**SKT 영상레터링/ 영상컬러링**

**개발진행 내용 정리**

**(Android 2.1)**

**2010.07. 29**



**MD 개발팀**

목차

[**1 Android 버전의 영상 컬러링/레터링 프로세스 설명도** 3](#_Toc268163714)

[1.1 영상 레터링 프로세스 설명도 3](#_Toc268163715)

[1.2 영상 컬러링 프로세스 설명도 3](#_Toc268163716)

[**2 Record type 정의(다운로드 URL 및 전화번호 정보) 4**](#_Toc268163717)

[**3 Record type 정의 (애칭 정보) 5**](#_Toc268163718)

[**4 VE 구동 예제 6**](#_Toc268163719)

[**5 영상 레터링 기능별 처리 방안 7**](#_Toc268163720)

[5.1 전화 수신에 대한 처리 방안 7](#_Toc268163721)

[5.2 전화 거절에 대한 처리 방안 7](#_Toc268163722)

[5.3 연결이 끊어진 경우에 대한 처리 방안 7](#_Toc268163723)

[5.4 전화 거절 후 메시지 전송 처리 방안 7](#_Toc268163724)

[**6 영상 컬러링 기능별 처리 방안** 8](#_Toc268163725)

[6.1 한뺨 통화 기능 (speaker on/off) 8](#_Toc268163726)

[6.2 통화 종료 기능 8](#_Toc268163727)

[6.3 통화 연결이 이루어진 경우에 대한 처리 방안 8](#_Toc268163728)

[**7 Amf 컨텐츠 저장 시나리오 9**](#_Toc268163729)

[**8 앞으로 개발 진행할 내용 정리 10**](#_Toc268163730)

1. Android 버전의 영상 컬러링/레터링 프로세스 설명도
   1. 영상 레터링 프로세스 설명도

전화 수신 시 RIL에서 parsing된 ss-Code가 VLTR인 경우 OEM은 OEM Phone App을 구동 하지 않아야 한다. 대신에 VE 클라이언트를 Intent을 통해서 구동 시켜주어야 한다.

* 1. 영상 컬러링 프로세스 설명도

전화 발신 시 RIL에서 parsing된 ss-Code가 VRBT인 경우 OEM은 OEM Phone App을 구동 하지 않아야 한다. 대신에 VE 클라이언트를 Intent을 통해서 구동 시켜주어야 한다.

1. Record type 정의(다운로드 URL 및 전화번호 정보)

Facility IE는 하나의 메시지에 모두 포함되어 전송되어야 한다. SETUP 메시지에 아래와 같은 Facility(24.080, 24.008) IE 를 포함한다.

Facility Information Element

|  |  |  |
| --- | --- | --- |
| Field | Reference | Value |
| Facility IEI | TS 24.080 [3.6] | 00011100’B |
| Length of Facility contents | TS 24.080 [3.6] |  |
| Component(s) | TS 24.080 [3.6] |  |
| Component type tag | TS 24.080 [3.6] | 10100001’B |
| Component length | X.690 |  |
| Invoke ID tag | TS 24.080 [3.6] | 00000010’B |
| Invoke ID length | X.690 |  |
| Invoke ID | TS 24.080 [3.6] |  |
| Operation Code tag | TS 24.080 [3.6] | 00000010’B |
| Operation Code length | X.690 |  |
| Operation Code | TS 24.080 [3.6] | 00010000’B |
| Parameters | TS 24.080 [4] |  |

표 VLTR Facility Informaion Element

* **Parameters 설정**

Parameter Format은 3GPP 24.080의 4.3.1. NotifySS Operationn 규격을 따른다. SS-Code 의 값에 따라 nameIndicator를 다음과 같이 서비스를 위한 컨텐츠 URL을 나타내는 용도로 사용한다.

- SS-Code = VLTR 서비스 구분자로 활용

- DataCodingScheme = ‘1000 0100’B (H’84) KSC5601

※ H’94 등 KSC5601를 지원하는 다른 값을 사용할 수 있음

- lengthInCharacters = nameString의 length

- nameString = Download URL

|  |  |
| --- | --- |
| IEI | Parameters |
| Facility | Operation Code = notifySS  Parameters  ss-Code = ‘1111 1001’B (H’F9) /\*VLTR\*/  name Indicator {  callingName {  namePresentationAllowed {  dataCodingScheme = 1000 0100 01)  lengthinCharacters  nameString = URL (1~ 160 Octet String)  }  }  } |

※ URL 길이는 MAX가 160 byte 이지만, 실제적으로는 60 bytes정도 사용 예상

표 VLTR Parameter 설정

|  |
| --- |
| Infomation element : Facility  IEI : 1C  Length : 40  Facility information :  Component type tag : 161  Component length : 38  Invoke ID tag : 2  Invoke ID length : 1  Invoke ID Value : 0  Operation Code tag : 2  Operation Code length : 1  Local Value : 16  Parameters  H : 0x30  Length : 30  SS-Code : 0x81  Length : 1  Contents : 249 (H’F9)  NameIndicator : 0xB4  Length : 25  CallingName : 0xA0  Length : 23  namePresentation Allowed : 0xA0  Length : 21  DataCodeScheme : 0x80  Length : 1  Contents : 132  LengthInCharacters : 0x81  Length : 1  Contents : 0x0D  NameString : 0x82  Length : 13 |

1. Record type 정의 (애칭 정보)

|  |  |
| --- | --- |
| 파 라 미 터 | 내 용 |
| *sKTDisplayText* 🞵) | * *NameSet* * *dataCodingScheme* * *USSD-DataCodingScheme* =   0100 11XX’B (KSC5601)   * *lengthInCharaters = ?* * *nameString* * *USSD-String* = “애칭 문자열” |

🞵) sKTDisplayText의 ASN.1 Notation은 아래와 같다.

|  |
| --- |
| sKTDisplayText-id OBJECT IDENTIFIER ::= { 0 3 4504 0 0 7 }  sKTDisplayText MAP-EXTENSION ::= {  &extensionId sKTOriginatingSFI-id,  &ExtensionType NameSet  -- NameSet Parameter Type is defined in TS 24.080.  } |

* CC*\_Setup* (TS24.008) 처리

단말로 *CC\_Setup* 전송시에 포함되는 파라미터는 아래와 같이 설정된다.

|  |  |
| --- | --- |
| 파 라 미 터 | 내 용 |
| *Facility* | * *OperationCode* = NotifySS * *Parameters* * *sS-Code* = CNAP * *NameIndicator* * *CallingName* * *NamePresentationAllowed* * *DataCodingScheme* * *USSD-DataCodingScheme* =   0100 11XX’B (KSC5601)   * *lengthInCharaters* = ? * *nameString* * *USSD-String* = “애칭 문자열” |

1. VE 구동 예제

Intent intent = **new** Intent();

intent.setClassName("com.mtelo.visualexpression", "Main");

intent.putExtra("service\_code", "VLTR");

intent.putExtra("download\_url", "http://www.example.com/1.am3");

intent.putExtra("caller\_num", "010-2057-3545");

intent.putExtra("caller\_name", "엠텔로"); intent.putExtra("nick\_name", "행복하세요~");

영상 레터링의 경우 OEM에서 Intent의 필드를 채우는 예제.

Intent intent = **new** Intent();

intent.setClassName("com.mtelo.visualexpression", "Main");

intent.putExtra("service\_code", "VRBT");

intent.putExtra("download\_url", "http://www.example.com/1.am3");

intent.putExtra("call\_num", "010-2057-3545");

intent.putExtra("call\_name", "엠텔로");

intent.putExtra("nick\_name", "행복하세요~");

영상 컬러링의 경우 OEM에서 Intent의 필드를 채우는 예제.

OEM에서는 위 예제의 내용대로 필드를 채우고 VE 클라이언트를 구동한다. 구동 시 OEM에서 벨소리 재생을 하지 않는다. VE 클라이언트가 구동이 되면서 단말에서 지정한 벨소리를 재생 시킨다.

이렇게 처리를 하는 이유는 컨텐츠가 SKM파일로된 경우 VE 클라이언트에서는 벨소리출력을 멈추게 하는 이벤트를 OEM으로 요청을 해야 하는데 이러한 내용 없이 VE 클라이언트 내에서 처리를 하도록 한다.

※ 예상치 못한 부분이 있을 수도 있으니 위 사항은 추후에 변경 가능함.

1. 영상 레터링 기능별 처리 방안
   1. 전화 수신에 대한 처리 방안

전화 수신을 하게 되는 경우 VE 클라이언트는 재생되는 컨텐츠를 멈추고 OEM으로 전화 수신을 하겠다는 이벤트를 보내고 VE 클라이언트는 스스로 종료를 시킨다.

**private** **final** **static** String ACTION\_VLTR\_ANSWER\_THE\_PHONE =

" com.mtelo.visualexpression.VLTR\_ANSWER\_THE\_PHONE";

sendBroadcast(**new** Intent(ACTION\_VLTR\_ANSWER\_THE\_PHONE));

finish();

VE 클라이언트에서 전화 수신 시 OEM으로 보내는 이벤트 예제.

* 1. 전화 거절에 대한 처리 방안

전화 수신을 거절 하게 되는 경우 VE 클라이언트는 재생되는 컨텐츠를 멈추고 OEM으로 전화 수신을 거절 하겠다는 이벤트를 보내고 VE 클라이언트는 스스로 종료를 시킨다.

**private** **final** **static** String ACTION\_VLTR\_REJECTION\_THE\_PHONE =

"com.mtelo.visualexpression.VLTR\_REJECTION\_THE\_PHONE";

sendBroadcast(**new** Intent(ACTION\_VLTR\_REJECTION\_THE\_PHONE));

finish();

VE 클라이언트에서 전화 수신 거절 시 OEM으로 보내는 이벤트 예제.

* 1. 연결이 끊어진 경우에 대한 처리 방안

전화 연결이 끊어지는 경우 OEM에서는 전화가 끊어졌다는 액션 Intent을 VE App에 전달 해주어야 한다. VE 클라이언트는 액션 값을 전달받으면 VE 클라이언트는 컨텐츠 재생을 멈추고 스스로 종료를 시킨다.

**public** **final** **static** String ACTION\_VLTR\_DISCONNECTED =

"com.mtelo.visualexpression.VLTR\_DISCONNECTED";

@Override

**protected** **void** onNewIntent(Intent intent) // 이 함수를 통해서 액션 값을 전달 받는다.

**{**

**if** (intent.getAction().equals(ACTION\_VLTR\_ANNOUNCEMENT))

{

//VE App 종료

}

OEM에서 보낸 액션을 VE 클라이언트에서 확인하는 예제.

* 1. 전화 거절 후 메시지 전송 처리 방안

삼성전자의 경우 전화 수신 상태에서 우측 카메라 버튼을 길게 누르면 전화 거절 후 메시지 전송 기능으로 진입을 한다.

@Override

**public** **boolean** onKeyLongPress(**int** keyCode, KeyEvent event)

{

**if** (keyCode == KeyEvent.KEYCODE\_CAMERA)

{

Intent intent = **new** ntent(ACTION\_CREATE\_REJECTED\_CALL\_WITH\_MSG, **null**);

intent.setFlags(Intent.FLAG\_ACTIVITY\_NEW\_TASK |

Intent.FLAG\_ACTIVITY\_EXCLUDE\_FROM\_RECENTS

| Intent.FLAG\_ACTIVITY\_NO\_USER\_ACTION);

intent.setClassName("com.android.phone","com.android.phone.RejectCallWithMsg");

intent.putExtra(PHONE\_NUMBER, "01020573545");

**this**.mContext.startActivity(intent);

전화 거절 후 메시지 전송 App을 띄우기 위한 예제.

※ 각 제조사 별로 구현이 다르게 되어 있으므로 참조용으로만 사용해야 한다.

1. 영상 컬러링 기능별 처리 방안
   1. 한뺨 통화 기능 (speaker on/off)

Toggle 형식으로 되어 있어 리시버와 스피커간의 통화연결 음/컨텐츠 내의 음원을 출력 할 수 있도록 한다.한뺨 통화 기능을 VE 클라이언트에서 조작 가능 하도록 OME에서 기능 제공을 해주어야 한다.

**private** **final** **static** String ACTION\_VRBT\_SPEAKER\_ON =

"com.mtelo.visualexpression. VRBT\_SPEAKER\_ON";

sendBroadcast(**new** Intent(ACTION\_VRBT\_SPEAKER\_ON));

VE 클라이언트에서 전화 발신 시 스피커 ON 시키는 예제.

**private** **final** **static** String ACTION\_VRBT\_SPEAKER\_OFF =

"com.mtelo.visualexpression. VRBT\_SPEAKER\_OFF";

sendBroadcast(**new** Intent(ACTION\_VRBT\_SPEAKER\_OFF));

VE 클라이언트에서 전화 발신 시 스피커 OFF 시키는 예제.

* 1. 통화 종료 기능

통화 종료 버튼을 누르면 통화가 종료 되어 VE 클라이언트는 종료되고 다시 Phone App을 OEM에서 구동을 시켜야 한다.

**private** **final** **static** String ACTION\_VRBT\_HANGUP =

"com.mtelo.visualexpression.VRBT\_HANGUP";

sendBroadcast(**new** Intent(ACTION\_VRBT\_HANGUP));

finish();

VE 클라이언트에서 전화 발신 시 통화 종료 시키는 예제.

* 1. 통화 연결이 이루어진 경우에 대한 처리 방안

전화를 걸고 통화연결을 시도 하는 중에 통화연결이 되었을 때 OEM에서는 VE 클라이언트에게 통화연결이 되었다는 것을 알려야 한다.

VE 클라이언트에서는 통화연결이 되었다는 사실을 OEM으로부터 전달 받은 후 재생을 멈추고 VE 클라이언트를 종료 시킨다.

**public** **final** **static** String ACTION\_ VRBT\_CONNECTED =

"com.mtelo.visualexpression.VRBT\_CONNECTED";

@Override

**protected** **void** onNewIntent(Intent intent) // 이 함수를 통해서 액션 값을 전달 받는다.

**{**

**if** (intent.getAction().equals(ACTION\_ VRBT\_CONNECTED))

{

// VE App 종료

}

OEM에서 보낸 액션을 VE 클라이언트에서 확인하는 예제.

1. Amf 컨텐츠 저장 시나리오

|  |  |  |
| --- | --- | --- |
| **저장 영역** | **세부시나리오** | **비고** |
| 1. 내장메모리 저장 공간 5M 확보 | 내장메모리 저장폴더 지정 후 저장 | 선입선출로 삭제, OEM 통화목록에서 다시보기 기능 가능 **(BEST)** |
| 5M Dummy content로 확보 | 단말 저장공간 이슈로 고객 VOC 발생 가능성 |
| SD카드/5M 동시 제공 시 사용자가 컨텐츠 저장공간 선택 | VE 관련 환경설정 기능 필요 |
| 1. 외장메모리(SD카드 저장) | SD카드 메모리 Full 될 때까지 저장 | 이미지파일 폰앨범에서 다시보기 가능  사용자가 파일 위치 변경 및 삭제 가능  SD카드 사용하지 않을 경우/ SD카드 Full로 사용할 경우에는  호 연결시마다 컨텐츠 다운로드 – 다운로드 속도 이슈 |
| 1. 내/외장 메모리 | 우선적으로 내장 메모리에 Amf 컨텐츠를 저장. 내장 메모리에 저장공간이 부족 시 외장메모리의 존재여부 확인  외장 메모리 존재한 경우 저장 공간 확인 후 컨텐츠 저장. | 여러 가지 체크 사항 때문에 App의 진행 속도가 다소 느려질 것으로 우려됨. |

1. 내장 메모리 영역에 5MB의 저장 공간을 미리 확보 하여 컨텐츠을 보다 안정적으로 저장한다. 메모리의 확보 방법으로는 5MB의 Dummy 파일을 VE App을 다운로드 후 설치 시점에서 VE App에서 로컬 영역에 생성을 한다.   
     
   Amf 컨텐츠를 다운로드 후 초기에 생성된 Dummy파일의 크기를 수정한다. 다운로드 받은 컨텐츠들의 크기를 뺀 남은 Dummy 메모리 영역의 크기로 Dummy파일을 수정 해나가면서 타 App에서 Amf 컨텐츠의 저장 영역을 침범하지 못하게 막는다.
2. SD카드의 저장 가능 범위 내에서는 컨텐츠들을 저장 할 수 있다. 문제는 스마트 폰 유저들이 모두 SD카드를 장착한다는 보장이 없다  
   Case 1: SD카드 장착 시, Amf 컨텐츠들을 저장 가능할 때까지 저장을 한다.   
     
   Case 2: SD카드 미 장착 시, Amf 컨텐츠들을 로컬 영역에 저장을 하도록 한다. (로컬 영역에 저장 공간이 여의치 않다면, 매번 컨텐츠를 받을 때 마다 기존 컨텐츠를 삭제 한다.)
3. 내/외장 메모리를 모두 사용해서 컨텐츠을 저장한다. 내/외장을 모두 사용한다는 측면에서 보면 유연한 방안이지만, 문제는 내/외장 메모리에 저장 정보를 체크해야 하는 부분은 VE App의 프로세스 속도 저하를 유발 할 가능성이 크다.
4. 앞으로 개발 진행할 내용 정리

|  |  |  |
| --- | --- | --- |
| **서비스** | **작업 내용** | **작업 영역** |
| 영상 컬러링 | 통화 연결 음 | VE App 에서 가지고 있는 음원 컨텐츠를 “통화 연결 음”을 출력하는 사운드 path에 연결 |
| DTMF | SKT 소리샘 진입 시에 DTMF을 ARS로 전달 처리 |
| 영상 레터링/컬러링 | Status Bar | VE App 구동 후 화면 상단의 Status Bar 비활성으로 처리 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |