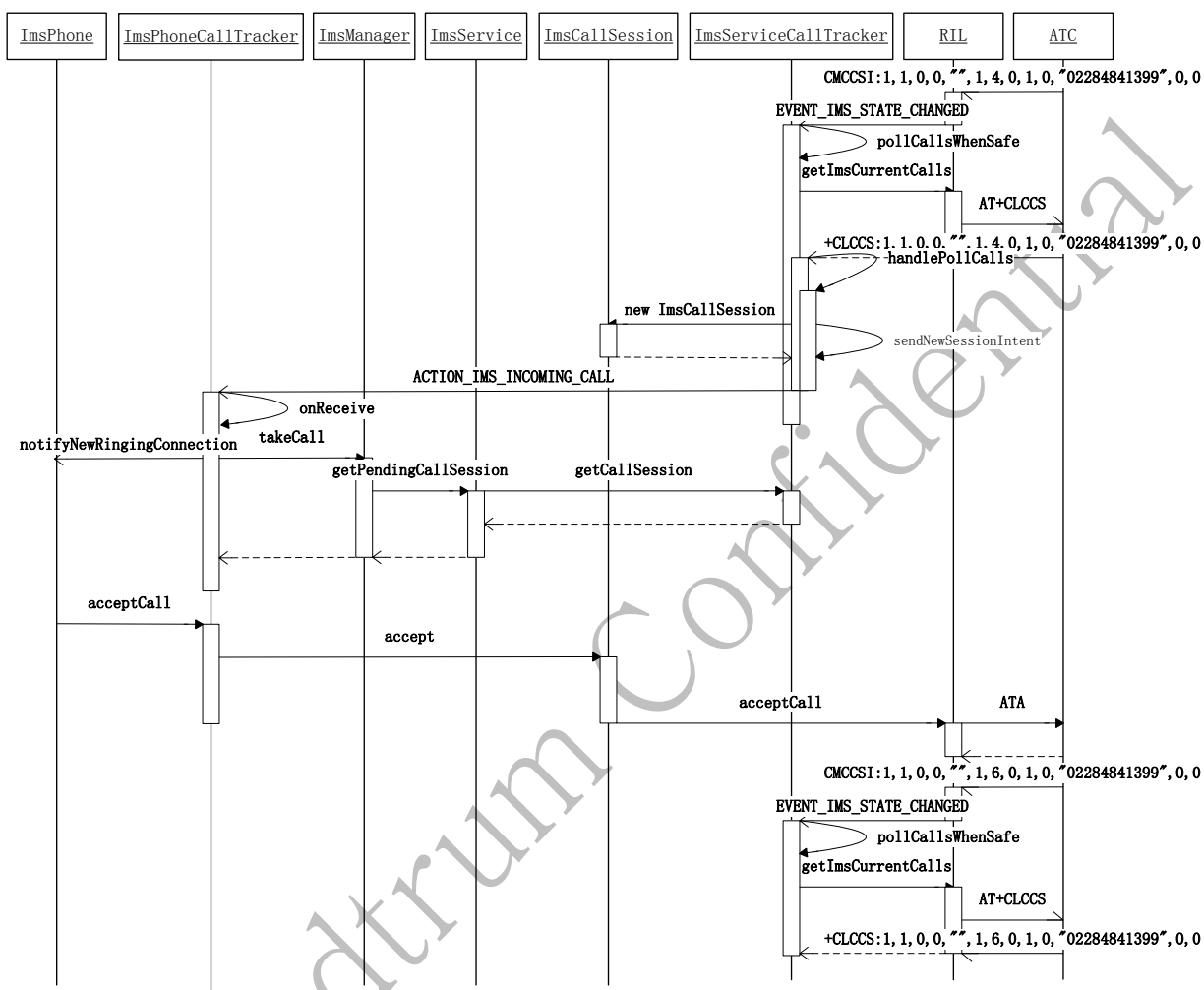
- Handles IMS voice/video call over LTE.

- Provides APIs for voice/Video call , such as initiation/termination session, hold/resume session, and so on..
- An IMS session that is associated with a SIP dialog which is established from/to INVITE request or a mid-call transaction to control the session.

4.2.2. MO Call



4.2.1. MT Call

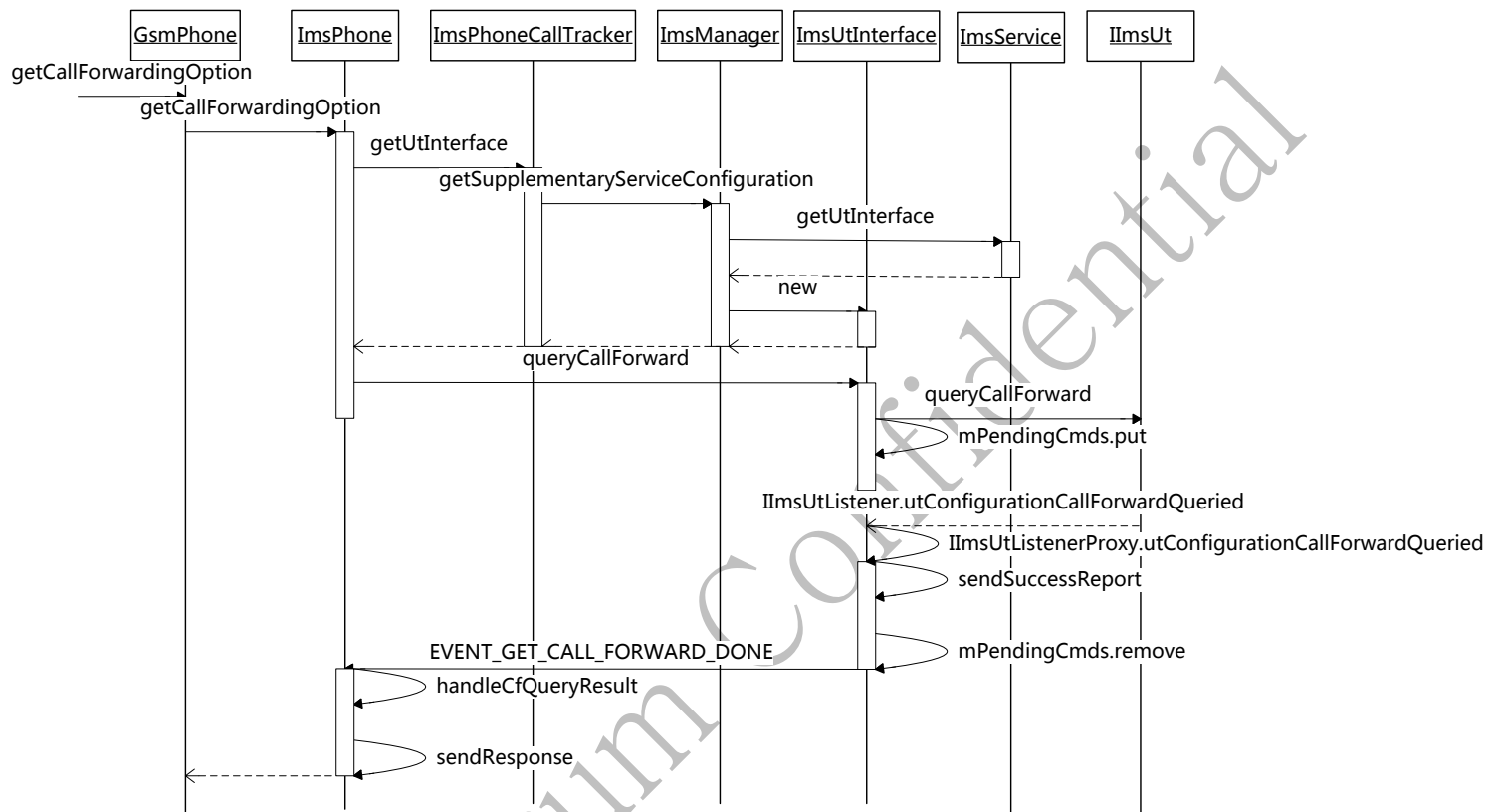


4.3. Supplementary Service

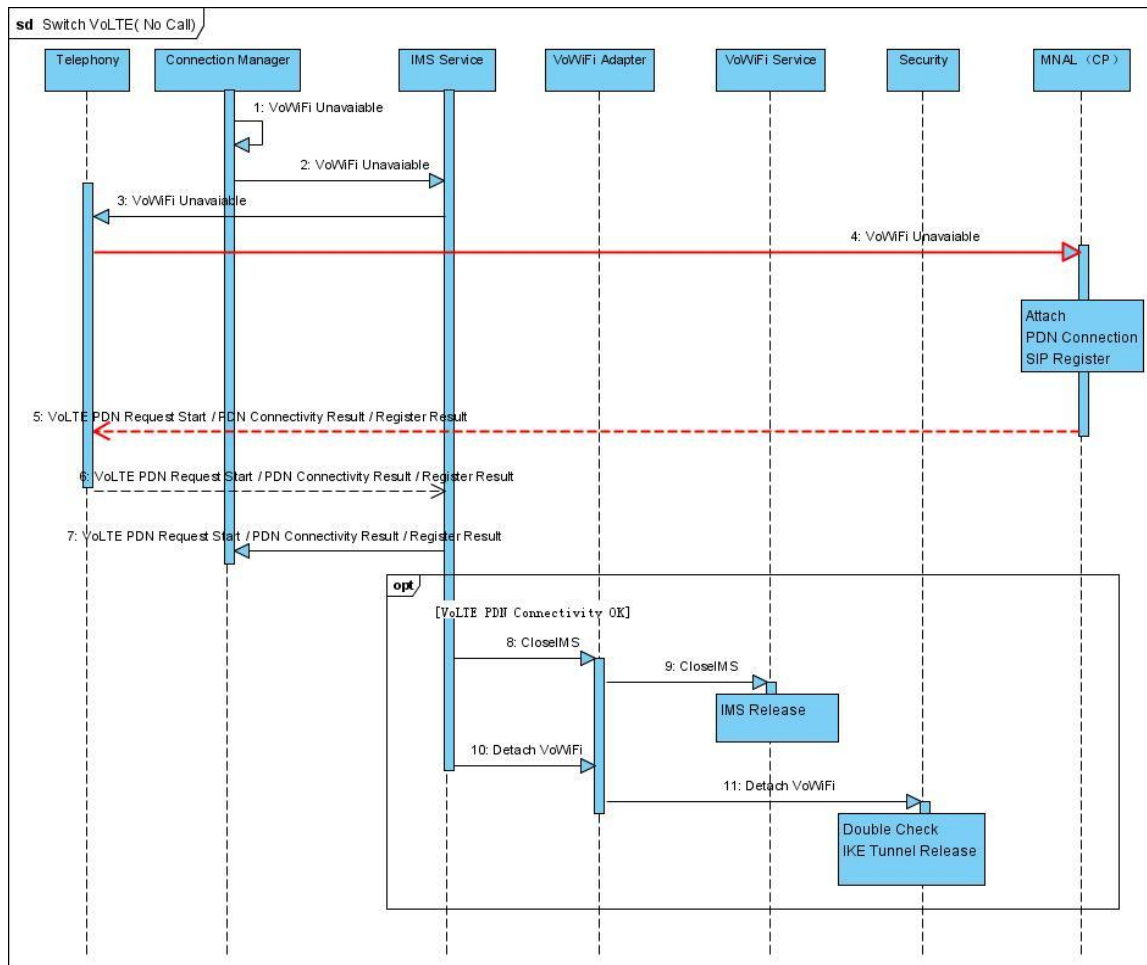
4.3.1. Feature description

- Provides APIs for the supplementary service settings using IMS (Ut interface).

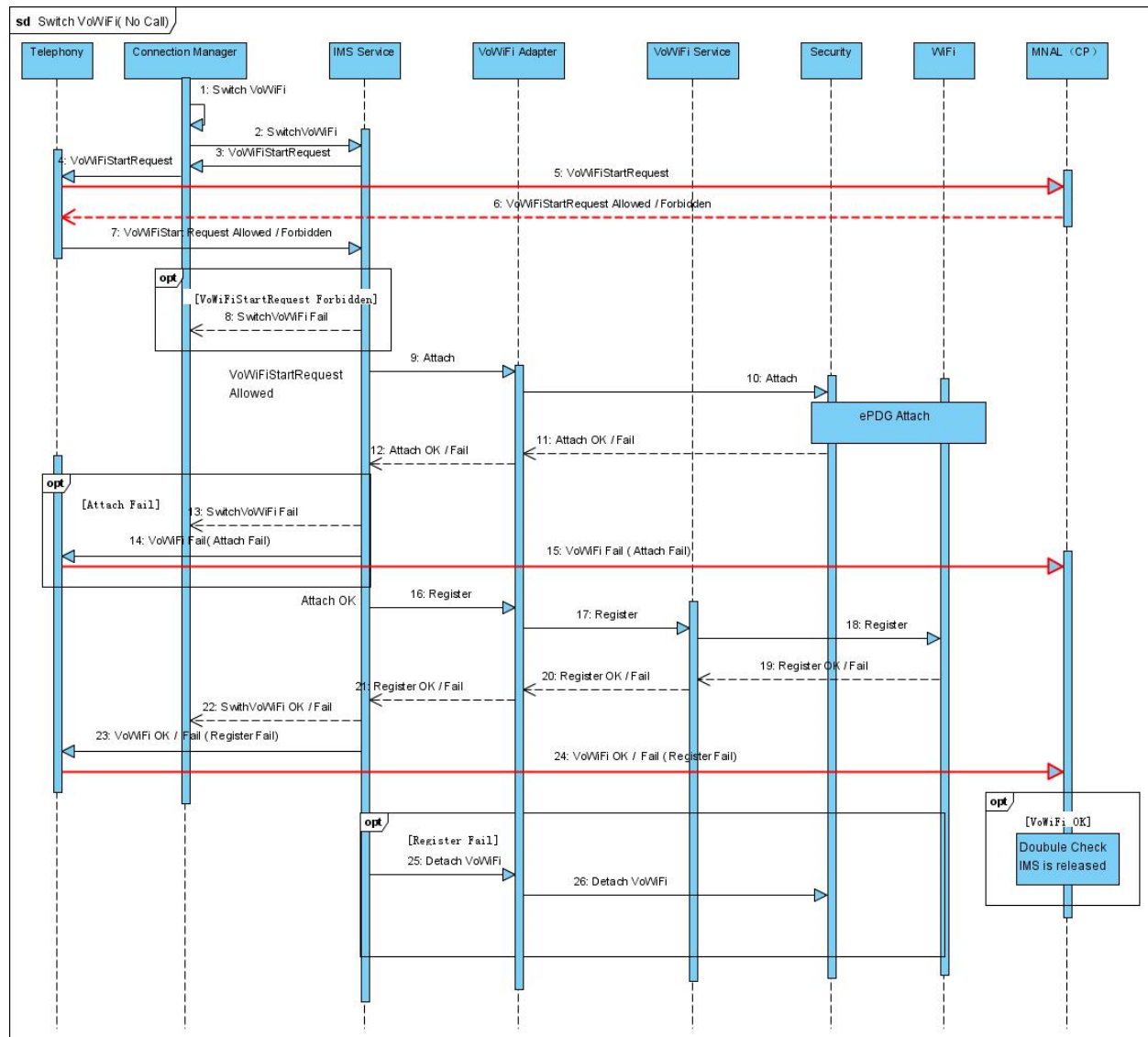
- It is created from 3GPP TS 24.623 (XCAP(XML Configuration Access Protocol) over the Ut interface for manipulating supplementary services).



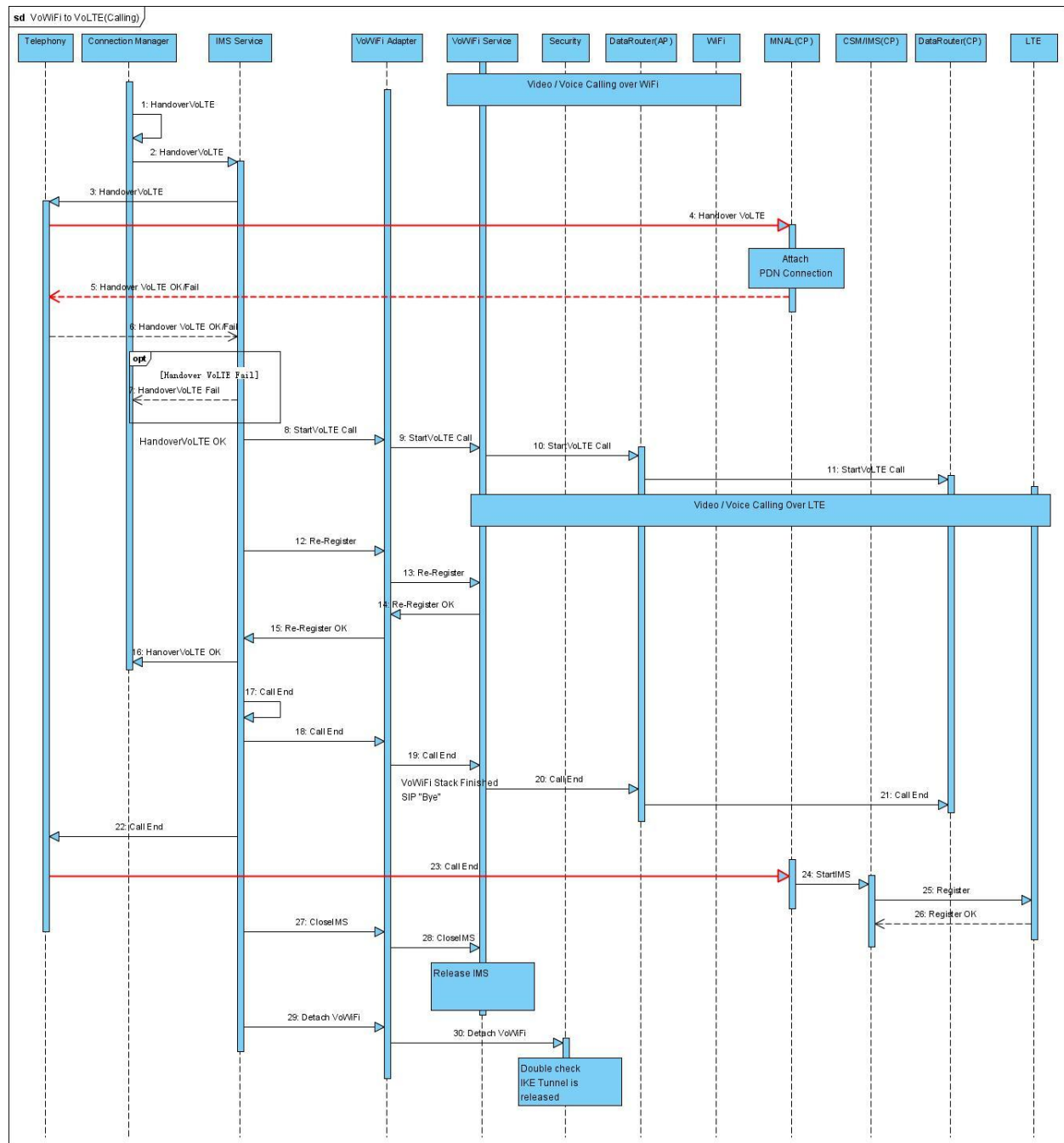
4.4. Switch to VoLTE (No Call)



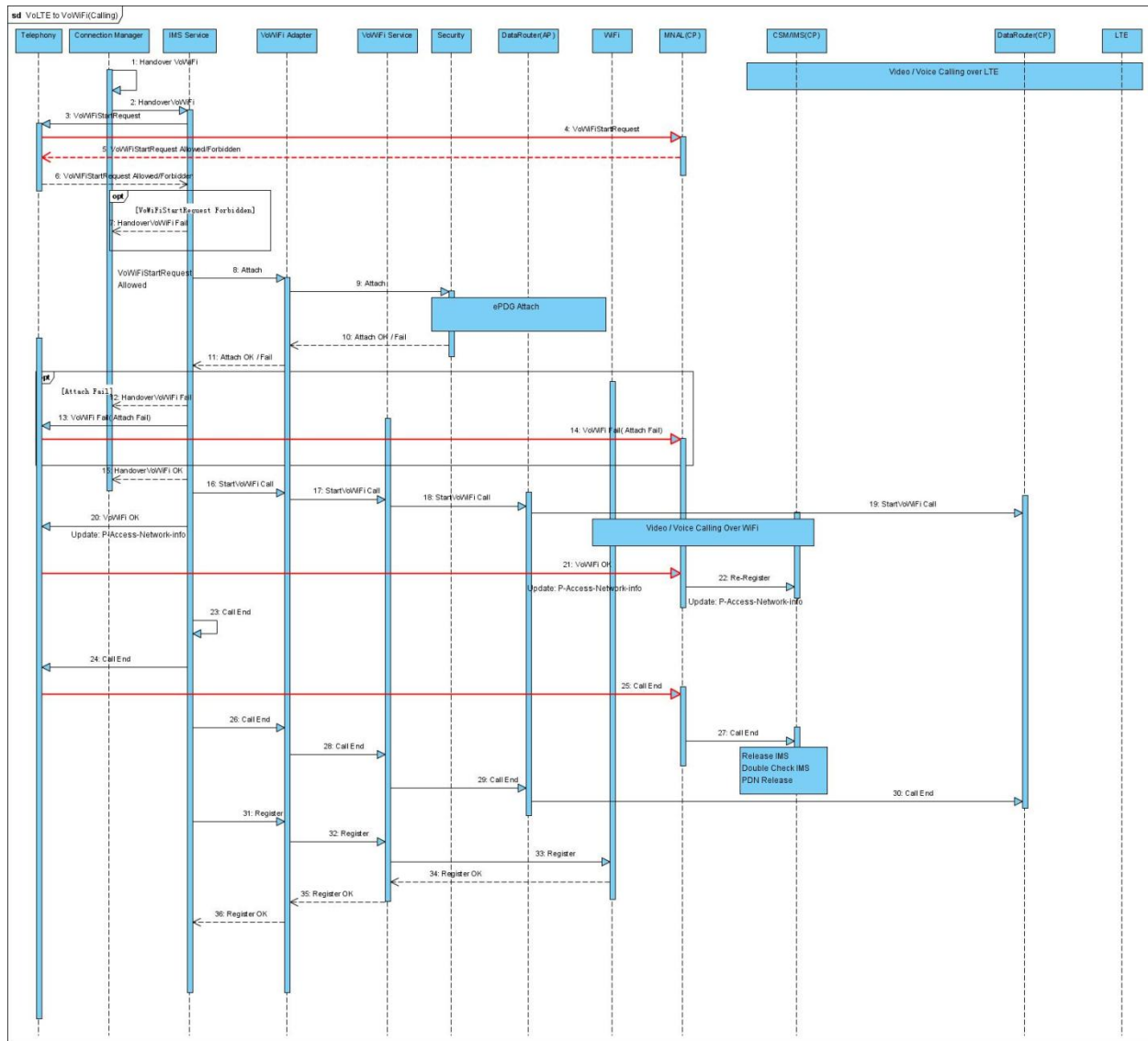
4.5. Switch to VoWiFi (No Call)



4.6. VoWiFi to VoLTE (Calling)



4.7. VoLTE to VoWiFi (Calling)



5. AT Commands

1.3.1 AP HANDOVER REQUEST

+IMSHO=<type>

type: 1, idle handover to VoWiFi

2, idle handover to VoLTE

3, handover to VoWiFi in Call

4, handover to VoLte in Call

Response:

OK, Allow

+CME ERROR: 3, Forbidden

1.3.2 CP HANDOVER REPORT

+IMSHOU:<type>

type: 1, idle handover to VoWifi

2, idle handover to VoLte

3, handover to VoWifi in Call

4, handover to VoLte in Call

1.3.3 IMS PDN STATUS REPORT(CP)

^CONN:11,2,<status>

status: 0, IMS PDN activate fail

1, IMS PDN ready

2, IMS PDN start

1.3.4 To VoWIFI HNDOVER RESULT SET

+IMSHORST=<result>

result: 0, VoWIFI register fail, unknow

1, VoWIFI register success

2, ims pdn connectivity fail

3,ims register fail

4-10,reserved

1.3.5 To VoLte HANDOVER RESULT REPORT

+IMSHORSTU:<result>

result: 0, unknow

1, success

2, ims pdn build fail

3, re-register fail

4-10, reserved

1.3.6 ACCESS NETWORK INFORMATION CHANGE(WIFI INFO)

+IMSHOWFINF=<inf>

inf: string

Ref the 34.229 7.2A.4.3

1.3.7 CALL END NOTIFICATION AFTER HANDOVER

+IMSHOCALLEND

1.3.8 VOWIFI IMS REGISTE STATUS

+VOWFREG=<status>

status: 0, register fail/

1, register success

2, in register progress

1.3.9 APN SYNC COMMAND FOR HANDOVER

+CGDCONT=11,..

Set the APN by the IMS PDN ID

+CGDCONT?

Get the CP current IMS APN by the IMS PDN ID

1.3.10 CALL STATUS SYNC

+CALLSYNC=<?>

[To be defined]

Use for sync the call status to CP for SRVCC after the handover to VoLte

1.3.11 VOWIFI ATTACH(bearer establish)STATUS

+IMSWFATT=<status>

status: 0, attach fail,

1, attach success, handover to VOWIFI success and the bearer is ready for Call

1.3.12 REPORT THE VoLte ACCESS NETWORK INFO

+IMSHOLTEINFU:<inf>

inf: string,

Ref the 34.229 7.2A.4.3

1.3.13 CONFIG THE IMS HANDOVER PARMATERS

+HOPARA=<rssi>,<sinr>,<brssi>,<chlult>

rss: int, WIFI rssi

sinr: int

brssi: int

chlult: int, channel Utilization

Spreadtrum Confidential