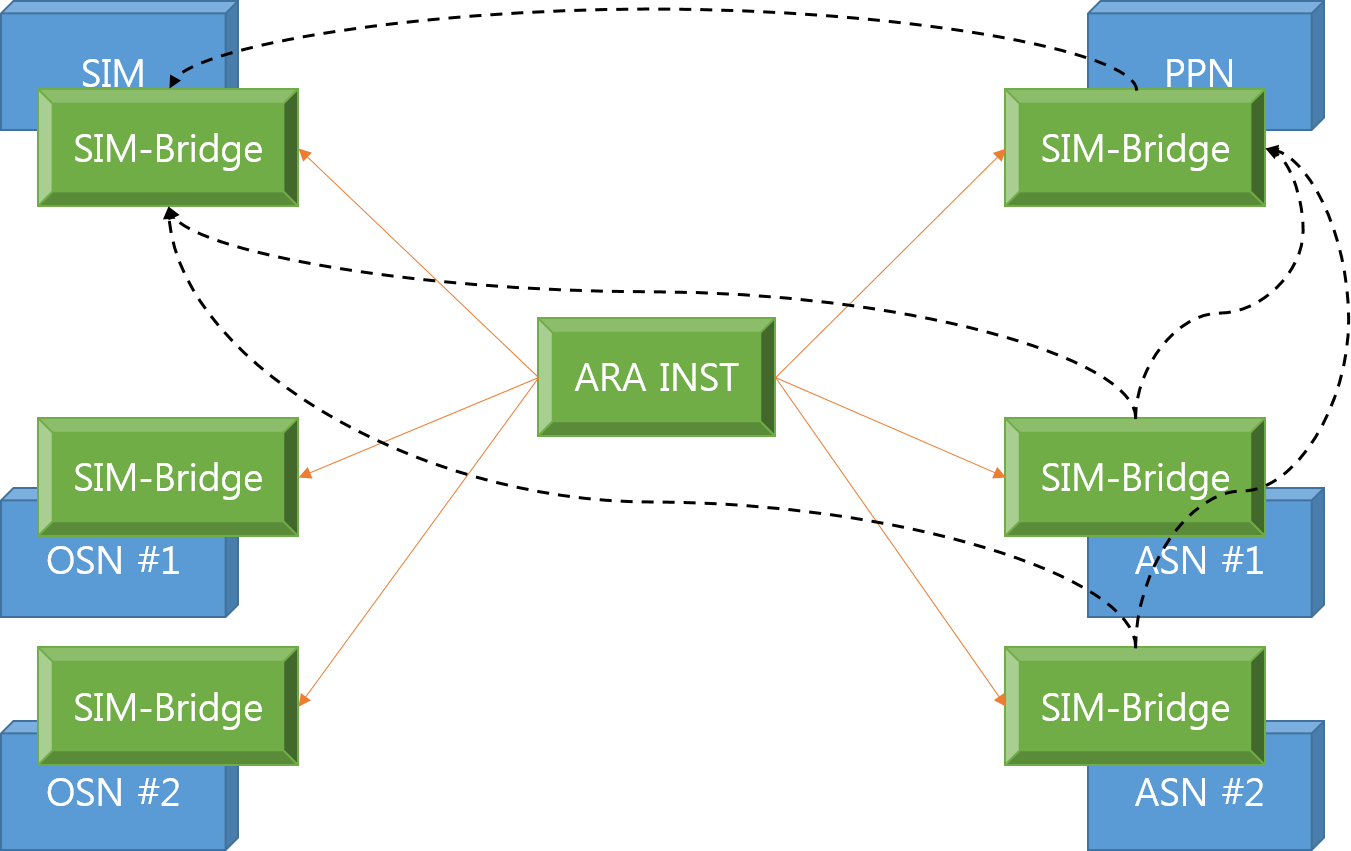
ARAInst V2.0 Profile

시스템 구성도



# ARAInst

## Open

1. ARAInst 에서 서버 Configuration 파일을 읽어들인다.
2. ARAInst 에서 등록된 서버로 접속한다.
3. 접속에 성공하면 서버의 상태를 Connected로 바꾼다.
4. 접속 실패시 서버의 상태를 Failed로 바꾸고 오류를 출력한다.
5. 각 서버에 초기화에 필요한 정보를 전송한다. (모델파일명 등등…)
6. Priority가 가장 낮은 서버에서 Simulation Time을 가져온다.
7. 다른 모든 서버에 Simulation Time을 전송한다.
8. 각 서버에 모든 서버의 정보를 전송한다.

## Download

1. Taglist 정보를 각 서버에 전송한다.
2. 각 서버는 tag mapping 정보를 보고 해당 서버에 연결한다. (Priority 높은 서버가 낮은 서버에 연결)
3. Download가 완료되면 서버 상태를 Paused로 바꾼다.

## Manual Run

1. 사용자가 입력한 time step 과 step num 만큼 simulation을 실행한다.
2. ARAInst 에서 매번 계산할 때 마다 time step을 전송한다.
3. 각 서버는 계산이 끝나면 Simulation Time을 전송한다.
4. 각 서버에 계산 요청을 전송하면 서버 상태를 Running으로 바꾼다.
5. 각 서버에서 계산 완료가 수신되면 서버 상태를 Paused로 바꾼다.

## Auto Run

1. 사용자가 입력한 time step 간격과 speed에 따라 매번 자동으로 계산한다.
2. 사용자가 limit time을 입력하면 그때까지만 계산한다.
3. 사용자가 계산 중 STOP 버튼을 누르면 중지한다.
4. Auto Run 중에는 Manual Run은 동작하지 않는다.

## View Taglist

1. 새로운 창을 띄워서 (새로운 탭) Taglist UI를 연다.
2. 필터링을 하여 목록을 출력한다. (서버이름, Tagname, Boundary, Malfunction, …)
3. 특정변수를 선택하여 Real Time Trend Graph를 출력한다.
4. 기존에 있는 그래프에 추가하거나 새로운 그래프를 만든다.
5. To Tag에 임의의 값을 (또는 함수, 스크립트) 를 설정한다. (Malfunction)

## Snapshot save

1. Auto Run 상태이면 Paused 상태로 바꾼다.
2. Server Status 를 Saving 으로 바꾼다.
3. 현재 서버의 모든 상태를 각 서버에서 파일로 저장한다.
4. 저장폴더는 환경설정에서 미리 설정해 놓는다. (또는 각 서버에서..)
5. Auto Run 상태였으면 다시 Auto Run 상태로 만든다.

## Snapshot restore

1. Auto Run 상태이면 Paused 상태로 바꾼다.
2. Server Status 를 Restoring 으로 바꾼다.
3. 각 서버의 snapshot 과 ARAInst 의 snapshot도 불러온다.

# SIMBridge

## Communication

1. Simulation Program을 실행하고 초기화 한다.
2. Simulation Server와 통신을 한다.
3. 통신 방법은 OPC, TCP/IP, DLL Function, etc
4. Simulation Program을 종료한다.
5. ARAInst 및 SIMBridge와 1:1 통신을 한다.

## Configuration

1. Profile에 환경변수(서비스 포트)가 설정되어 있으면 값을 읽어온다.
2. 환경 설정을 변경하면 Profile 에 저장한다.
3. 그외의 환경설정은 ARAInst 로부터 받아서 동작한다. (Home, Sim Program, etc)

## Start

1. Server Name은 시스템 정보를 읽어서 보여준다.
2. Host IP는 시스템 정보를 읽어서 ComboBox에서 선택할 수 있게 한다.
3. Service Port는 직접 입력을 한다.
4. Start 버튼을 누르면 서비스를 시작한다.
5. 환경설정이 되어 있는 경우 자동으로 시작한다.
6. 자동실행 시 Tray Icon으로 표시한다. (추후)

## Current Simulation Time

1. 현재 Simulation Time을 가져온다.
2. 현재 Simulation Time을 설정한다.

## Node Information

1. ARAInst 로부터 각 서버의 정보를 받는다.

## Taglist

1. ARAInst 에서 Taglist 정보를 받아서 관리한다.
2. ARAInst 에서 임의로 Tag 정보를 변경하거나 추가, 삭제할 수 있다.
3. Taglist 정보를 분석하여 해당 서버에 접속한다.
4. Normal 상태의 Tag Value를 해당 서버에 전송한다.
5. Tag Value를 수신하여 Tag 값을 변경한다. (From ARAInst or Server)

# Scenario

## SIMBridge Start

1. SIMBridge를 설치하고 실행한다.
2. Profile 에서 환경 정보를 읽어온다.
3. Hostname을 바꾸고 싶으면 바꾼다. (기본값은 COMPUTERNAME)
4. Service Port를 확인한다. (기본값은 2014)
5. Server Type 을 설정한다. (미 설정시 ARAInst 에서 설정한다.)
6. Priority를 설정한다. (미 설정시 ARAInst 에서 설정한다.)
7. Start 버튼을 누른다.
8. 환경 정보를 Profile에 저장한다.

## ARAInst Start

1. ARAInst 를 설치하고 실행한다.
2. SIMBridge Host IP 를 입력한다.
3. SIMBridge Service Port를 입력한다. (기본값 2014)
4. Server Type 을 설정한다. (미 설정시 SIMBridge 에서 가져온다.)
5. Hostname을 설정한다. (미 설정시 SIMBridge 에서 설정한다.)
6. Priority를 설정한다. (미 설정시 SIMBridge 에서 설정한다.)
7. Add 버튼을 누르면 목록에 추가하고 파일에 저장한다.
8. SIMBridge에 정상적으로 접속하면 상태를 Connected 로 한다.
9. 접속에 실패하면 상태를 Failed 로 한다.
10. Open.. 버튼을 누른후 설정파일을 찾아서 정보를 읽어들인다.
11. 서버 정보를 목록에 출력한다.
12. SIMBridge 접속에 성공하면 Connected로 바꾼다.
13. SIMBridge 접속에 실패하면 Failed로 바꾼다.
14. ARAInst에서 SIMBridge에 접속에 성공하면 HELLO ARAInst를 보낸다.
15. SIMBridge는 HELLO를 수신하면 HELLO Hostname ServerType Priority 을 보낸다. ServerType 은 “HYSYS.DLL”, “HYSYS.OPC”, “DeltaV.OPC”, “Dummy.TCP” 중 하나이고 Priority는 정수값을 갖는다.

## Download

1. Taglist 정보를 파일에서 읽어서 한줄씩 해당하는 정보를 SIMBridge에 전송한다.
2. 정보를 다 전송하였으면 완료 정보를 전송한다. (인수 개수를 0으로 보냄)
3. SIMBridge는 Read Only, Read-Write Tag 목록을 만든다.
4. 정보를 다 받았으면 필요에 따라 그룹을 만들고 Simulation Server에 등록한다.
5. Taglist 정보에 따라 통신이 필요한 SIMBridge에 연결한다.
6. SIMBridge 와 SIMBridge 사이에는 Block으로 데이터를 전송한다.
7. ARAInst 는 SIMBridge에 매번 Tagname 목록을 보내고 그 결과를 받는다.

## 계산

1. Download가 완료되면 SIMBridge-to-SIMBridge가 연결이 완료된다.
2. 각 시뮬레이션 프로그램과 연결하여 초기값을 읽는다.
3. 결과를 ARAInst에 전송한다. (Simulation Program Connection Complete)
4. 각각의 SIMBridge가 해당하는 서버의 SIMBridge에 초기값을 전송한다.
5. 수신한 초기값과 Taglist에 있는 초기값, 경계값을 시뮬레이션 프로그램에 쓴다.
6. 결과를 ARAInst에 전송한다. (Simulation Ready)
7. ARAInst는 SIMBridge가 보낸 상태나 오류를 출력한다.
8. ARAInst가 Simulation Time이 없으면 Main 서버에 시간을 요청한다.
9. Main 서버는 ARAInst에 Simulation Time을 전송한다.
10. 사용할 SIMBridge가 모두 Ready가 되었으면 ARAInst에서 계산 명령을 전송한다.
11. SIMBridge는 Simulation Program에 계산을 수행한다. (Sync? Or Async & Callback)
12. 계산이 완료되면 SIMBridge가 시뮬레이션 프로그램의 Tag값을 읽는다.
13. 각각의 SIMBridge가 각 SIMBridge에 Tag값을 전송한다.
14. 각 SIMBridge에서 받은 Tag 값을 시뮬레이션 프로그램에 쓴다.
15. ARAInst에 완료 메시지를 전송한다.

프로토콜

# 기본 포맷

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Header | | | Body | | | | |
| STX(4) | CMD(8) | LEN(4) | NUM(4) | TYPE(2) | LEN(2) | VALUE(N) | … |
| #!RS | String | DWORD | DWORD | ENUM | WORD | Any | … |

## Header

1. STX: 메시지의 시작을 알리는 문자열이다. 요청(#!RS), 응답(#!RR), 전송(#!RO)
2. CMD: 메시지의 명령이다. Null Terminated String이고 대소문자 구별한다.
3. LEN: 전체 메시지의 길이이다. (Header + Body)

* 0인 경우:
* 15인 경우:

## Body

1. NUM: Body의 인수 개수다.
2. TYPE: 인수의 종류이다.

* 0: None, Null, Empty
* 1: Char(1), ASCII 문자열에 사용된다.
* 2: UChar(1..N), UTF-8 문자열에 사용된다.
* 3: Bool(1), True(Odd Number) / False(Even Number)
* 4: Short(2), 2byte integer value
* 5: Integer(4), 4byte integer value
* 6: Long(8), 8byte integer value
* 7: Single(4), 4byte float value
* 8: Double(8), 8byte float value
* 9: Binary(1), 이진 데이터.

1. LEN: 인수의VALUE에 해당하는 배열의 길이이다. (INT 2개 이면 8이 아닌 2)
2. VALUE: 배열로 이루어진 값. 최대 배열의 개수는 65535 이다.

* 데이터는 4의 배수가 되도록 Null Padding을 붙인다.
* 2바이트 이상의 숫자인 경우는 Little Endian으로 전송한다.

# ARAInst-to-SIMBridge

## ERROR

1. SIMBridge는 어느 시점이던지 오류가 발생하면 ARAInst에 오류를 전송한다.
2. ARAInst는 오류 내용을 출력한다.

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| SIMBridge | #!RO | NUM: 1  TYPE: 1 (Char)  LEN: nnnn  VALUE: Error String  Padding: 0byte~3byte |

## HELLO

1. ARAInst는 등록된 정보를 이용하여 SIMBridge 에 접속한다.
2. ARAInst를 알리고 SIMBridge정보를 요청하는 메시지이다.
3. SIMBridge는 자신의 정보를 담아서 응답해야 한다.

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| ARAInst | #!RS | NUM: 1  TYPE: 1 (Char)  LEN: 8  VALUE: ARAInst  Padding: 0byte~3byte |

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| SIMBridge | #!RR | NUM: 3  TYPE: 1 (Char)  LEN: length of hostname + 1  VALUE: hostname  Padding: 0byte~3byte  TYPE: 1 (Char)  LEN: length of type string  VALUE: HYSYS.DLL / HYSYS.OPC / DeltaV.OPC / Dummy.TCP  Padding: 0byte~3byte  TYPE: 4 (Short)  LEN: 1  VALUE: 0 ~ 65535 (Little endian)  Padding: 0x00 0x00 |

## SET

1. ARAInst 에서 SIMBridge의 정보를 설정하거나 변경할 때 사용한다.
2. SIMBridge는 정보를 수정하고 결과를 ARAInst에 응답한다.

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| ARAInst | #!RS | NUM: 3  TYPE: 0 (None) ; Hostname  LEN: 0  TYPE: 1 (Char) ; Server Type  LEN: 10  VALUE: Dummy.TCP  Padding: 0x00 0x00  TYPE: 4 (Short) ; Priority  LEN: 1  VALUE: 0 ~ 65535  Padding: 0x00 0x00 |

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| SIMBridge | #!RR | NUM: 1  TYPE: 0 (None)  LEN: 0 |

## INFOR

1. ARAInst 에서 모든 Host의 정보를 모든 SIMBridge에 전송한다.
2. SIMBridge는 Host 정보를 재설정하고 결과를 응답한다.

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| ARAInst | #!RS | NUM: nn ; number of host  TYPE: 1 (Char) ; Host information  LEN: length of host information  VALUE: hostname,host\_ip,port\_no  Padding: 0byte~3byte  TYPE: 1 (Char) ; Host information  LEN: length of host information  VALUE: hostname,host\_ip,port\_no  Padding: 0byte~3byte  … repeat for nn |

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| SIMBridge | #!RR | NUM: 1  TYPE: 0 (None for Success) / 1 (Char for Error)  LEN: 0 / nn  VALUE: Error String if occur. 자신의 정보가 ARAInst가 보낸 것과 다르면 오류메시지를 응답한다. |

## SIMINIT

1. ARAInst에서 SIMBridge가 Simulation Program을 초기화하기 위한 정보를 전송한다.
2. SIMBridge는 Simulation Program을 초기화 하고 결과를 응답한다.
3. Simulation Program 초기화시 오류가 발생하면 오류를 전송한다.
4. ARAInst 는 결과를 전송 받아 상태를 변경하거나 오류 내용을 출력한다.

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| ARAInst | #!RS | NUM: nn ; number of parameter  TYPE: 1 (Char) ; working home folder  LEN: length of home folder  VALUE: home folder path  Padding: 0byte~3byte  TYPE: 1 (Char) initial snapshot file  LEN: length of file path  VALUE: snapshot file path  Padding: 0byte~3byte  … repeat |

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| SIMBridge | #!RR | NUM: 1  TYPE: 0 (None for Success) / 1 (Char for Error)  LEN: 0 / nn  VALUE: Error String if occur. 자신의 정보가 ARAInst가 보낸 것과 다르면 오류메시지를 응답한다. |

## SIMTIME

1. ARAInst에서 snapshot을 읽지 않아서 (초기상태) Simulation Time이 없는 경우 Priority가 가장 낮은 서버에 Simulation Time을 요청한다.
2. SIMBridge는 Simulation Program에서 Simulation Time을 읽어서 응답한다.

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| ARAInst | #!RS | NUM: 1  TYPE: 0 (None) |

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| SIMBridge | #!RR | NUM: 1  TYPE: Integer / Long / Single / Double / 1 (Char for Error)  LEN: 4 / 8 / nn  VALUE: 4바이트 또는 8바이트 시각정보를 응답한다.  오류일 경우 오류메시지를 응답한다. |

## TAGINIT

1. ARAInst에서 모든 SIMBridge에 Download 초기화 를 요청한다.
2. SIMBridge는 각각의 SIMBridge와 연결을 종료하고 Tag 정보를 초기화한다.
3. SIMBridge는 결과를 ARAInst에 전송한다.

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| ARAInst | #!RS | NUM: 1  TYPE: 0 (None) |

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| SIMBridge | #!RR | NUM: 1  TYPE: None(0) / 1 (Char for Error)  LEN: 0 / nn  VALUE: 오류일 경우 오류메시지를 응답한다. |

## TAGDOWN

1. ARAInst에서 Taglist 파일을 한줄한줄 읽으면서 해당서버에 정보를 전송한다.
2. SIMBridge는 Tag정보를 받으면 분석하여 관리한다.
3. ARAInst는 Tag 정보를 다 전송하면 완료표시로 None을 전송한다.
4. SIMBridge는 None을 받으면 통신을 해야할 각각의 SIMBridge에 접속을 한다.
5. Simulation Program에 접속하여 Tag 정보를 확인한다.
6. SIMBridge-to-SIMBridge Tag정보를 검증한다. (이름순서, 변수형, …)
7. SIMBridge는 Tag 숫자 또는 오류 메시지를 응답한다.

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| ARAInst | #!RS | NUM: 1  TYPE: Char(1) ; Taglist one line  LEN: nn  VALUE: tag 정보 |

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| ARAInst | #!RS | NUM: 1  TYPE: 0 (None) ; 완료 메시지 |

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| SIMBridge | #!RR | NUM: 1  TYPE: Integer (Success) / Char (Error)  LEN: 4 / nn  VALUE: # / 오류일 경우 오류메시지를 응답한다. |

## SEND

1. ARAInst는 값을 보내야 할 모든 SIMBridge에 값을 보내라는 메시지를 송신한다.
2. SIMBridge는 Simulation Program의 모든 Tag값을 읽는다.
3. 전송해야 할 Tag 값을 해당 SIMBridge에 전송한다. (RO)
4. 값을 전송받은 SIMBridge는 Tag 의 write값을 Simulation Program에 쓴다.
5. 값을 전송한 결과를 ARAInst에 응답한다.

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| ARAInst | #!RS | NUM: 1  TYPE: 0 (None) |

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| SIMBridge | #!RO | NUM: NN  TYPE: Any  LEN: nn  VALUE: Any \* nn  … repeat |

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| SIMBridge | #!RR | NUM: 1  TYPE: Integer (Success) / Char (Error)  LEN: 4 / nn  VALUE: # / 오류일 경우 오류메시지를 응답한다. |

## RUN

1. ARAInst 에서 모든 서버가 Ready 상태이면 RUN 메시지를 송신한다.
2. 메시지를 받은 SIMBridge는 Simulation Program에 계산을 시킨다.
3. 실행 결과를 ARAInst에 전송한다.

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| ARAInst | #!RS | NUM: 1  TYPE: 0 (None) |

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| SIMBridge | #!RR | NUM: 1  TYPE: None / Char  LEN: 0 / nn  VALUE: Error string |

## TAGADD

1. ARAInst에서 임의로 Tag정보를 추가할 경우 사용한다.
2. Taglist에 등록은 되지 않았지만 Event capture로 추가된 tag이다.
3. ARAInst에서 Taglist 내용을 추가하는 경우 사용
4. SIMBridge는 TAGDOWN finish 일때처럼 동작한다.

## TAGMOD

1. ARAInst에서 Malfunciton을 실행하여 강제로 값을 바꿀 때 사용한다.

## TAGDEL

1. ARAInst 에서 Taglist에서 해당 Tag를 삭제할 때 사용한다.

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| ARAInst | #!RS | NUM: 1  TYPE: Char  LEN: nn  VALUE: taglist one line string |

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| SIMBridge | #!RR | NUM: 1  TYPE: None / Char  LEN: 0 / nn  VALUE: Error string |

## SAVE

1. ARAInst 에서 Snapshot save 메시지를 전송한다.
2. SIMBridge는 Simulation Program에 Snapshot save 명령을 내린다.
3. Snapshot이 저장된 것이 확인되면 ARAInst에 응답한다.

## RESTORE

1. ARAInst에서 Snapshot restore 메시지를 전송한다.
2. SIMBridge는 Simulation Program에 Snapshot restore 명령을 내린다.
3. Snapshot이 로딩된 것이 확인되면 ARAInst에 응답한다.

## DELETE

1. ARAInst에서 Snapshot delete 메시지를 전송한다.
2. SIMBridge는 snapshot 파일을 삭제한다.
3. ARAInst에 응답한다.

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| ARAInst | #!RS | NUM: 1  TYPE: Char  LEN: nn  VALUE: snapshot file path |

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| SIMBridge | #!RR | NUM: 1  TYPE: None / Char  LEN: 0 / nn  VALUE: Error string |

# SIMBridge-to-SIMBridge

## HELLO

1. Priority 값은 높은 SIMBridge#1이 낮은 SIMBridge#2로 접속을 한다.
2. 접속에 성공하면 SIMBridge#1은 HELLO 메시지를 전송한다.
3. HELLO메시지를 수신한 SIMBridge#2는 HELLO메시지를 응답한다.
4. ARAInst에서 받은 정보와 SIMBridge#1이 송신한 정보가 다르면 ARAInst에 Error메시지를 전송한다.
5. ARAInst에서 받은 정보와 SIMBridge#2에서 받은 정보가 다르면 ARAInst에 Error 메시지를 전송한다.

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| SIMBridge | #!RS | NUM: 1  TYPE: Char  LEN: nn  VALUE: Hostname |

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| SIMBridge | #!RR | NUM: 1  TYPE: Char  LEN: nn  VALUE: Hostname |

## CONFORM

1. ARAInst로부터 Tag정보를 다 받았으면 Simulation Program에 접속하여 Tag정보를 확인한다.
2. 통신해야 할 각 SIMBridge에 Tag Name과 Value Type을 전송 (RO)한다.
3. 수신한 SIMBridge는 자신이 가진 Tag Name, Value Type, Ordering을 확인한다.
4. 오류가 있으면 ARAInst에 Error를 전송한다.

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| SIMBridge | #!RO | NUM: N  TYPE: Char  LEN: nn  VALUE: Tagname,#  … repeat |

## SEND

1. ARAInst로부터 SEND 메시지를 받으면 Simulation Program의 Tag값을 읽는다.
2. 전송해야 할 각 SIMBridge로 Tag Value를 전송(RO)한다.
3. 값을 전송받은 SIMBridge는 해당 값을 Simulation Program의 Tag에 쓴다.

|  |  |  |
| --- | --- | --- |
| Sender | Type | Argument |
| SIMBridge | #!RO | NUM: NN  TYPE: Any  LEN: nn  VALUE: Any \* nn  … repeat |

# SIMBridge-to-Dummy.TCP

## Tag Add

1. Simulation Program에 통신을 위한 Tag를 등록한다.

## Tag Delete

1. 등록한 Tag를 삭제한다.

## Tag Read

1. 등록된 Tag의 값을 읽는다.

## Tag Write

1. 등록된 Tag에 값을 쓴다.

## Snapshot Save

1. Simulation Time과 변수값을 파일에 저장한다.

## Snapshot Load

1. 파일로부터 Simulation Time과 변수 값을 읽어온다.

## Snapshot Remove

1. Snapshot 파일을 삭제한다.

Program Design

# SIMBridge

## Read Only Tags (for Send)

|  |  |  |
| --- | --- | --- |
| tagname | type | value |
|  |  |  |
|  |  |  |
|  |  |  |

1. Tagname은 Simulation Program에 등록된 이름이다.
2. Type은 변수의 형이다.
3. Value 는 Simulation Program에서 읽은 값이다.

## Read Write Tags (For Receive)

|  |  |  |  |
| --- | --- | --- | --- |
| tagname | type | Read value | Write value |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. Tagname은 Simulation Program에 등록된 이름이다.
2. Type은 변수의 형이다.
3. Read value 는 Simulation Program에서 읽은 값이다.
4. Write value은 다른 SIMBridge또는 ARAInst에서 받은 값이다.

## Send Tags

|  |  |
| --- | --- |
| tagname | Tag\_index |
|  |  |
|  |  |
|  |  |

1. 각 SIMBridge 마다 하나씩 생성한다.
2. 패킷을 인코딩하기 위해 사용한다.
3. Tagname 은 대상 SIMBridge의 tag 이름이다.
4. Tag index는 연결되는 read only tags의 index 이다.

## Recv Tags

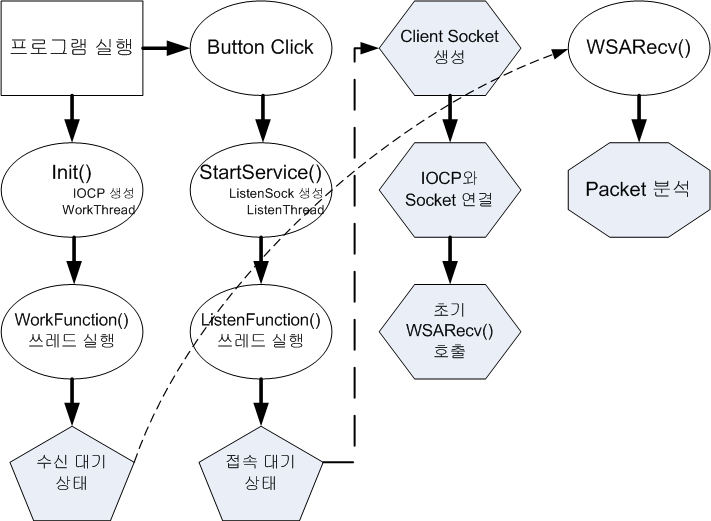
|  |  |
| --- | --- |
| Tagname | Tag\_index |
|  |  |
|  |  |
|  |  |

1. 수신해야 할 SIMBridge 마다 하나씩 생성한다.
2. 패킷을 디코딩하여 값을 저장하기위해 사용한다.
3. Tagname 은 대상 SIMBridge의 tag 이름이다.
4. Tag index는 연결되는 read write tags의 index 이다.

# Class diagram

## Packet

* Char buffer[]
* Char \*cmd
* Int packet\_length
* Int argument\_num
* Argument \*argument



# Technology

## Thread

## IOCP

## Timer

# Coding Rule

## Debug Log

파일에 기록

라이브러리에서는 모드 디버그 로그로 출력

#define PRINTLOG(a) g\_Logger.PrintLog a

#define PRINTLOGV(a,b) g\_Logger.PrintLogV(Logger::DebugLog,a,b)

#define PRINTDUMP(a,b) g\_Logger.PrintDump(a,b)

## Info Log

로그보기 화면에 기록

## Error Log

메인 화면에 출력