2000 춘계 논문집
한국원자력학회

차세대 원자로 주제어실 동적 모형 개발

Development of a Prototype for KNGR Main Control Room.

정연섭, 이승호, 성찬호

전력연구원
대전광역시 유성구 문지동 305-380

요약

차세대원자로 주제어실은 CRT 기반의 감시 제어 시스템이다. 범용 공정제어에서 활용되는 MMI 기법의 적용뿐만 아니라 다양한 운전원 상호작용을 실험하고 평가하기 위하여 개발의 유연성이 풍부한 개발 툴을 선택하고 분산 시스템 구조로 설계 개발하였다. 감시변수는 초고속 이더넷에 멀티캐스팅으로 전송되고, 제어 명령은 원거리 함수호출 기법을 사용했다. 응용 프로그램은 독자 언어로 작성 후 C 언어로 변환 후 실행된다. 감시 변수는 1 초 이내, 제어 신호 0.3 초, 경보 등도 1 초 이내에 처리된다. 운전원 조작행위와 발전소의 중요 이벤트는 설계 분석을 위하여 기록되고 실험 후에 분석된다.

KNGR main control room is based on compact workstations. During the design a prototype of KNGR MCR has been developed to evaluate new design features with more flexible developing tools rather than commercial MMI tools. Data transfer is achieved with multicasting on the fast Ethernet, and control commands are delivered by remote procedure call. Application programs are coded in a simple language and translated in C language. This prototype shows good response for all plant events. The important parameters and operator's actions can be recorded and analyzed after experimental session.

1. 서론.

범용 공정에서는 MMI 상용 소프트웨어를 사용하여 주제어실을 개발한다. 원자력 발전소의 주제어실도 정도의 차이는 있지만 불가능하지 않다. MMI 상용 소프트웨어는 감시 제어 화면 작성과 실시간 데이터처리를 위한 전용 소프트웨어이기 때문이다. 그러나 차세대 주제어실의 설계는 다양한 운전자 화면과 상호작용을 끊임없이 추구하여 최적의 인터페이스를 만들어내는데 있으므로 기능이 고정된 상용 소프트웨어 보다는 프로그래밍 작업은 다소 많

더라도 유연성이 풍부한 개발 툴을 선택할 필요가 있다.

공정제어의 중요한 실시간 요건을 만족시키기 위하여 네트웍의 구조의 각종 소프트웨어 모듈간 통신 최적화가 필요하다. 현재 시장의 DCS 등의 소프트웨어는 이런 경험이 축적되어 나온 제품이다. 그러나 그 계층적인 구조가 개선의 여지가 없지는 않으며 이 프로토타입과 같은 개발 경험을 통하여 한층 최적화 될 수가 있다. 이 프로토타입은 서넛 차례의 최적화를 통하여 실시간 요건을 만족시킨다. 이런 구조는 범용 공정제어의 계층적인 네트웍 구성의 정형화의 한 사례로써 의미를 가진다.

발전소 변수는 현장에서 직접 생성되는 데이터뿐만 아니라 경보의 처리 및 파생 변수의 계산도 중요한다. 약 1200 여 개의 변수에 쉽게 접근하여 필요변수를 검색 후에 파생 변수의 값을 생성시키는 메커니즘은 실시간 계산의 중요한 부분이다. 이 시스템에서는 두 번 컴파일하는 기법으로 컴파일 시간과 실행시간을 줄일 수가 있었다.

마지막으로 많은 자료의 공유는 필연적으로 발전소 데이터 베이스의 효율적인 관리를 요한다. 이 프로토타입에서는 범용 데이터베이스 같은 고가의 소프트웨어 대신에 단순한 파일로서 발전소 데이터의 관리 및 일관성을 유지하였다.

2. 차세대 주제어실 Mockup 의 구조

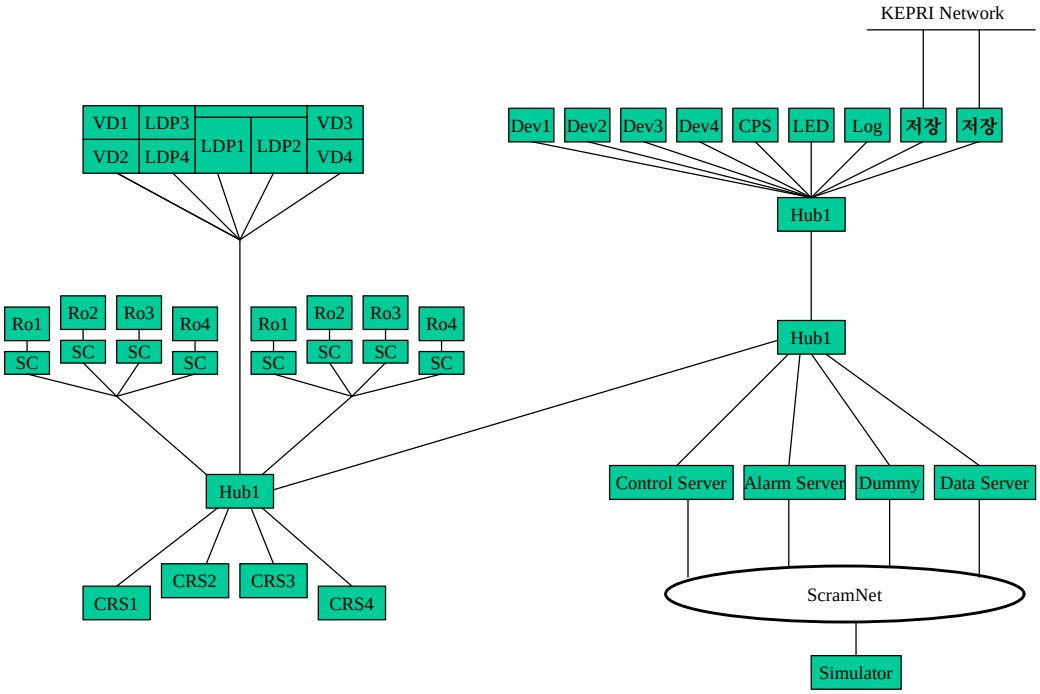

그림 1 차세대 주제어실 Mockup 의 구조

그림 1은 네트웍에 분산된 각종 컴퓨터를 나타낸다. 발전소 정보의 두개의 원천은 시뮬레이터와 운전원의 제어 행위이다. 생성시키는 정보의 양은 시뮬레이터가 운전원보다 많으므로 일반적으로 시뮬레이터가 정보의 생산자이고 운전원이 정보의 소비자로 간주된다.

시뮬레이터는 영광 3,4 호기 전범위 시뮬레이터이다. 시뮬레이터의 데이터는 ScramNet 이라는 메모리 공유기법을 통하여 제어서버, 경보 서버, 자료서버로 공유된다. 시뮬레이터는 4HZ 로 ScramNet 에 접근하고 각종 서버들도 1Hz 로 ScramNet 에 접근한다. 이렇게 공유되고 있는 변수의 개수는 약 12000 개 정도이다. 차세대 프로토타입에서는 이들 변수를 두개의 범주로 나누었다. 일반 발전소 상태 변수와 경보 변수이다. 약 12000 개의 변수 중에 4000 개는 경보 변수이다.

이 시뮬레이터는 기존 발전소 제어반의 입출력 장치를 염두에 두고 설계되어 디지털 통신으로 연결하기에 부적절한 부분이 있으나 ScramNet 의 접속빈도를 높이여 정확한 데이터를 받을 수가 있었다. 특히 빠르게 점멸하는 ON/OFF 신호는 주기적으로 데이터를 읽기가 용이하지 않다. 앞으로 차세대 원전 시뮬레이터가 개발될 때에 I&C 의 하위 계층만 개발하고 MMI 논리를 제외시키는 것이 오히려 나중에 MMI 화면 개발에 유리할 수 있다는 사례이기도 하다.

데이터 서버는 각종 센스의 값(아날로그)과 계기의 상태를(디지털) ScramNet 에서 읽어 소비자 층으로 방송한다. 멀티캐스팅을 활용하여 멀티캐스팅 그룹에 참여한 모든 소비자는 주기적으로 데이터를 받을 수가 있다. 또 데이터 서버는 계기의 제어 가능 여부도 똑같은 방법으로 보내준다. 서버는 타이머를 사용하여 주기적인 자료전송이 가능하고 클라이언트는 이벤트 방식으로 방송신호가 잡히면 깨어나 자료를 수신한다.

각종 값과 제어가능 표현은 일반 아스키 값으로 전달된다. 바이너리 데이터보다 통신량은 많으나 Mockup 에 설치되어 있는 네트워크의 폭이 넓어 실시간 자료 전송에는 무리가 없었다. 한번에 전송하는 응용층 프레임의 길이는 40K 정도이다. 이 프레임은 데이터 베이스에 나타난 변수의 순서대로 보내지므로 생산자와 소비자 사이에 변수의 순서만 일치한다면 변수 명을 전달할 필요가 없으므로 프레임의 길이가 작아지는 효과가 있다.

경보서버는 시뮬레이터에서 나온 경보의 발생 및 해제 신호와 운전원에게서 유래한 경보 인지 및 Flush 신호를 처리하여 소비자에게 전달하는 역할을 한다. 그림 2에 상 전이도가 표시되어 있다. 이와 같은 상태 전이도는 ISA 에서 언급하는 상태 전이도와 틀리다. 이유는 CRT 기반에서 경보 목록을 보여줄 때 빠른 갱신을 막기 위함이다. 이 상태 전이도는 경보 변수에 모두 적용되지 않으며 정보의 성격이 강한 경보에 대해서는 인지 신호가 다소 완화된 상태 전이도를 적용한다.

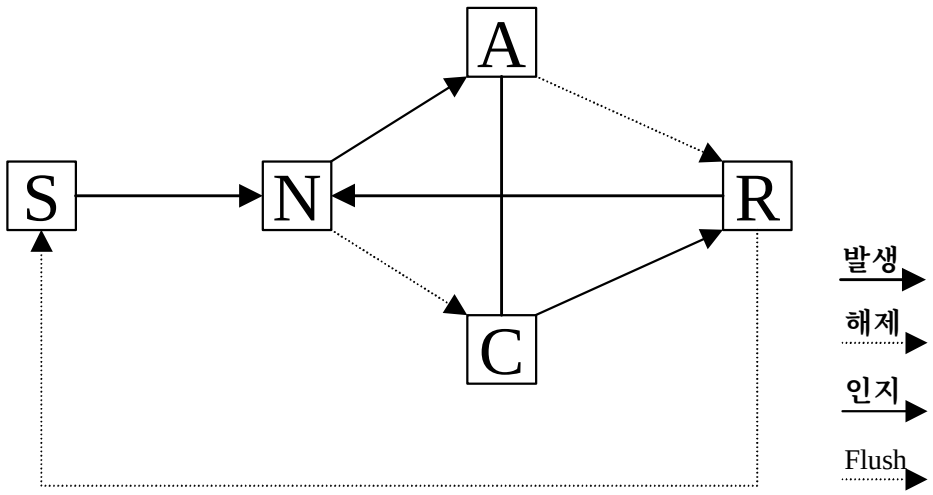

그림 2 경보의 일생

경보 서버는 경보 변수 중 하나라도 상태가 바뀌면 있으면 바뀐 경보 뿐만 아니라 다른 모든 경보도 소비자에게 전송한다. 그렇게 함으로서 나중에 참여한 소비자도 경보정보의 일관성을 유지시킬 수가 있기 때문이다. 경보의 상태 전이 루틴은 노르웨이 HRP 의 경보처리 루틴 COAST 를 활용하였다. 초기에는 Coast 에 많은 기능을 할애하였지만 최적화 시켜 가면서 COAST 의 기능을 완화 시키고 COAST 에 들어 가는 입력 값도 값에서 변경이 있을 경우에만 COAST 를 호출하였다. 최적화 후에 설계 기준 사고와 같은 경보 폭주 상태에서도 경보는 실시간으로 반응하였다.

경보 서버는 경보의 일생 처리뿐만 아니라 많은 파생 변수를 만들고 있다. 예를 들어 과냉각도, 부적절한 냉각 감시 변수 생성, 기준사고에 근거한 CFM/SPM 계산, 발전소 운전 모드, 냉각속도 등이 그것이다. 또 시뮬레이터에서 생성되지 않는 경보의 생성과 운전모드, 원인과 결과에 따른 경보의 억제 기능도 여기서 처리된다. 또 변수의 증감 여부를 처리하기 위해 약 1 분간의 자료를 원형 버퍼에 담아두고 증감을 계산한다. 이중의 일부는 데이터 서버에서도 똑 같은 알고리듬으로 계산된다. 이런 계산량에도 불구하고 충분한 응답 속도를 보였다. 초기의 개발에서는 HRP 의 COASTt 로 계산을 시도했지만 COAST 에서 처리하는 속도가 나빠 이 계산은 C 언어로 처리하였다.

변수 명을 배열의 색인으로 바꾸는 PreComplier 를 도입하여 빠른 컴파일 속도와 실행중의 빠른 응답속도를 얻을 수가 있었다. 해시 테이블 등의 기법을 시도하여 보았지만 컴파일 때에 너무 많은 메모리와 시간을 낭비하였다. 이 기법은 정형화되지 않는 파생 변수의 처리에 탁월한 기능을 보여준다. PreCompiler 는 변수이름을 데이터베이스의 색인으로 바꾸는 기능이다. 그림 3 에 예시되어 있다.

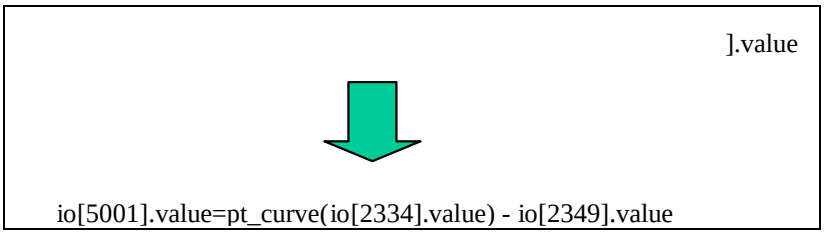

그림 3 Precompiler 의 기능

경보 서버에서 소비자로 자료를 전송하는 방식 또한 데이터 서버에서와 같이 멀티캐스팅 방법이다. 운전원으로부터 인지신호는 TCP/IP 통신 방식으로 받으며 소켓에 자료가 오면 깨어나는 이벤트 방식이다. 하나의 인지신호 뿐만 아니라 그룹 인지도 가능하며 이 경우에도 충분한 응답속도가 보장된다. 이 경보 서버는 각종 컴퓨터의 동기화 신호를 보내는 서버 역할도 한다.

3. 경보의 생성 및 통합 표현

앞에서도 언급했듯이 Mockup 에서는 일반 변수와 경보 변수로 양분되어 있다. 그렇지만 경보 변수는 일반적으로 센서에서 직접 생성되지 않고 일반 변수에서 유도되는 변수이기 때문에 일반 변수로부터 분리가 오히려 이상하게 여겨진다. 그러나 한 개의 일반 변수로부터 여러 개의 경보변수가 파생되는 경우나 혹은 여러 개의 일반 변수로부터 한 개의 경보 변수가 만들어지는 경우가 허다하기 때문에 오히려 경보 변수를 독립된 실체로 간주하는 것이 여러 면에서 유리하다.

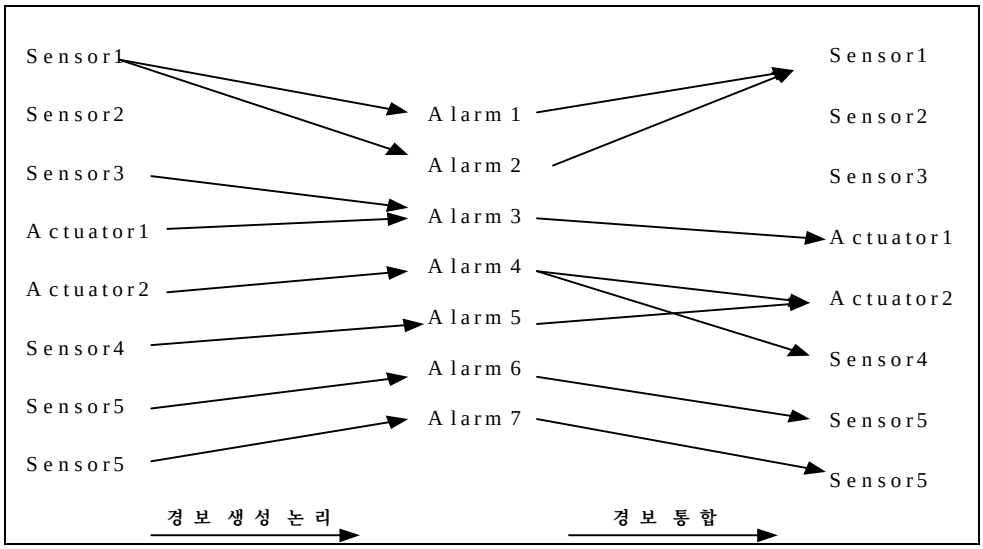

그림 4 경보의 생성 논리와 경보 표현을 위한 통합

그림 4 는 일반 변수로부터 경보가 발생되고 발생된 경보가 다시 일반 변수에 통합되어 표시되는 그림을 보여준다. 즉 생성 논리와 통합 논리가 다대다의 대응관계에 있을 때 경보 변수의 도입은 타당하고 편리하다. 현재의 Mockup 에서 경보는 이미 시뮬레이터에서 생성되어 있으므로 오로지 경보의 통합만 고려하면 된다. 이를 위해 각 일반 변수마다 대응되는 경보 변수의 목록을 지니고 있으며 이중에 하나라도 경보가 발생하면 일반 변수에도 경보 표시가 나타나도록 구현하였다.

뿐만 아니라 경보 목록을 보여주는 CRT 에서는 경보 변수만 나타나고 이 목록에서만 경보의 인지가 가능하도록 하였다. 일반 변수에 통합된 경보는 일반 변수의 인지에 의하여 경보 변수가 인지되지는 않는다. 운전모드에 따라 당연히 발생하는 경보가 있다면 억제는 되지만 운전원 요구에 의해서 볼 수는 있도록 처리된다.

4. 운전원 화면 개발 및 갱신

화면 작성기는 로컬 데이터베이스를 두고 화면 작성을 한다. 이 로컬 데이터 베이스는 서버에서 받는 발전소 정보에 의하여 끊임없이 갱신이 된다. 그러나 갱신이 되지 않는다고 하더라도 화면 작성기로 화면을 그리기에 아무 문제가 없다. 또 로컬 데이터 베이스 뿐만 아니라 화면마다 새로운 변수를 선언하여 사용할 수도 있다.

화면 작성기의 중요한 또 하나의 기능의 발전소 계기의 각종 심벌을 작성하여 반복적으로 사용하는 기능이다. Mockup 에서 사용되는 화면 작성기는 HRP 에서 만들어진 Picasso-3 를 사용하였다. 현 Mockup 의 개발에는 100 여종 이상의 심벌을 사용하고 있다. 이들 심벌은 객체지향 특성을 중 상속의 특성을 가지기 때문에 심벌 자체에서 내재된 동적 특성뿐만 아니라 심벌을 선택하여 화면에 삽입할 때에 심벌의 특성을 변경시킬 수도 있다.

화면 작성기의 이런 특성으로 인하여 화면 작성자는 프로그래머와는 독립적으로 자신의 화면을 생성할 수 있다. 현재의 Mockup 에서는 서버에 있는 데이터 베이스와 각종 운전원 CRT 에서 사용하는 로컬 데이터 베이스를 일치시키는 작업이 필요하며 새로운 변수가 추가될 때마다 데이터 베이스 파일을 일치시켜 나아간다.

차세대 설계의 중요한 특성 중의 하나는 강력한 화면 전환이다. 자신의 CRT 뿐만 아니라 옆 CRT, LDP 의 화면을 제어할 수가 있다. 또 제어를 위한 제어 화면을 호출할 수도 있다. 이런 기능은 화면작성기에서 외부 프로그램을 호출할 수 있는 기능이 있을 때 가능하다. Picasso-3 의 기능은 화면작성기와 외부 호출 프로그램을 마치 단일 프로세스처럼 결합시킬 수 있는 기능이 있고 이 기능을 많이 활용하였다.

하나의 실행 파일로 RO,TO,CRS 등 여러 환경에서 작동하기 위하여 프로그램 초기 실행 때 환경 설정 파일을 읽어 자신의 CRT, 오른쪽 CRT, 왼쪽 CRT 등의 정보를 얻도록 하였다. 이런 환경 설정은 실행파일을 단일화 시킴으로 개발시간의 이점, 시스템의 안정성을 획기적으로 증대 시킬 수 있다.

5. 수동 디지털 제어와 프로세스의 응답 속도.

PLC 혹은 자동 제어기에 의하여 제어되는 계기들은 빠른 PLC 의 주사 속도로 인하여 충분한 피드백이 나타난다. 물론 시간 지연이 없는 아날로그에 비할 수는 없겠지만 많은 응용분양에서 타당성이 검증되었다. 그러나 MMI 를 통한 제어 명령은 제어의 피드백까지 다소의 시간 지연이 있다. 제어 신호는 이벤트 방식으로 ScramNet 까지 지연 없이 전달되지만 시뮬레이터가 그 신호를 받기까지 최대 0.25 초, 피드백이 계산되기까지 0.25 초, 서버에서 이를 읽어 제어기에 나타나기까지 0.5 초의 시간 지연이 있다. 일반적으로 MMI 와 DCS 혹은 PLC 와의 통신은 주기적으로 1 초 정도가 보편적이므로 1 초보다 빠른 응답 속도를 보이는 프로세스는 MMI 를 통한 제어에는 다소 무리가 예상된다. 현재의 Mockup 에서는 SoftControl 의 경우 다소 빠른 속도의 통신부하를 주면 컴퓨터 자원에 다소 부하가 걸렸다. 그러나 대부분의 이런 프로세스는 수동보다는 자동으로 설정치를 입력할 수 있으므로 제어에는 아무런 문제가 없어 보인다.

6. 운전원 행위 기록 시스템.

운전 행위 기록 시스템은 운전원의 화면 전환, 제어 행위, 경보의 상태전이, 절차서 단계 이동 등이 기록되며 또 중요한 발전소 변수가 기록된다. 또 이런 이벤트가 생길 때마다 운전원 CRT 와 화면이름 계기의 이름과 시간도 동시에 저장된다.

이벤트가 발생하는 컴퓨터가 산재 되어 있으므로 각종 시각의 동기화가 필요하다. 현 Mockup 에서는 마스터 시각 서버를 두고 각 컴퓨터를 동기화 시키고 있다. 그러나 동기화 시점이 프로그램의 초기 실행 때 한차례 뿐으로 시간이 지나면서 컴퓨터간 시계차이가 발생한다. 즉 동기화에도 불구하고 운전원 행위의 순서를 완벽하게 재현하기 위해서는 Mockup 시간을 사용하지 못하고 운전원 행위가 기록되는 로컬 컴퓨터의 시간을 기준으로 삼았다.

기록된 자료는 경보의 경우에 발생에서부터 인지가 되기까지의 시간을 측정하는 분석을 하였다. 그러나 이와 같은 작업을 통하여 얻은 경험은 방대한 자료를 처리하기 위해서 분석용 프로그램의 필요성이다. 현재 이 프로그램의 개발이 진행 중에 있다.

7. 참조 문헌

1. HRP, User Manuals for Coast and Picasso-3

2. W. Richard Stevens, Advanced Programming in the Unix Environment, Addison Wesley Longman, 1993