**블록체인 기반 생산 및 품질이력**

**정보 관리 기술 검증**

**응용서비스 연계 프로토콜 정의서**

|  |  |
| --- | --- |
| **문서코드** | HD-P-MW-002 |
| **버전** | V0.1.3 |
| **작성일** | 2019-06-25 |
| **작성자** | 강 대현 |

**개정 이력**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **버전** | **개정일자** | **¹개정사유** | **²개정내역** | **개정자** | **승인자** |
| **0.1.0** | **2019/06/04** | **추가** | **최초 작성** | **강대현** |  |
| **0.1.1** | **2019/06/19** | **수정** | **조회, 일괄정지 Reponse에 event.id 기재 위치가 잘못되어 있어 결과의 배열 항목으로 이동** | **강대현** |  |
| **0.1.2** | **2019/06/24** | **삭제, 추가** | **프로토콜상의 publish를 위한 수집명령별 통신포트는 페이로드의 topic으로 관하도록 변경되어 publish 통신포트는 프로토콜상 불필요하여 삭제 및 수집정보 프로토콜 내용 기재.** | **강대현** |  |
| **0.1.3** | **2019/06/25** | **수정** | **PLC수집 일괄정지 프로토콜의 stop.all의 plc.ip, plc.port, polling.period가 불필요하여 삭제 후 event.id 및proc.date만 넣도록 수정.**  **Method “stop”을 “stop.all”과의 계층 구조 일치화를 위해 “stop.part”로 변경** | **김동범** |  |
| **0.1.4** | **2019/09/04** | **추가** | **요구사항 추가로 인한 PLC읽기 프로토콜 추가(6번 항목)** | **강대현** |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**¹개정 사유 : 개정 또는 개정 내용이 이전 문서에 대해 추가/수정/삭제인지 선택 기입**

**²개정 내역 : 개정이 발생하는 페이지 번호와 변경 내용을 기술**

목차

1.PLC수집 시작 프로토콜 4

1.1.PLC수집 시작 Request 프로토콜(응용서비스 -> M/W) 4

1.2.PLC수집 시작 Response 프로토콜(M/W -> 응용서비스) 6

2.PLC 수집 정지 프로토콜 8

2.1.PLC수집 정지 Request 프로토콜(응용서비스 -> M/W) 8

2.2.PLC수집 정지 Response 프로토콜(M/W -> 응용서비스) 9

3.PLC 수집 조회 프로토콜 11

3.1.PLC수집 조회 Request 프로토콜(응용서비스 -> M/W) 11

3.2.PLC수집 조회 Response 프로토콜(M/W -> 응용서비스) 12

4.PLC 수집 일괄정지 프로토콜 14

4.1.PLC수집 일괄정지 Request 프로토콜(응용서비스 -> M/W) 14

4.2.PLC수집 일괄정지 Response 프로토콜(M/W -> 응용서비스) 15

5.PLC 수집 정보 Publish 프로토콜 16

5.1.PLC수집 정보 Publish 프로토콜(M/W -> 응용서비스) 16

6.PLC 읽기 프로토콜 18

6.1.PLC 읽기 Request 프로토콜(응용서비스 -> M/W) 18

6.2.PLC 읽기 Response 프로토콜(M/W -> 응용서비스) 19

# PLC수집 시작 프로토콜

* 응용서비스와 미들웨어 간 PLC수집 시작 명령을 위한 MQ(REQ-RES)기반의 프로토콜로 해당 명령 수신 시, 공정 정보 수집을 위한 미들웨어 Handler 시작과 정보 전달을 위한 MQ PUB 서버 기동 및 Publish를 개시(Event Context)한 후 처리 결과(Response)를 응답한다.
* 전송 데이터는 JSON표준 형식으로 전송
* 전송 문자 타입은 UTF-8 인코딩 사용.
* 전송 종료 코드는 CR LF ('\r\n')를 사용.

## PLC수집 시작 Request 프로토콜(응용서비스 -> M/W)

* **Request JSON 스키마**

{

“jsonrpc” : “2.0”,

“method” : “start”,

“id” : {“type” : “string”},

“param” : {

“protocol.version” : {“type” : “string”},

“event.id” : {“type” : “string”},

“plc.ip” : {“type” : “string”},

“plc.port” : {“type” : “string”},

“polling.period” : {“type” : “string”},

~~“publish.port” : {“type” : “string”},~~

“items” : [

{ “key” : {“type” : “string”},

“device.code” : {“type” : “string”},

“device.num” : {“type” : “string”},

“device.score” : {“type” : “string”}

}]

}

}\r\n

* **Request 항목 상세**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **항목명** | | | **항목명(한글)** | **필수** | **설명** | **비고** |
| 1 | jsonrpc | | | Json rpc 버전 | O | Json rpc 프로토콜 버전 | 2.0 |
| 2 | method | | | 기능명 | O | 처리 | start |
| 3 | id | | | 통신ID | O | Req-Res 통신 ID |  |
| 4 | param | | | 파라미터 | O | 기능별 입력 파라미터 | Object |
| 5 |  | protocol.version | | 프로토콜 버전 | O | 프로토콜 버전 |  |
| 6 | event.id | | 이벤트 ID | O | 명령처리의 이벤트ID | 수집 시작/정지 및 조회 처리에 대한 Identifier로 사용 |
| 7 | plc.ip | | PLC 아이피 주소 | O | PLC연결을 위한 IP Address |  |
| 8 | plc.port | | PLC 아이피 포트 | O | PLC연결을 위한 Port Number |  |
| 9 | polling.period | | 수집주기 | O | 정보 수집 주기(s) |  |
| ~~10~~ | ~~publish.port~~ | | ~~수집정보 통신 포트~~ | ~~O~~ | ~~수집정보 전달을 위한 MQ 통신 포트~~ |  |
| 11 | items | | 아이템 정보 배열 | O | 정보 수집을 위한 아이템 정보 배열 | Array |
| 12 |  | key | 아이템명 | O | 수집 항목의 이름 |  |
| 13 |  | device.code | PLC 디바이스 코드 | O | PLC 디바이스 영역에 따른 코드 | \*액세스 가능 디바이스 일람 참조 |
| 14 | device.num | PLC 디바이스 번호 | O | PLC 디바이스의 메모리 번호(16진수 어드레스) | \*액세스 가능 디바이스 일람 참조 |
| 15 |  | device.score | PLC 디바이스 스코어 | O | PLC 디바이스의 메모리 스코어 | \*최대 사용가능 스코어 참조 |

* **Request JSON 샘플**

{

“jsonrpc” : “2.0”,

“method” : “start”,

“id” : “1”,

“param” : {

“protocol.version” :”1.0”,

“event.id” : “event1”,

“plc.ip” : “127.0.0.1”,

“plc.port” : “8192”,

“polling.period” : “3”,

~~“publish.port” : ”5000”,~~

“items” : [

{ “key” : “lot”,

“device.code” : “D\*”,

“device.num” : “10000”,

“device.score” : “4”

},

{ “key” : “quality”,

“device.code” : “D\*”,

“device.num” : “10004”,

“device.score” : “1”

}]

}

}\r\n

## PLC수집 시작 Response 프로토콜(M/W -> 응용서비스)

* **Response JSON 스키마**

{

“jsonrpc” : “2.0”,

“id” : {“type” : “string”},

“result” : {

“protocol.version” : {“type” : “string”},

“event.id” : {“type” : “string”},

“proc.data” : {“type” : “string”}

},

“error” : {

“code” : {“type” : “number”},

“message” : {“type” : “string”}

}

}\r\n

* **Response 항목 상세**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **#** | **항목명** | | **항목명(한글)** | **필수** | **설명** | **비고** |
| 1 | jsonrpc | | Json rpc 버전 | O | Json rpc 프로토콜 버전 | 2.0 |
| 2 | id | | 통신ID | O | Req-Res 통신 ID |  |
| 3 | result | | 결과정보 | X |  | 장애 발생시 ‘null’ |
| 4 |  | protocol.version | 프로토콜 버전 | O | 프로토콜 버전 |  |
| 5 |  | event.id | 이벤트 ID | O | 명령 처리의 이벤트ID |  |
| 6 |  | proc.data | 처리시간 | O | 명령 처리시간 | yyyymmdd hh:mm:ss.SSS |
| 7 | error | | 에러정보 | X | 장애 발생시 에러정보 | 정상 처리시 ‘null’ |
| 8 |  | code | 에러코드 | O | 에러코드 |  |
| 9 |  | message | 에러메세지 | O | 에러메세지 |  |

* **Response JSON 샘플**

{

“jsonrpc” : “2.0”,

“id” : “1”,

“result” : {

“protocol.version” : “1.0”,

“event.id” : “event1”,

“proc.data” : “20190801 17:32:54.100”

},

“error” : {

“code” : -32000,

“message” : “PLC connection failed.”

}

}\r\n

# PLC 수집 정지 프로토콜

* 응용서비스와 미들웨어 간 PLC수집 정지 명령을 위한 MQ(REQ-RES)기반의 프로토콜
* 전송 데이터는 JSON표준 형식으로 전송
* 전송 문자 타입은 UTF-8 인코딩 사용.
* 전송 종료 코드는 CR LF ('\r\n')를 사용.

## PLC수집 정지 Request 프로토콜(응용서비스 -> M/W)

* **Request JSON 스키마**

{

“jsonrpc” : “2.0”,

“method” : “stop.part”,

“id” : “1”,

“param” : {

“protocol.version” :”1.0”,

“event.id” : “event1”,

}

}\r\n

* **Request 항목 상세**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **#** | **항목명** | | **항목명(한글)** | **필수** | **설명** | **비고** |
| 1 | jsonrpc | | Json rpc 버전 | O | Json rpc 프로토콜 버전 | 2.0 |
| 2 | method | | 기능명 | O | 처리 | stop.part |
| 3 | id | | 통신ID | O | Req-Res 통신 ID |  |
| 4 | param | | 파라미터 | O | 기능별 입력 파라미터 | Object |
| 5 |  | protocol.version | 프로토콜 버전 | O | 프로토콜 버전 |  |
| 6 | event.id | 이벤트 ID | O | 명령처리의 이벤트ID | 수집 시작/정지 및 조회 처리에 대한 Identifier로 사용 |

* **Request JSON 샘플**

{

“jsonrpc” : “2.0”,

“method” : “stop.part”,

“id” : “1”,

“param” : {

“protocol.version” :”1.0”,

“event.id” : “event1”,

}

}\r\n

## PLC수집 정지 Response 프로토콜(M/W -> 응용서비스)

* **Response JSON 스키마**

{

“jsonrpc” : “2.0”,

“id” : {“type” : “string”},

“result” : {

“protocol.version” : {“type” : “string”},

“event.id” : {“type” : “string”},

“proc.data” : {“type” : “string”}

},

“error” : {

“code” : {“type” : “number”},

“message” : {“type” : “string”}

}

}\r\n

* **Response 항목 상세**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **#** | **항목명** | | **항목명(한글)** | **필수** | **설명** | **비고** |
| 1 | jsonrpc | | Json rpc 버전 | O | Json rpc 프로토콜 버전 | 2.0 |
| 2 | id | | 통신ID | O | Req-Res 통신 ID |  |
| 3 | result | | 결과정보 | X |  | 장애 발생시 ‘null’ |
| 4 |  | protocol.version | 프로토콜 버전 | O | 프로토콜 버전 |  |
| 5 |  | event.id | 이벤트 ID | O | 명령 처리의 이벤트ID |  |
| 6 |  | proc.data | 처리시간 | O | 명령 처리시간 | yyyymmdd hh:mm:ss.SSS |
| 7 | error | | 에러정보 | X | 장애 발생시 에러정보 | 정상 처리시 ‘null’ |
| 8 |  | code | 에러코드 | O | 에러코드 |  |
| 9 |  | message | 에러메세지 | O | 에러메세지 |  |

* **Response JSON 샘플**

{

“jsonrpc” : “2.0”,

“id” : “1”,

“result” : {

“protocol.version” : “1.0”,

“event.id” : “event1”,

“proc.data” : “20190801 17:32:54.100”

},

“error” : {

“code” : -32000,

“message” : “PLC connection failed.”

}

}\r\n

# PLC 수집 조회 프로토콜

* 응용서비스와 미들웨어 간 PLC수집 상태 조회를 위한 MQ(REQ-RES)기반의 프로토콜
* 전송 데이터는 JSON표준 형식으로 전송
* 전송 문자 타입은 UTF-8 인코딩 사용.
* 전송 종료 코드는 CR LF ('\r\n')를 사용.

## PLC수집 조회 Request 프로토콜(응용서비스 -> M/W)

* **Request JSON 스키마**

{

“jsonrpc” : “2.0”,

“method” : “status”,

“id” : {“type” : “string”},

“param” : {

“protocol.version” : {“type” : “string”}

}

}\r\n

* **Request 항목 상세**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **#** | **항목명** | | **항목명(한글)** | **필수** | **설명** | **비고** |
| 1 | jsonrpc | | Json rpc 버전 | O | Json rpc 프로토콜 버전 | 2.0 |
| 2 | method | | 기능명 | O | 처리 | status |
| 3 | id | | 통신ID | O | Req-Res 통신 ID |  |
| 4 | param | | 파라미터 | O | 기능별 입력 파라미터 | Object |
| 5 |  | protocol.version | 프로토콜 버전 | O | 프로토콜 버전 |  |

* **Request JSON 샘플**

{

“jsonrpc” : “2.0”,

“method” : “status”,

“id” : “1”,

“param” : {

“protocol.version” :”1.0”

}

}\r\n

## PLC수집 조회 Response 프로토콜(M/W -> 응용서비스)

* **Response JSON 스키마**

{

“jsonrpc” : “2.0”,

“id” : {“type” : “string”},

“result” : {

“protocol.version” : {“type” : “string”},

“status” : [{

“event.id” : {“type” : “string”},

“plc.ip” : {“type” : “string”},

“plc.port” : {“type” : “string”},

“polling.period” : {“type” : “string”}

~~“publish.port” : {“type” : “string”}~~

}]

},

“error” : {

“code” : {“type” : “number”},

“message” : {“type” : “string”}

}

}\r\n

* **Response 항목 상세**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **항목명** | | | **항목명(한글)** | **필수** | **설명** | **비고** |
| 1 | jsonrpc | | | Json rpc 버전 | O | Json rpc 프로토콜 버전 | 2.0 |
| 2 | id | | | 통신ID | O | Req-Res 통신 ID |  |
| 3 | result | | | 결과정보 | X |  | 장애 발생시 ‘null’ |
| 4 |  | protocol.version | | 프로토콜 버전 | O | 프로토콜 버전 |  |
| 5 |  | status | | 상태정보 | O | 수집(기동)중인 PLC정보 | Array  수집중인 PLC가 없을 경우 ‘null’ |
| 6 |  |  | event.id | 이벤트 ID | O | 명령 처리의 이벤트ID |  |
| 7 |  |  | plc.ip | PLC 아이피 주소 | O | PLC연결을 위한 IP Address |  |
| 8 |  |  | plc.port | PLC 아이피 포트 | O | PLC연결을 위한 Port Number |  |
| 9 |  |  | polling.period | 수집주기 | O | 정보 수집 주기(s) |  |
| ~~10~~ |  |  | ~~publish.port~~ | ~~수집정보 통신 포트~~ | ~~O~~ | ~~수집정보 전달을 위한 MQ 통신 포트~~ |  |
| 11 | error | | | 에러정보 | X | 장애 발생시 에러정보 | 정상 처리시 ‘null’ |
| 12 |  | code | | 에러코드 | O | 에러코드 |  |
| 13 |  | message | | 에러메세지 | O | 에러메세지 |  |

* **Response JSON 샘플**

{

“jsonrpc” : “2.0”,

“id” : “1”,

“result” : {

“protocol.version” : “1.0”,

“status” : [{

“event.id” : “event1”,

“plc.ip” : “192.168.0.100”,

“plc.port” : “8001”,

“polling.period” : “3”

~~“publish.port” : “5000”~~

},

{

“event.id” : “event2”,

“plc.ip” : “192.168.0.101”,

“plc.port” : “8002”,

“polling.period” : “5”,

“publish.port” : “5005”

{

}

}\r\n

# PLC 수집 일괄정지 프로토콜

* 응용서비스와 미들웨어 간 PLC수집 일괄 정지 명령을 위한 MQ(REQ-RES)기반의 프로토콜
* 전송 데이터는 JSON표준 형식으로 전송
* 전송 문자 타입은 UTF-8 인코딩 사용.
* 전송 종료 코드는 CR LF ('\r\n')를 사용.

## PLC수집 일괄정지 Request 프로토콜(응용서비스 -> M/W)

* **Request JSON 스키마**

{

“jsonrpc” : “2.0”,

“method” : “stop.all”,

“id” : {“type” : “string”},

“param” : {

“protocol.version” : {“type” : “string”},

}

}\r\n

* **Request 항목 상세**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **#** | **항목명** | | **항목명(한글)** | **필수** | **설명** | **비고** |
| 1 | jsonrpc | | Json rpc 버전 | O | Json rpc 프로토콜 버전 | 2.0 |
| 2 | method | | 기능명 | O | 처리 | stop.all |
| 3 | id | | 통신ID | O | Req-Res 통신 ID |  |
| 4 | param | | 파라미터 | O | 기능별 입력 파라미터 | Object |
| 5 |  | protocol.version | 프로토콜 버전 | O | 프로토콜 버전 |  |

## PLC수집 일괄정지 Response 프로토콜(M/W -> 응용서비스)

* **Response JSON 스키마**

{

“jsonrpc” : “2.0”,

“id” : {“type” : “string”},

“result” : {

“protocol.version” : {“type” : “string”},

“stop.all” : [{

“event.id” : {“type” : “string”},

“proc.data” : {“type” : “string”}

~~“plc.ip” : {“type” : “string”},~~

~~“plc.port” : {“type” : “string”},~~

~~“polling.period” : {“type” : “string”}~~

~~“publish.port” : {“type” : “string”}~~

}]

},

“error” : {

“code” : {“type” : “number”},

“message” : {“type” : “string”}

}

}\r\n

* **Response 항목 상세**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **항목명** | | | **항목명(한글)** | **필수** | **설명** | **비고** |
| 1 | jsonrpc | | | Json rpc 버전 | O | Json rpc 프로토콜 버전 | 2.0 |
| 2 | id | | | 통신ID | O | Req-Res 통신 ID |  |
| 3 | result | | | 결과정보 | X |  | 장애 발생시 ‘null’ |
| 4 |  | protocol.version | | 프로토콜 버전 | O | 프로토콜 버전 |  |
| 5 |  | stop.all | | 일괄정지 PLC정보 | O | 일괄정지한 PLC정보 | Array  정지 대상 PLC가 없을 경우 ‘null’ |
| 6 |  |  | event.id | 이벤트 ID | O | 명령 처리의 이벤트ID |  |
| 10 |  |  | proc.date | 처리시간 | O | 명령 처리시간 | yyyymmdd hh:mm:ss.SSS |
| 11 | error | | | 에러정보 | X | 장애 발생시 에러정보 | 정상 처리시 ‘null’ |
| 12 |  | code | | 에러코드 | O | 에러코드 |  |
| 13 |  | message | | 에러메세지 | O | 에러메세지 |  |

# PLC 수집 정보 Publish 프로토콜

* 응용서비스와 미들웨어 간 PLC수집정보 전달을 위한 MQ(PUB-SUB)기반의 Broadcast용 publish 프로토콜.
* 전송 데이터는 JSON표준 형식으로 전송
* 전송 문자 타입은 UTF-8 인코딩 사용.
* 전송 종료 코드는 CR LF ('\r\n')를 사용.
* topic은 event.id를 사용.

## PLC수집 정보 Publish 프로토콜(M/W -> 응용서비스)

* Publish JSON 스키마

{

“jsonrpc” : “2.0”,

“id” : {“type” : “string”},

“result” : {

“protocol.version” : {“type” : “string”},

“event.id” : {“type” : “string”},

“proc.data” : {“type” : “string”}

“items” : [

{ “key” : {“type” : “string”},

“value” : {“type” : “string”}

}],

}

“error” : {

“code” : {“type” : “number”},

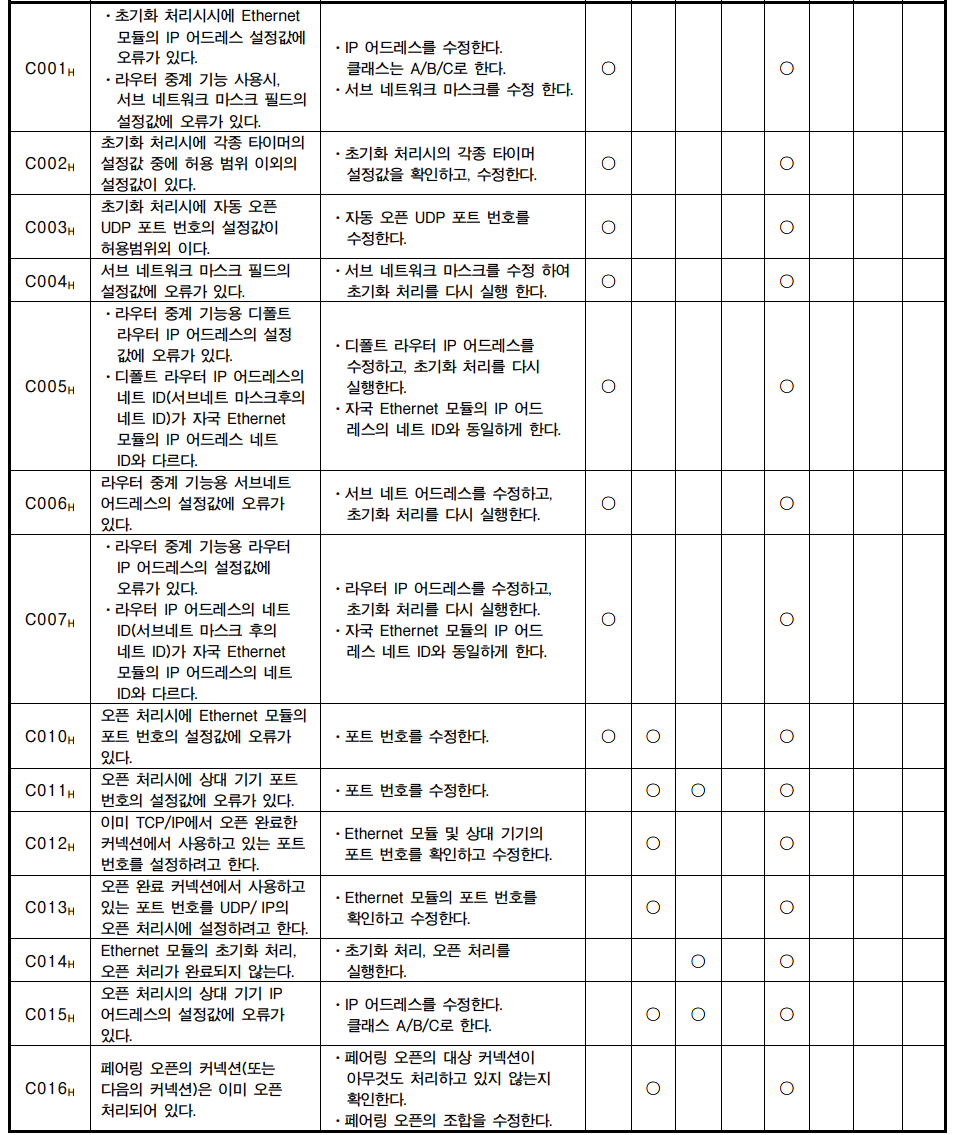
“message” : {“type” : “string”}

}

}\r\n

* Publish 항목 상세

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **항목명** | | | **항목명(한글)** | **필수** | **설명** | **비고** |
| 1 | jsonrpc | | | Json rpc 버전 | O | Json rpc 프로토콜 버전 | 2.0 |
| 2 | id | | | 통신ID | O | Pub 통신 ID | auto increment |
| 3 | result | | | 결과정보 | O |  |  |
| 4 |  | protocol.version | | 프로토콜 버전 | O | 프로토콜 버전 |  |
| 5 |  | event.id | | 이벤트 ID | O | 명령 처리의 이벤트ID |  |
| 6 |  | proc.data | | 처리시간 | O | 명령 처리시간 | yyyymmdd hh:mm:ss.SSS |
| 7 |  | items | | 아이템정보 | X | 수집아이템 정보 | 장애 발생시 ‘null’ |
| 8 |  |  | key | 키 | O | 수집 아이템 이름 | 수집시작 명령 시 전달받은 아이템 이름과 동일 |
| 9 |  |  | value | 값 | O | 수집 아이템 값 | 16진수 HEX값 |
| 10 | error | | | 에러정보 | X | 장애 발생시 에러정보 | 정상 처리시 ‘null’ |
| 11 |  | code | | 에러코드 | O | 에러코드 |  |
| 12 |  | message | | 에러메세지 | O | 에러메세지 |  |



# PLC 읽기 프로토콜

* 응용서비스와 미들웨어 간 PLC수집 시작 명령을 위한 MQ(REQ-RES)기반의 프로토콜로 해당 명령 수신시, PLC로 부터 수집한 정보를(Response)를 즉시 응답한다.
* 전송 데이터는 JSON표준 형식으로 전송
* 전송 문자 타입은 UTF-8 인코딩 사용.
* 전송 종료 코드는 CR LF ('\r\n')를 사용.

## PLC 읽기 Request 프로토콜(응용서비스 -> M/W)

* **Request JSON 스키마**

{

“jsonrpc” : “2.0”,

“method” : “readonce”,

“id” : {“type” : “string”},

“param” : {

“protocol.version” : {“type” : “string”},

“plc.ip” : {“type” : “string”},

“plc.port” : {“type” : “string”},

“items” : [

{ “key” : {“type” : “string”},

“device.code” : {“type” : “string”},

“device.num” : {“type” : “string”},

“device.score” : {“type” : “string”}

}]

}

}\r\n

* **Request 항목 상세**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **항목명** | | | **항목명(한글)** | **필수** | **설명** | **비고** |
| 1 | jsonrpc | | | Json rpc 버전 | O | Json rpc 프로토콜 버전 | 2.0 |
| 2 | method | | | 기능명 | O | 처리 | start |
| 3 | id | | | 통신ID | O | Req-Res 통신 ID |  |
| 4 | param | | | 파라미터 | O | 기능별 입력 파라미터 | Object |
| 5 |  | protocol.version | | 프로토콜 버전 | O | 프로토콜 버전 |  |
| 7 | plc.ip | | PLC 아이피 주소 | O | PLC연결을 위한 IP Address |  |
| 8 | plc.port | | PLC 아이피 포트 | O | PLC연결을 위한 Port Number |  |
| 11 | items | | 아이템 정보 배열 | O | 정보 수집을 위한 아이템 정보 배열 | Array |
| 12 |  | key | 아이템명 | O | 수집 항목의 이름 |  |
| 13 |  | device.code | PLC 디바이스 코드 | O | PLC 디바이스 영역에 따른 코드 | \*액세스 가능 디바이스 일람 참조 |
| 14 | device.num | PLC 디바이스 번호 | O | PLC 디바이스의 메모리 번호(16진수 어드레스) | \*액세스 가능 디바이스 일람 참조 |
| 15 |  | device.score | PLC 디바이스 스코어 | O | PLC 디바이스의 메모리 스코어 | \*최대 사용가능 스코어 참조 |

* **Request JSON 샘플**

{

“jsonrpc” : “2.0”,

“method” : “start”,

“id” : “1”,

“param” : {

“protocol.version” :”1.0”,

“plc.ip” : “127.0.0.1”,

“plc.port” : “8192”,

“items” : [

{ “key” : “lot”,

“device.code” : “D\*”,

“device.num” : “10000”,

“device.score” : “4”

},

{ “key” : “quality”,

“device.code” : “D\*”,

“device.num” : “10004”,

“device.score” : “1”

}]

}

}\r\n

## PLC 읽기 Response 프로토콜(M/W -> 응용서비스)

* **Response JSON 스키마**

{

“jsonrpc” : “2.0”,

“id” : {“type” : “string”},

“result” : {

“protocol.version” : {“type” : “string”},

“proc.data” : {“type” : “string”}

“items” : [

{ “key” : {“type” : “string”},

“value” : {“type” : “string”}

}],

}

“error” : {

“code” : {“type” : “number”},

“message” : {“type” : “string”}

}

}\r\n

* **Response 항목 상세**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **항목명** | | | **항목명(한글)** | **필수** | **설명** | **비고** |
| 1 | jsonrpc | | | Json rpc 버전 | O | Json rpc 프로토콜 버전 | 2.0 |
| 2 | id | | | 통신ID | O | Pub 통신 ID | auto increment |
| 3 | result | | | 결과정보 | O |  |  |
| 4 |  | protocol.version | | 프로토콜 버전 | O | 프로토콜 버전 |  |
| 5 |  | proc.data | | 처리시간 | O | 명령 처리시간 | yyyymmdd hh:mm:ss.SSS |
| 6 |  | items | | 아이템정보 | X | 수집아이템 정보 | 장애 발생시 ‘null’ |
| 7 |  |  | key | 키 | O | 수집 아이템 이름 | 수집시작 명령 시 전달받은 아이템 이름과 동일 |
| 8 |  |  | value | 값 | O | 수집 아이템 값 | 16진수 HEX값 |
| 9 | error | | | 에러정보 | X | 장애 발생시 에러정보 | 정상 처리시 ‘null’ |
| 10 |  | code | | 에러코드 | O | 에러코드 |  |
| 11 |  | message | | 에러메세지 | O | 에러메세지 |  |

* **Response JSON 샘플**

{

“jsonrpc” : “2.0”,

“id” : “001”,

“result” : {

“protocol.version” : “1.1”,

“proc.data” : “20190904 13:54:06.289”

“items” : [

{ “key” : “lot”,

“value” : “3132333435363839”

},

{ “key” : “result”,

“value” : “3F55”

}],

}

}\r\n