

공학석사 학위논문

감시제어 시스템의 신뢰성 향상을 위한 서버 및 네트워크 이중화

A Redundancy of Servers and Networks for Reliability
Improvement of a Monitoring and Control System

지도교수 류 길 수

2003 년 2 월

한국해양대학교 대학원

컴퓨터공학과 임 대 근

목 차

Abstract	
제 1 장 서 론	1
제 2 장 감시제어 시스템	4
2.1 감시제어 시스템의 개요	4
2.2 기존 감시제어 시스템의 구조	5
2.3 기존 감시제어 시스템의 구조의 개선점	7
제 3 장 제안하는 감시제어 시스템의 이중화 구조	9
3.1 감시제어 시스템의 하드웨어 이중화	9
3.2 오류 판단과 처리	10
3.3 감시제어 시스템의 모듈화	14
제 4 장 통신 모듈의 구현	16
4.1 메모리 관리	16
4.2 데이터 관리 모듈	19
4.3 장비 접속 모듈	21
4.3.1 접속 관리 쓰레드	23
4.3.2 데이터 수집 쓰레드	28
4.4 서버 상태 관리 모듈	29
4.4.1 활성서버의 서버 상태 관리 모듈	32
4.4.2 대기서버의 서버 상태 관리 모듈	34
4.5 데이터 저장 모듈	36
제 5 장 실험 및 고찰	38
제 6 장 결 론	44
참고문헌	45

A Redundancy of Servers and Networks for Reliability Improvement of a Monitoring and Control System

Dae-Geun Lim

Department of Computer Engineering, Graduate School,
Korea Maritime University, Pusan, Korea

Abstract

Recently, together with development of communications technology, monitoring and control systems have been used in many fields. When monitoring and control systems are developed, one of important things is considered about a redundancy so that the system can be operated normally whenever hardware faults are occurred partly.

In this paper, we introduce a monitoring and control system with a redundancy function for servers and networks. The system, therefore, has an active server and standby server. Each server also has 3 communication ports, and 2 ports are connected to all field units and 1 port is connected to another server. In fields, all field units have to be constructed 2 communication ports connected each server through communication lines. Also, each server's communication program is implemented by analyzing and handling fault elements, and is modularized for linking easily with a monitoring and control module. An experiment with 2 servers and 2 field units is constructed to demonstrate its effectiveness.

제 1 장 서 론

최근에 걸쳐 이루어진 컴퓨터의 데이터 처리능력과 통신기술의 발달로 각종 감시제어 시스템들이 더욱 중요하게 활용되고 있다[1,2]. 이러한 감시제어 시스템은 원거리의 광범위한 지역에 분포된 설비를 유기적으로 연결하여 원격감시, 제어, 계측, 운용하는 많은 사업 분야에 응용되고 있다[3,4,5,6]. 이 시스템은 종래 인간이 운영하던 방식을 컴퓨터기술을 이용하여 실시간 개념으로 여러 종류의 원격지 시설장치를 중앙집중식으로 감시 또는 제어하여 무인화를 가능하게 하고, 또한 고장시 신속한 원인규명과 고장복구를 지원할 수 있도록 한다[7,8,9]. 초기에는 감시기능과 제어기능을 별도로 구성하여 감시기능은 컴퓨터 측에서 작동되고, 제어기능은 현장에서 작동되도록 구성하였다. 그러나 최근에는 컴퓨터 통신기술의 발달에 힘입어 컴퓨터에서 감시와 제어를 동시에 할 수 있는 감시제어 시스템으로 발전되고 있다[10,11,12].

이러한 감시제어 시스템의 개발시에 가장 고려하여야 할 사항은 첫째 개발의 편의성, 둘째 하드웨어 오류에 대비한 하드웨어의 이중화(Redundancy) 기능, 셋째 단순 감시기능만이 아닌 고장진단기능을 포함한 시스템의 오류에 대한 지능적인 처리기능들이 있다[13]. 이중에서 시스템의 신뢰성을 향상시키기 위해서는 두 번째와 세 번째 기능이 필요하다. 또한 최근에는 많은 SCADA 시스템들이 개발되어 감시제어 시스템의 개발편의성을 향상시키고 있다. 일반적으로 감시제어 시스템을 개발하기 위해서는 컴퓨터 공학에 관한 지식과 산업현장에 대한 지식이 모두 필요하게 된다. 이러한 개발의 복잡한 과정을 간단하게 하여 산업현장의 전문가들이 쉽게 개발에 동참할

수 있도록 SCADA 시스템들이 이용되고 있는 것이다[13].

이러한 SCADA 시스템을 개발하여 제공하는 업체가 많아지고 있으며, 대표적인 회사들에는 KDT Systems, 여의시스템, 대명스카다, ABB, Citect 등이 있다. 그러나 이들 시스템에서는 하드웨어의 이중화나 지능적인 시스템에 관한 고려들이 제대로 이루어지고 있지 않다. 또한 산업용에서 실용화되어 있는 많은 감시제어 시스템들은 이러한 SCADA 시스템을 이용하는 것보다는 산업설비 관리업체에서 독자적으로 하드웨어를 구성하여 시스템을 폐쇄적으로 개발하고 있는 실정이다. 이것은 대부분의 SCADA 시스템이 너무 개발자의 편의성만을 고려한 나머지 산업분야의 특정기능들에 대해서는 구현되어 있지 않기 때문이다[14,15,16,17].

예를 들어 선박용 감시제어 시스템의 경우, 삼성, 현대, NORCON 등에서 독자적인 시스템을 개발하여 제공하고 있지만, 기술 공개를 우려하여 완성된 시스템 형태로만 제공하고 있다. 따라서 선박엔진마다 별개의 서로 다른 감시제어 시스템을 운용해야 하는 어려움이 있으며, 선주 입장에서는 고가의 비용을 지불해야만 한다. 그렇다고 시스템 구축비용을 절감하기위해서 일반 SCADA 시스템을 선박엔진분야에 적용하게 되면 특수한 많은 기능들을 제공할 수 없어서 실용적인 시스템으로 구축되기가 어렵다[18,19].

이에 본 논문에서는 선박용과 같이 실용적인 감시제어 시스템 구축을 지원하기 위한 SCADA 시스템의 통신 모듈 개발을 목표로 하고 이 시스템에 신뢰성을 제공하기위해 하드웨어의 이중화, 즉 서버 및 통신라인을 이중화하는 것에 관하여 연구한다. 먼저 감시제어 시스템을 활성서버(Active Server)와 대기서버(Standby Server)로 구축하고 현장의 필드유닛(Field Unit)들이 모두 2개의 통신 포트를

가하도록 하여, 이것들을 각각 활성서버 및 대기서버에 연결시키는 하드웨어 구조로 구성한다. 다음 각 서버의 통신 모듈을 구성하는 내부 모듈들을 구현하고 각 구성요소의 고장개소를 분석하고 처리하는 알고리즘을 작성한다. 또한 향후 감시제어 기능과의 유연한 연계를 고려하여 프로그램을 모듈별로 분류 개발하여 모듈의 재활용성과 구성의 편의성을 높일 수 있도록 한다.

본 논문의 구성은 다음과 같다. 2장에서는 기존의 감시제어 시스템에 대하여 소개하고, 3장에서는 감시제어 시스템을 향상시키는 방법에 대하여 설명한다. 4장에서는 시스템을 구현한 모듈들에 대하여 설명한다. 5장에서는 결론과 향후 과제에 대하여 논한다.

제 2 장 감시제어 시스템

본 장에서는 감시제어 시스템의 개요와 기존 시스템들의 구조에 대하여 설명하고 기존에 개발되어 왔던 감시제어 시스템에서 발견되는 문제점을 제시하도록 한다.

2.1 감시제어 시스템의 개요

ANSI(美 표준연구소)/IEEE(전기전자기술자협회)의 권고안에 명시된 감시제어 시스템의 주요 기능으로는 원격장치의 경보상태에 따라 미리 규정된 동작을 하는 감시 기능인 경보기능, 원격 외부장치를 선택적으로 수동, 자동 또는 수·자동 복합으로 동작시키며, 각 기기의 설정값을 조절하기 위한 감시제어기능, 원격장치의 상태정보를 수신, 표시·기록하는 감시시스템의 지시(표시)기능, 디지털 펄스 정보를 수신, 합산하여 표시·기록에 사용할 수 있도록 하는 누산기능 등이 있다[20].

SCADA 시스템은 이러한 감시제어 시스템을 현장의 엔지니어 즉 비프로그래머도 쉽게 구축할 수 있도록 하기위한 산업용 개발툴이다. 현재 개발되어진 SCADA시스템에는 Citect의 CitectSCADA 와 KDT Systems의 CIMON 등이 있으며 발전 및 전력계통, 정수 및 하수처리, 플랜트 자동화, 공정제어, 열병합 발전계통, 가스, 석유, 빌딩관리 등에 응용되어 사용되고 있다.

2.2 기존 감시제어 시스템의 구조

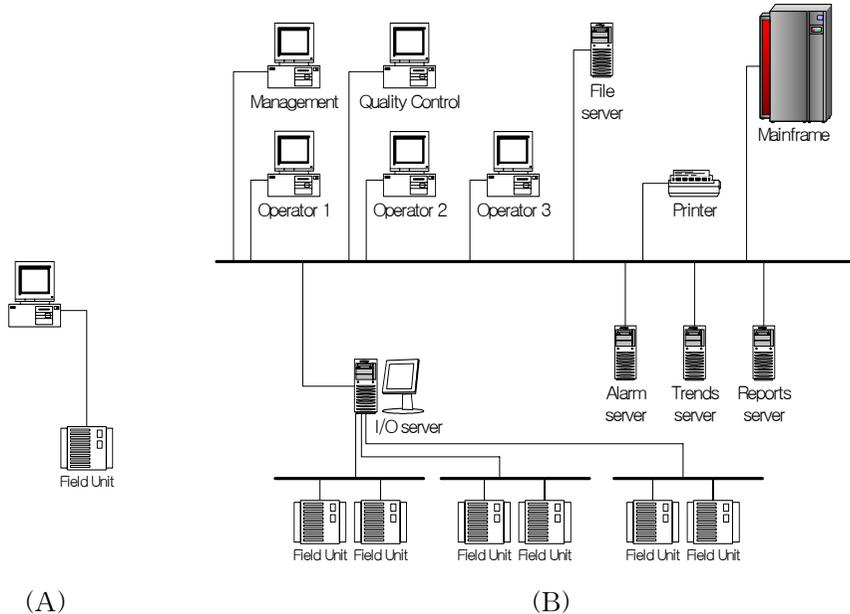


그림 2.1 Citech 감시제어 시스템의 구조

그림 2.1은 Citech이라는 회사의 감시제어 시스템의 구조를 보여 주고 있으며 다른 회사에서 제공하는 감시제어 시스템에서도 비슷한 구조가 나타난다. 그림 2.1의 (A)는 하나의 필드유닛과 하나의 서버가 있을 경우의 시스템 구조이고, (B)는 여러 개의 필드유닛과 여러 개의 서버가 있을 경우의 구조이다. 하나의 필드유닛만 있는 경우에는 소규모의 감시와 제어가 필요할 경우 사용되는 시스템으로 한 대의 서버로 모든 감시와 제어를 담당하게 된다. 그러나 많은 필드유닛이 필요할 경우 그림 2.1의 (B)처럼 데이터 처리를 위한 여러 개의 서버를 두고 각각 다른 기능을 담당하여 처리하는 구조로 확장을 한다.

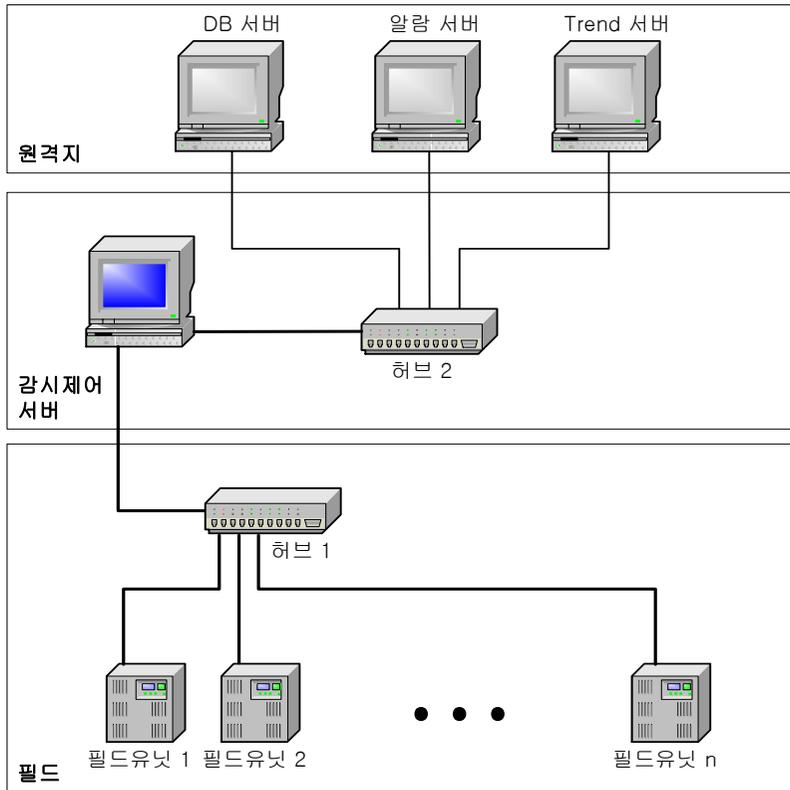


그림 2.2 기존 감시제어 시스템의 구조

그림 2.1의 시스템 구조를 필드와 감시제어 서버와 원격지로 나누어 단순화하여 표현하면 그림 2.2와 같다. 이러한 구조는 필드유닛이 하나의 통신라인을 통해 데이터를 보내면 하나의 서버에서 데이터를 받고 이 데이터를 처리하여 원격지의 데이터를 저장하거나, 알람을 설정하는 기능을 가진 다른 서버에 데이터를 보낸다. 또한 감시제어 서버는 필드유닛을 제어하는 신호를 보내는 기능을 모두 겸하게 된다. 소규모의 감시제어 시스템에서는 전송되는 데이터의 양이 적기 때문에 처리가 가능하지만 큰 규모의 감시제어 시스템의 경우에는 처리하여야 하는 데이터의 양이 많으므로 서버를 여러 개

로 나누어 서버의 부하를 줄이는 방법을 사용하고 있다. 그림 2.2에서 나타나 있는 것처럼 이 시스템의 필드유닛들은 I/O 서버에 해당하는 감시제어 서버를 통해서 데이터를 주고 받으며 다른 데이터를 처리하는 서버에서도 감시제어 서버를 통해서 데이터를 주고 받는다.

감시제어 시스템의 규모를 나누는 일반적인 방법은 표 2.1에서 분류되어 있듯이 필드유닛의 수와 필드유닛이 보내는 데이터의 개수로 구분한다[21].

표 2.1 감시제어 시스템의 규모

규모	총 감시 포인트 수
소규모	500개 이하
중규모	2,000개 이하
대규모	2,000개 이상

2.3 기존 감시제어 시스템의 구조의 개선점

감시제어 시스템에서 중요시되는 부분은 필드유닛, 감시제어 서버와의 통신, 감시제어 서버이다. 감시제어 서버에서 필드유닛으로 통신이 중요한 이유는 이전의 시스템에서는 필드유닛에서 하던 장비들의 제어를 원격지인 감시제어 서버에서 하기 때문이다. 그리고 감시제어 서버를 통하여 데이터가 전송되므로 감시제어 서버에서 오류가 날 경우 필드유닛과 원격지의 다른 서버들과는 전혀 통신이 되지 않기 때문에 필드유닛이 정상적인 동작을 할 수 없게 된다. 따라서 기존의 감시제어 시스템의 구조에서 문제점은 감시제어 서버

나 통신라인의 오류가 발생하였을 경우 어느 부분에서 오류가 발생했는지 판별하고 대처하기가 힘들다. 예를 들어 그림 2.2에서 필드유닛 1과 허브 1과 연결되는 통신라인에서 오류가 발생하였을 경우 필드유닛 1과의 통신은 불가능해지며 필드유닛과 허브 1을 연결하는 통신라인을 복구하는 방법 외에는 다른 방법이 없게 된다. 그리고 통신라인을 복구하는 동안에는 필드유닛 1의 감시제어를 할 수가 없게 된다.

감시제어 서버에서 오류가 발생하였을 경우에는 더욱 심각해진다. 모든 필드유닛의 감시제어가 불가능하게 될 뿐만 아니라 원격지의 다른 서버들의 작업도 중지 된다. 통신라인 오류를 해결하기 위해 필드유닛과 연결되는 통신라인을 이중화한 시스템도 있으나 여전히 I/O서버에 대한 신뢰성은 해결하지 못한다[16,21].

따라서 그림 2.2와 같은 구조는 감시제어에 대한 신뢰성을 보장하지 못하며 제 3 장에서 이와 같은 문제점들을 해결하기 위한 구조를 제시한다.

제 3 장 제안하는 감시제어 시스템의 이중화 구조

3.1 감시제어 시스템의 하드웨어 이중화

앞장에서 제시한 오류에 대한 문제는 그림 3.1과 같이 활성 서버와 대기서버를 두어 서버를 이중화하고 각 필드유닛과 허브, 허브와 서버를 연결하는 통신라인을 이중화하여 해결한다.

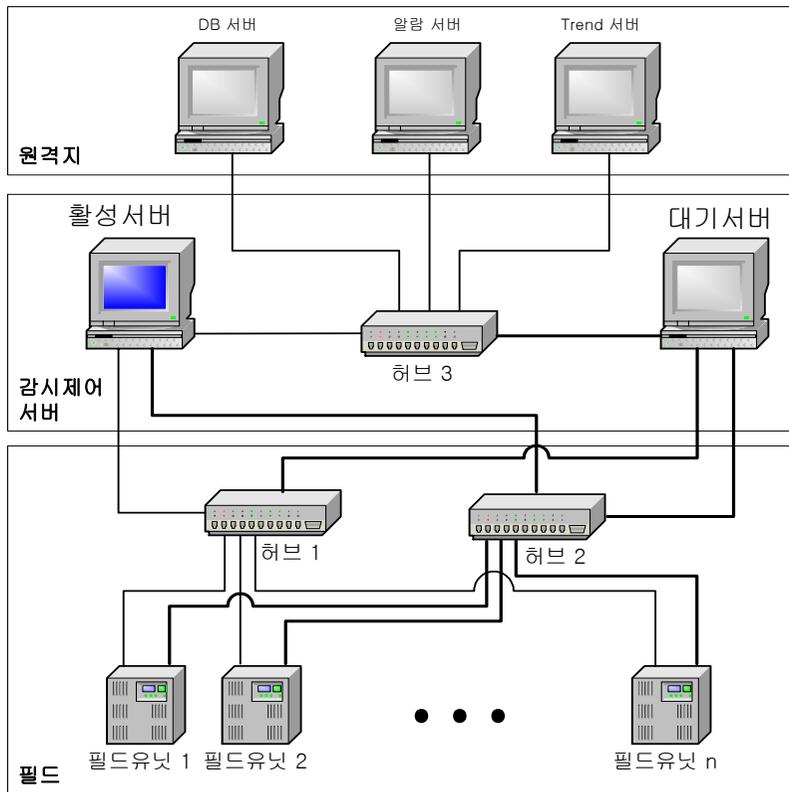


그림 3.1 이중화된 감시제어 시스템의 구조

서버의 오류 상황을 확인하기 위하여 활성서버와 대기서버 사이에는 상태확인 메시지를 주고받으며 오류가 발생하면 대기서버가 활성서버로 서버전환을 하여 감시제어가 계속 될 수 있도록 한다. 통신라인의 오류는 두 통신라인 중에서 오류가 없는 통신라인으로 변경하여 감시제어가 계속 되도록 한다.

3.2 오류 판단과 처리

제안한 감시제어 시스템의 서버와 통신라인 중에서 오류가 일어날 수 있는 위치는 그림 3.2에 표시된 것과 같이 활성서버, 대기서버, 허브, 필드유닛, 통신라인으로 분류 할 수 있다.

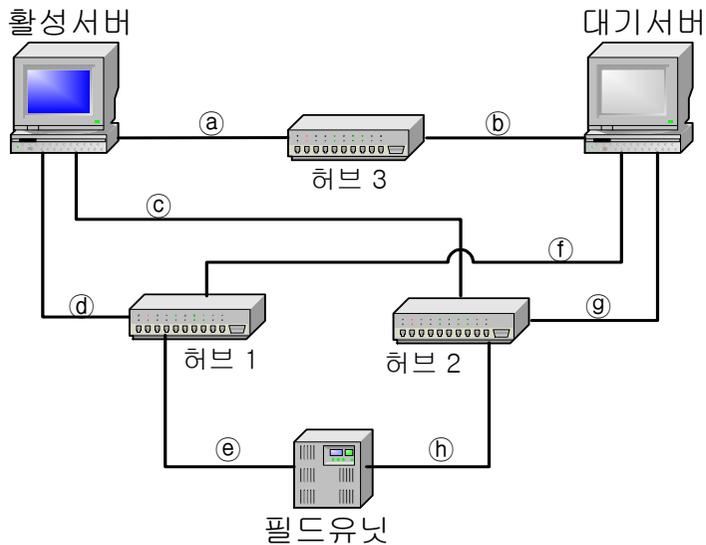


그림 3.2 감시제어 시스템의 오류위치

활성서버의 오류일 경우 대기서버에서 활성서버의 상태확인이 되지 않을 것이다. 활성서버의 상태확인 방법은 활성서버와 연결되는 세 가지 종류의 방법으로 확인할 수 있다. 통신라인의 오류에 대비하여 세 가지 방법으로 모두 확인할 수 있으며 활성서버의 상태확인을 할 수 있는 세 가지 연결 방법은 다음과 같다.

- (1) 대기서버 → ⑥ → 허브3 → ① → 활성서버
- (2) 대기서버 → ② → 허브1 → ④ → 활성서버
- (3) 대기서버 → ③ → 허브2 → ⑤ → 활성서버

위의 세 가지 방법으로 상태확인을 하여 응답이 없는 경우는 활성 서버가 다운 등의 문제로 작동할 수 없는 경우이므로 대기서버가 자동으로 활성서버로 전환되고 사용자에게 활성서버가 응답이 없어서 서버전환이 되었음을 알린다. 반대로 대기서버의 오류일 경우에는 활성서버의 상태확인 방법의 역순으로 확인하며, 응답이 없다면 대기서버가 응답이 없다는 오류 메시지를 보여준다.

통신라인의 오류는 응답시간으로 판단을 한다. 활성서버와 허브 사이의 통신라인 오류는 모든 필드유닛의 접속을 끊고 허브를 전환하여 감시제어를 계속한다.

표 3.1 통신 증상에 따른 오류위치와 처리(활성서버와 필드유닛)

증 상	오류의 위치	처 리
활성서버와 필드유닛 사이의 통신이 되지 않음	㉔ or ㉕	1. 통신라인을 변경 2. 통신라인 알람 설정
	허브 1 or 허브 2	1. 통신라인을 변경 2. 통신라인과 허브 알람 설정
	㉔ or ㉕	1. 통신라인 변경 2. 허브와 통신라인 알람 설정 3. 대기서버의 라인오류 정보가 수신되면 허브 알람 제거

표 3.2 통신 증상에 따른 오류위치와 처리(대기서버와 필드유닛)

증 상	오류의 위치	처 리
대기서버와 필드유닛 사이의 통신이 되지 않음	㉔ or ㉕	1. 통신라인 알람 설정 2. 활성서버로 라인오류 정보 전송
	허브 1 or 허브 2	1. 통신라인과 허브 알람 설정 2. 활성서버로 라인오류 정보 전송
	㉖ or ㉗	1. 허브와 통신라인 알람 설정 2. 활성서버의 라인오류 정보가 수신되면 허브 알람 제거

표 3.3 통신 증상에 따른 오류위치와 처리(활성서버와 대기서버)

증 상	오류의 위치	처 리
서버간의 통신이 되지 않음	Ⓐ, Ⓑ or Ⓒ, Ⓔ or Ⓓ, Ⓕ	1. 나머지 2 가지 방법으로 서버 상태 확인 2. 라인 오류 설정
	Ⓐ, Ⓒ, Ⓓ	1. 활성서버가 통신을 할 수 없는 상황 2. 대기서버가 활성서버로 전환 3. 활성서버 오류 알람 설정
	Ⓑ, Ⓕ, Ⓔ	1. 대기서버가 통신을 할 수 없는 상황 2. 대기서버 오류 알람 설정

활성서버와 필드유닛을 연결하는 통신라인의 오류를 점검하고 처리하는 방법은 표 3.1과 같으며, 대기서버와 필드유닛을 연결하는 통신라인의 오류를 점검하고 처리하는 방법은 표 3.2와 같다. 표 3.3은 활성서버와 대기서버를 연결하는 통신라인의 오류를 점검하고 처리하는 방법이다. 이러한 방법들에 대한 구현은 4장에서 설명한다.

3.3 감시제어 시스템의 모듈화

그림 3.3은 감시제어 시스템의 구조를 모듈별로 나누어 표현하고 있다. 이렇게 모듈화가 필요한 이유는 감시제어 시스템에서 소프트웨어 확장성과 개발의 편의성을 위해서 필요한 부분이다. 기존의 시스템에서는 확장성과 편의성을 고려하지 못한 구조를 가진다.

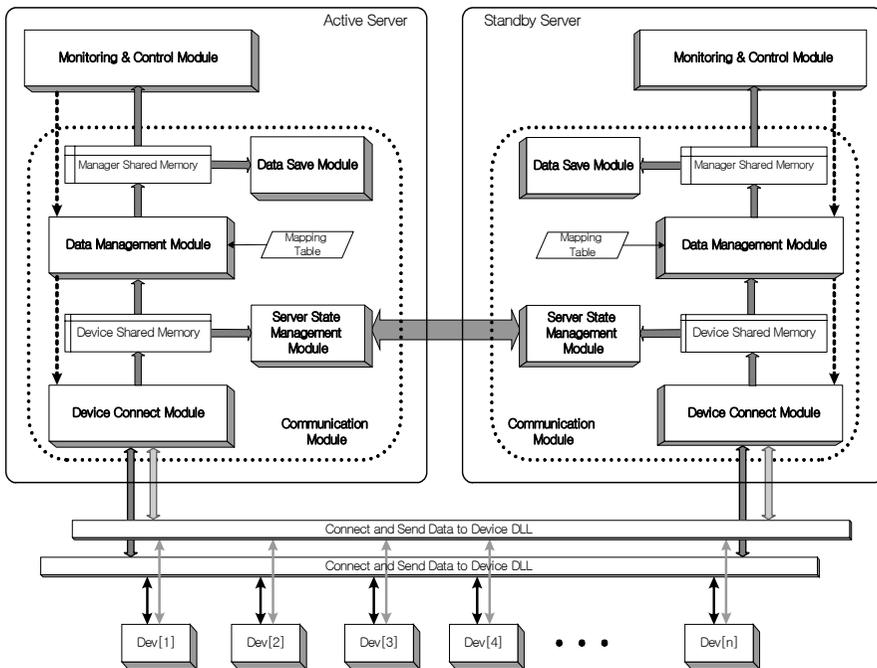


그림 3.3 감시제어 시스템의 소프트웨어적인 구성

그림 3.3에서 실선으로 나누어진 부분이 감시제어 서버 부분이며 Dev[1]~Dev[n]은 필드유닛 부분이다. 서버는 크게 감시제어 모듈(Monitoring & Control Module)과 통신 모듈(Communication Module)로 나누어지며 통신 모듈은 다시 세분화된 모듈들로 나누어

진다. 각 모듈들은 공유 메모리를 통하여 감시데이터들을 주고받는다. 장비 접속 모듈(Device Connect Module)이 펠드유닛으로부터 받은 데이터는 장비 공유 메모리(Device Shared Memory)에 기록하고 데이터 관리 모듈(Data Management Module)에서는 변환 테이블(Mapping Table)을 참조하여 관리 공유 메모리(Manager Shared Memory)에 기록한다. 서버 상태 관리 모듈(Server State Management Module)은 서버사이의 통신을 담당하고 데이터 저장 모듈(Data Save Module)은 데이터들을 저장장치에 기록한다.

제 4 장 통신 모듈의 구현

본 장에서는 감시제어 시스템의 구성 모듈들 중에서 통신 모듈의 구현에 대하여 설명한다. 감시제어 시스템의 통신 모듈은 데이터 관리 모듈(Data Management Module), 장비 접속 모듈(Device Connect Module), 서버 상태 관리 모듈(Server State Management Module), 데이터 저장 모듈(Data Save Module)로 구성되며 이 모듈들은 감시제어 모듈(Monitoring & Control Module)에서 제어된다.

4.1 메모리 관리

모듈들은 장비들의 정보와 데이터들을 공유 메모리를 이용하여 주고 받는다. 장비접속 모듈은 필드유닛과의 통신과 통신라인의 오류 검출에 부하를 줄이기 위해 필드유닛으로부터 받은 데이터를 그대로 장비 공유 메모리(Device Shared Memory)에 저장을 하고 데이터 관리 모듈에서 데이터 처리를 한다. 장비 공유 메모리의 데이터를 관리 공유 메모리(Manager Shared Memory)로 옮기기 위해서는 맵핑 테이블이 필요하게 된다.

1) 관리 공유 메모리의 구조

앞부분은 장비들의 정보가 저장되고 그 뒤부터 데이터의 종류에 따라 분류되어 저장된다.

Device Line Status Device Information Value Information	Value[1]	Value[2]	Value[3]	●●●	Value[n]
---	----------	----------	----------	-----	----------

Device Line Status = 필드유닛의 수 * 2 (Byte)

Device Information = 필드유닛의 수 * 1 (Byte)

Value Information = 분류된 값의 수 * 1 (Byte)

Value[i] = 필드유닛이 보내는 데이터의 수 * 데이터 크기 (Byte)

데이터 크기는 아날로그 값일 경우는 3 바이트이고 디지털 값일 경우는 1 바이트이다. Device Line Status는 필드유닛과 통신라인의 상태를 저장하는 부분이고 Device Information은 필드유닛의 오류를 저장하는 부분이다. Value Information은 데이터의 값이 디지털인지 아날로그인지를 저장하는 부분이다.

2) 장비 공유 메모리의 구조

앞부분은 장비들의 정보가 저장되고 그 뒤부터 장비별로 값이 저장된다.

Device Line Status Device Information Data Information	Device[1]	Device[2]	Device[3]	● ● ●	Device[n]
--	-----------	-----------	-----------	-------	-----------

Device Line Status = 필드유닛의 수 * 2 (Byte)

Device Information = 필드유닛의 수 * 1 (Byte)

Data Information = 필드유닛의 데이터의 수 * 필드유닛의 수 (Byte)

Device[i] = 아날로그 데이터 수 * 3 + 디지털 데이터 수 * 1 (Byte)

Device Line Status는 필드유닛과 통신라인의 상태를 저장하는 부

분이고 Device Information은 필드유닛의 오류를 저장하는 부분이다. Data Information은 데이터의 값이 디지털인지 아날로그인지를 저장하는 부분이며 이 값은 필드유닛의 수가 늘어나면 그 크기도 늘어나게 된다.

3) 변환 테이블의 구조

변환 테이블은 데이터의 종류에 따른 장비 공유 메모리의 위치를 저장하고 있다. 데이터 관리 모듈에서 장비 공유 메모리의 데이터를 관리 공유 메모리로 저장할 때 변환 테이블을 참조하여 장비 공유 메모리의 위치를 찾고 그 위치의 데이터를 관리 공유 메모리에 기록한다.

표 4.1 변환 테이블의 구조

Dev 1's data 1	Dev 2's data 1	...	Dev n's data 1
Dev 1's data 2	Dev 2's data 2	...	Dev n's data 2
Dev 1's data 3	Dev 2's data 3	...	Dev n's data 3
...
Dev 1's data n	Dev 2's data n	...	Dev n's data n

예를 들어 관리 공유 메모리의 Value[2]의 두 번째로 들어갈 장비 공유 메모리의 데이터의 위치는 변환 테이블에 2 열 2 번째 줄에 저장되어 있다. 장비 공유 메모리의 변환 테이블의 2 열 2 번째 줄에 저장되어 있는 위치에서 Data Information을 참조하여 그 크기만큼 읽어 관리 공유 메모리의 Value[2]의 두 번째 위치에 기록한다.

4.2 데이터 관리 모듈

데이터 관리 모듈은 메인 프로그램의 설정에 따라 여러 모듈을 실행 또는 정지시키는 기능을 한다. 이 모듈에서는 자신이 대기서버인지 활성서버인지를 판별하여 장비 접속 모듈을 실행시킨다. 활성서버인 경우에는 장비 접속 모듈을 실행시키고 대기서버인 경우는 장비 접속 모듈을 실행시키지 않는다. 그림 4.1에서 나타난 데이터 관리 모듈은 하위의 장비 접속 모듈과 상위의 감시제어 모듈의 중계자 역할을 하며 필드유닛에서 전송되어져 장비 공유 메모리에 저장된 데이터를 감시제어 모듈에서 사용하기 편하게 변경하여 관리 공유 메모리에 기록하는 기능을 가진다. 이때 사용되는 것이 변환 테이블이다. 감시제어 모듈과 데이터 관리 모듈 사이의 데이터 교환은 메모리 맵을 이용하며 명령전달은 함수의 호출을 이용한다.

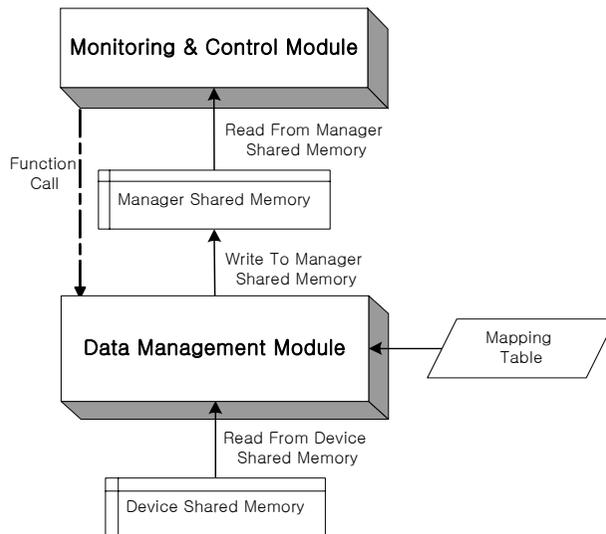


그림 4.1 데이터 관리 모듈의 구성

그림 4.1의 데이터 관리 모듈은 아래와 같은 기능을 가지며 그 알고리즘은 그림 4.2와 같다.

- ① 관리 공유 메모리와 장비 공유 메모리 생성
- ② 활성서버/대기서버 판별 후 장비 접속 모듈 실행
- ③ 변환 테이블을 이용하여 장비 공유 메모리의 데이터 변환 후 관리 공유 메모리 갱신
- ④ 감시제어 모듈로부터의 함수 호출 수행

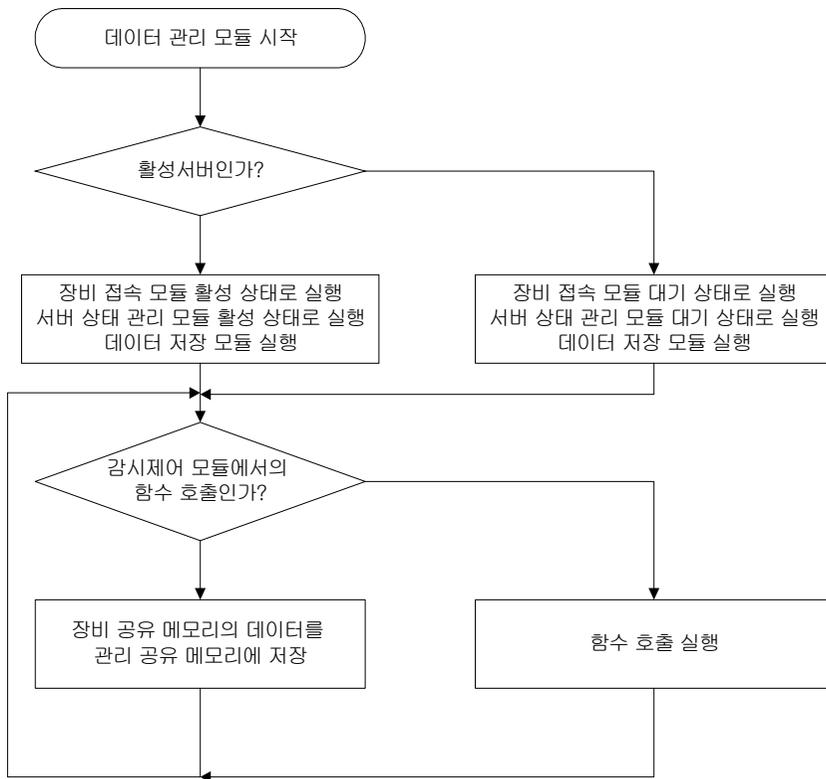


그림 4.2 데이터 관리 모듈

4.3 장비 접속 모듈

장비 접속 모듈에서는 여러 장비들로부터 데이터를 받고 제어신호를 보내는 작업을 한다. 필드유닛과의 통신은 LON, LAN, RS485 등으로 할 수 있으며 통신의 구조는 비슷하다. 본 논문에서는 LAN 환경에서 TCP/IP 프로토콜을 사용하는 것을 대상으로 하여 설명한다.

장비들과의 통신구조는 그림4.3과 같이 장비들로부터 데이터를 수신 받아 장비 공유 메모리에 저장한다.

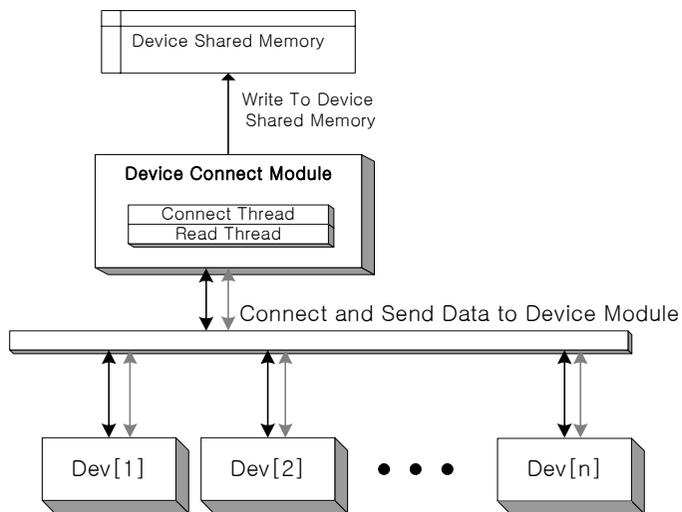


그림 4.3 장비 접속 모듈의 구조

장비 접속 모듈은 두 가지의 쓰레드가 실행되어 통신을 하며 그림 4.4와 같은 순서로 메시지를 교환 한다.

접속 관리 쓰레드(Connect Thread)는 장비의 접속과 통신 라인의 오류를 관리하며, 데이터 수집 쓰레드(Read Thread)는 필드유닛과

통신하여 데이터를 수집하는 기능을 가진다.

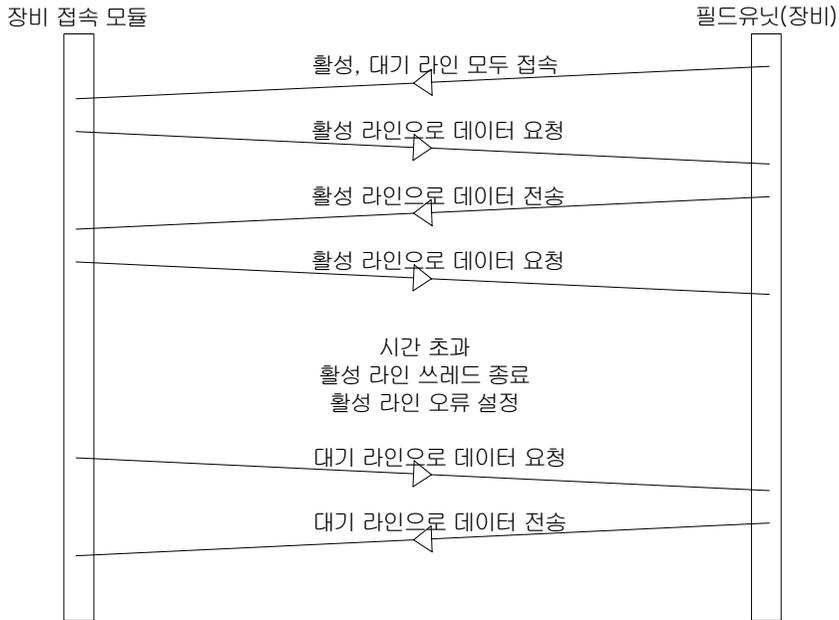


그림 4.4 장비 접속 모듈과 필드유닛과의 메시지 교환

데이터 수집 쓰레드와 필드유닛과의 통신 프로토콜의 구조는 다음과 같은 구조를 가진다.

Head(1 Byte)	Function(1 Byte)	Data(n Byte)	Error Check(1 Byte)	Tail(1 Byte)
--------------	------------------	--------------	---------------------	--------------

여기에서 Head, Function, Error Check, Tail은 모두 1 바이트이고 Data 부분은 메시지의 종류에 따라 그 크기가 변한다. Head와 Tail은 메시지의 처음과 끝을 알리는 부분이고 Function은 메시지의 종류를 나타낸다. Function에 따라서 데이터의 요청인지 상태정보의 요청인지 장비의 제어 정보인지를 판별한다.

4.3.1 접속 관리 쓰레드

장비 접속 모듈이 실행되면 그림4.5와 같은 순서로 실행된다.

- ① 장비들이 접속하면 접속된 장비들의 개수만큼 데이터 수집 쓰레드를 생성하고 필드유닛 접속 리스트에 추가한다.
- ② 필드유닛 접속 리스트에 저장된 순서대로 데이터 수집 쓰레드에 읽기 권한을 준다.
- ③ 필드유닛 접속 리스트의 모든 데이터 수집 쓰레드에게 읽기 권한을 주고 받은 뒤에 통신라인의 오류를 점검 한다.
- ④ 통신라인의 오류가 검출되면 서버 전환이 필요함을 장비 공유 메모리에 기록한다.

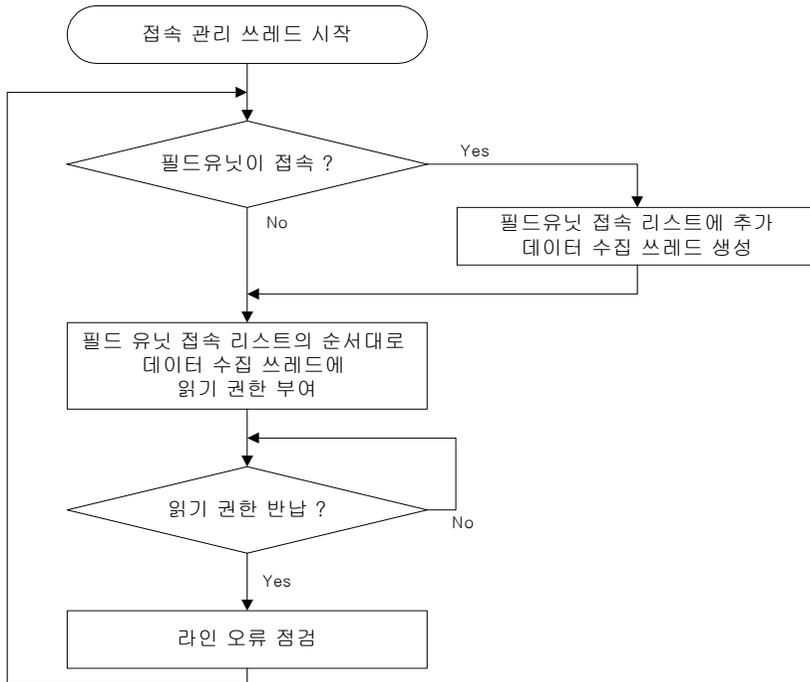


그림 4.5 장비 접속 모듈의 실행순서

필드유닛과 서버와의 통신라인의 오류를 검출하기 위해 본 논문에서는 상태저장 변수를 이용한다. 네트워크 상태에 따라 데이터 수집 쓰레드에서 이 변수의 값을 변경시키도록 하고 이 변경된 값에 따라 통신라인의 오류를 판단한다. 상황에 따른 변수 값은 표 4.2와 같다.

표 4.2 통신라인의 오류 상황에 따른 상태저장 변수 값

상태	상황
활성	활성 통신라인으로 통신 중
대기	대기 통신라인으로 통신 중
오류	활성/대기 통신라인 모두 고장

통신라인의 상태를 저장중인 상태저장 변수의 값을 보고 그림 4.6의 통신라인 오류 위치를 판단하여 조치를 내린다. 상황에 따른 조치는 표 4.3과 같으며 그림 4.7과 같이 증상을 판별한다.

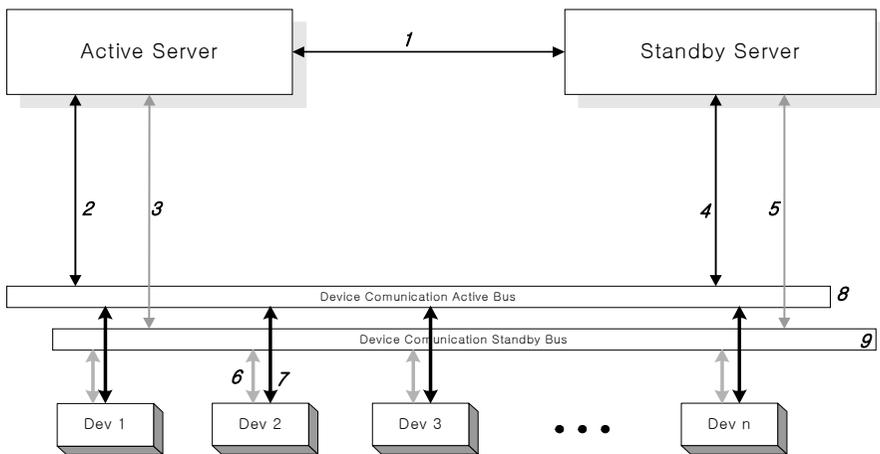


그림 4.6 통신라인 오류의 위치

오류의 위치가 1인 경우에는 서버사이의 통신이 되지 않으며 이것은 데이터 통신 모듈에서 관리를 한다. 2번이나 8번에서 오류가 발생하였을 경우에는 3번 라인으로 변경하여 통신을 계속하며 2번과 3번이 동시에 통신이 되지 않으면 서버전환을 한다.

표 4.3 상태저장 변수에 따른 판단과 조치

네트워크 증상	변수값	판단	조치
정상	활성	1. 아무런 문제가 없는 상황 2. 활성 통신라인으로 통신	필요 없음
정상	대기	1. 활성 통신라인의 버스 오류 2. 모니터링 시스템의 활성 통신장비 오류 3. 활성 통신라인으로 통신	통신 라인 오류설정
정상	활성 대기	1. 장비의 활성 통신라인 오류	통신 라인 오류설정
일부 통신 불가능	활성 대기 오류	- 오류 1. 활성/대기 통신라인 모두 오류 2. 장비에 문제가 생긴 경우 - 대기 1. 활성 통신라인 오류	서버전환
통신 불가능	오류	1. 모니터링 시스템의 통신장비의 문제가 생긴 경우 2. 활성/대기 두 개의 버스에 문제가 생긴 경우	서버전환

표 4.3과 같이 상태저장 변수의 값이 모두 활성이면 아무런 문제가 없이 필드유닛들과 통신이 되고 있는 상태이다. 그러나 오류나 대기 값이 포함되어 있는 경우는 통신라인에 문제가 생긴 경우이므로 알람을 설정하고 그 상태에 따라 통신라인을 변경하거나 서버를 전환한다.

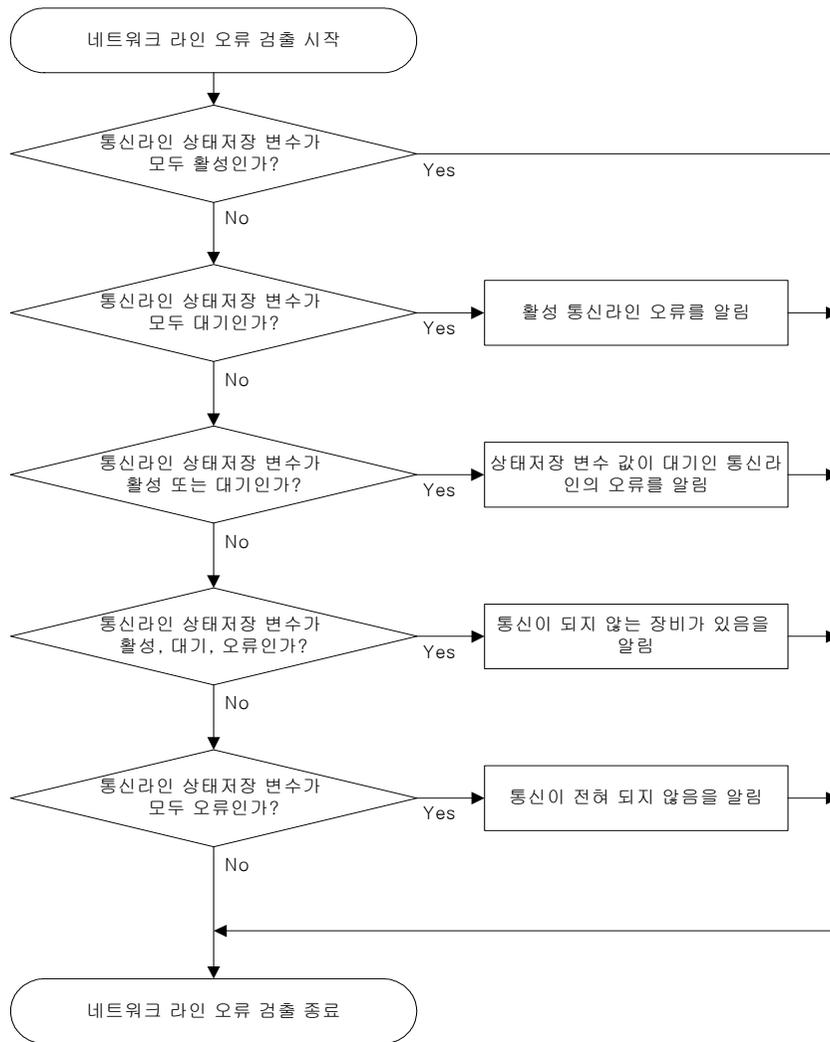


그림 4.7 통신라인 오류 검출 순서도

통신라인의 오류 검출을 위해 그림 4.7과 같은 순서로 상태저장 변수의 값을 확인 한다. 이것은 접속 관리 쓰레드에서 데이터수집 쓰레드로 읽기 권한을 순차적으로 부여하고 돌려받는 작업이 끝난 뒤에 데이터수집 쓰레드에 의해 설정되어진 상태 값으로 확인을 한다.

4.3.2 데이터 수집 쓰레드

데이터 수집 쓰레드는 그림 4.8과 같이 실행된다.

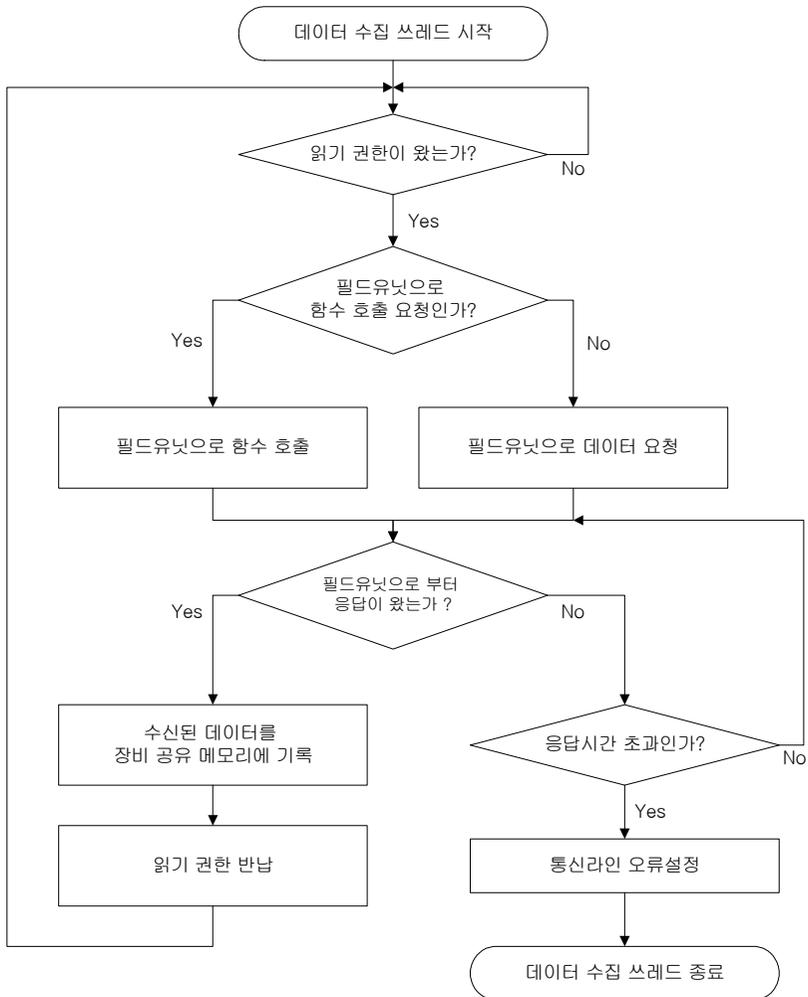


그림 4.8 데이터 수집 쓰레드의 구조

데이터 수집 쓰레드는 다음과 같은 기능들을 수행한다.

- ① 데이터 수집 쓰레드가 실행되면 각 변수들을 초기화하고 접속 관리 쓰레드에 의해 읽기 권한이 오기를 계속 기다린다.
- ② 읽기 권한이 왔을 경우 명령(Command)의 전송인지를 판별하고 그에 따라 명령전송이나 데이터 요청을 하고 요청한 데이터를 기다린다.
- ③ 데이터가 오면 데이터를 장비 공유 메모리에 기록하고 읽기 권한을 반환한다.
- ④ 기다리는 동안 응답시간 초과가 되는지 확인을 한다. 응답이 시간이 초과되면 상태저장 변수의 값을 대기상태로 설정하여 통신라인에 문제가 있음을 나타내고 쓰레드를 종료한다. 응답시간이 초과되지 않았다면 계속 데이터를 기다린다.

4.4 서버 상태 관리 모듈

서버 상태 관리 모듈에서는 활성서버와 대기서버 사이에서 정보를 주고받는 역할을 한다. 서버의 이중화가 되기 위해서는 이 모듈의 기능이 중요하다.

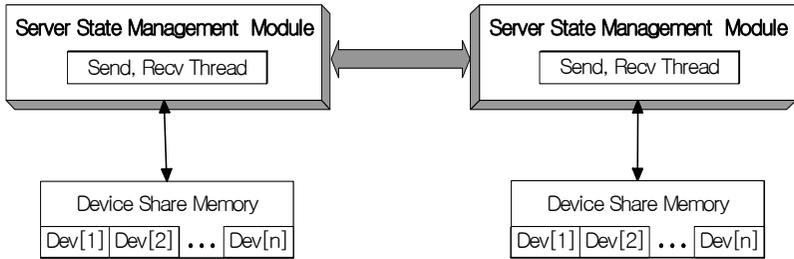


그림 4.9 서버 상태 관리 모듈의 구조

그림 4.9와 같이 대기서버의 서버 상태 관리 모듈이 활성서버의 서버 상태 관리 모듈로 접속 통신을 한다. 그리고 그림 4.10과 같이 메시지를 주고 받으며 서버의 상태를 확인하고 필요에 따라 서버전환을 한다. 서버 상태 관리 모듈의 주된 기능은 아래와 같다.

- ① TCP/IP를 이용하여 통신을 한다.
- ② 활성서버의 서버 상태 관리 모듈에서는 장비들로부터 읽어 들인 데이터를 대기서버의 서버 상태 관리 모듈로 보낸다.
- ③ 두 서버 사이의 상태를 확인한다.
- ④ 서버 전환 메시지를 주고 받는다.

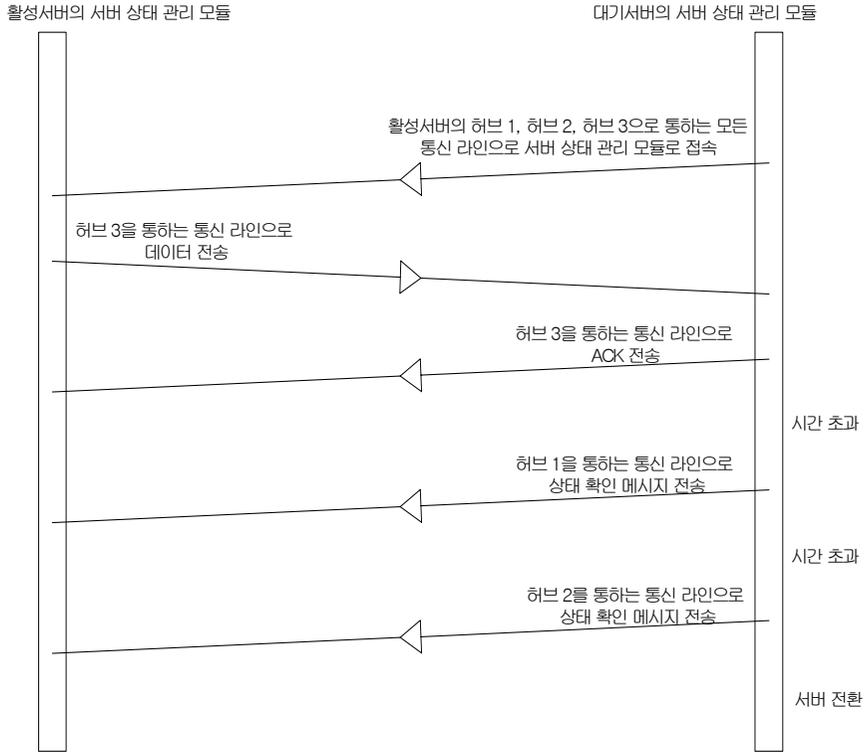


그림 4.10 서버 상태 관리 모듈의 메시지 교환

활성서버와 대기서버의 서버 상태 관리 모듈 사이에 주고 받는 메시지의 구조는 장비 접속 모듈의 데이터 수집 쓰레드와 필드유닛과의 통신 프로토콜 구조와 동일하다. 메시지의 종류로는 서버전환, 데이터 전송, 통신라인 상태전송 그리고 상태 확인 메시지 들이 있으며, 통신 프로토콜의 Function은 이 메시지의 종류를 나타낸다. 또한 Data 부분은 장비 공유 메모리의 데이터나 통신 라인 상태 데이터를 보내며 상태 확인 메시지의 경우에는 데이터 부분이 생략 된다. 서버전환 메시지의 경우에는 Data 부분은 사용자 요청 또는 통신라인 오류에 따라 서로 다른 상태 값을 전송한다.

4.4.1 활성서버의 서버 상태 관리 모듈

활성서버의 서버 상태 관리 모듈은 그림 4.11과 같은 알고리즘을 가지며 아래와 같은 순서로 실행된다. 활성서버의 서버 상태 관리 모듈은 대기서버로의 데이터 전송과 대기서버의 상태 확인 등이 있으며 활성서버의 서버 상태 관리 모듈에서는 항상 대기서버의 서버 상태 관리 모듈이 접속하기를 기다린다.

- ① 대기서버의 서버 상태 관리 모듈이 접속되면 장비 공유 메모리에서 데이터를 읽어서 대기서버로 보내고 ACK를 기다린다.
- ② ACK가 오지 않을 경우 접속을 끊고 다시 대기서버가 접속하기를 기다린다.
- ③ 대기서버의 서버 상태 관리 모듈에서 서버전환 메시지가 수신되어진 경우에는 ACK를 보내고 접속을 끊고 대기서버로 전환한다.
- ④ 감시제어 모듈에서 서버전환 함수가 호출되면 서버전환 메시지를 대기서버의 서버 상태 관리 모듈로 보내고 ACK를 기다린다.
- ⑤ ACK가 일정 시간 동안 오지 않을 경우 서버전환을 하지 않고 접속을 끊은 뒤 다시 접속하기를 기다린다.
- ⑥ ACK가 수신되면 접속을 끊고 대기서버로 전환한다.

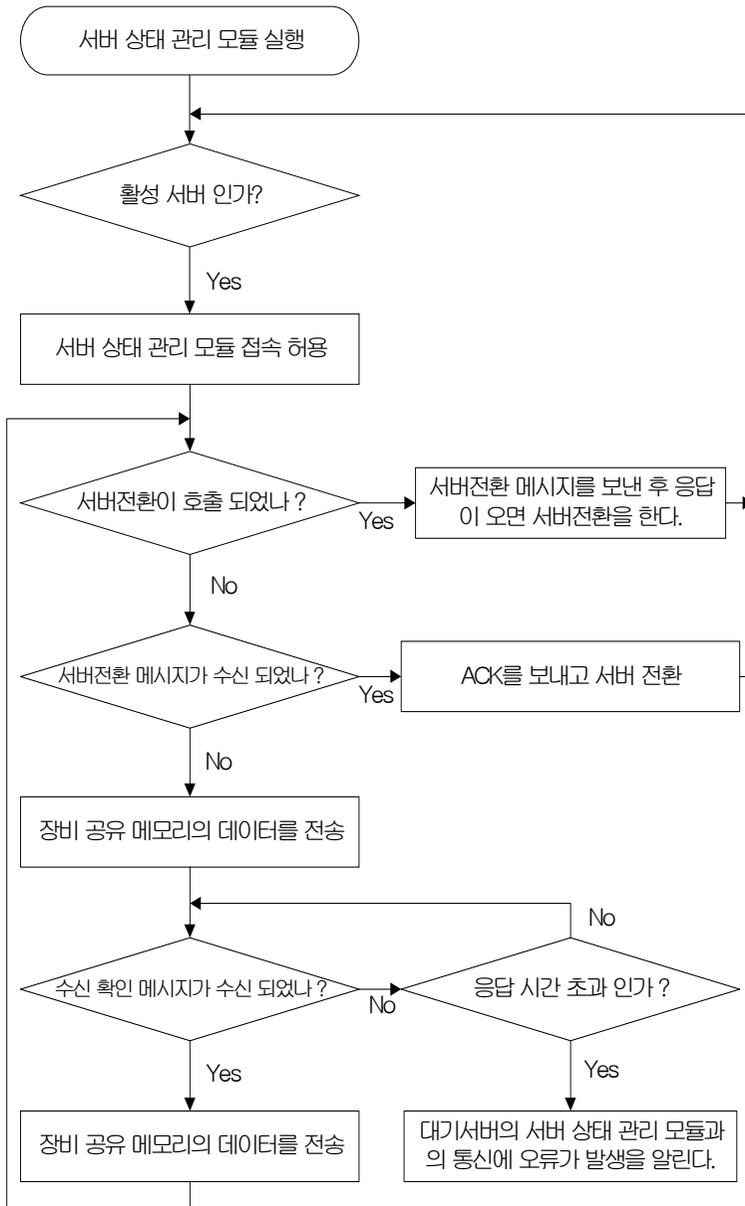


그림 4.11 활성서버의 서버 상태 관리 모듈

4.4.2 대기서버의 서버상태 관리 모듈

대기서버의 서버 상태 관리 모듈은 그림 4.12와 같은 알고리즘을 가지며 대기서버가 실행되면 서버 상태 관리 모듈도 실행되어 항상 활성서버의 서버 상태 관리 모듈로 접속을 시도하게 되고 접속이 되면 활성서버로부터 전송되어진 데이터를 처리하고 활성서버의 상태를 계속 점검한다.

- ① 활성서버가 보내는 데이터를 변경하지 않고 장비 공유 메모리에 저장한다.
- ② 활성서버로부터 일정 시간 동안 데이터 수신에 없을 경우 재접속을 한다.
- ③ 서버전환 메시지가 수신되어진 경우에는 ACK를 보내고 접속을 끊은 뒤 활성서버로 전환한다.
- ④ 감시제어 모듈에서 서버전환 함수가 호출되면 서버전환 메시지를 활성서버로 보내고 ACK를 기다린다.
- ⑤ 일정 시간 동안 ACK가 수신되지 않으면 접속을 끊고 대기서버인 상태로 재접속을 한다.
- ⑥ ACK가 수신되면 접속을 끊고 활성서버로 전환한다.

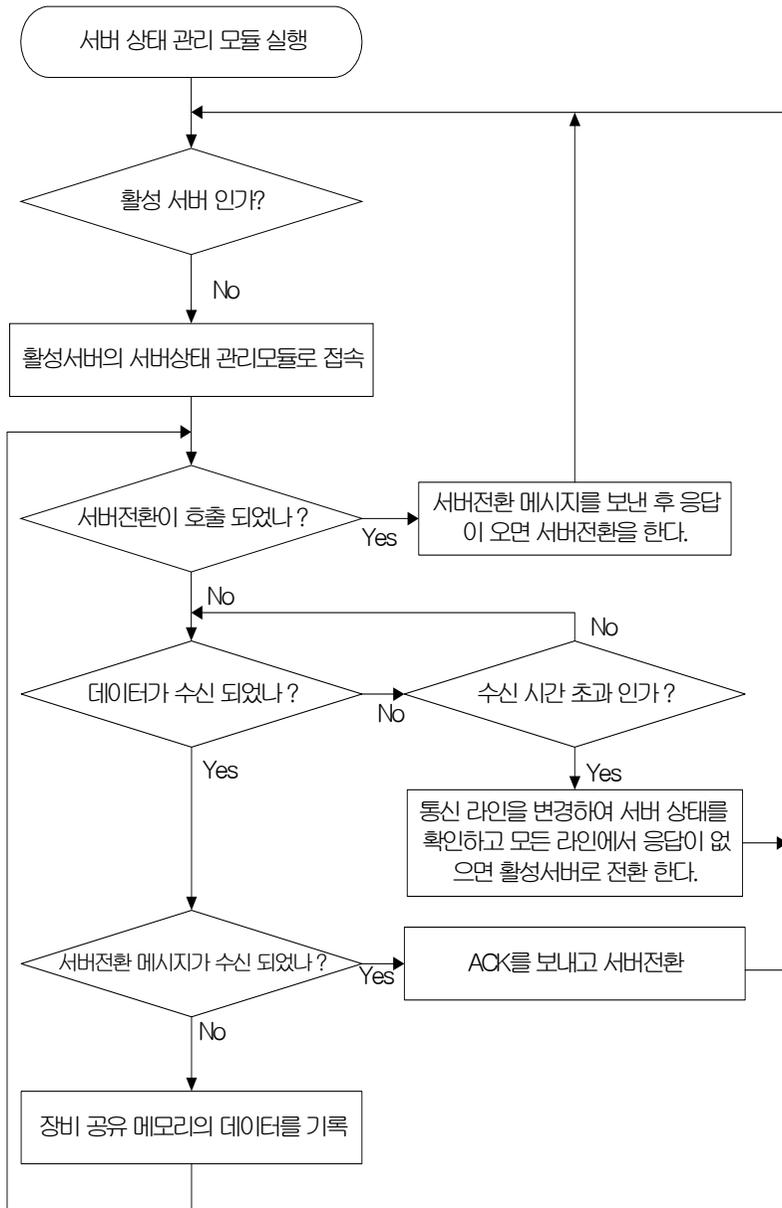


그림 4.12 대기서버의 서버 상태 관리 모듈 통신 순서도

4.5 데이터 저장 모듈

데이터 관리 모듈에서 수정되어 저장된 데이터를 관리 공유 메모리에서 읽어 들여 파일로 저장을 한다. 데이터파일은 시간단위로 생성되며 날짜별로 디렉터리를 만들어 저장된다. 저장된 데이터들은 감시제어 모듈에서 과거 데이터의 경향(Trend)을 확인할 경우에 사용된다. 그림 4.13은 데이터 저장 모듈의 구성도 이고 그림 4.14는 그 순서도이다.

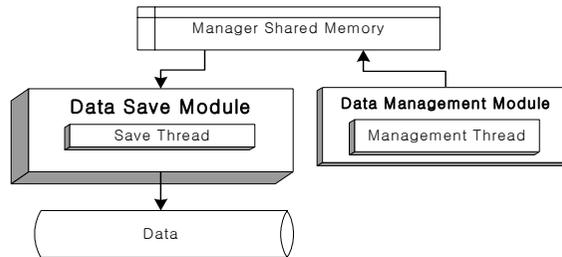


그림 4.13 데이터 저장 모듈의 구성

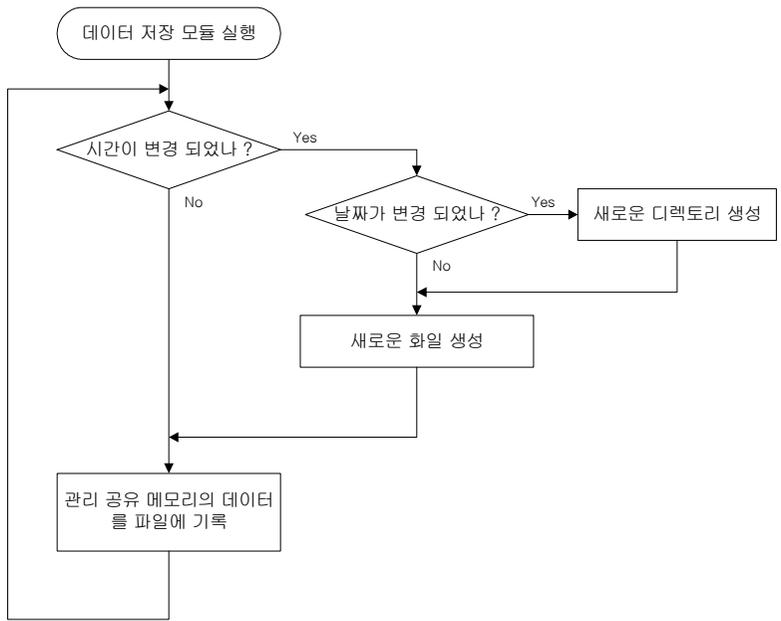


그림 4.14 데이터 저장 모듈의 순서도

제 5 장 실험 및 고찰

본 논문에서 제시한 감시제어 시스템의 통신 모듈의 실험을 위하여 그림 5.1과 같이 구성하였다.

서버로 사용될 컴퓨터 2대에 랜카드를 각각 2개씩 설치하고, 필드유닛 대신으로 사용될 컴퓨터 2대에도 랜카드를 2개씩 설치하였다. 그리고 8포트 허브 2개로 구성 하였다. 필드유닛은 제공 받기로 하였던 업체의 장비와 프로토콜이 일치하지 않아 컴퓨터로 대체하여 실험하였다. 서버와 필드유닛에 해당하는 컴퓨터 모두 윈도우 2000 프로페셔널이 설치된 컴퓨터를 이용하였다. 서버간의 상태확인용 허브 1과 허브 2로만 확인하도록 하였다. 활성서버에서 장비들로부터 받은 데이터를 대기서버로 보내는 기능을 생략하였으므로 허브 1과 허브 2를 통하여 서버간의 상태확인을 하는 경우에도 필드유닛과의 통신에 별다른 영향을 주지 않기 때문에 허브 3은 생략하였다.

오류검출과 서버전환을 위한 실험은 다음과 같이 실행하였다.

- (1) 허브전원 끄
- (2) 활성서버 종료
- (3) 필드유닛 1에 해당하는 컴퓨터의 통신 중지
- (4) 통신라인 단락

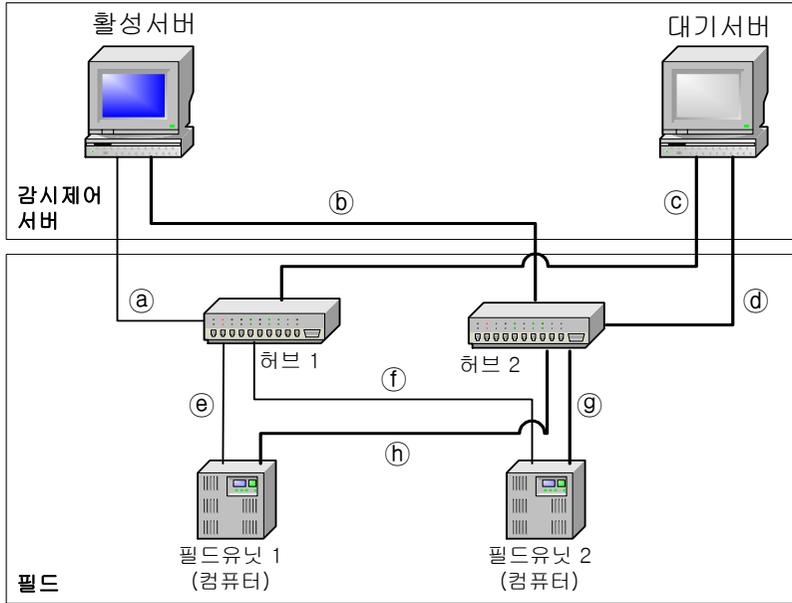
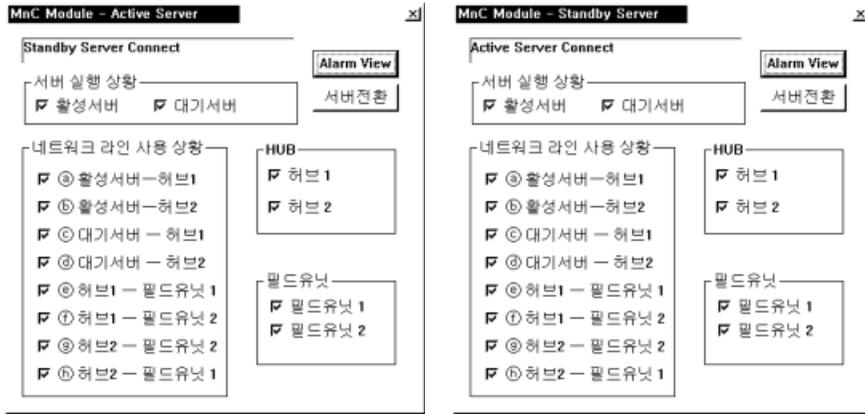


그림 5.1 통신 모듈 실험을 위한 구성

서버와 필드유닛 프로그램의 초기화면은 그림 5.2와 같다. 서버 실행상황 부분은 서버가 동작중이면 체크되고 동작중일 경우에는 체크박스에 체크가 되어 표시되고 오류가 발생하면 발생한 부분의 체크가 표시되지 않도록 구현 하였다. 아무런 오류가 없는 상황에서 대기서버 통신 프로그램이 활성서버 통신 프로그램으로 접속을 하게 되면 그림 5.2와 같이 실행된다. 이때에는 통신라인은 모두 연결되어 있는 상황이며 (a), (c)라인과 (a), (f)라인을 통해 필드유닛 으로부터 데이터를 받는다.

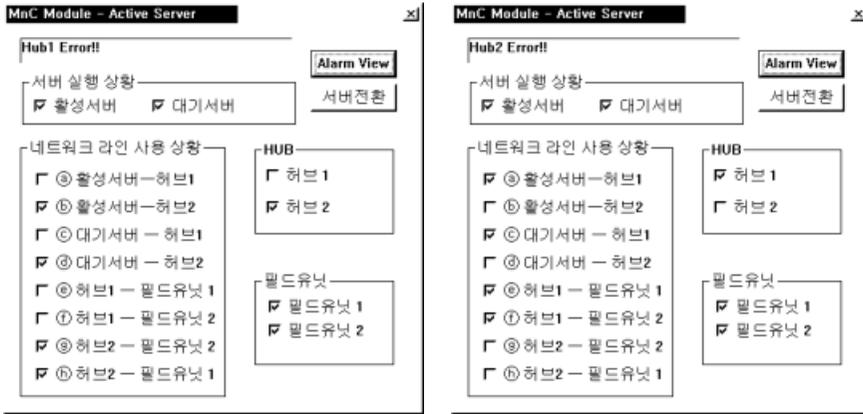


(A) 활성서버 초기화면

(B) 대기서버 초기화면

그림 5.2 모듈 실험 서버 프로그램 초기화면

오류가 없는 상황에서 허브의 전원을 내렸을 경우 그림 4.18처럼 표시 된다. 그림 5.3의 (A)는 허브 1의 전원을 내렸을 경우이고 (B)는 허브 2의 전원을 내렸을 경우이다. 허브1의 전원이 꺼지면 ㉠, ㉡, ㉢, ㉣라인이 동작을 하지 않게 되고 허브 2의 전원이 꺼지면 ㉤, ㉥, ㉦, ㉧라인이 동작을 하지 않게 되어 활성서버에서는 그림 4.18과 같은 화면이 나타난다. 대기서버에서도 똑같이 표시되어 나타난다. 허브 1에서 오류가 발생하면 허브 2를 통하는 ㉤, ㉥라인과 ㉦, ㉧라인을 통해 필드유닛으로부터 데이터를 받는다.



(A) 허브 1 오류

(B) 허브 2 오류

그림 5.3 허브 오류시의 활성서버 실행화면

활성서버에 오류가 발생하였을 경우에 서버전환이 되는지 확인하기 위하여 활성서버 프로그램을 종료 시켰을 경우 대기서버에서는 그림 5.4와 같은 화면이 나타났다. 활성서버가 응답이 없으므로 대기서버에서는 활성서버와 연결된 통신라인인 ㉠과 ㉡에 오류 표시가 나타나고 서버전환이 되어 대기서버가 활성서버 역할을 하게 된다. 따라서 대기서버가 ㉢, ㉣라인과 ㉤, ㉥라인을 통해 필드유닛으로부터 데이터를 받는다.

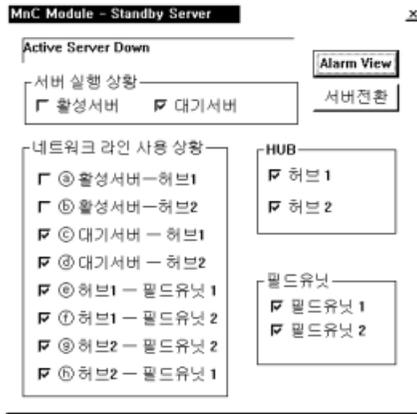


그림 5.4 활성서버의 오류시 대기서버측 화면

필드유닛1에 해당하는 컴퓨터의 프로그램을 종료시키면 그림 5.5와 같은 화면이 활성서버 쪽에 나타나게 된다. 이때에는 ㉔와 ㉕라인이 오류로 나타나게 된다. 이때에는 필드유닛이 오류임을 나타내고 필드유닛 2와는 계속 통신을 하게 된다.



그림 5.5 필드유닛의 오류시 활성서버의 화면

활성서버와 연결된 ㉔라인에 오류가 발생하면 그림 5.6과 같이 나

타나며 통신은 ㉑라인을 이용한다. 따라서 활성서버는 ㉑, ㉒라인과 ㉑, ㉒라인을 통해 필드유닛의 데이터를 받는다. 오류가 발생한 시점에서는 대기서버, 허브 1, 필드유닛 1, 필드유닛 2, ㉑, ㉒, ㉓라인도 오류가 발생한 것으로 인식하여 체크표시가 사라지며, 대기서버로부터 라인오류 정보가 수신된 후에 정상적으로 체크표시가 나타났다.



그림 5.6 ㉑ 라인 오류시 활성서버의 화면

제 6 장 결 론

본 논문에서는 감시제어 시스템의 신뢰성 향상을 위한 통신 모듈들을 개발하였다. 감시제어 시스템을 활성서버와 대기서버로 이중화하여 오류가 발생하면 상황에 따라 서버를 전환하도록 하고, 각 하드웨어와 연결되는 통신라인을 이중화하여 통신라인에 오류가 발생하면 오류증상에 따라 적절하게 동작하도록 하였다. 본 시스템의 구성방법을 통하여 다음과 같은 특징을 얻을 수 있었다.

- (1) 서버와 통신라인의 이중화 알고리즘을 구현하여 감시제어 시스템의 신뢰성을 향상시켰다.
- (2) 감시제어용 소프트웨어를 모듈별로 개발하여 차후 업그레이드를 할 경우 개발이 용이하도록 하였다.
- (3) 각 모듈간의 데이터 전송을 메모리 공유방법 중에서 메모리 맵핑 방법을 사용하여 시스템 모듈의 추가 개발이 필요할 경우 분리된 모듈로서 개발이 가능하도록 하였다.

향후 연구과제로는 개발되어진 모듈들을 이용하여 실용적인 SCADA 시스템을 개발해야 할 것이며 이에 따라 모듈들을 추가 또는 변경하여야 할 것이다. 그리고 이중화된 시스템에서의 더욱 지능적인 서버전환 알고리즘이나 통신라인 전환 알고리즘의 개발이 필요하며, 고장진단에 관한 전문가시스템의 도입이 필요할 것으로 사료된다.

참 고 문 헌

- [1] 권태상, “Ethernet 기반의 통합 네트워크 시스템”, pp. 70-74, 월간 자동제어계측, 2000년 5월호.
- [2] 이상범, 김정호, 산업용 네트워크 기술, 도서출판기술, 1993.
- [3] 최병철, 강창수, “SCADA/RTU의 기술동향”, pp. 25-31, 월간 자동제어계측, 1996년 12월호.
- [4] 손장근, “SCADA/RTU 시스템의 기술동향”, pp. 7-13, 월간 자동제어계측, 1996년 12월호.
- [5] 황철상, “HMI/SCADA 시장의 새로운 기술동향”, pp. 44-49, 월간 자동제어계측, 2000년 5월호.
- [6] 최경일, “HMI/SCADA SYSTEM의 기술동향”, <http://www.autotech.co.kr/new/0002/journal.htm>
- [7] 최욱현, 선박 엔진 고장진단 전문가시스템의 구현에 관한 연구, 한국박용기관학회 추계학술대회 논문집, pp. 79-84, 1998.
- [8] 최욱현, 하이브리드 지식 표현을 이용한 선박엔진 고장진단 시스템의 개발에 관한 연구, 공학석사학위 논문, 1999.
- [9] 김달현, 선박엔진용 실시간 고장진단시스템의 구현에 관한 연구, 공학석사학위 논문, 2001.
- [10] 최재곤, 선박기관실 모니터링 시스템을 위한 실시간 데이터 처리에 관한 연구, 공학석사학위 논문, 1998
- [11] 최정남, “PC-Based Control”, pp. 30-34, 월간 자동제어계측, 2000년 5월호.
- [12] 김정열, “고기능 산업용 HMI System”, <http://www.autotech.c>

o.kr/new/0002/journal.htm

- [13] 김형일, 이승룡, 전태웅, 박영택, “확장성과 개방성을 지원하는 SCADA 시스템 설계 및 구현”, pp. 753-763, 제어자동화 시스템공학 논문지, 제5권 제6호, 1999년 8월호.
- [14] http://kr.dir.yahoo.com/Business_and_Economy/Business_to_Business/Manufacturing/Factory_Automation/
- [15] KDT System의 CIMON, <http://www.komosys.co.kr/cimon.htm>
- [16] Citect User's Guide, Citect (C).
- [17] 김영경, “최고의 SCADA개발 도구인 WIZCON 소개”, pp. 73-77, 월간 자동제어계측, 1996년 10월호.
- [18] 문태석, “HMI 소프트웨어를 이용한 전력설비 감시제어 시스템”, pp. 50-55, 월간 자동제어계측, 2000년 5월호.
- [19] 최종선, “CIMON을 이용한 전력감시 시스템”, pp. 35-43, 월간 자동제어계측, 2000년 5월호.
- [20] 웹 기반 대형 광역 SCADA 시스템, http://www.infothe.com/main/nsub.asp?idx=521&choice_num=6
- [21] <http://www.citopia.co.kr/citect/kscalable.htm>