

ASN.1을 활용한 BIS시스템 사례

2016. 10. 20.

(주) 동인 S/W 연구소
윤 상 영

1. ASN.1의 이해

- ASN.1 이란
- ASN.1의 필요성
- ASN.1 이용 사례
- ASN.1 컴파일러

1. ASN.1의 이해

ASN.1 이란

- ✓ ASN.1은 구조화된 정보를 기술하기 위한 언어
- ✓ BER(Basic Encoding Rule)과 DER(Distinguished Encoding Rule)
 - ASN.1은 추상적인 데이터 형식이기 때문에 그것을 그대로 전송 할 수 없음
 - BER과 DER은 각 ASN.1 형태의 데이터 형식을 네트워크 상에서 전송할 수 있는 형태로 Encoding, Decoding해주는 방법에 대하여 정의

Initiate ::= SEQUENCE {

datex-Sender-txt

UTF8String (SIZE (0..40)),

datex-Destination-txt

UTF8String (SIZE (0..40))

}

```
static int memb_datex_Sender_txt_constraint_1(asn_TYPE_descriptor_t *td, const void *sptr,asn_app_constraint_failed_f *ctfailcb, void *app_key) {
    const UTF8String_t *st = (const UTF8String_t *)sptr;
    size_t size;

    if(!sptr)
        _ASN_CTFAIL(app_key, td, sptr,"%s: value not given (%s:%d)",td->name, __FILE__, __LINE__);        return -1;

    size = UTF8String_length(st);
    if((ssize_t)size < 0)
        _ASN_CTFAIL(app_key, td, sptr,"%s: UTF-8: broken encoding (%s:%d)",td->name, __FILE__, __LINE__);        return -1;

    if((size <= 40))        return 0;
    else {        _ASN_CTFAIL(app_key, td, sptr,"%s: constraint failed (%s:%d)",td->name, __FILE__, __LINE__);        return -1;    }
}
```

1. ASN.1의 이해

ASN.1의 필요성

- ✓ 이기종 시스템(Windows, UNIX 등)간 데이터 전송시 발생하는 문제점을 해결하기 위하여 공통화된 표현방법이 필요
- ✓ 데이터 표기법이 시스템마다 다름
 - Windows는 리틀엔디언(Little Endian)이고, UNIX는 빅엔디언 (Big Endian)이어서 데이터 해석방법이 틀림
 - 0x23이 32비트 빅엔디언에서는 (0x23,0x00,0x00,0x00), 리틀엔디언에서는 (0x00,0x00,0x00,0x23)으로 해석됨으로 서로 다른 데이터로 인식
- ✓ 운영하는 장비가 8,16,32,64Bit 컴퓨터이냐에 따라 데이터크기가 다름
 - 32비트 장비에서 int *형은 4바이트이고, 64비트 장비에서는 8바이트
ex) int *, char *, size_t, time_t

1. ASN.1의 이해

ASN.1의 필요성

✓ 사용하는 Language에 따라서 데이터의 정의가 틀릴수 있음

- C++과 Ada로 개발된 프로그램이 통신을 할 경우 서로 정의 된 데이터가 틀리기 때문에 다르게 해석될 수 있음

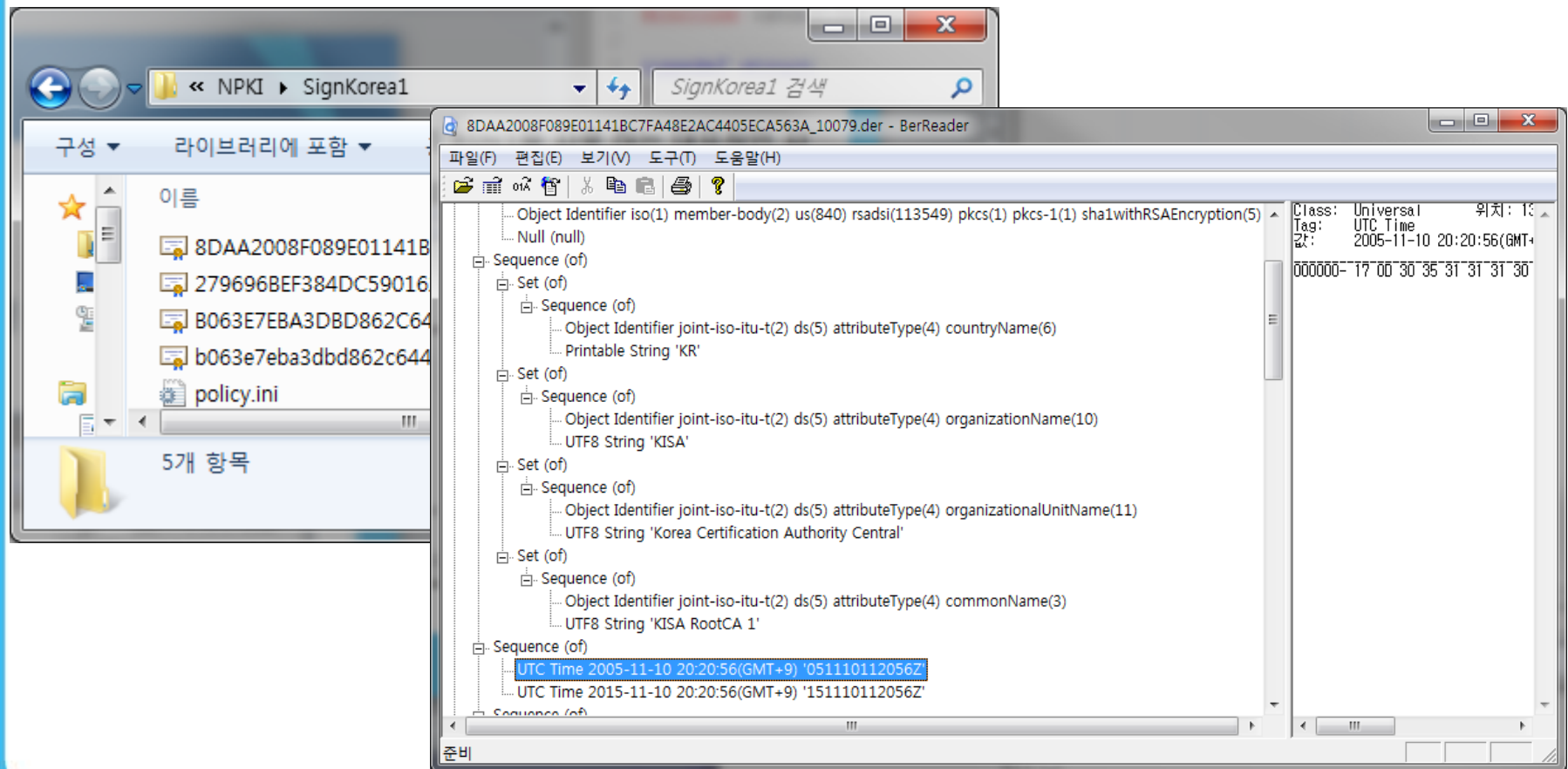
ex) struct {	struct {	struct {	struct {
char temp1;	char temp1;	char temp1;	int temp1;
	char temp2;	int temp2;	char temp2;
int temp2;	int temp3;	char temp3;	int temp3;
} => 8byte	} => 8byte	} => 12byte	} => 12byte

1. ASN.1의 이해

ASN.1의 이용 사례

✓ 인터넷 뱅킹 이용시 인증서

– 인터넷 뱅킹 사용시 사용 하는 인증서가 ASN.1을 사용하는 대표 사례

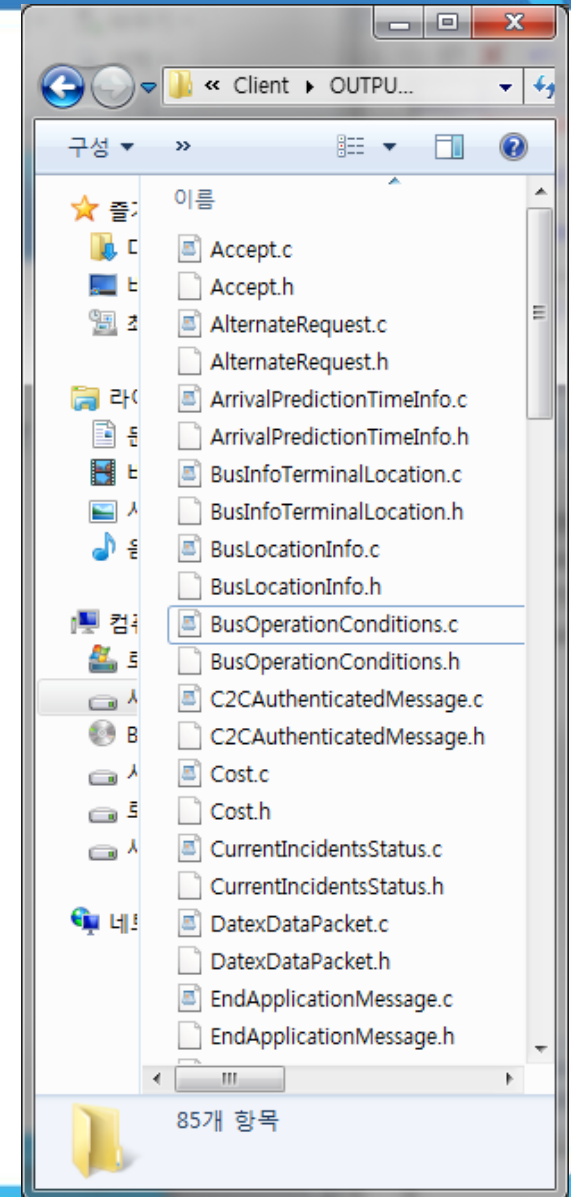
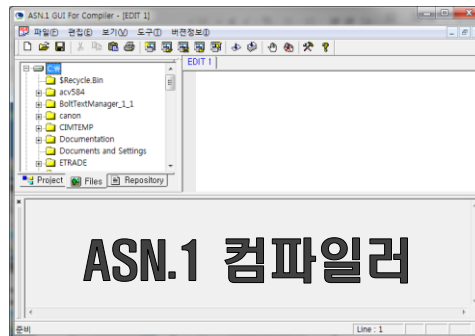
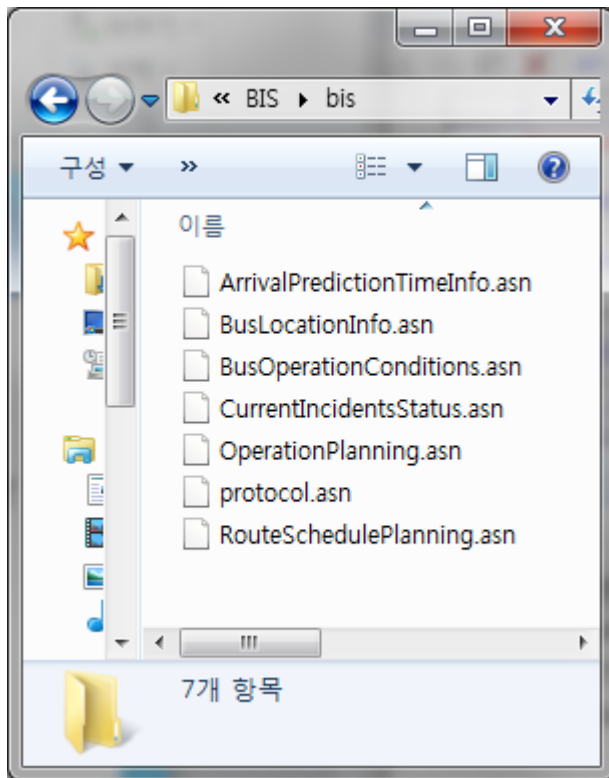


1. ASN.1의 이해

ASN.1 컴파일러

✓ ASN.1 컴파일러

- ASN.1의 규격을 읽어 이에 맞는 소스파일(c언어의 경우 C File, header File)을 만들어 내는 Tool



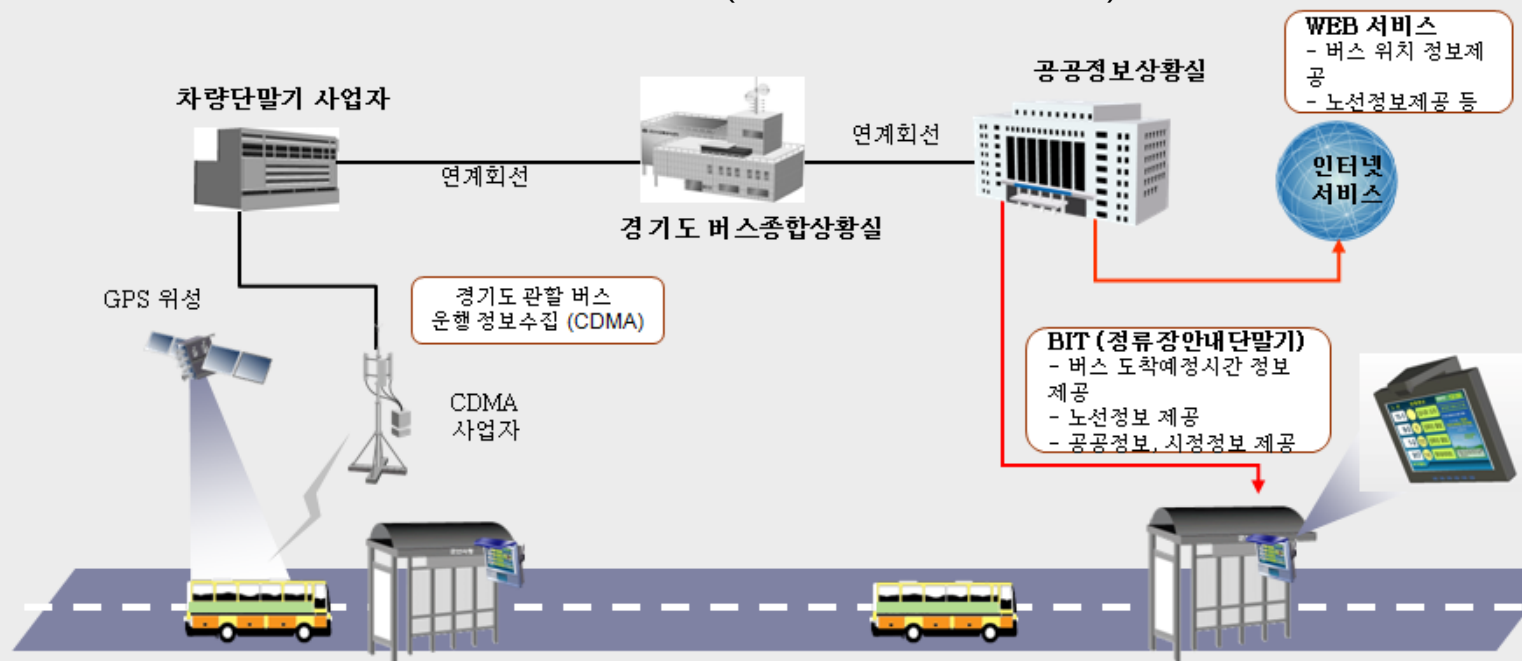
2. 어플리케이션 개발 활용

- BIS 시스템 구성
- BIS 센터 소프트웨어
- 대중교통 정보교환 기술기준
- 교통정보 교환을 위한 국가표준
- SAMPLE 데이터 패킷분석
- 데이터 접속 및 전송절차 SAMPLE
- ASN.1사용시 주의할 점

2. 어플리케이션 개발 활용

BIS 시스템 구성

- 현재 경기도에 구축된 시스템을 나타낸 것으로 경기도 버스종합 상황실에서 경기도 관내에 있는 모든 시내버스 및 광역버스에 대한 정보(버스 정류장 도착, 출발, 교차로 통과, 돌발상황정보 등)를 수집하여 각 지자체로 전송(ASN.1)해 주면 각 지자체에서는 이를 가공하여 도착예측정보를 BIT(정류장 안내 단말기)로 전송하는 시스템



2. 어플리케이션 개발 활용

BIS 센터 소프트웨어

- 연계서버 : 경기도 BMS센터로부터 정보수집
- 주운영서버 : 수집된 정보를 가공하여 예측정보 생성
- 정보제공서버(BIT, WEB) : 이용자에게 버스도착정보 및 생활정보를 제공
- 운영단말 : 버스관련 시설물을 관리하고 운영상황을 모니터링
- 상황판 운영서버
- GIS 서버
- 마을버스 통신서버
- 노선관리 서버
- SMS 서버
- ARS 서버
- OPEN API 서버
- 모바일 서버



2. 어플리케이션 개발 활용

대중교통 정보교환 기술기준

- 대중교통 정보교환 기술기준에 따라 201~206까지 6개의 정보가 정의 됨
- 이중 201, 205, 206이 현재 경기도 BMS센터로 부터 지자체로 데이터 전송
- 버스위치정보(201)의 정류장도착, 정류장출발, 교차로 통과 정보를 활용하여 도착에
측정보를 가공하고 이를 BIT에 정보 전송

ID	정보명	교환 주기	정보내용
201	버스위치 정보	실시간	차량ID, 노선ID, 막차 여부, 막차의 최종 도착정류장ID -. 이벤트정보 : 메시지발생시각, 수집노드ID, 노드진입, 진출시각, 노드통행시간 -. 정주기정보 : 차량위치정보(GPS), 수집시각, 수집주기
202	도착예정 정보	실시간	정류장ID, 노선ID, 차량ID, 도착예정시간(출발정류장ID 및 진출, 진입시각, 통과시간), 평균통행속 도, 막차정보
203	운행계획 정보	변경시	노선기본정보(노선ID, 노선명칭, 기.종점정류장ID), 노선부가정보, 노선운행정보(첫차, 막차 출발 시각, 첨두/비첨두 배차간격), 차량운행회수, 운행계획정보 갱신시각, 갱신내용
204	운행지시 정보	필요시	차량ID, 다음정차정류장ID, 차간거리조정, 운행지시정보
205	운행관리 정보	필요시	차량ID, 노선ID, 정보수집노드ID, 차량위치정보(GPS), 운행상태정보, 무단결행노선ID, 무단결행발 생대수, 노선명
206	긴급상황 정보	유고발 생시	차량ID, 노선ID, 이벤트정보수집노드(zone, 구역)ID, 차량위치정보(GPS), 돌발상황발생시각, 돌발 상황유형, 발생위치(도로명칭, 관령교차로, 돌발상황발생위치설명), 돌발상황부연설명, 돌발상황 긴급정도, 돌발상황긴급정도 부연설명

2. 어플리케이션 개발 활용

Ex : 버스 위치정보(201)

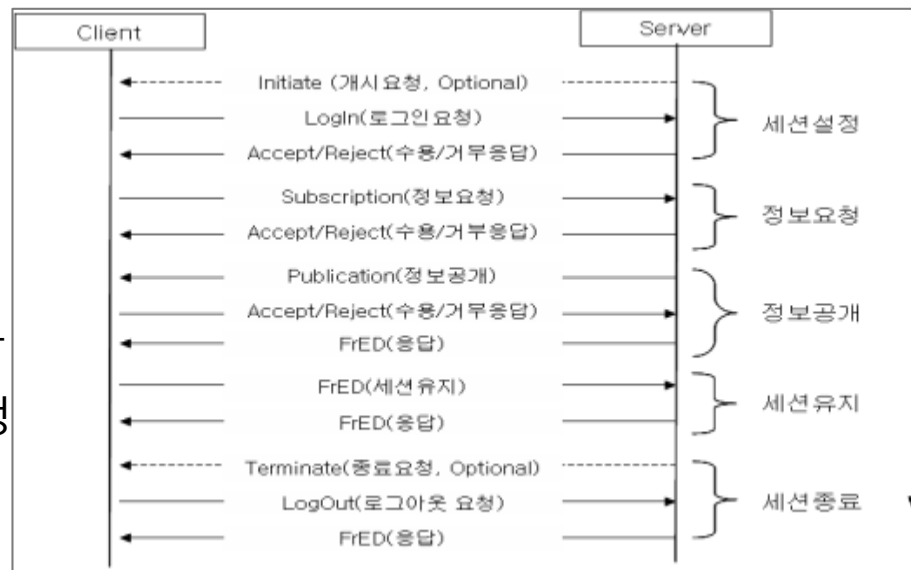
```
BUSLOCATIONINFO {iso(1) standard(0) std14827(14827) part2(2) trafficInformation(0) 1} DEFINITIONS AUTOMATIC TAGS ::= BEGIN
BusLocationInfo ::= SEQUENCE
{
    tsmg-MessageGenerationTime GeneralizedTime,
    tsfc-PTVehicleIDNumber UTF8String(SIZE(1..255)),
    tsvh-NodeZoneIDNumber UTF8String(SIZE(1..40)),
    tsvh-NodeZoneEntryTime GeneralizedTime,
    tsvh-NodeZoneExitTime GeneralizedTime,
    tsvh-NodeZoneTripTime INTEGER(0..1200) OPTIONAL,
    tsfc-LastPTVehicle BOOLEAN OPTIONAL
}
BusInfoTerminalLocation ::= SEQUENCE
{
    tsfc-PTVehicleIDNumber UTF8String(SIZE(1..255)) OPTIONAL,
    tsfc-PTVehicleLocationLatitude INTEGER(-4294967295..-3000000000) OPTIONAL,
    tsfc-PTVehicleLocationLongitude INTEGER(3000000000..4294967295) OPTIONAL,
    tsfc-PTVehicleCollectedTime GeneralizedTime OPTIONAL,
    tsfc-PTVehicleCollectedCycleTime INTEGER(0..3600) OPTIONAL
}
END
```

2. 어플리케이션 개발 활용

교통정보 교환을 위한 국가 표준

- 통신방식 : 센터간 통신방식은 TCP/IP방식으로 규정
- 데이터교환절차 : 국가표준(KS X ISO 14827-2)에 정의된 통신절차를 따름

- 세션설정 : 로그인 과정
- 세션유지 : 주기적으로 Fred 패킷을 전송하여 접속유지
- 세션종료 : 접속 종료
- 정보요청 : Client에서 필요한 정보를 주기적, 이벤트성, 한번요청 인지를 명시하여 요청
- 정보공개 : 요청한 정보를 전송



- 데이터 패킷구성 : 국가표준(KS X ISO 14827-2)에 ASN.1으로 정의된 Datex 데이터 패킷 구조를 따름

Header 부					Data부	Tail부
Datex Version No.	Authentication Info	Data Packet No.	Data Packet Priority No.	Header Option	PDU	CRC

2. 어플리케이션 개발 활용

DATEX 데이터 패킷

Header 부					Data부	Tail부
Datex Version No.	Authentication Info	Data Packet No.	Data PacketPriority No.	Header Option	PDU	CRC

DatexDataPacket ::= SEQUENCE

```

{
    datex-Version-number          ENUMERATED
    {
        experimental (0),
        version1 (1),
        ...
    },
    datex-Data                    OCTET STRING,
    datex-Crc-nbr                 OCTET STRING (SIZE (2))
}

```

C2CAuthenticatedMessage ::= SEQUENCE {

```

    datex-AuthenticationInfo-text    OCTET STRING (SIZE (0..255)),
    datex-DataPacket-number          INTEGER (0..4294967295),
    datex-DataPacketPriority-number  INTEGER (0..10),
    options                          HeaderOptions,
    pdu                              PDUs
}

```

2. 어플리케이션 개발 활용

DATEX 데이터 인코딩 소스 샘플

```
sprintf(strAuthenticationInfo, "TestQuthenticationInfoText");
OCTET_STRING_fromBuf(&p_AuthenticationMessage->datex_AuthenticationInfo_text, strAuthenticationInfo, -1);
strDataPacketNbr = "1";
asn_long2INTEGER(&p_AuthenticationMessage->datex_DataPacket_number, (long)1);
p_AuthenticationMessage->datex_DataPacketPriority_number = iPacketPriorityNbr;
p_AuthenticationMessage->options = *p_Headeroptions;
p_AuthenticationMessage->pdu = *p_PDUs;

ec = der_encode_to_buffer(&asn_DEF_C2CAuthenticatedMessage, p_AuthenticationMessage, buff, &size);
if(ec.encoded == -1) //Encode fail
    ireturn = -1;
else //Encode Success
{
    p_DatexDataPacket->datex_Version_number = datex_Version_number_version1;
    OCTET_STRING_fromBuf(&p_DatexDataPacket->datex_Data, buff, size);
    OCTET_STRING_fromBuf(&p_DatexDataPacket->datex_Crc_nbr, "01", 2);

    ec = der_encode_to_buffer(&asn_DEF_DatexDataPacket, p_DatexDataPacket, buff, &i_buff_size);
    if(ec.encoded == -1) //Encode fail
        ireturn = -1;
    else //Encode Success
    {
        .....
    }
}
```

SAMPLE 데이터 패킷분석 – Fred (1/4)

Intelligent Transport Society of Korea
한국지능형교통체계협회

SAMPLE 데이터 패킷분석 – Fred (2/4)

Protocol	데이터
<pre> DatexDataPacket ::= SEQUENCE { datex-Version-number ENUMERATED { experimental (0), version1 (1), ... }, datex-Data OCTET STRING, datex-Crc-nbr OCTET STRING (SIZE (2)) } </pre>	<pre> 80 01 01 -. 0x80 : Tag -. 0x01 : Length(1) -. 0x01 : Value -> version1 81 81 98 30~~~~~01 -. 0x81 : Tag -. 0x81 0x98 : Length(152) -. 0x30~0x01 : datex-Data 82 02 D5 9F -. 0x82 : Tag -. 0x02 : Length(2) -. 0xD5 0X9F : datex-Crc-nbr </pre>

```
register unsigned short crc16 = 0;
while(len--) {
    crc16 = crctable[ ((crc16 ^ *buf) & 0xff) ] ^ (crc16 >> 8);
    buf++;
}
```

SAMPLE 데이터 패킷분석 – Fred (3/4)

Protocol	데이터
C2CAuthenticatedMessage ::= SEQUENCE { datex-AuthenticationInfo-text OCTET STRING (SIZE (0..255)), datex-DataPacket-number INTEGER (0..4294967295), datex-DataPacketPriority-number INTEGER (0..10), options HeaderOptions, pdu PDUs }	30 81 95 80~~~~~ 01 (데이터 인코딩) -. 0x30 : Tag -. 0x81 0x95 : Length(149) -. 0x80~~~~~0x01 : Value 80 01 03 -> Fred 81 01 05 82 01 01 A3 81 84 80~~~00 A4 03 83~01

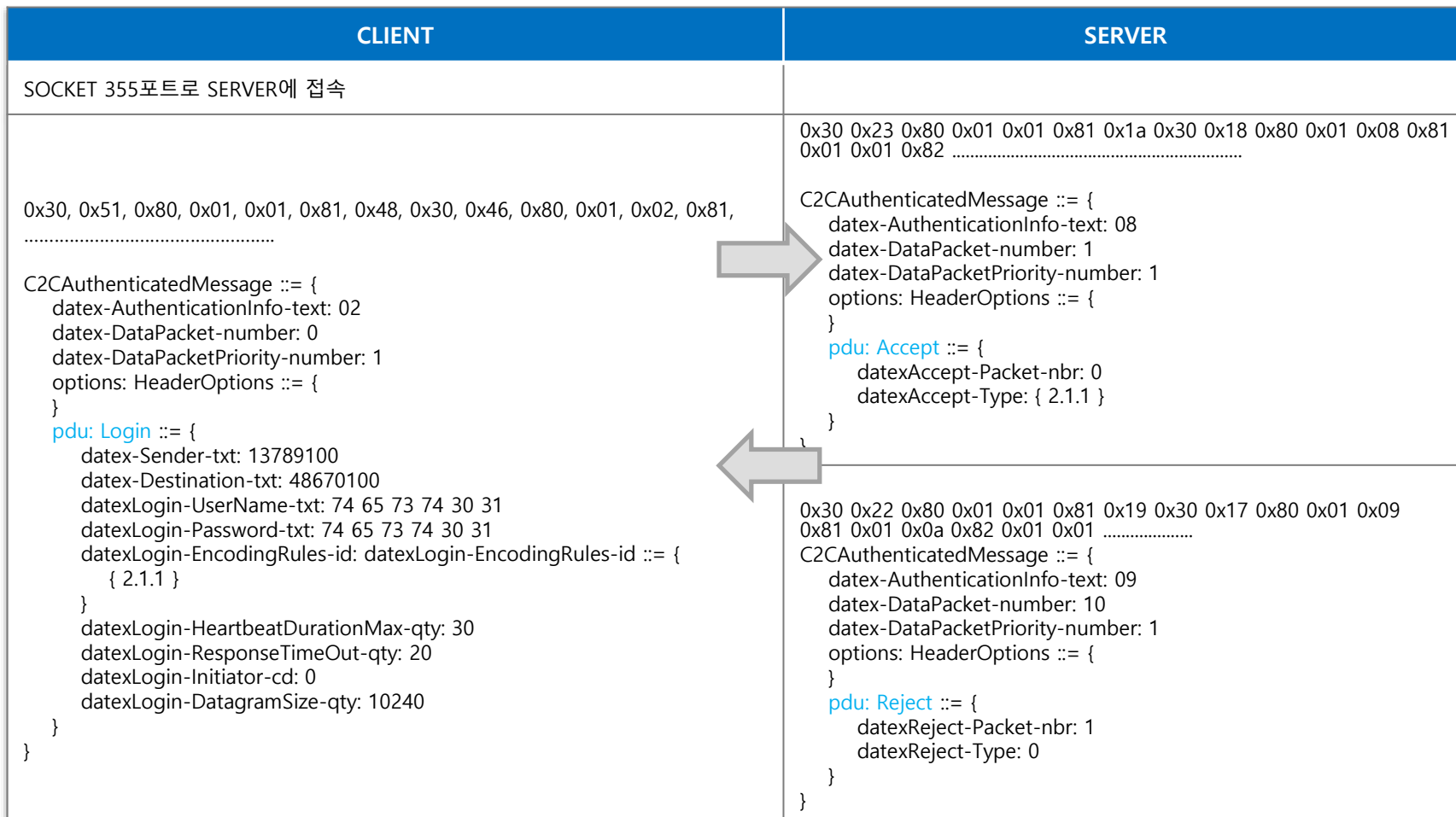
0x01 : Initiate	0x06 : Subscription
0x02 : Login	0x07 : TransferDone
0x03 : Fred	0x08 : Accept
0x04 : Terminate	0x09 : Reject
0x05 : Logout	0x50~0x60 : Publication

SAMPLE 데이터 패킷분석 – Fred (4/4)

Protocol	데이터
HeaderOptions ::= SEQUENCE { datex-Origin-text UTF8String (SIZE (0..40)) OPTIONAL, datex-OriginAddress-location OCTET STRING OPTIONAL, datex-Sender-text UTF8String (SIZE (0..40)) OPTIONAL, datex-SenderAddress-location OCTET STRING OPTIONAL, datex-Destination-text UTF8String (SIZE (0..40)) OPTIONAL, datex-DestinationAddress-location OCTET STRING OPTIONAL, datex-Cost Cost OPTIONAL, datex-DataPacket-time Time OPTIONAL}	80 14 32 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ 00 81 14 32 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ 00 82 14 32 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ 00 83 14 32 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ 00 84 14 32 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ 00 85 14 32 ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ 00 *Cost는 없음 *Time은 없음
PDUs ::= CHOICE { datex-Initiate-null[1] Initiate, login[2] Login, fred[3] FrED, terminate[4] Terminate, logout[5] Logout, subscription[6] Subscription, publication[7] Publication, transfer-done[8] TransferDone, accept[9] Accept, reject[10] Reject }	A4 03 83 01 01 83 01 01 -> Fred에 1을 설정
FrED ::= INTEGER (0..4294967295) -- datexFrED-ConfirmPacket-nbr	

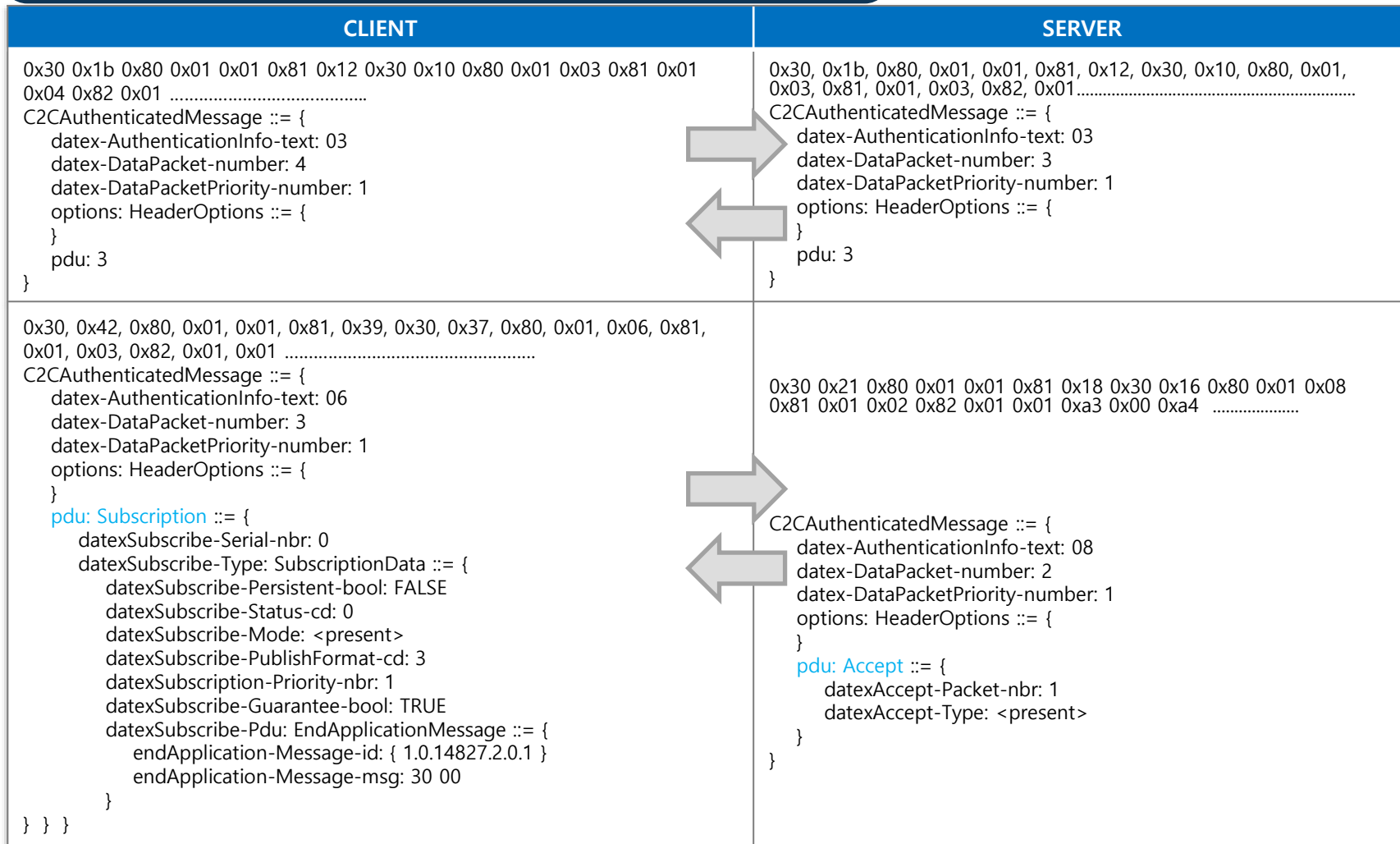
2. 어플리케이션 개발 활용

데이터 접속 및 전송절차 SAMPLE (1/3)



2. 어플리케이션 개발 활용

데이터 접속 및 전송절차 SAMPLE (2/3)



2. 어플리케이션 개발 활용

데이터 접속 및 전송절차 SAMPLE (3/3)

CLIENT	SERVER
<p>0x30, 0x21, 0x80, 0x01, 0x01, 0x81, 0x18, 0x30, 0x16, 0x80, 0x01, 0x08, 0x81, 0x01, 0x04, 0x82, 0x01, 0x01, 0xa3, 0x00, 0xa4,</p> <pre> C2CAuthenticatedMessage ::= { datex-AuthenticationInfo-text: 08 datex-DataPacket-number: 4 datex-DataPacketPriority-number: 1 options: HeaderOptions ::= { } pdu: Accept ::= { datexAccept-Packet-nbr: 4 datexAccept-Type: <present> } } </pre>	<p>0x30 0x23 0x80 0x01 0x01 0x81 0x1a 0x30 0x18 0x80 0x01 0x08 0x81 0x01 0x01 0x82</p> <pre> C2CAuthenticatedMessage ::= { datex-AuthenticationInfo-text: 51 datex-DataPacket-number: 3 datex-DataPacketPriority-number: 1 options: HeaderOptions ::= { } pdu: Publication ::= { datexPublish-Guaranteed-bool: TRUE datexPublish-Format: datexPublish-Data ::= { PublicationData ::= { datexPublish-SubscribeSerial-nbr: 1 datexPublish-Serial-nbr: 1 datexPublish-LatePublicationFlag: TRUE datexPublish-Type: EndApplicationMessage ::= { endApplication-Message-id: { 1.0.14827.2.0.1 } endApplication-Message-msg: 30 67 30 65 80 09 56 48 5F 49 44 5F } } } } } </pre>
<p>0x30, 0x1b, 0x80, 0x01, 0x01, 0x81, 0x12, 0x30, 0x10, 0x80, 0x01, 0x05, 0x81, 0x01, 0x04, 0x82, 0x01, 0x01,</p> <pre> C2CAuthenticatedMessage ::= { datex-AuthenticationInfo-text: 05 datex-DataPacket-number: 4 datex-DataPacketPriority-number: 1 options: HeaderOptions ::= { } pdu: 4 } </pre>	<p>0x30 0x21 0x80 0x01 0x01 0x81 0x18 0x30 0x16 0x80 0x01 0x08 0x81 0x01 0x02 0x82 0x01 0x01 0xa3</p> <pre> C2CAuthenticatedMessage ::= { datex-AuthenticationInfo-text: 08 datex-DataPacket-number: 2 datex-DataPacketPriority-number: 1 options: HeaderOptions ::= { } pdu: Accept ::= { datexAccept-Packet-nbr: 1 datexAccept-Type: <present> } } </pre>

2. 어플리케이션 개발 활용

ASN.1 사용시 주의 할 점

✓ Length 산출시 주의점

- ber_extract_length() 함수의 리턴값은 Value의 길이만을 리턴한 값으로 Length의 가변적인 길이 값을 추가하여 산출하여야함

✓ Long Long형 데이터 입력

- 현재 Long Long형(4바이트 이상) 데이터를 입력할 수 있는 방법이 없으므로 기본 함수를 수정하여 신규 함수를 만들어야 함
- int asn_long2INTEGER(INTEGER_t *st, long value) =>
int asn_long2INTEGER_EDIT(INTEGER_t *st, long long value)

✓ UTF8 Encoding, Decoding

- 한글의 경우 UTF8타입에서 입력 및 출력하는 경우 별도의 인코딩, 디코딩이 필요

✓ 패킷 메모리 해제

- 메모리 해제를 수행하지 않으면 기하급수적으로 메모리 사용량이 늘어나기 때문에 인코딩 및 디코딩 후에는 항상 메모리 해제를 수행

Q & A

(주)동인 : 윤상영

E-mail : young68@donginsm.com