발전소 시뮬레이터의 다이나믹 모델과 디스플레이 모델간 데이타전송

김동욱°

전력연구원(Tel: 865-5737; Fax: 865-5748; E-mail: dongwook@hanbit.kepri.re.kr)

Data Transporting between Dynamic Model and Display Model of Power Plant Simulator

Dong Wook Kim
Korea Electric Power Research Institute

Abstracts : The safety and reliability of nuclear power plant operations relies heavily on the plant operators ability to respond to various emergency situations. It has become standard industry practice to utilize simulators to improve the safety and reliability of nuclear power plants operations. The simulators built for Younggwang#3,4, which is the basic model of the Korean Nuclear Power Plant design, has been developed precisely for this purpose. Dynamic Model and Display Model are develped under US3(UNIX Simulation Software Support System) environment in simulator for Younggwang#3,4. Since these two models are developed under each own operating system, it is necessary to develop a method for transporting data between these two systems. This paper descrives communication environment between Dynamic Model and Display Model, and addresses a file generation method for the Display Model, which will be necessary for designing MMI of MCR(Main Control Room) in the furture.

Keywords Simulator, Dynamic Model, Display Model, Shared Memory

1. 개요

영광#3,4호기 발전소 운전원 훈련용 시뮬레이터에서는 개발 및 운영환경을 제공하기위해 US3 (UNIX Simulation Software Support System)기반하에 이루어진다. US3 환경에서는 Dynamic Model 과 Display Model은 그림1과 같이 공유 메모리를 매개로 point value를 주고 받는다. 공유메모리를 통한 communication 구현방법은 가장 간단하면서 강력한 통신 방법의 일종이다. 그러나 이 방법은 같은 시스템 내에서의 통신방법으론 유용하지만 Dynamic model 과 Display Model 이 각기 다른 시스템에 존재하면서 Lan을 통해 연결되어야 할 경우는 사용될 수가 없다.

기본적으로 unix 에서 제공하는 공유 메모리 기법은 같은 시스템 상에서의 통신방법이기 때문이다. 공유메모리 기법은 구현자체가 간단하기 때문에 이를 사용하여 모델을 개발하게 되면 모델개발자는 공유메모리를 접근하는 것만으로 통신이 이루어지는 것으로 생각할 수 있기 때문에 모델개발 자체에 집중할 수 있다. 각각의 Dynamic model 과 Display

그림 1 공유메모리 개념
Fig. 1 Simplified view of shared memory

Model들이 통신을 염두에 두고 그 구현을 위한 코드를 각자가 갖고 있어야 한다면 이는 모델개발자에겐 부담이 아닐 수 없다. 비록 Dynamic model 과 Display Model 이 다른 호스트에서 구현되어 있고

Lan 을 통해 연결된다할지라도 개발자들에게 이러한 physical environment 대신에 모든 모델과 Display Model들이 한 시스템 내에서 구현된 것과 같은 logical environment를 제공할 수 있다면 개발자들은 공유메모리만을 상대하면서 시스템을 개발할 수 있을 것이다. 그림2는 그림1의 공유 메모리의 논리적 내부구조를 보여준다. 모델개발자들이 공유메모리의 내부구조나 그 구현방식으로 부터 자유로워질 수 있도록 개발자들의 개발 환경은 그림 3과 같다.

그림 2 공유메모리의 논리적 내부 구조
Fig 2 Inside Logical View of Shared Memory

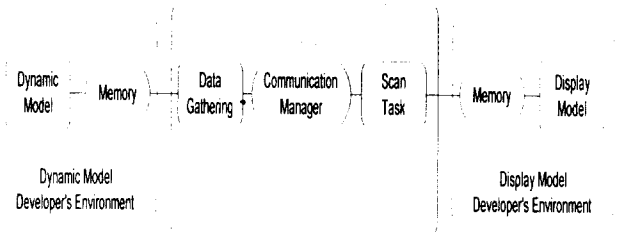

그림 3 개발자에게 숨겨진 통신 환경
Fig. 3 Hidden communication environment from developer

그림3의 developer's environment 영역이 개발자들이 보게되는logical environment가 된다. 이러한 logical environment를 제공하는 핵심부분인 그림 3의 hidden area는 data gathering task, communication manager, scanning task의 3부분으로 구성되어 있다. 그림 4는 Dynamic Model 과 Display Model간의 데이터 흐름을 보여준다.

2. Data Gathering & Scanning Task

Data Gathering Task는 모델에서 계산된 값들을 정해진 시간 주기로 그림4의 공유메모리로부터 읽어내어 communicatIon manager 의 sender를 통해 remote host에 위치한 display model에 값을 전달한다. data gathering task가 공유 메모리에서 읽어낸

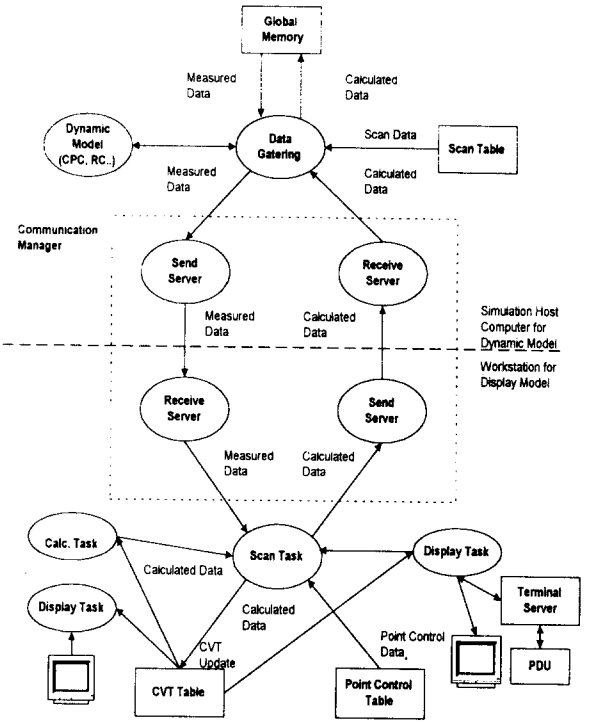

그림 4 Dynamic Model 과 Display Model 간의 데이터 흐름
Fig. 4 Data flow Between Dynamic Model and Display Model

값들을 communication manager의 sender에게 건네주면 communication manager의 sender는 display model이 위치한 remote host의 communication의 receiver에 이 값들을 전달한다. 이렇게 전달된 값들은 scanning task을 통해 display model developer의 logical environment에 위치한 공유메모리에 write되며 display model은 해당 값들을 참조할 수 있게 된다. 이런 방식으로 Dynamic model 개발자나 display model 개발자는 공유 메모리만을 통해 communication이 이루어지는 것과 같은 logical environment를 제공받게 된다.

2.1 Data Gathering Task의 역할

Data Gathering Task는 Dynamic Model로부터 계산된 정보들을 매초 단위로 Scan Task로 보내며 Display Model에서 Update된 정보들을 다시 Dynamic Model로보내며 중요 역할은 다음과 같다.

- Scan Rate에 의해 MA(Measured Analog) Value를 Scan Task에 송신
- 1초의 주기마다 MB(Measured Binary) Value를 Scan Task에 송신
- SOE(Sequence of Events) Points의 Event 발생 시마다 Scan Task에 송신

- Scan Task로부터 수신한 CA(Calculated Analog) Value를 공유 메모리에 Set
- Scan Task로부터 수신한 CB(Calculated Binary) Value를 공유 메모리에 Set
- MST의 Status에 따른 Point Value 송수신

2.2 Scanning Task의 역할

Scanning Task는 Data Gathering Task가 보내온 정보들을 CVT에 Update한다. 그리고 각 Point들의 Alarm 상태를 Check하여 Alarm Task로부터 Alarm Status를 보낸다. 또한 Display Task가 보내온 정보들을 Data Gathering Task와 Alarm Task로 보내며 중요 역할은 다음과 같다.

- Gathering Task로 부터 수신한 MA Value를 CVT에 Set
- Gathering Task로 부터 수신한 MB Value를 CVT에 Set
- Gathering Task로 부터 수신한 SOE Value를 CVT에 Set
- Calculation Task로 부터 수신한 CA Value를 CVT에 Set하고 Gathering Task에 송신
- Calculation Task로 부터 수신한 CB Value를 CVT에 Set하고 Gathering Task에 송신
- 각 Point들의 Alarm을 Check하고 Alarm이 발생한 Point들에 대한 정보를 Alarm Task에 송신

3. File Generation

Power Plant Simulator는 다수의 Dynamic model과 Display Model들로 구성된다. 한 Dynamic model에서 계산된 값이 여러 개의 Display Model에서 참조되는 경우가 있으며 한 Display Model이여러 개의 Dynamic model에서 계산된 값들로 구성되기도 한다. US3에서 각 Dynamic model은 자신의 reference environment로서 global memory 영역을 할당받아 사용하고 있다. 공유 메모리는 Display Model이 필요로 하는 point를 기준으로 생성된다. Display Model에서 사용되는 point가 결정되면 이 point들이 계산되는 Dynamic model의 global memory에서 값을 모아 Display Model을 위한 global memory 영역이 잡힌다. Data gathering task는 Dynamic model의 reference environment 의 global memory를 직접 접근하지 않고 해당 Display Model이 필요로 하는 point 값들을 위해 할당된 global memory로부터 값들을 읽어낸다. 그림5는 Display Model YMM을 기준으로 호스트 컴퓨터에 할당되는 메모리 구조를 보여준다. Data gathering task는 Display Model YMM이 필요로 하는 point a1, b2, c3을 해당 point를 계산하는 Dynamic model의 global 메모리(global_a, global_b, global_c)를 접근하는 대신 YMM을 위해 할당된 메모리(global_ymm)에서 point 값을 읽어낸다. 그림5의 global_ymm은 그림 3의 math model developer's environment 안의 메모리에 해당하며 D_YMM은 Display Model developer's environ- ment안의 메모리이다. Display Model 각각에 대해서 그림5와 같은 메모리들이 호스트 컴퓨터에 할당된다. Data gathering task가 각각의 Display Model을 위해 할당된 공유 메모리를 접근하기 위해서 필요한 정보는 File generation을 통해서 생성된다.

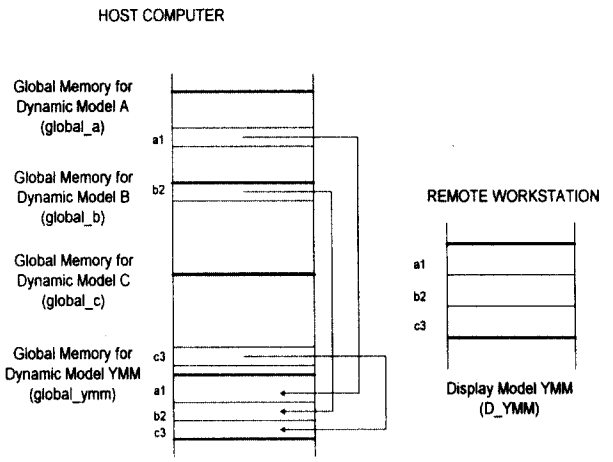

그림 5 디스플레이 모델을위한 호스트의 메모리 구조

Fig. 5 Memory configuration in Host computer for Display Model

3.1 공유 메모리 접근 방법

Data gathering task 는 File generation을 통해서 생성된 Offset table과 Scan table을 이용해서 공유 메모리의 값을 읽을 수가 있다. 그림 6은 data gathering task가 공유메모리에서 값을 읽어오는 과정을 보여준다. Data gathering task가 실행되면 File generator 가 생성해낸 오프셋과 스캔정보의 파일을 읽어들여 배열로 저장한다. 그림 6의 Local memory array가 이 정보를 갖고 있다. ma_scan_table에는 전송될 analog measured point들의 pri가 전송 rate에 따라 정렬되어 있는데 이 pri는 오프셋 테이블인 ana_off_table에서 offset을

찾는 인덱스로 사용된다. ana_off_table에는 US3의 DBM(Data Base Manager)에 등록된 point들의 공유 메모리 내에서의 오프셋이 정렬되어 있다. 스캔 rate에 따라 ma_scan_table을 참조 해당 point의 오프셋이 저장된 ana_off_table의 인덱스(pri)를 취득한 후 ana_off_table[pri]에서 저장된 오프셋 ana_off_table[pri].offset를 공유 메모리 시작번지 global_ppc(global_ymm)와 조합하여 point value를 읽어 낼 수 있다. 실제로 data gathering task의 source code상의 일부분을 보이면 다음과 같다.

```
pri = ma_scan_table->pri[base+i];
offset = ana_off_table[pri].offset;
data[i] = *(ANA_VAL_TYPE *)(global_ppc
        + offset);
```

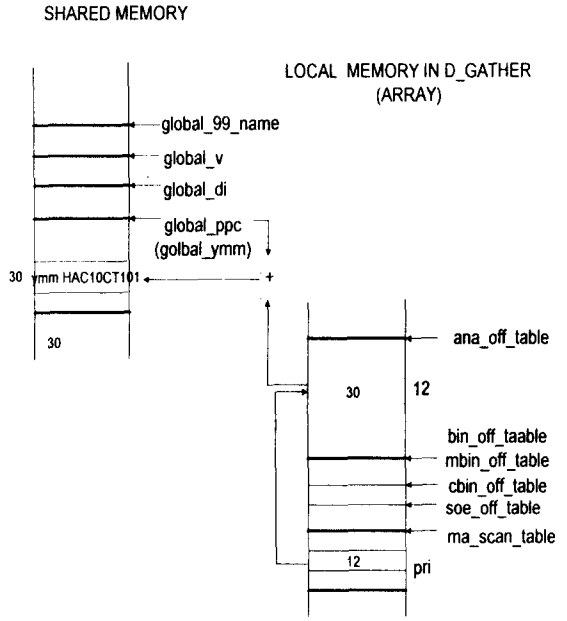

그림 6 Shared Memory 접근 방법

Fig. 6 The method to access to shared memory

3.2 Procedure for File Generation

File generation에 의해서 생성되는 파일은 US3 DB에 데이타를 등록하기 위한 dbm명령어 파일인 두 개의 card file 과 PPC control, Point ID table, Point control table, CVT table, Offset table, Scan table의 5개 테이블이며 이들은 파일형태로 디스크

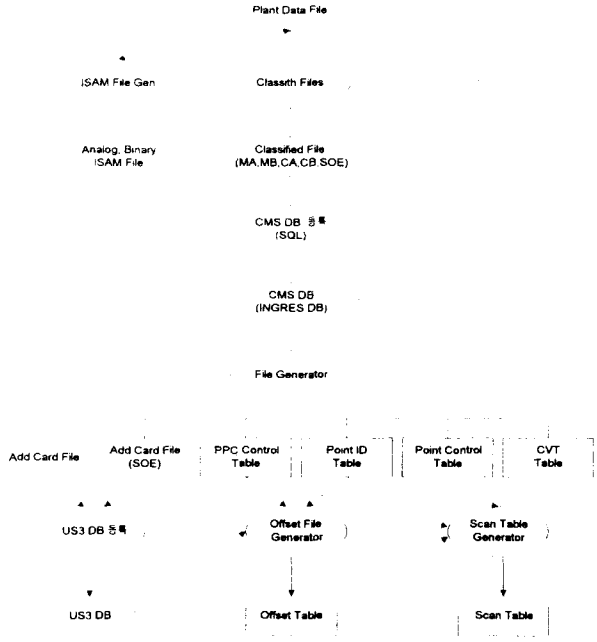

그림 7 디스플레이 모델을 위한 파일 생성 절차

Fig. 7 File generation procedure for Display model

에 저장된다. 전체 procedure는 그림 7과 같다. 그림에서 볼 수 있듯이 모두 4개의 file generator가 있어서 각 단계에서 해당되는 테이블들을 생성해낸다. 새로운 Display Model 이 추가되면 이 모듈이 필요로 하는 데이터에 관한 정보를 담은 plant data file을 만든다. 추가된 모듈의 존재(SYS_ID의 등록)와 새로운 모듈이 파일 생성 과정에서 사용하게될 파일명(6개의 테이블명, 2개의 카드화일명, ISAM화일명) 그리고 기타의 정보들(새로운 모듈이 사용하게될 공유메모리 key값 등록, communication manager와의 자료 전달을 위해 사용될 Queue의 생성을 위한 key값 등)의 수정 및 등록이 필요하다. 이러한 정보의 등록은 US3 가 제공하는 header 화일을 통해 이루어진다. 헤더화일의 적절한 수정과 새로운 모듈을 위한 정보의 추가만으로 file generator의 source code의 수정을 최소화하며 새로운 시스템의 추가가 가능하다. Data gathering task, communication maneger, scan task는 file generation procedure동안에 수정된 header 파일들을 include하여 추가된 모듈을 지원할 수 있게 Update될 수 있다.

4. 결론

영광3,4호기 시뮬레이터에는 Dynamic Model이 1차측 13 계통 과 2차측 15 계통 합하여 28개이며

Display Model이 원자로 노심감시계통등 8개로서 시뮬레이션 규모나 복잡성이크다. 또한 원자력 발전소의 특성상 안전을 위해 기존 계통에대한 변경이나 새로운 계통이 추가되기도한다. 그럴 때 Dynamic Model 과 Display Model 간의 인터페이스 과정은 매우 복잡하여 Dynamic Model 과 Display Model 개발자는 어려움을 겪게된다. 이때 두 개발자들로 하여금 같은 구조적 환경에서 개발하는 것으로 느끼며 자유롭게 개발 할수있도록 하는 것이 필요하며 이를 위해 file generation 방법을 통한 데이터 전송으로 해결하였으며 개발자는 통신에대한 부담없이 새로운 Display Model 추가가 쉽게 이루어짐을 확인하였다. 따라서 향후 발전소에서 운전원을 위한 디스플레이 환경이 추가되어질 때 시뮬레이터에서도 디스플레이 모듈 알고리즘 개발만 하면 쉽게 추가할 수 있다.

참고 문헌

[1] S3T,"US3(UNIX Simulation Software Support System) User Guide", June 1994.

[2] Burton J. Smith, "Interconnection Networks for Shared Memory Parallel Computers", IEEE, 1995.