범용 SCADA 게이트웨이 개발을 위한 DNP 3.0 응용 계층 분석

김건웅*, 송병권**, 김새벽**
*목포해양대학교 해양전자통신공학부1),
**서경대학교 정보통신공학과
e-mail:kgu@mmu.ac.kr

A Study on the Application Layer of the DNP3.0 for Development of General SCADA Gateways

Geonug Kim*, Byung-Kwen Song**, Sei-Byuck Kim**
*Division of Electronic & Communication Eng., Mokpo Maritime University
**Information Communication Engineering, SeoKyeong University

요 약

SCADA 시스템과 관련된 많은 표준 프로토콜이 등장하고 있으며, 이들 프로토콜들이 공존하는 형태로 발전할 것으로 예상된다. 본 논문은 범용 SCADA 게이트웨이 개발을 위해 DNP 3.0 응용 계층을 분석한 결과를 소개하고 있다. 여기에는 게이트웨이 설계 시 반드시 고려되어야 하는, DNP3.0 응용 계층의 객체 모델링과 객체 의미, 그리고 통신 기능이 포함된다.

1. 서론

감시제어설비(SCADA: Supervisory Control And Data Acquisition) 시스템은 통신 경로상의 아날로그 또는 디지털 신호를 이용하여 원격장치의 상태 정보 데이터를 RTU(remote terminal unit)로 수집하고 이를 저장, 표시하며, 이를 바탕으로 중앙 제어 시스템에서 원격 장치를 감시 제어하는 시스템이다. 현재 광산, 상하수도, 전력, 연료 관련 업계에서 가장 중요한 영역이라고 볼 수 있다. 현재 SCADA를 위한 많은 프로토콜이 표준화되고 적용되고 있는데, 대표적인 예로 DNP (Distributed Network Protocol) 3.0[1][2][3], IEC(International Electrotechnical Commission)-61850[4][5],, DLMS(Device Language Message Specification)[6][7][8] 등을 들 수 있다.

DNP 3.0은 1990년 IEC875-5에 기초하여 DNP1.0/2.0이 개발되었고, 1993년 DNP3 Basic 4 문서가 발표되면서, DNP User Group이 결성되었으며, 현재 산업계의 사실상의 표준으로서 전기, 석유, 가스 분야에서 널리 사용되고 있다. 특히 한전 SCADA 시스템의 표준 프로토콜로 선정되어 중요도가 증가되고 있다.

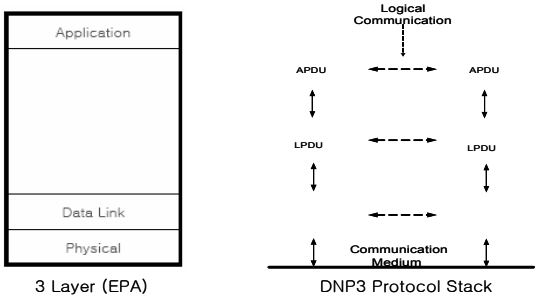
1990년대 초반 미국에서 EPRI를 중심으로 UCA2.0을 개발하였는데, 유럽에서 IEC TC57 WG10, 11, 12를 발족 한 후 IEC 60870, UCA 2.0 기술자들을 모아 개발한 것이 IEC 61850이다. 북미 지역에서 현재 가장 많이 이용하고 있는 프로토콜, 신규 도입시 가장 많이 고려하고 있는 프로토콜이 DNP(Serial/LAN)인데 반해, 북미를 제외한 다른 국가들에서 가장 많이 도입을 고려중인 프로토콜이 IEC61850이다[9]. 따라서 DNP3.0과 IEC61850이 공존하는 상황이 예측되며, 이에 대한 대비가 필요하다.

본 논문에서는 SCADA 시스템의 통신 기능 테스트 및 다른 통신 프로토콜과의 혼용을 위한 게이트웨이 시스템 개발에 필요한 항목을 도출하기 위해 분석한 DNP 3.0 프로토콜 내용을 담고 있다. 본 논문의 구성은 다음과 같다. 먼저 2장에서는 DNP 3.0 프로토콜의 개요와 특징을 살펴본 후 3장에서는 DNP3.0의 응용 계층을 정리한다. 4장에서는 응용 계층의 상태 천이를 소개하며, 5장에서 결론과 향후 연구 방향을 제시한다.

본 연구는 서울시 산학연지원센터 신기술개발과제 연구비로 수행되었음

2. DNP 3.0 프로토콜 개요

DNP3.0 프로토콜은 전기 관련 업계에서 RTU(remote terminal unit)들, IED(Intelligent Electronic Device)들과 마스터 스테이션(master station)들 간에 공개된 표준을 기반으로 상호운용성을 확보하려는 노력의 결과로 만들어진 프로토콜이다[1][2]. DNP3.0은 원래 OSI 7 계층 모델에서 3 가지 계층(물리 계층, 데이터 링크 계층 및 응용 계층)을 기반으로 하여 설계되었다. 다음 그림은 일반적인 시리얼 통신 상에서의 DNP3.0 프로토콜 스택을 보여주고 있다.

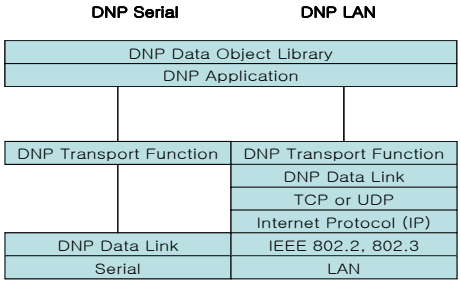


(그림 1) DNP3.0 프로토콜 스택 구조

응용 계층에서는 가장 일반적인 데이터 형태를 지원하기 위해여 객체(object)를 기반으로 한다. 데이터 링크 계층에서는 클래스(class) 및 객체의 변화를 폴링(polling) 하는 몇 가지 데이터 수

집 방법을 제공한다. 물리 계층에서는 가장 일반적으로 사용되는 단순한 RS-232 또는 RS-485 인터페이스를 정의한다.

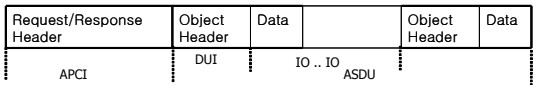
또한 Advanced RTU function들과 기본적으로 IEC document 8702-5-1에 의해 정의된 프레임 크기(최대 16bytes)보다 더 큰 메시지(최대 255bytes)를 전송할 수 있도록 가상 트랜스포트 계층을 사용할 수 있으며, LAN 환경에서의 프로토콜 스택도 고려되어 있다. 다음 그림은 DNS 시리얼과 DNP LAN의 프로토콜 스택을 보여주고 있는데, 원래 DNP 시리얼로 표준이 개발되고 나중에 DNP LAN이 개발된 관계로, TCP 또는 UDP 계층 위에 DNP 데이터 링크 계층이 있는 문제점이 있다.



(그림 2) DNP LAN 프로토콜 스택

3. DNP 3.0 응용 계층

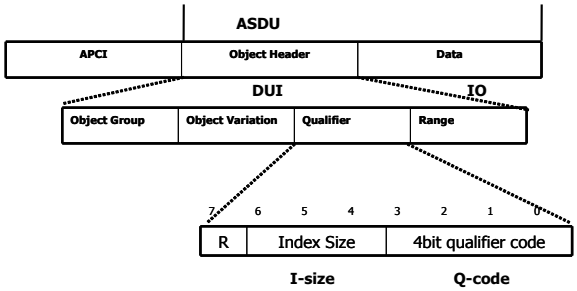
응용계층에서는 사용자 데이터를 받아 ASDU(Application Service Data Unit)을 생성하는데, 하나의 사용자 데이터는 여러 개의 ASDU로 분리도 가능하다. 각각의 ASDU는 헤더에 해당하는 APCI(Application Protocol Control Information)와 결합하여 APDU(Application Protocol Data Unit)를 생성하며, 수신측 응용에서는 하나의 ASDU를 수신했을 경우, 앞의 APCI를 제거하고 ASDU를 어셈블(assemble)하여 완벽한 데이터를 생성한다. 메시지 형식은 다음과 같다. 여기서 객체 헤더는 DUI(Data Unit Identifier)로서 뒤따라오는 데이터 객체를 식별하기 위한 헤더이며, 데이터에는 객체 헤더에서 규정된 형태의 IO(Information Object)들이 포함된다[1][2].



(그림 3) APDU 형식

요청 헤더는 AC(application control)와 FC(function control)로 구성되고, 응답 헤더에는 거기에 IIN(Internal Indication)이 추가된 형태이다. AC에는 첫 번째 분할된 ASDU나 마지막 ASDU를 지칭하는 FIR, FIN과 확인을 요청하는 CON, 그리고 순서번호가 포함된다. FC는 메시지의 목적을 나타내는데, 마스터의 요청과 와 슬레이브의 응답에서 이용되는 코드들은 다음 표1과 같다. IIN은 응답시 처리결과를 나타내기 위한 목적으로 이용된다.

객체 헤더의 DUI는 객체 그룹(Object Group), 객체 변화(Object Variation), 한정값(Qualifier), 범위(Range)로 구성되는데, 여기서 객체 그룹은 일반적인 데이터 클래스를 의미한다. 객체 변화는 요청을 보낼 때는 0으로 설정되며, 응답시 특정한 타입에 따라 그것을 표현하게 된다. 예를 들면 요청에서는 Analog Input으로 변화가 0으로 설정되어 전달되면, 응답에서는 16bits Analog Input으로 구체적인 형식에 대한 변화 값이 돌려진다. 한정값은 뒤에 나오는 범위 값의 의미를 구분하며, 범위에서는 인텍스의 시작값과 종료값을 가지고 있다.



(그림 4) Object Header 형식

표 1. DNP ASDU FC 코드

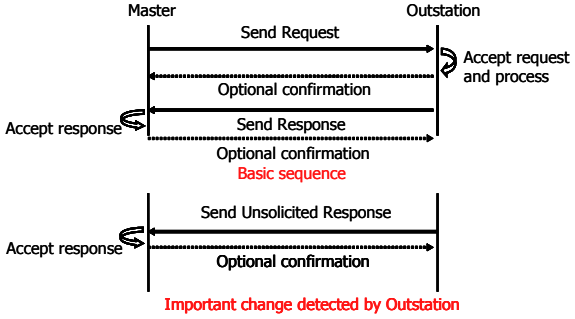
코드	마스터 요청	슬레이브 응답
0x00	Confirm	Confirm
0x01	Read	Response
0x02	Write	Unsolicited Response
0x03	Select	
0x04	Operate	
0x05	Direct Operate	
0x06	Direct Op. No Ack	
0x07	Immediate Freeze	
0x08	Immediate Freeze No Ack	
0x09	Freeze and Clear	
0x0A	Freeze and Clear No Ack	
0x0B	Freeze with Time	
0x0C	Freeze with Time No Ack	
0x0D	Cold Restart	
0x0E	Warm Restart	
0x0F	Init Data to Default	
0x10	Initialize Application	
0x11	Start Application	
0x12	Stop Application	
0x13	Save Configuration	
0x14	Enable Unsolicited Messages	
0x15	Disable Unsolicited Messages	
0x16	Assign Class	
0x17	Delay Measurement	
0x18	Record Current Time	

FC=0x00(CONFIRM) 메시지는 단일 분할 메시지에 대한 확인 응답과 요청된 메시지 수신에 대한 확인 메시지로 이용된다. FC=0x01(READ) 메시지는 Outstation으로부터 Data Object의 정보를 얻어내는데 이용되는데, 다양한 경우에 대해 요청과 응답이 정의되어 있다. FC=0x02(WRITE) 메시지는 Master station이 Out station으로 object를 전송할 때 이용된다. FC=0x03(SELECT) 메시지는 제어 장치를 선택할 때 사용되는데, 이것은 특정 장치를 선정하여 대기상태로 만들고 그 결과를 보고받는다. 이를 받은 Out station은 타이머를 개시하는데, 타임아웃 전에 FC=0x04 (OPERATE) 메시지가 수신되어야 정상적으로 활성화된다. FC=0x04(OPERATE)는 하나 이상의 제어장치를 활성화 시키는데 이용된다. FC=0x05(DIRECT OPERATE)의 경우 제어장치가 SELECT 메시지를 미리 수신할 필요 없이 바로 활성화 시키는데 이용된다. FC=0x06(Direct OPERATE. No Ack)는 Out station이 응답할 필요가 없는 직접 제어 메시지이다. FC=0x07, 0x08(IMMEDIATE

FREEZE) 메시지는 버퍼에 특정 객체의 값을 복사하는데 이용되는데 응답의 필요여부에 따라 구별된다. FC=0x09, 0x0A(FREEZE AND CLEAR)는 FC=0x07, 0x08과 동일한데, 복사 후 객체 값을 0으로 설정하는 점이 다르다. FC=0x0B, 0x0C(FREEZE WITH TIME)는 특정 시간이나 주기적으로 객체 값을 얻는데 이용된다. FC=0x0D(COLD START)는 Out station에서 전원 차단과 연결을 통해 응용을 다시 시작하도록 하는 것이며, FC=0x0E(WARM RESTART)는 전원 차단 없이 응용만 재시작 시키는데 이용된다. FC=0x0F(INITIALIZE DATA)는 설정 가능한 데이터를 초기화하거나 기본 값으로 설정하는데 이용되며, FC=0x10(INITIALIZE APPLICATION)은 응용을 초기화시키는데 이용하고, FC=0x11(APPLICATION START), FC=0x12(APPLICATION STOP)은 특정 응용의 시작과 종료에 이용되는데, Out station은 이러한 동작의 성공 여부를 응답으로 보내야 한다. FC=0x13(SAVE CONFIG)는 특정 구성정보를 비휘발성 저장공간에 저장하도록 하며, FC=0x14, 0x15는 객체들에 대한 정보를 자동으로 보고하는 기능을 활성화하거나 비활성화하는데 이용된다. FC=0x16(ASSIGN CLASS)는 클래스에 객체를 할당하는데 이용하며, FC=0x17(DELAY MEASUREMENT)는 통신 지연 측정에 이용되며, FC=0x18(RECORD CURRENT TIME)은 현재 시각의 저장에 이용된다.

객체 클래스는 고정 데이터 객체와 사건 데이터 객체로 분류되며, 고정 데이터 객체는 외부 스테이션의 현재 값을 반영하며 클래스 0이 할당된다. 사건 데이터 객체는 데이터의 변화나 기타 다른 요인으로 인해 생성되는 객체로서 클래스 1,2,3으로 할당이 가능하다. 외부 스테이션 장치 프로파일에는 클래스들로 구성되며, 데이터를 할당하는 정보와 설정이 기술된다.

응용 계층 메시지 교환은 2가지가 제공되는데, 마스터에서 요청을 하고, 이에 대한 응답을 외부 스테이션에서 보내는 방식과, 외부 스테이션에서 감지된 변화를 스스로 보내는 방식이 있다.



(그림 5) DNP 3.0 응용 계층 메시지 교환

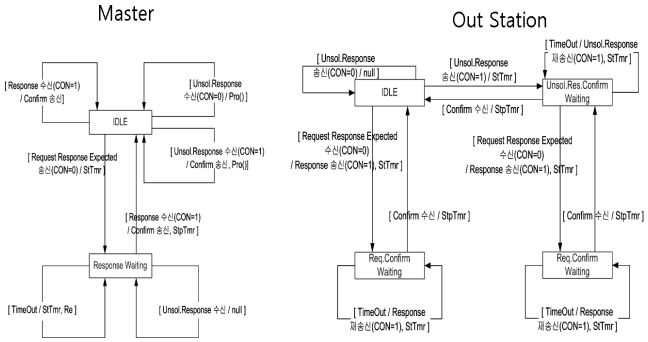
마스터와 슬레이브간 요청/응답 트랜잭션은 다른 요청이 슬레이브에 전달되기 전 완전히 처리되어야 하며, 마스터는 요청을 내보낸 후 응답을 받기 전에 자발적으로 온 메시지를 받을 수도 있다.

4. DNP 3.0 응용 계층 상태 천이

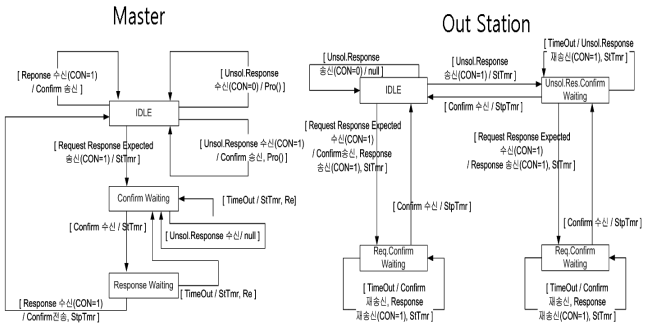
DNP 응용 계층에서 메시지 처리 모드는 2가지가 존재하는데, IMMEDIATE_PROCESS_mode에서는 마스터의 요청과 외부 스테이션의 Unsol.response가 동시에 발생(충돌)한 경우, 외부 스테이션에서 마스터의 요청을 우선 처리한다. 그러나 이진 입력 데이터나 카운터 이벤트 데이터 등과 같은 시스템 데이터에 대한 읽기 요구는 제외된다. PROCESS_AFTER_CONFIRM_mode에서는 Unsol.response에 대한 확인을 기다렸다가 수신된 후 처리한다[1][2].

요청 메시지에서 응답이 요구되는 경우 Application Control field의 CON값은 자유롭게 설정할 수 있지만 응답을 기대하지 않는 요청에서는 반드시 CON값을 1로 설정해야 한다. 응답 메시지에서는 요청에 대한 응답인 경우 반드시 CON값을 1로 설정해야 하며, Unsol.response인 경우에는 자유롭게 설정할 수 있다.

다음은 IMMEDIATE_PROCESS_mode에서 응답이 요구되는 요청시 CON 값이 0으로 설정된 경우와 CON이 1로 설정된 경우의 상태 천이도이다.

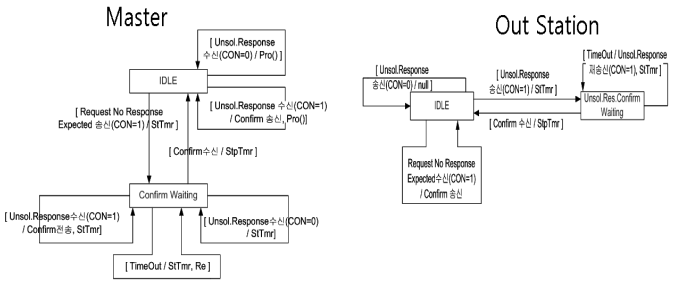


(그림 6) Request Response Expected & CON=0



(그림 7) Request Response Expected & CON=1

다음은 PROCESS_AFTER_CONFIRM_mode에서 응답이 요구되지 않는 요청을 처리하는 경우의 상태 천이도이다.



(그림 8) Request No Response Expected & CON=1

전체적으로 나타나는 상태(state)는 다음과 같은 4가지이다.

- IDLE => 0
- Request confirm waiting =>1

- Request Response waiting => 2
- Unsol.Response confirm waiting => 3

또한 사건(event)은 다음과 같이 14가지가 있다.

- Request Response Expected송신 (CON=0)
- Request Response Expected송신 (CON=1)
- Request No Response Expected송신 (CON=1)
- Request Response Expected수신 (CON=0)
- Request Response Expected수신 (CON=1)
- Request No Response Expected수신 (CON=1)
- Response송신 (CON=1)
- Response수신 (CON=1)
- Unsol.Response송신 (CON=0)
- Unsol.Response송신 (CON=1)
- Unsol.Response수신 (CON=0)
- Unsol.Response수신 (CON=1)
- Confirm 수신
- TimeOut

또한 동작(action)은 다음과 같이 12가지가 있다.

- Request Response Expected재송신 (CON=0)
- Request Response Expected재송신 (CON=1)
- Request No Response Expected재송신 (CON=1)
- Response송신 (CON=1)
- Response재송신 (CON=1)
- Confirm송신
- Confirm재송신
- StTmr
- StpTmr
- Pro() – 내부적인처리
- Null
- Unsol.Response재송신 (CON=1)

다음은 예시로 제시한 IMMEDIATE_PROCESS_mode에서 응답이 요구되는 요청시 CON이 1로 설정된 경우 외부 스테이션의 상태 천이표이다.

사건 상태	RequestResponseExpected수신 (CON=0)	Unsol.Response발생 (CON=0)	Unsol.Response 발생 (CON=1)	Confirm 수신	TimeOut
IDLE	Response 송신(CON=1), StTmr / Request Confirm Waiting	NULL / IDLE	StTmr / Unsol.Response Confirm Waiting	예외처리 (오류) / IDLE	예외처리 (오류) / IDLE
RequestConfirmwaiting	예외처리 (오류) / Request Confirm waiting	예외처리 (오류) / Request Confirm waiting	예외처리 (오류) / Request Confirm waiting	StpTmr/ IDLE	Response 재송신 (CON=1), StTmr / Request Confirm waiting
Unsol.ResponsConfirmWaiting	Response 송신(CON=1), StTmr / Request Confirm waiting	예외처리 (오류) / Unsol.Response Confirm Waiting	예외처리 (오류) / Unsol.Response Confirm Waiting	StpTmr/ IDLE	Unsol.Response 재송신 (CON=1), StTmr / Unsol.Response Confirm Waiting

5. 결론 및 추후 연구 방향

게이트웨이의 역할은 A 표준을 이용하는 시스템과 B 표준을 이용하는 시스템간 의미있는 데이터를 교환할 수 있도록 하는 것이다. 이를 위해서는 2가지가 고려되어야 하는데, 먼저 교환하는 데이터의 의미를 일치시켜야 하며, 다음 두 표준의 프로토콜을 동시에 지원하여 매핑을 시켜야 한다.

DNP3.0에서의 객체 그룹은 서로 유사한 데이터 타입의 집합으로 볼 수 있다. 외부 스테이션 구현 시 동일한 타입을 갖는 데이터들이 배열 형태로 모여서 관리하고, 이러한 데이터를 요구할 때 데이터 타입과 범위를 지정하여 원하는 정보를 수집하고 있다. 이러한 객체들과 의미 정보를 기술한 것이 외부 스테이션 장치 프로파일이다. DNP3.0은 호환성보다는 구현의 용이성이나 자원의 효율적인 이용에 초점이 맞추어져 있어서 장치마다 데이터 구성이 다를 수 있다. 따라서 장치별로 데이터간 매핑을 위한 표가 따로 존재할 필요가 있다.

두 개의 통신 프로토콜을 동시에 지원하여 통신을 중계하고자 하는 경우, 요청에 대한 응답 방식은 일대일 매핑으로 바로 처리될 수 있지만, Unsol-Response는 또 다른 처리가 요구될 수 있다. DNP3.0에서 Unsol-Response를 지원하여 기존의 주기적인 폴링 방식에서 자발적인 보고 방식으로 진화한 것은 관리를 용이하게 한 측면이 있으나, 그러한 보고 방식을 설정하거나 조건을 부여하는 방식, 보고 주기 등이 표준마다 서로 상이하며, 따라서 이들을 일치시키는 작업이 필요할 수 있다.

본 연구의 최종 목표인 범용 SCADA 시스템 개발을 위해 먼저 프로토콜간 매핑 기능을 제공하고, 다음 관리하고자 하는 데이터들의 의미를 일치시키는 작업을 진행하고자 한다. 현재 DNP3.0, IEC61850, DLMS 등 SCADA 표준의 분석과 통신 기능의 구현 작업이 진행 중이며, 이를 기반으로 범용 SCADA 시스템을 설계할 예정이다.

참고문헌

[1] DNP User Group, "Distributed Network Protocol DNP 3.0 BASIC 4 DOCUMENT SET"
[2] DNP User Group, "DNP3 Protocol Primer"
[3] www.dnp.org
[4] IEC Technical Report 61850-(1 ~ 10), 2003
[5] Ralph Mackiewicz, "IEC61850 & ICCP-TASE.2 Technical Overview", SISCO
[6] IEC, COSEM Application layer, IEC 62056-53, 2006
[7] IEC, Object identification system (OBIS), IEC 62056-61. 2006
[8] Parasanth Gopalakrishnan, "Need for Open Communication Standard for metering and suitability of DLMS-COSEM", Kalki Comm., 2005
[9] "Increases in Substation Related Automation and Integration Program Spending Reported by World's Major Electric Power Utilities", Substation Automation News – March 2006