# 자동화시스템을 위한 컴포넌트 기반의 통신 Emulate 구현

정 화 영
예원대학교 전자상거래학과

## 요 약

현재, 자동화 시스템을 위한 통신분야는 TCP/IP를 기반으로 하는 네트워크 기반 원격제어 부분과 시스템의 각 내부 장치들간의 통신을 위한 간단한 직렬통신으로 나뉠 수 있다. 네트워크의 비약적인 발전에도 불구하고 내부 제어를 위한 통신부분은 현재까지 간단한 RS232기반을 사용하고 있다.
또한, 시스템의 개발기법은 각 기능단위의 모듈러 프로그래밍 기법에서 객체지향 프로그램으로 발전하였으며, 현재는 소프트웨어의 부품단위인 컴포넌트 기반 개발기법으로 발전하였다. 이는, 비즈니스 로직을 포함하는 독립적인 운영단위를 조립하며 새로운 시스템의 개발로 이어지는 기법으로서 소프트웨어 개발기법의 새로운 대안으로 제시되고 있다.
따라서, 본 연구는 컴포넌트 기반의 개발 기법을 적용한 GUI기반의 RS232C에서의 내부통신 Emulate를 구현하였다. 이는, 수신부와 송신부로 나뉘어지는 통신 제어부분을 컴포넌트화 하였으며, 이를 합성함으로서 제어부에서는 송,수신 데이터의 핸들링을 담당하도록 하였다.

# The Implementation of Communication Emulate Based on Component For Automation System

Hwa-Young Jeong

## ABSTRACT

Currently, communication field for automation system can be divided by simple serial communication for communication between each internal devices and network base remote control system that is based on TCP/IP. In spite of great development of network, communication part for internal control is using simple RS232 base until present.
Also, development techniques of system developed by object oriented program in modular programming techniques of each function unit. Currently, it developed by component base development technique that is parts unit of software. This is presented by the new alternative of software development techniques as techniques to composition independent operation unit including business logic and is connected to development of new system.
Therefore, this research implemented internal communication Emulate in RS232C based on GUI that apply development techniques of component base. that is, I maked component to commnication control part between receiving and sending and, as composite it, Control part did to handling between send and receive data.

# Ⅰ. 서 론

PC를 자동화 설비의 제어 시스템으로 활용하면서 개발기간이 짧아지고 개발비용이 저렴하며 보다 효율적인 개발환경과 시스템의 안정된 구성이 가능해졌다. 즉, PC 기반의 GUI 시스템은 강력하고, 저렴하며, 충분한 교육을 받은 인수자들이 수정하기 쉽다는 장점을 주었다[1]. 또한, 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. 이에, 제어 시스템의 개발은 제어와 GUI 두 개의 부분으로 나뉘어져 개발되며 이들 사이에는 간단한 RS232C 통신이 이용된다.

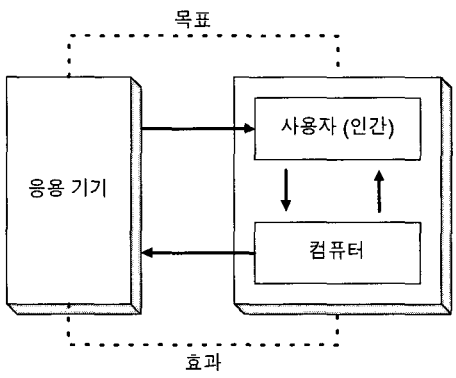
이러한 시스템 개발기법은 기능단위를 라이브러리화 하는 기존의 모듈러 프로그래밍에서 소프트웨어를 하나의 부품으로 보는 컴포넌트 기반 개발 방법론으로 연구 및 변화되고 있다. 컴포넌트 기술은 소프트웨어 프로그래밍에서 하드웨어 개발 환경처럼 소프트웨어 Plug-and-Play 방식으로 시스템을 구축하는 '합성을 통한 시스템 구축'으로의 전환을 목적으로 한다[3, 4]. 따라서, CBD를 효과적으로 지원하기 위해서는 응용 컴포넌트들이 서로 정확하게 결합하여 작동할 수 있는 아키텍처를 기반으로 컴포넌트의 생성과 합성작업이 이루어질 수 있어야한다[5]. 이를 위하여 , 컴포넌트 합성방법에서는 컴포넌트간의 인터페이스 불일치를 해결할 수 있어야하며, 독립적인 컴포넌트의 메소드 수정 없이 합성할 수 있어야 한다[6].

따라서, 본 논문에서는 컴포넌트를 기반으로 하는 GUI기반의 통신 시스템을 구현하였다. 즉, 통신의 중요 로직인 송,수신부는 C++언어 기반의 컴포넌트로 구현하며, 이를 제어하는 제어부는 합성된 통신 시스템에서 처리하도록 한다. 또한, 송·수신되는 데이터를 누적하여 저장함으로써 자동화 시스템의 운용상황을 분석하는 자료로 활용할 수 있도록 하였다.

# Ⅱ. 관련 연구

## 2.1 자동화 시스템

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[7].




(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

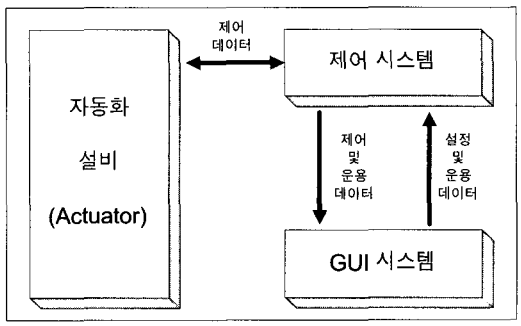
대응도 중요한 요건이었는데 그 중심은 안정된 품질을 확보하면서 메이커로서의 경제적, 인적, 자원적인 효율을 추구하려는 것이었고 또한, 풍부한 생산량의 확보가 주요과제였으며, 점차 자금의 투자규모와 제조 코스트를 최소화하면서 대량생산에 의한 양의 확보와 동일제품의 반복생산에 의한 품질 조성기술의 학습효과에 의하여 제품의 안정성을 확보하려는 것이다[8]. 즉, 자동화는 설계나 생산 또는 관리기능에 컴퓨터의 복합기술을 적용하여 생산성과 유연성을 동시에 달성할 수 있도록 하는 생산활동 전체의 시스템화 경향 및 이에 따른 경제적, 사회적 효과의 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.



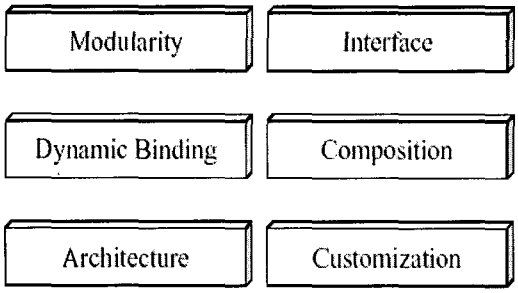

(그림 2) 자동화 시스템 구성

즉, 동작 제어 시스템은 멀티 태스킹이 가능한 운영체제를 사용하여 자동화 시스템의 동작제어를 담당하였고, GUI 시스템은 윈도우 기반에서 오류 핸들링과 운용 데이터의 산출을 담당하였으며 이를 위하여 PC를 사용하였다. (그림 2)는 자동화 시스템 구성을 나타낸다.

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 동작상황은 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.

### 2.3 컴포넌트 기반 소프트웨어 개발기법

CBSD(Component-Based Software Development)는 이미 존재하는 소프트웨어 컴포넌트를 조립함으로써 시스템을 개발하는 방법이다[10]. 컴포넌트를 이용하여 시스템을 개발하는 것은 소프트웨어 개발시간과 비용을 줄이며, 재사용 및 유지보수, 신뢰성을 높일 수 있다. 컴포넌트의 일반적인 요소는 다음 (그림 3)과 같다[11].



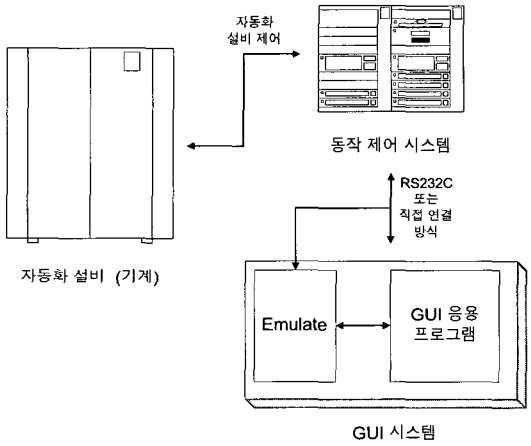

(그림 3) 컴포넌트의 일반적인 요소

컴포넌트 개발기법은 크게 3가지 관점으로 나타낼 수 있다. 이미 만들어진 컴포넌트를 이용하여 새로운 소프트웨어를 구성하는 통합자 관점, 재사용될 수 있는 컴포넌트를 만드는 제공자 관점, 제공자가 만들어놓은 컴포넌트를 통합자가 효과적으로 이용할 수 있도록 도와주는 중간자 관점이 있다[12].

## Ⅲ. 컴포넌트 기반 통신 Emulate 개발

### 3.1 자동화 시스템에서의 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 없게된다. 그러나, 하나의 하드웨어에서 두 시스템을 포함하여도 멀티 태스킹이 요구되는 운영체제 기반의 제어부와 윈도우 운영체제 기반의 GUI부분 사이의 자료교환은 필수적인 요소가 된다.



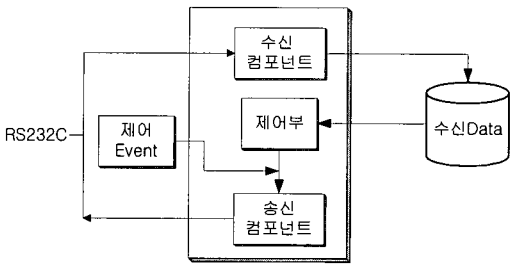

(그림 4) 자동화 시스템의 내부 통신망

동작 제어 시스템과 GUI 시스템으로 분류되는 기존의 방식에서 RS232C로 시스템을 연결하는 방법은 추가적인 개발비용이 들지 않는다는 장점으로 인하여 주로 이용되고 있다. (그림 4) 자동화 시스템의 내부 통신망 구성은 이를 도식화한 것이다.

### 3.2 통신 Emulate 구성 및 분석

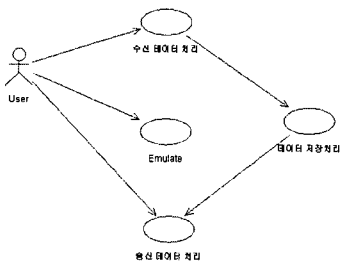
본 연구의 Emulate 분석 및 설계를 위해 UML을 이용하였다. 다음 〈그림 5〉는 Emulate의 구성

을 나타낸다.



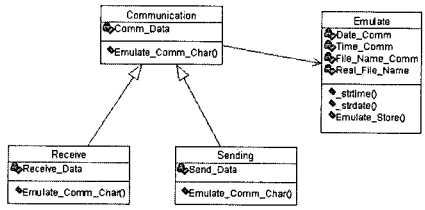

(그림 5) 컴포넌트 기반 통신 Emulate 구성

즉, RS232C로 수신되는 제어 데이터는 수신 컴포넌트가 수신 데이터베이스에 전송하고, 제어부는 이를 데이터베이스에서 가져와 처리한다. 또한, 제어 Event가 발생시 제어부에서는 송신 컴포넌트에 이를 보내고, 송신 컴포넌트는 정해진 프로토콜에 따라 RS232C를 통하여 제어 데이터를 보낸다.



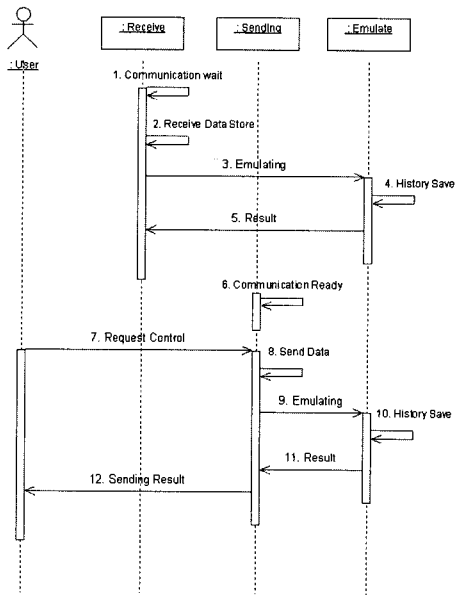

(그림 6) Usecase Diagram

(그림 6)은 이에 관한 Usecase Diagram을 나타낸다. 이는, 송,수신 데이터 처리시 모든 통신 데이터에 대한 히스토리를 저장하며, 사용자는 필요시에 이를 열람할 수 있도록 하였다.



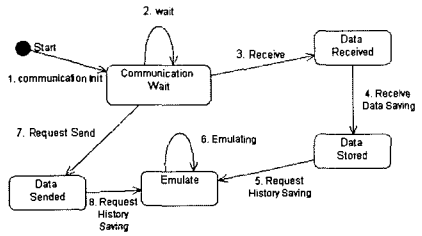

(그림 7) Class Diagram

이에 따른, 각 클래스는 (그림 7)과 같이 설계하였다. 또한, 송,수신 컴포넌트와 Emulate내의 프로세스들간의 제어 순서의 흐름은 다음 (그림 8)과 같이 나타낼 수 있다.




(그림 8) Sequence Diagram

즉, 수신 컴포넌트의 Receive와 송신 컴포넌트의 Sending에서 해당 데이터의 처리 후 Emulate에 통신 처리된 데이터를 저장하였다.




(그림 9) State Diagram

이에 따라, 통신상의 데이터 흐름을 사용자가 추후의 요구시에 확인할 수 있도록 하였다. 또한, 이와 연계된 상태변화는 (그림 9)과 같이 나타낼 수 있다.

### 3.3 통신 Emulate 구현

본 연구에서는 두 시스템간의 통신 Emulate를 위하여 C++ 언어로 구현하였으며 이를 위하여 Windows2000 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
        }
```

수신 컴포넌트는 다음과 같이 Receive에서 구현하였으며, 컴포넌트의 입력포트인 Inport에서 수신되는 데이터를 받아 상위 클래스의 비즈니스 로직인 Communication_ Receive에서 처리하도록 하였다. 즉, 수신이 이루어지는 수신부의 Emulate는 Communication_Receive(); 에 위치하여 수신 데이터를 다음과 같이 저장하게된다. 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.

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

Class Receive::Rev_prcess {
        public void Inport(Char *ReceiveData)
                {
Rev_prcess.Communication_Receive(); }
        public bool Outport();
}
```

또한, 송신 컴포넌트는 다음과 같이 구현하였다. 컴포넌트의 입력포트인 Inport에서 출력포트인 Outport로 제어를 넘기고, 이에 관한 결과를 Boolean형식으로 반환한다. 또한, 통신데이터의 저

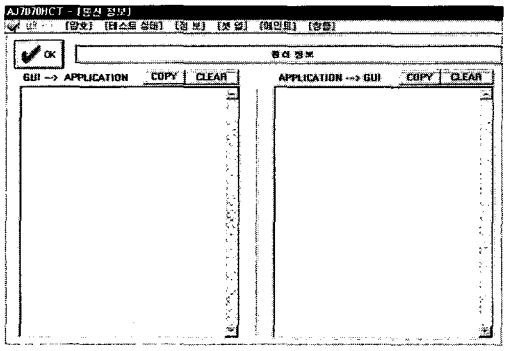
장을 위하여 Emulate_File_Comm_ Char을 이용한다. 즉, GUI 시스템에서 동작 제어 시스템으로 데이터를 송신하고 이를 확인한 후에 Emulate에 송신 데이터를 저장한다.

```
Class Rev_prcess {
        Bool Sending_Process(송신 데이터) {
                SendDataa(송신 데이터); }
}

Bool SendDataa(송신 데이터)
{
   송신 데이터의 분류;
   동작 제어 시스템으로 데이터 송신;
   송신 데이터 확인;
   Emulate_File_Comm_Char(송신 데이터);
}

Class Sending::Send_prcess {
        public void Inport(송신 데이터)
          { Outport(송신 데이터); }
        public bool Outport(송신 데이터)
          {Send_prcess.Sending_Process(송신
데이터); }
}
```

Emulate에서는 아직 처리되지 않은 수신 데이터를 날짜별 파일명으로 저장한다. 이때, 데이터의 저장방식은 누적을 사용하여 이전 데이터의 손실을 막는다.

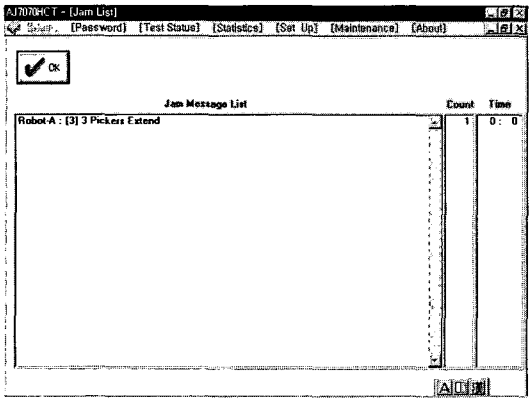
```
void        Emulate_File_Comm_Char(char
*Comm_Emulate_Char)
{
 char Date_Comm[10], Time_Comm[10],
 File_Name_Comm[50];
 FILE *FileHand;
 char Real_File_Name[5];
 int i;
 _strtime(Time_Comm);
 _strdate(Date_Comm);
 for(i=0; i<2; i++)
  Real_File_Name[i] = Date_Comm[i];
 for(i=3; i<5; i++)
  Real_File_Name[i-1] = 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);
}
```

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. 이에 따라, 다음 (그림 10)은 Emulate 화면을 나타낸다.



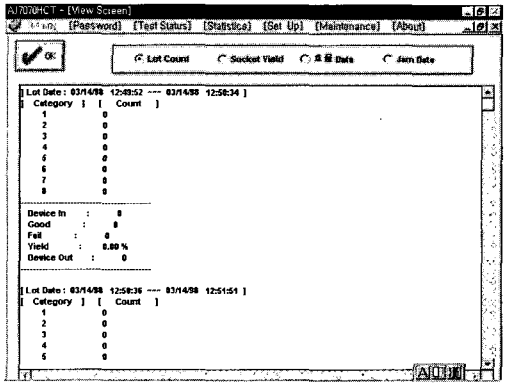

(그림 10) Emulate 화면

좌측에는 GUI에서 송신되는 송신부 화면이며, 우측은 GUI로 수신되는 수신부의 화면이다. 또한, 다음 〈그림 11〉은 GUI로 수신되는 자동화 시스템의 오류 데이터에 관한 오류발생화면을 나타낸다.




(그림 11) 오류 발생 화면

각 통신데이터에 따른 히스토리 데이터의 저장된 결과를 확인 및 검증하기 위하여 Emulate 확인 결과는 다음 (그림 12)과 같다. 이는, 통신으로 수신되는 각 상황별 데이터를 구분하여 저장되며, 선택되는 종류에 따라 통신되는 데이터를 확인할 수 있다.




(그림 12) Emulate 검증 화면

## V. 결 론

본 연구에서는 산업 자동화 시스템의 통신 Emul ate를 구현하였다. 재사용성과 효율성을 높이기 위하여 통신부분의 핵심인 송, 수신부는 컴포넌트화 하였으며, 간단한 통신방법인 RS232C을 이용하였다. 또한, 통신 데이터의 확인 및 검증을 위하여 송, 수신되는 데이터 히스토리를 자동으로 저장하여 이를 확인할 수 있도록 하였다. 따라서, 송, 수신부의 통신 데이터 처리부분은 이에 관한 다른 프로젝트에도 적용할 수 있으며, 기능 독립 단위로 구현함으로써 프로세스 처리 및 관리가 용이해졌다. 또한, Emulate는 통신상의 데이터에 대한 실시간적인 확인이 가능하였으며, 히스토리 데이터의 지원으로 통신검증이 가능하였다. 즉, 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 데

이터를 이용하여 분석하고자하는 시점에서의 자동화 시스템에 관한 동작상황을 알 수 있었다.

그러나, 본 시스템은 통신상 누락될 수 있는 데이터의 처리부분에 대한 충분한 고려가 이루어지지 않았고, 파일을 통한 통신 히스토리 데이터의 저장관리 또한 효율적이지 못하였다. 따라서, 향후 연구방향은 이에 관한 통신 및 저장 데이터의 효율적인 처리방안이 충분히 고려되어야 할 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] F. Brosard, D. Bryan, W. Kozaczynski, E. S. Liongorari, J. Q. Ning, A. Olafsson, and J. W. Wetterstrand, "Toward Software Plug-and-Play", in Proc. of the 1997 Symposium on Software Reusability, 1997.

[4] P. C. Clements, "From Subroutines to Subsystem : Component-Based Software Development", Component Based Software Engineering, IEEE CSPress, 1996.

[5] 신동익 외6인, "C2 스타일의 아키텍처 기술을 지원하는 ADL 지원도구의 개발", 한국정보처리학회 논문지 Vol. 8-D, No 6. 2001.

[6] 최유희, 권오천, 신규상, "C2 스타일을 이용한 EJB 컴포넌트 합성방법", 한국정보처리학회 논문지, Vol. 8-D, No 6. 2001.

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

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

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

[10] SEI in Carnegie Mellon University, "Component Based Software Development/COTS Intergration", http://www,sei.mcu.edu/str/descriptions/cbsd_body.html

[11] 김철진, 김수동, "컴포넌트 워크플로우 커스터마이제이션 기법", 한국정보과학회 논문지 : 소프트웨어 및 응용 제27권 제5호, 2000, 5.

[12] 윤희진, 최병주, "컴포넌트 기반 개발 개념을 활용한 테스트 프로세스 tailoring" 한국정보과학회 논문지 : 소프트웨어 및 응용 제27권 제12호, 2000, 12.

**정 화 영**

1994년 경희대학교 전자계산공학과(석사)

1998년~2000년 한남대학교 컴퓨터공학과(박사과정)

2000년~2001 경희대학교 전자계산공학과(박사수료)

1994년~1998년 아주시스템(주) 부설기술연구소 전임연구원

1998년-1999년 CNA Research(주) 전임연구원

2000년-현재 예원대학교 정보경영학부 전임강사

관심분야 : 소프트웨어공학, OOP/S, S/W 재사용, 컴포넌트기반 소프트웨어개발 방법론.