자동화 기기를 위한 제어 시스템에서의 통신 Emulate 개발

Development of Communication Emulate Technique in control system for Automatic Machine.

이 범 석, 정 화 영*

혜천대학 컴퓨터통신계열, *예원대학교 커뮤니케이션디자인학과

<요 약>

자동화 기기 분야에서 컴퓨터의 적용 및 응용은 하드웨어 발달에 따라 매우 빠르고 민감하게 반영되어왔다. 이는 컴퓨터 하드웨어의 빠른 개발 주기에 맞춰 저 가격, 고 효율성, 높은 신뢰성, 호환성 등의 장점을 가진 PC가 현대의 컴퓨터 흐름을 주도하게 되면서 자동화 산업분야 또한 이를 적용하여 왔기 때문이다. 이에 따라, 자동화 기기 분야에서는 고 가격, 긴 개발기간 등을 필요로 했던 과거와 달리 저 가격, 짧은 개발기간, 다양한 개발환경 등을 이룰 수 있었다. 또한, 생산량 증가에만 의존하던 과거와 달리 현대에 이르러서는 시스템의 최적화, 효율의 극대화, 시스템의 안정성, 운용의 편리성, 호환성 등의 개념들이 도입되고 있는 것이다.

자동화 기기를 구성하는 요인으로는 크게 시스템의 틀을 이루는 기계부분과 이를 제어하는 제어 시스템부로 나뉠 수 있다. 제어 시스템에서는 기계부분의 동작을 제어하는 동작 제어부와 이에 관한 정보를 화면에 나타내는 GUI(Graphical User Interface)부분으로 나뉘게된다. 현재에는 이를 통합하여 하나의 하드웨어에서 제어부와 GUI를 모두 담당하는 방법이 연구 진행되고 있으나, 하드웨어를 둘로 나누거나 하나로 하여도 제어부와 GUI 사이의 통신부분은 빼놓을 수 없는 요소가 된다.

따라서, 본 논문에서는 시스템의 안정성을 위하여 두 시스템간에 송·수신되는 데이터를 추적할 수 있도록 하는 Emulate 기법을 구현 및 개발하고자 한다. 이는, 두 시스템간의 통신 데이터를 실시간으로 누적, 저장하여 사용자로 하여금 시스템의 운용상태를 분석할 수 있게 하였으며, 시스템 오류발생 시 Emulate 자료를 근거로 시스템의 운용상태를 파악할 수 있게 하였다.

1. 서 론

산업 사회에서 자동화 시스템의 도입은 새로운 시대로의 도약을 의미했다. 또한, 컴퓨터 하드웨어의 급속한 발달로 인하여 컴퓨터의 사용이 일반화되면서 자동화 시스템 분야는 일대 변화가 일어났다. 특히, 고가의 시스템을 복잡하게 장착했던 고전적인 방식에서 값싼 PC로 대체되었으며 사용자의 요구조건도 다양하게 변화되었다. 즉, PC를 자동화 설비의 제어 시스템으로 활용하면서 개발기간이 짧아지고 개발비용이 저렴하며 보다 효율적인 개발환경과 시스템의 안정된 구성이 가능해졌다. 또한, 사용자 친화적인 운용환경을 위하여 윈도우 환경의 GUI 기반으로 구현되어 사용자는 보다 쉽게 자동화 시스템을 운영할 수 있었다. 즉, PC 기반의 GUI 시스템은 강력하고, 저렴하며, 충분한 교육을 받은 인수자들이 수정하기 쉽다는 장점을 주었다[1].

자동화 시스템 개발에 있어서는 시스템의 틀을 이루는 기계부분과 제어 시스템 두 가지가 모두 필요했고, 운영 체제로는 실시간 제어가 요구되었다[2]. 제어 시스템은 실시간 제어를 기반으로 기계부분을 제어하는 동작 제어부분과 사용자에게 자동화 시스템의 운용정보를 처리하여 나타내는 GUI 시스템으로 나뉜다. 즉, GUI 시스템은 동작 제어 시스템으로부터 전해지는 운용 데이터를 근거하여 이를 단계별로 통합 산출하여 사용자에게 나타내는 것이다. 이에, 제어 시스템의 개발은 제어와 GUI 두 개의 부분으로 나뉘어져 개발되며

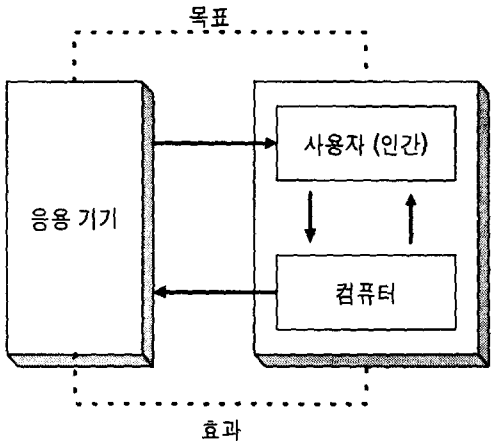
이들 사이에는 통신의 수단으로 연결된다. 따라서, 두 시스템 사이의 통신은 자동화 시스템의 구성요인에 중요한 요소가 된다. 실제, 자동화 시스템이 개발되어 현장에서 운용시 통신부분의 단절로 인하여 동작이 중지되는 경우가 발생된다. 또한, 자동화 시스템의 운용환경에서 소수의 사용자가 많은 자동화 시스템을 관리, 운용하고 있기 때문에 사용자 없이 동작 중에 정지되는 경우 이의 원인을 추적할 수 있는 데이터가 없다.

따라서, 본 논문에서는 두 시스템간의 송·수신되는 데이터를 실시간으로 저장하여 자동화 시스템의 갑작스런 운용정지에 대하여 그 원인을 추적할 수 있도록 하였다. 또한, 송·수신되는 데이터를 누적하여 저장함으로써 자동화 시스템의 운용상황을 분석하는 자료로 활용할 수 있도록 하였다.

2. 자동화 시스템의 개요

2.1. 자동화의 개요

컴퓨터의 도입으로 인류는 많은 부분을 의지하고 응용하였다. 즉, 인간이 수 작업으로 처리하던 과정을 기계가 대신하고 이를 컴퓨터가 제어하면서 보다 빠르고 정확한 작업 및 생산능력을 높일 수 있었다. 이에 관하여 다음 <그림 1>은 컴퓨터와 사용자(인간)의 관계를 도식화한 것이다[3].



< 그림 1 > 응용 기기에 대한 사용자와 컴퓨터의 관계

즉, 자동화의 개념은 단위 자동화 시스템과 이를 이용하여 생산성과 유연성을 달성할 수 있도록 하는 생산공정의 시스템화를 말한다. 자동화의 발전을 보면, 경영전략의 중심은 시장동향에 대한 대응도 중요한 요건이었는데 그 중심은 안정된 품질을 확보하면서 메이커로서의 경제적, 인적, 자원적인 효율을 추구하려는 것이었고 또한, 풍부한 생산량의 확보가 주요과제였으며, 점차 자금의 투자규모와 제조 코스트를 최소화하면서 대량생산에 의한 양의 확보와 동일제품의 반복생산에 의한 품질 조성기술의 학습효과에 의하여 제품의 안정성을 확보하려는 것이다[4]. 즉, 자동화는 설계나 생산 또는 관리기능에 컴퓨터의 복합기술을 적용하여 생산성과 유연성을 동시에 달성할 수 있도록 하는 생산활동 전체의 시스템화 경향 및 이에 따른 경제적, 사회적 효과의 총체라고 할 수 있다. 따라서, 자동화 시스템의 도입효과를 분석하는데 사용되는 지표는 일반적으로 노동력 절감효과, 설비 가동율 향상효과, 품질향상효과, 노동내용 또는 노동환경의 개성효과, 설비의 유연화 효과, 생산효율의 향상효과, 공정관리의 용이화 효과 등으로 구분된다.

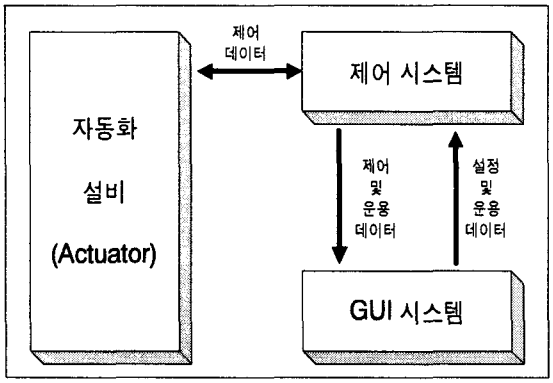
2.2 자동화 시스템 통신망 기술의 구성

자동화 시스템 구성 및 개발에 대한 전통적인 접근방법은 입·출력을 위한 PLC(Programmable Logic Controller)와, 제어 순서, 동작 제어 시스템 및 UI(User Interface)를 위한 PC등을 사용하는 것이다. 이를 위하여 다음과 같은 사항이 요구된다[2].

1. 운영체제 , 2. 멀티 태스킹(Multi-tasking) 및 멀티 스레딩(Multi-threading)이 가능한 운영체제 지원, 3. PC 기반의 하드웨어, 4. 객체지향 소프트웨어개발, 5. 구조적인 오류 핸들링(Error Handling), 6. 윈도우 기반의 UI.

그러나, 본 논문에서는 이를 크게 동작 제어 시스템과 GUI 시스템의 두 부분으로 나누었다. 즉, 동작제어 시스템은 멀티 태스킹이 가능한 운영체제를 사용하여 자동화 시스템의 동작제어를 담당하였고, GUI 시스템은 윈도우 기반에서 오류 핸들링과 운용 데이터의 산출을 담당하였으며 이를 위하여 PC를 사용하였다. 다음 <그림 2>는 자동화 시스템 구성을 나타

낸다.



< 그림 2 > 자동화 시스템 구성

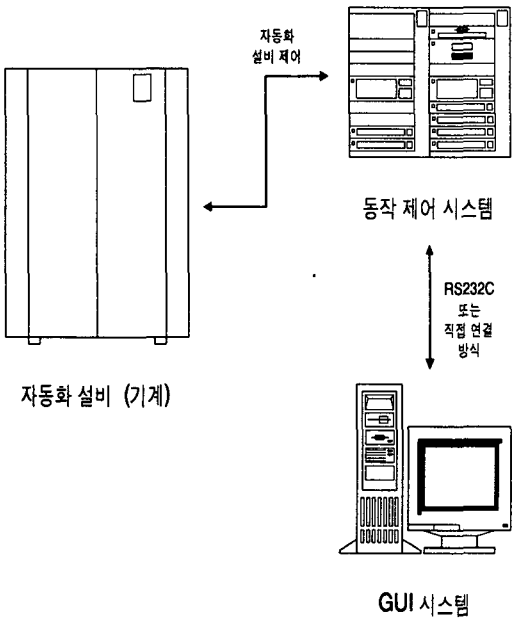
따라서, 동작 제어 시스템과 GUI 시스템 사이의 데이터 교환을 위한 통신수단이 필요하게된다. 이를 위하여 본 논문에서는 RS232C를 이용하였다. 이를 통하여, 동작 제어 시스템에서는 GUI 시스템에서 자동화 시스템의 운용 데이터 산출에 필요한 제어 및 운용 데이터를 전송하며, GUI 시스템에서는 동작 제어 시스템이 원활한 동작제어를 수행하도록 사용자의 운용 설정 데이터 및 동작중의 오류 핸들링을 위한 운용 데이터를 전송하였다. 따라서, 자동화 시스템의 운용 중에 발생되는 동작 상황은 동작 제어 시스템으로부터 GUI 시스템을 거쳐 사용자에게 제공되고, 오류 발생시 사용자는 이를 적절한 상황에 맞게 데이터를 입력하면 GUI 시스템을 거쳐 동작 제어 시스템으로 전송되어 자동화 시스템의 동작 제어에 반영할 수 있는 것이다.

3. 자동화 시스템 통신망

3.1. 기존의 자동화 시스템 내부 통신망

기계(자동화 설비), 동작 제어 시스템, GUI 시스템으로 분류되는 자동화 시스템에서 각 시스템을 연결하는 내부 구조는 RS232C를 이용하거나 사용자 장치를 이용한 직접 연결방식을 들 수 있다. 물론, 현재에는 GUI시스템과 동작 제어 시스템을 하나의 하드웨어에서 처리하는 방식이 연구·개발되고 있으며, 이 경우 두 시스템간의 외부 연결 통신망은 필요 없게 된다. 그러나, 하나의 하드웨어에서 두 시스템을 포함하여도 멀티 태스킹이 요구되는 운영체제 기반의 제어부와 윈도우 운영체제 기반의 GUI부분 사이의 자료교환은 필수적인 요소가 된다.

동작 제어 시스템과 GUI 시스템으로 분류되는 기존의 방식에서 RS232C로 시스템을 연결하는 방법은 추가적인 개발비용이 들지 않는다는 장점으로 인하여 주로 이용되고 있다. <그림 3> 자동화 시스템의 내부 통신망 구성은 이를 도식화한 것이다.



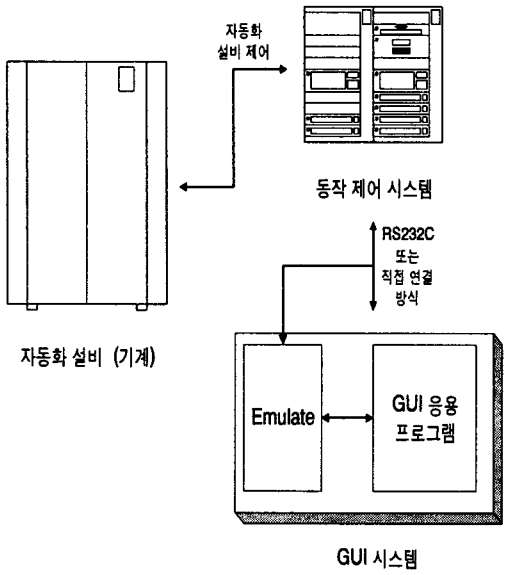
<그림 3> 자동화 시스템의 내부 통신망 구성

두 시스템간에 연결된 RS232C 통신에서는 정해진 프로토콜에 의하여 데이터의 송·수신이 이루어진다. 따라서, 자동화 시스템의 초기 설치시 GUI시스템은 운영에 관한 설정 데이터를 동작 제어 시스템으로 전송하고 동작이 이루어지면 운영 및 임의 동작 정지 데이터를 동작 제어 시스템이 GUI 시스템으로 전송한다. GUI 시스템은 이를 분석하여 오류 수정 명령을 동작 제어 시스템으로 전송하여 자동화 시스템이 원활히 동작하도록 한다. 그러나, 동작 제어 시스템이 갑작스런 오류가 발생하거나 어떠한 요인으로 인하여 통신 연결이 끊기는 경우 자동화 시스템은 동작을 중지하게 되며, 사용자가 없는 경우 자동화 시스템의 동작중지 요인을 찾을 수 없게 된다. 따라서, 자동화 시스템의 안정된 운용을 위하

여 통신상태를 주시할 수 있는 Emulate 기법이 필요하게된다.

3.2. 자동화 시스템 통신망에서의 Emulate 기법

본 논문에서는 두 시스템간에 연결된 통신상태를 주시하기 위하여 GUI 시스템 내부에 Emulate 기능을 추가하였다. 따라서, 동작 제어 시스템과 GUI 시스템 사이에 송·수신되는 모든 데이터는 GUI 시스템 내부의 Emulate에 모두 저장된다. 다음 <그림 4> 자동화 시스템 통신망에서의 Emulate는 이를 도식화한 것이다.



<그림 4> 자동화 시스템 통신망에서의 Emulate

Emulate는 GUI 시스템 내부에서 데이터의 송·수신이 이루어지는 부분에 위치하며, 통신이 이루어지는 시점마다 송·수신 데이터를 지정된 디스크의 위치에 저장하게된다. 또한, 송·수신되는 시점에서 날짜가 변경되면 다시 해당 날짜의 파일명으로 통신 데이터를 저장한다. 따라서, GUI 시스템의 오류가 발생하지 않는 한, 저장된 통신 데이터를 근거로 현재까지 자동화 시스템 내부의 동작 제어 시스템이 어떠한 상황이었는지 분석할 수 있다. 또한, 두 시스템간의 통신 데이터가 올바르게 이루어지는지 검색할 수 있으며, 날짜가 변경된 경우 해당 날짜의 파일명으로 다시 생성하여 통신 데이터를 저장함으로써 자동화 시스템의 날짜별 상황도 분석할 수 있다.

4. 자동화 시스템의 통신 Emulate 기법 구현

본 논문에서는 두 시스템간의 통신 Emulate를 위하여 C 언어로 구현하였으며 이를 위하여 Windows98 운영체제를 사용하도록 하였다. 통신 Emulate는 GUI 시스템의 송·수신되는 위치에서 구현되어 데이터의 송·수신이 이루어지는 시점에서 모든 통신 데이터를 저장하게된다. 이에, GUI 시스템은 RS232C 통신에 의하여 동작 제어 시스템으로부터 제어 및 운용 데이터를 수신 받기 위하여 다음과 같이 윈도우 메시지를 받아 사용하였다.

```
int NotifyStatus;

case WM_COMMNOTIFY:
        NotifyStatus = LOWORD(lParam);
        if(NotifyStatus & CN_RECEIVE) {
         Communication_Receive();
         //제어 및 운용 데이터 받는 루틴
        }
```

이 경우, 수신이 이루어지는 수신부의 Emulate는 Communication_Receive(); 에 위치하여 수신 데이터를 다음과 같이 저장하게된다. 즉, GUI 시스템에서는 다음과 같이 동작 제어 시스템으로부터 전송되는 데이터를 처리하기 전에 수신큐의 데이터를 받아 Emulate에서 처리하도록 한다. 이때, 수신큐에 있는 데이터가 정상적으로 수신이 완료되었는지 확인한 후에 Emulate 기능을 수행한다.

```
void Communication_Receive(void)
{
   데이터의 수신 확인;
   Emulate_File_Comm_Char(통신 수신큐의
데이터);
   수신 데이터 처리;
}
```

Emulate에서는 아직 처리되지 않은 수신 데이터를 날

짜별 파일명으로 저장한다. 이때, 데이터의 저장방식은 누적을 사용하여 이전 데이터의 손실을 막는다.

```
void Emulate_File_Comm_Char(char
*Comm_Emulate_Char)
{
        char Date_Comm[10],
Time_Comm[10], File_Name_Comm[50];
        FILE *FileHand;
        char Real_File_Name[5];
        int i;

        _strtime(Time_Comm);
        _strdate(Date_Comm);
        for(i=0; i<2; i++)
          Real_File_Name[i] =
Date_Comm[i];
        for(i=3; i<5; i++)
          Real_File_Name[i-1] =
Date_Comm[i];
        Real_File_Name[4] = 0x00;

        sprintf(File_Name_Comm,
"%s%s.txt", Daily_Comm_Copy,
Real_File_Name);

FileHand=fopen(File_Name_Comm,"a+");
        fprintf(FileHand, "%s  %s : %s₩n",
Date_Comm, Time_Comm,
Comm_Emulate_Char);
        fclose(FileHand);
}
```

GUI 시스템에서 동작 제어 시스템으로 전송하는 부분에서도 위와 같이 Emulate 부분을 삽입하여 송신 데이터를 저장하도록 한다.

```
BOOL SendDataa(HWND hDlg, 송신 데이터)
{
    송신 데이터의 분류;
    동작 제어 시스템으로 데이터 송신;
    송신 데이터 확인;
    Emulate_File_Comm_Char(송신 데이터);
}
```

즉, GUI 시스템에서 동작 제어 시스템으로 데이터를 송신하고 이를 확인한 후에 Emulate에 송신 데이터를 저장한다. 이로써, 두 시스템간의 데이터가 송·수신되는 시점마다 통신 데이터를 저장할 수 있었으며, 파일명을 날짜로 함으로서 통신 데이터를 관리할 수 있었다. 이는, GUI 시스템 내부에서 구현됨으로써 두 시스템 중 어느 한 시스템에서 오류가 발생하기 전까지 통신상에서 송·수신되는 모든 데이터를 자동으로 저장할 것이다.

5. 결 론

자동화 시스템이 개발되어 도입, 생산, 폐기되기까지 시스템의 개발자는 시스템의 안정화 및 효율화에 지속적인 수정, 보완에 노력하여야 한다. 또한, 자동화 시스템의 도입이후 사용자도 지속적으로 수정 및 보완에 관한 요구조건을 제시하고 있다. 이는, 많은 자동화 시스템을 소수의 사용자가 관리, 운용하고 있고 자동화 시스템에 의해 생산된 제품에 관하여 일반적으로 신뢰하고 있기 때문에 자동화 시스템의 안정화, 효율화 및 유지보수가 매우 중요한 요소가 되기 때문이다. 따라서, 자동화 시스템의 안정된 운용은 시스템의 개발자가 고려하여야하는 중요한 요소가 된다.

이를 위하여, 본 논문에서는 제어 시스템이 동작 제어 시스템과 GUI 시스템으로 분류되는 자동화 시스템에 있어서 제어 시스템의 안정된 운용을 위하여 내부 통신망을 분석하는 Emulate 기법을 추가하였다. 그 결과, Emulate에서 산출된 데이터는 자동화 시스템의 동작 중 갑작스런 오류 발생이나 임의 정지시에 이를 추적할 수 있는 분석자료가 되었다. 또한, 개발자는 이를 분석하여 자동화 시스템의 운용상황을 검색할 수 있었고, 사용자는 누적된 날짜별 데이터를 이용하여 분석하고자하는 시점에서의 자동화 시스템에 관한 동작상황을 알 수 있었다. 그리고, 자동화 시스템의 설치 이전에 정상적으로 통신이 이루어지고 있는지 확인할 수 있었다. 본 논문에서 제안된 Emulate 기법은 하나의 하드웨어에서 두 개의 시스템을 사용하는 방법에서도 같은 형식으로 적용될 수 있다. 이 경우, 두 시스템 사이의 데이터 교환을 위하여 일반적으로 시스템의 메모리를 이용하게 되는데 이때 시스템의 메모리를 Emulate하는 기법을 위와 같은 방법으

로 적용할 수 있다.

그러나, 본 논문에서 구현된 Emulate 기법은 통신이 이루어지는 시점마다 디스크에 저장함으로써 데이터의 처리 속도가 늦어지며 과도한 디스크의 사용을 유발한다. 즉, 자동화 시스템의 운용 중에 이루어지는 통신 데이터는 짧은 시간에 고속으로 많은 데이터가 송·수신되며 이때마다 송·수신 데이터를 디스크에 저장하게되는 것이다. 따라서, 자주 반복되는 디스크의 사용으로 인하여 디스크의 수명 단축과 내부 데이터의 처리속도 저하가 발생된다. 이러한 문제점을 보완하면 자동화 시스템에서 적용되는 Emulate를 보다 효율적으로 구현할 수 있을 것이다.

참고문헌

[1] Kevin Borthwick, Pardip Thind, and Philip Fransen, "PC-Based Operator Interface", IEEE Industry Application Vol 4 No4 July/August 1998.

[2] R.L.Anderson, J.M.Reagin, T.D.Garner, T.E.Sweeny, "Open-architecture controller solution for custom machine system", SPIE Vol. 2912, 1997, 9.

[3] John Long, "Specifying relations between research and the design of human-computer interactions", International Journal of Human-computer Studies, 44, 875-920, 1996

[4] 문희화, "국내 공장자동화 현황조사 보고서 FA 조사 연구보고서 '92-01호", 한국 생산성본부, 1992, 12.

[5] 임기평, "공장자동화 기술의 도입 및 활용에 관한 실용분석", 한국중소기업학회, Vol. 19, No. 1, 1997, 6.