

생산정보 관리를 위한 클라이언트-서버 모델

송기석*, 임성락**
호서대학교 컴퓨터공과
e-mail:srrim@office.hoseo.ac.kr

A Model of Client-Server for the Product Information Management

Ki-Seok Song*, Seong-Rak Rim**
Dept of Computer Engineering, Hoseo University

요 약

오늘날 대부분의 중소제조업체들은 생산 공정의 불합리한 요소제거, 현장의 투명성 및 생산성 향상을 위한 체계적인 생산정보 관리를 요구하고 있다. 본 논문에서는 생산현장에서 발생하는 데이터를 수집, 분석하여 경영자 및 작업자의 의사결정을 지원하는 생산정보 관리를 위한 클라이언트-서버 모델을 제시한다.

1. 서론

오늘날 모든 기업들은 국내/외적인 경쟁심화와 기술혁신 및 신제품 개발, 정보화시대 도래에 따른 기업정보의 중요성 증대 등 급변하는 기업 환경 속에 있다. 생산정보화는 이와 같은 경쟁시대에서 기업의 경쟁력을 확보하는 방법 중 하나이다.

생산정보화는 정보기술(IT)을 활용하여 생산현장에서 발생하는 정보를 수집/분석하고, 생산공정을 제어 감시하여 경영자 및 작업자의 의사결정을 지원하는 생산현장의 정보화를 의미한다[1].

모든 산업은 생산현장에 산재되어 있는 생산자원을 실시간으로 생산정보로 변환하고 이 정보를 통합화하는 것이 가장 중요하다. 이를 구현하지 못하면, 생산에 대한 계획과 수행의 시간, 제어, 정보 및 동기화 차이가 발생한다. 그래서 생산정보화로 생산현장의 작업을 통제하고, 작업 및 자재를 추적, 분석, 연계와 품질을 관리 하는 것이 요구 된다. 생산정보화는 공급자 및 고객과 연계의 중요성이 필수적이므로 생산공정의 시각화, 공급사슬과 연계 및 신제품의 시장 출시에 대한 리드타임을 단축하는 제품 설계와 연계의 기능이 요구된다. 이러한 정보화는 정보소요

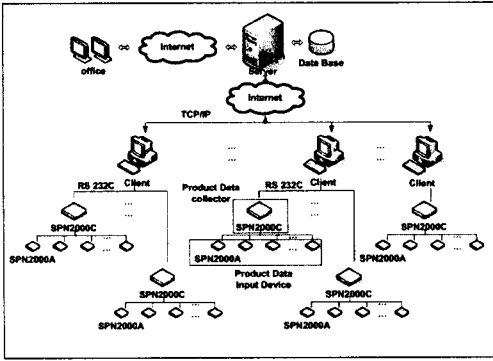
비용의 극소화와 복수 공장을 관리할 수 있는 관리성을 기반으로 한 구성이 필수적이다.

본 논문에서는 공장 작업자들이 손쉽게 생산데이터를 입력할 수 있도록 제작된 입력장치를 사용한다. 이 장치를 사용하여 실시간으로 데이터를 수집하고 정보처리에 적합한 데이터로 변환하여 서버로 전송하는 IOCR(I/O Completion Routine) 모델 기반의 클라이언트를 설계한다. 그리고 이러한 클라이언트를 복수로 처리하는 IOCP(I/O Completion Port) 모델 기반의 서버를 설계한다.

2. 시스템 구성

중소기업체의 생산정보화 구축 우수사례를 살펴보면 생산정보 관리 시스템은 공통적으로 생산데이터 입력장치, 뷰어, 클라이언트 그리고 서버가 구성요소로 되어 있다.

본 논문에서 제시하고자 하는 생산정보 관리 시스템의 구성은 (그림 1)과 같이 생산데이터 입력장치와 뷰어 기능을 포함한 클라이언트 그리고 서버로 구성되어 있다.



(그림 1) 시스템 구성도

2.1. 생산데이터 입력

생산데이터 입력장치는 작업자부터 직접 입력을 받는 모듈(SPN2000A)과 이 모듈들로부터 들어온 생산데이터를 모아서 클라이언트로 전송하는 모듈(SPN2000C)을 사용한다. 생산데이터 입력장치에 들어오는 입력은 실적카운트(양품/불량), 가동/비가동 상태, 불량수량/불량원인, 정체시간/정체요인, 준비시간/양산시간 등의 생산데이터가 있다.

2.2. 클라이언트

클라이언트의 기능은 첫째로 입력장치로 들어오는 다양한 생산데이터를 생산정보로 변환한다. 둘째로 공장책임자로부터 로그인, 리포트, 에디터, 뷰어 등의 요청을 입력받는다. 마지막으로 입력 받은 생산정보와 공장책임자의 요청을 서버로 전송하고 서버로부터 응답을 전송 받아 생산정보를 클라이언트 화면에 출력한다.

2.3. 서버

서버의 기능은 클라이언트로부터 요청을 받아 요청에 따른 응답을 해주는 것이다. 클라이언트의 요청과 이를 처리하는 서버의 응답은 다음과 같다.

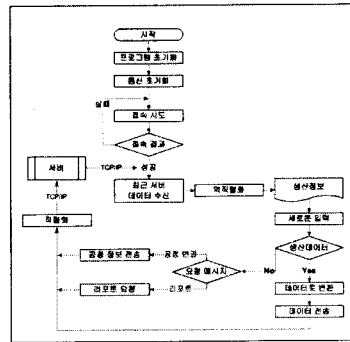
- 로그인 : 사용자를 검사 하여 검사 결과를 클라이언트로 전송한다.
- 리포트 : 특정 생산정보를 요청하여 DB를 검색하여 리포트 파일을 클라이언트로 전송한다.
- 에디터 : 클라이언트에서 공정 변경되면, DB에 저장하고 변경된 정보를 모든 클라이언트에게 전송한다.
- 생산데이터 : 클라이언트로부터 새로운 생산데이터가 들어오면, 이를 DB에 저장하고 변경된 데이터를

모든 클라이언트에게 전송한다.

3. 시스템 설계

3.1. 클라이언트

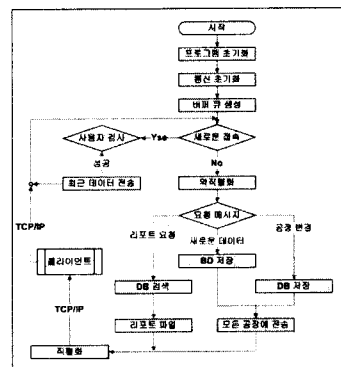
클라이언트의 흐름은 (그림 2)와 같다. 서버와 통신 초기화 후 로그인을 시도한다. 로그인에 성공하면 서버의 DB에서 최근 데이터를 전송 받아 역직렬화한다. 전송 받은 데이터를 클라이언트 화면에 출력하고 새로운 입력이 있을 때마다 입력의 종류에 따라 생산데이터와 요청 메시지로 분류하여 처리 한다. 처리한 데이터는 직렬화되어 서버로 전송된다.



(그림 2) 클라이언트의 흐름도

3.2. 서버

서버의 흐름은 (그림 3)와 같다. 통신의 초기화 후 버퍼풀을 생성한다. 클라이언트로부터 접속이 들어오면 새로운 클라이언트의 경우 사용자를 검사하여 승인결과나 비승인결과를 클라이언트로 전송한다. 승인된 클라이언트에서 들어오는 메시지를 역직렬화하여 요청한 메시지에 따라 DB를 검색하거나 저장한다. 처리한 데이터는 직렬화를 하여 클라이언트로 전송한다.



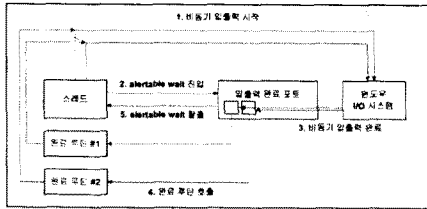
(그림 3) 서버 흐름도

3.3. 통신 소켓

본 논문에서는 비동기 입출력 방식인 IOCR 모델과 IOCP 모델을 기반의 클라이언트와 서버를 설계한다.

3.3.1. 클라이언트 : IOCR 모델

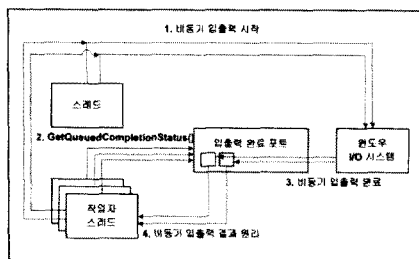
IOCR 모델은 IOCP 모델보다 성능을 떨어지지 않던 NT계열의 환경에서만 사용할 수 있는 IOCP보다 더 다양한 환경에서 사용할 수 있다. 이런 장점 때문에 클라이언트 모델은 IOCR을 많이 선호한다. IOCR 모델은 완료 루틴을 통해 비동기 입출력 결과를 (그림 4)와 같이 처리한다. 여기서 완료 루틴이란 애플리케이션이 정의한 일종의 콜백 함수로, 운영체제가 적절한 시점에서 자동으로 호출하도록 되어 있다.



(그림 4) IOCR 모델의 동작 원리

3.3.2. 서버 : IOCP 모델

멀티 스레드를 관리하기 위하여 만들어진 IOCP는 모든 소켓 입출력 모델 중 가장 뛰어난 성능을 제공한다. IOCP 모델의 장점은 CPU 개수에 따라 적정 수의 작업 스레드를 유지한다. I/O 완료 시에 IOCP가 LIFO 방식으로 가장 최근에 대기 중인 작업 스레드에게 스케줄링 하여 문맥교환을 최소화 시킨다. 또한, 블로킹을 줄이기 위해 OS의 소켓 버퍼를 거치지 않고 사용자 주소 공간의 버퍼를 바로 이용해 I/O를 한다.

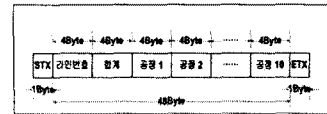


(그림 5) IOCP 모델의 동작 원리

3.4. 통신 메시지

3.4.1. 생산데이터

생산데이터 입력장치는 시리얼통신(RS232C)이나 USB 통하여 생산데이터를 클라이언트에 입력한다. 시리얼 통신을 하기 위해서는 운영체제에 시리얼 포트를 사용하겠다는 선언을 한다. 데이터가 외부로부터 사용할 포트에 들어오면 운영체제는 임시로 자신의 버퍼에 저장한다. 운영체제는 생산데이터의 입력을 클라이언트에게 알려주어서 처리한다.



(그림 6) 메시지 프레임

S T X : 메시지 프레임의 시작. 'S' 입력.

E T X : 메시지 프레임의 끝. 'E' 입력.

라인번호 : 입력이 들어오는 공정기계 번호.

합 계 : 모든 공정의 데이터의 합. (오류 체크)

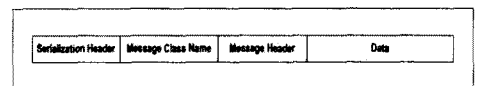
공 정 : 실제 공정의 데이터.

입력된 데이터의 프레임 크기는 50바이트이다. 가장 앞쪽과 뒤쪽의 1바이트는 프레임의 시작과 끝을 나타낸다. 시작과 끝을 제외한 나머지 부분은 실제 데이터부분으로 4바이트씩 48바이트로 구성된다.

3.4.2. 객체 직렬화(Object Serialization)

클라이언트와 서버간의 통신을 위하여 객체 직렬화를 사용한다. 객체 직렬화란 객체의 내용을 바이트 단위로 변환하여 파일 또는 네트워크를 통해서 송수신(스트림)이 가능하게 만들어 주는 것을 말한다. 여기서 객체란 멤버 변수의 메모리만으로 구성된 것을 말한다.

객체 직렬화의 장점은 객체 자체의 내용을 입출력 형식에 구애받지 않고 객체 자체를 네트워크를 통하여 손쉽게 교환할 수 있도록 하며, 또한 코딩도 짧아지고 작업이 수월해 진다는 것이다.

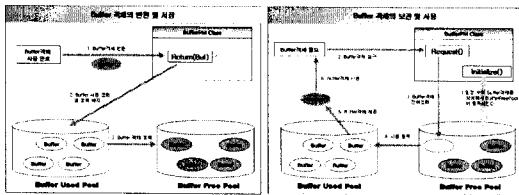


(그림 7) TCP/IP의 메시지 프레임

Serialization Header : 직렬화 식별자
 Message Class Name : 클래스 이름
 Message Header : 데이터 클래스 구성 정보
 Data : 실제 데이터

3.5. 버퍼풀

프로그램에서 버퍼 객체를 생성 및 해제하면 이 시간적인 오버헤드는 전체 시스템의 성능을 느리게 만들 수 있다. 그래서 버퍼풀이란 기법을 적용하여 프로그램의 실행과 동시에 미리 버퍼 객체를 일정 수만큼 생성하여 놓고, 이 버퍼풀을 제어/관리하는 버퍼 매니저 클래스 객체에 의해 필요에 따라 버퍼 객체를 할당/반납하도록 하는 기법을 사용한다. 이 기법의 사용으로 시간적 오버헤드를 조금 줄일 수 있으며, 이 기법을 확대하여 기타 자원들에 적용하면 필수 자원관리가 용이해지는 이점을 얻을 수 있다.



(그림 9) 반환 및 저장 (그림 8) 보관 및 사용

4. 구현 및 테스트

4.1. 구현

제시한 모델의 타당성을 검토하기 위하여 IOCR 기반 클라이언트와 IOCP 기반 서버를 각각 설계하고 Windows 환경에서 실험하였다.

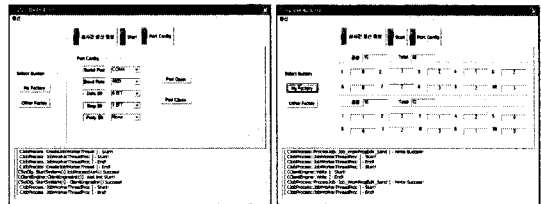
생산데이터 입력장치의 입력방법은 작업자가 손쉽게 입력을 할 수 있도록 Key-Pad 방법을 사용하였다. 생산데이터 입력장치로부터 들어오는 데이터는 연속적인 단방향 전송이므로 유실된 데이터가 생긴다. 논문에서는 이런 데이터의 사용을 메시지 프레임의 STX와 ETX를 체크하여 막았다. 그리고 버퍼 오버플로우로 인한 유실된 데이터도 코드의 간결화를 통한 프로그램 속도 향상으로 줄였다.

4.2. 실험

본 논문에서는 다음과 같은 테스트 순서에 따라 생산데이터의 입력과 클라이언트와 서버간의 통신을 테스트 하였다.

- ① 서버와 클라이언트를 구동한다. ② 클라이언트의 초기 설정을 한다.
- ③ 등록되어 있는 ID와 암호를 입력한다.
- ④ 생산데이터를 입력한다.
- ⑤ 요청을 입력한다.

클라이언트의 초기 설정은 (그림 10)과 같이 서버의 IP와 포트, 통신 속도 등을 설정한다. 설정 후 로그인을 시도하여 서버와 접속을 한다. 생산데이터 입력장치를 이용하여 생산데이터를 입력하고 이를 처리하는 클라이언트와 서버간의 접속 상태와 생산데이터 입력장치의 입력 상태 등은 (그림 11)에서 보여주고 있다.



(그림 10) 초기화 설정 (그림 11) 통신 상태

5. 결론

본 논문에서는 효율적인 생산정보 관리를 위하여 IOCR 기반의 클라이언트와 IOCP 기반으로 서버를 설계하였다. 테스트를 통하여 실시간으로 생산데이터 입력장치와 클라이언트 그리고 서버간의 통신이 원활하게 이루어 졌다. 앞으로 이를 사용하여 다양한 생산정보를 확인할 수 있도록 인터페이스를 설계 구현하고 이를 사용할 사용자들의 편의를 제공하는 연구가 따라야 할 것으로 본다.

참고 문헌

[1] 중소기업청, “생산정보화 시스템”, <http://www.smba.go.kr/>
 [2] A simple IOCP Server/Client Class, <http://www.codeproject.com/>
 [3] Windows Sockets 2.0 : Write Scalable Winsock Apps Using Completion Ports, <http://msdn.microsoft.com/library/>
 [4] 김선우, “윈도우 네트워크 프로그래밍”, 한빛미디어, 2004