무중단의 풍력 발전 시스템 관리를 위한 고가용성 데이터베이스 시스템

김영환*, 손재기*, 함경선*
*전자부품연구원 지능형 시스템 제어
e-mail: yhkim93@keti.re.kr

Fault-Tolerant Database System for managing Wind Turbine Control System

Young-Hwan Kim*, Jae-Gi Son*, Kyung-sun Ham*
*Dept of Intelligent Control System, Korea Electronic Technology Institute

요 약

최근 신재생 에너지 분야 가운데 풍력발전에 대한 관심이 높아지면서 관련 산업 분야의 연구 또한 활발히 수행되고 있다. 풍력발전기의 경우 초기 설치에 고가의 비용이 필요하며, 고장 발생 시 일반적으로 풍력발전기의 동작을 멈추게 한다. 본 논문에서는 풍력발전기의 Nacelle에서 발생하는 다양한 센서 데이터를 수집하는 Bottom에서의 제어시스템을 다중화하여 고장 발생 시에도 풍력발전기의 중단 없이 지속적으로 에너지 생산이 가능하게 하는 고가용성 데이터베이스 시스템에 관한 것이다. 본 논문을 통해 개발된 고가용성 시스템을 통해 풍력 발전기의 높은 안정성을 보장하며, 수집된 데이터를 분석함으로서 고장에 대한 예측이 가능하다.

1. 서론

신재생에너지 분야에 대한 관심이 높아지면서 관련 산업 분야 연구 또한 활발히 진행되고 있다. 그 중 풍력발전 산업은 다른 발전 시스템에 비해 상대적으로 낮은 투자비용이 소모되며 무공해이다. 이러한 풍력발전기의 중1)요한 구성 요소의 하나인 로터 블레이드는 발전 용량을 높이기 위해 대형화 하는 추세이며 가벼우면서도 강도가 높은 복합재료를 이용하여 개발하고 있다. 이 풍력 복합재 블레이드는 외부의 돌풍, 조류의 충돌 등에 의하여 블레이드 내부에 접착부 탈착, 모재 손상, 층간 분리, 베어링 마모 현상 등의 손상이 발생하게 된다. 이러한 손상을 그대로 방치하게 되면 블레이드, 너셀, 타워 등에 심각한 손상을 초래하는 원인이 될 수 있다. 그러므로 풍력 발전기의 복합재 블레이드의 손상을 조기에 모니터링 하는 것은 매우 중요하다.

다양한 방법으로 풍력발전기의 상태를 체크하는 센서들에 관한 연구는 많은 성과를 이루고 있는 반면, 수집한 데이터를 지속적으로 관리 및 유지할 수 있는 고가용성 시스템에 대한 연구는 미비한 실정이다.

따라서, 풍력발전기의 상태에 대한 센싱 데이터를 높은 신뢰성과 안정성을 갖는 시스템으로 항시 모니터링 할 수 있어야 하며, 해당 시스템은 실시간성과 고가용성을 보장하여야 한다.

지식경제부 신재생에너지기술개발의 "풍력발전시스템용 제어기술 및 기기 개발(2010T100200257)" 지원을 받아 수행된 연구임

2. 관련 연구

2.1 고가용성 시스템

고가용성(High availability) 시스템이란 장애를 최소화하도록 관리하거나 예상된 시스템 다운타임(Down Time)을 최소화함으로써 서비스의 손실을 줄일 수 있도록 고안된 시스템을 의미한다.

서비스 중단은 크게 계획된 중단(Planned Down)과 계획되지 않은 중단(Unplanned Down)으로 나눌 수 있다[2]. 계획된 중단은 백업(Backup), 업그레이드(Upgrade), 유지보수(Maintenance) 등 계획에 의해 서비스가 중단되는 경우이고, 계획되지 않은 서비스 중단은 정전, 하드웨어 또는 소프트웨어 결함, 네트워크 장애, 물리적인 시스템 파괴, 자연 재해 등에 의해 서비스가 중단되는 경우이다.

(1) Heartbeat

Heartbeat은 Alan Robertson이 공개적으로 진행 중인 리눅스 고가용성 프로젝트로서 고가용성 시스템 내의 결함여부를 상호간에 체크하여 필요한 조치를 취하도록 한다. Heartbeat에는 시스템 점검 기능과 IP takeover 기능이 있으며, 상태 체크를 위한 heartbeat 메시지는 시리얼 라인, UDP, PPP/UDP 등을 통하여 전송될 수 있다.

Heartbeat 메시지는 연결된 다른 노드에게 전달 되어 상대 노드가 정상적이면 응답을 하지만 응답 메시지가 정

해진 시간 이내에 도착되지 않으면, timeout 카운트가 시작되고 계속해서 정해진 개수만큼 heartbeat 메시지에 대하여 응답 메시지를 받지 못하면 상대 노드는 장애가 발생한 것으로 간주한다. 이러한 heartbeat 메시지의 간격은 환경 설정을 통해 조절할 수 있다. 시스템 에러 체크를 단시간 내에 하기 위해 메시지 간격을 줄이면 그 만큼 heartbeat 메시지를 위한 비용이 증가하는 반면, 메시지 간격을 늘리면 메시지를 위한 비용은 감소하겠지만 그만큼 에러 체크하는 시간이 크게 지연되므로 메시지가 전달되는 하드웨어 장비의 성능과 시스템의 처리 능력을 고려하여 heartbeat 메시지 간격을 설정하여야 한다.

(2) Mon

Mon은 서버에서 제공하는 서비스의 가용성을 모니터링 한다. Mon은 서비스의 상태를 모니터 하여 서비스가 중단되거나 재실행되었을 때 적절한 조치를 취할 수 있다. 이러한 기능은 Mon 스케줄러에서 수행하며, Mon 스케줄러는 계속적으로 각 서비스들의 상태를 모니터링하고 서비스 중단이 발견될 경우 적절한 경고 작업(alert)을 호출한다.

Mon은 모니터 요소와 경고(alert) 요소 두 가지 형태로 구성되어있다.

- Monitors : Monitors는 서비스의 상태를 체크하고, 정해진 출력장치를 통해 서비스의 상태를 보고하는 프로그램이다. Mon 스케줄러와는 독립적으로 동작하기 때문에 새로운 서비스 모니터를 추가하려면, 해당되는 모니터 스크립트를 작성하여야 한다.
- Alerts : Alerts는 Mon이 서비스 중단을 감지 했을 때, 경고 메시지를 보낸다거나, 그 밖의 다른 행동을 취하는 스크립트이다. 또한, 정지되었던 서비스가 다시 실행되었을 때 upalert 를 통해 행동을 정의하는 것도 가능하다.

(3) MySQL Replication

Replication은 하나의 MySQL 데이터베이스 서버(마스터)에 있는 데이터를 하나 이상의 다른 MySQL 데이터베이스 서버(슬레이브)에 복제하는 기술이다. Replication을 사용하는 주요 목적은 다음과 같다.

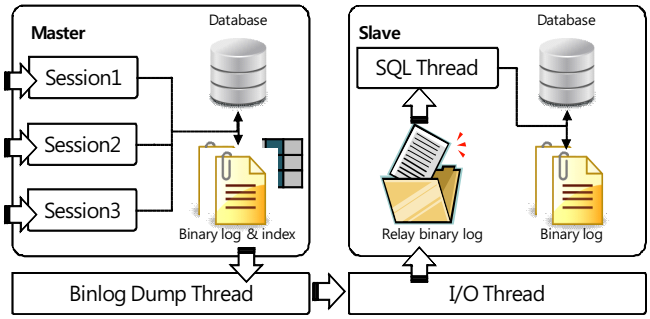
- 시스템 확장 : 여러 대의 서버에 로드를 분산함으로써 전체 시스템 성능을 개선한다. 이와 같은 환경에서는 모든 쓰기 및 업데이트 연산이 마스터 서버에서만 실행되어야 한다. 반면, 읽기 연산은 하나 이상의 슬레이브에서 수행될 수 있다.
- 데이터 안전성 : 데이터가 슬레이브에 복제되고, 슬레이브가 replication 프로세스를 잠시 중단 시킬 수 있기 때문에, 마스터 데이터와는 상관없이 슬레이브에서 안전하게 백업을 실행 할 수 있다.
- 분석 : 실제 데이터는 마스터에서 생성되기 때문에, 마스터 서버의 성능에는 영향을 주지 않은 채로 슬레이브에서 데이터 분석이 가능하다.

MySQL은 이러한 Replication 기능을 Thread를 통해 운영한다.

[표 2] Thread를 통한 Replication 운영

스레드	생성위치	용도
Binlog Dump Thread	Master	생성된 Binary Log를 Slave로 보냄
I/O Thread	Slave	Master로부터 Binary log를 받아 Relay Log로 작성해 저장
SQL Thread	Slave	생성된 Relay Log를 읽고 실행

MySQL은 <표1>의 세 가지 스레드는 비동기적으로 실행되어 Replication이 진행되도록 한다.



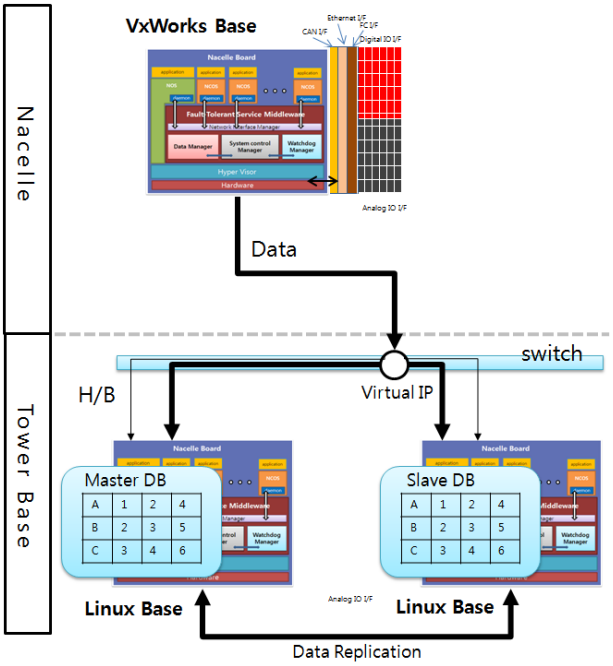
[그림 1] MySQL의 Replication 구조

MySQL의 Replication 동작은 [그림 1]과 같이 나타낼 수 있다. Slave에서 slave 모드가 시작되면 Master에는 Binlog Dump Thread가 생성되고 Slave에는 I/O Thread, SQL Thread가 생성된다. Binlog Dump Thread는 Binary log가 생성될 때마다 log를 slave로 보낸다. 이때 Slave의 I/O Thread는 받은 Binary log를 Relay log로 기록하게 되고, SQL Thread는 Relay log를 읽어 Slave 측에 저장한다.

3. 풍력발전기를 위한 고가용성 시스템 구축

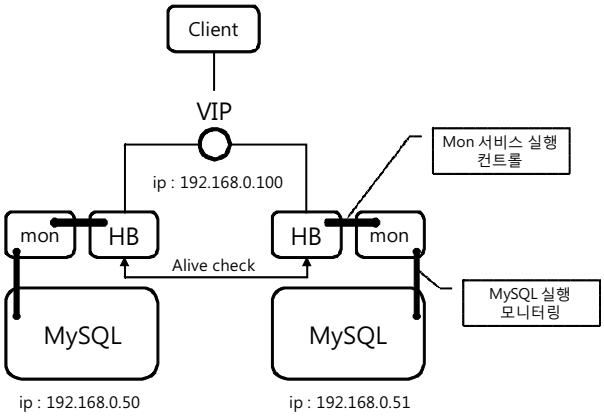
3.1 시스템 구조도

[그림 2]는 본 논문에서 구축한 풍력발전기를 위한 고가용성 모니터링 시스템의 전체 구조도 이다. 본 시스템은 Nacelle에 실시간 운영체제 VxWorks를 기반으로 하여 센서들의 데이터를 실시간으로 처리 가능하도록 하였으며, Tower Base에는 Linux를 기반한 고가용성 시스템을 구축하여 전송 받은 데이터를 안전하게 유지하고 효율적으로 활용할 수 있도록 하였다.



[그림 2] 고가용성 시스템 구조

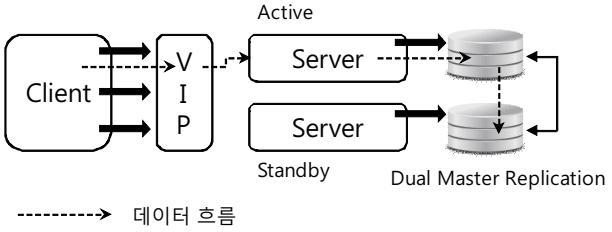
3.2 고가용성 데이터베이스 시스템 구축



[그림 3] Heartbeat, Mon, MySQL연동 구조

고가용성 데이터베이스 시스템 구축을 위한 방법으로 Linux에 Heartbeat, Mon, MySQL을 연동할 수 있다.

관련 연구에서 설명하였듯이 Heartbeat는 IP takeover 기능을 포함하는데 이는 가상 IP를 클라이언트에게 제공하여 실제 서비스를 하는 플랫폼이 어떤 것인지 관계없이 가상 IP로만 접근하여 서비스를 받을 수 있도록 한다. 이때, 서비스를 제공하던 하드웨어가 정지하는 사태가 발생하면 IP를 takeover하여 Standby 하드웨어가 Active 모드로 변환하여 가상 IP로 서비스를 제공한다. 또한, 서비스 객체인 MySQL의 정상 동작확인을 위해 Mon은 데몬을 띄워 mysql.monitor 스크립트를 동작시켜 MySQL이 정상적인 기능을 하는지 체크한다. <그림 3>은 고가용성 데이터베이스 시스템 구축을 위해 Heartbeat, Mon, MySQL을 연동한 구조를 나타낸다.

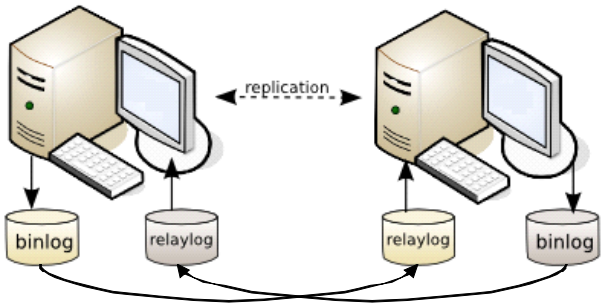


[그림 4] 데이터 저장 흐름

<그림 3>과 같은 시스템 구축을 통해 클라이언트는 서버가 제공하는 서비스를 VIP를 통해 사용 가능하다. <그림 4>와 같이 클라이언트가 전송하는 데이터는 가상 IP를 활성화 시킨 Active 서버의 Server프로그램에 전달된다. 전달된 데이터를 받은 Server 프로그램은 로컬 연결을 통해 MySQL 데이터베이스에 데이터를 저장한다.

(1) Dual Master Replication

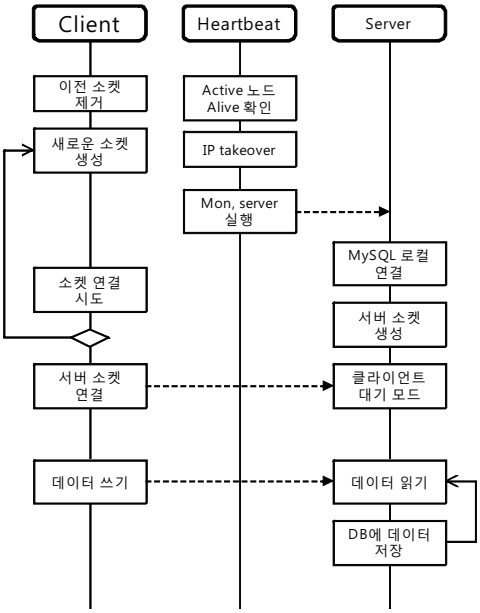
<그림 3>과 같이 시스템을 구성하였을 경우 클라이언트가 전송하는 데이터는 Active 모드에 있는 데이터베이스에만 데이터가 저장된다. 하지만, Replication으로 인해 Standby 모드의 데이터베이스에도 Master와 동일한 데이터가 저장된다. 반면, Master 데이터베이스가 동작하는 시스템이 오류로 인해 정지하는 경우가 발생한다면, Standby로 있던 시스템이 Active 시스템이 되면서, 클라이언트가 전달하는 데이터가 Slave 데이터베이스에 저장된다. 이러한 경우 Master 데이터베이스가 존재하는 시스템이 다시 정상 작동하였을 때, 두 개의 Master, Slave 데이터베이스는 서로 다른 데이터를 갖게 된다.



[그림 5] Dual Master Replication

(2) 고가용성을 위한 장애 극복

서비스를 제공하던 Active 서버가 계획 되지 않은 중단을 하였을 때, 이는 heartbeat에 의해 인지되어 IP takeover를 통해 Standby 서버가 Active 모드로 동작하게 한다. 따라서, 클라이언트는 동일한 IP 주소를 통해 서비스를 제공받을 수 있다. 하지만, 이전에 생성되었던 소켓은 유효한 상태가 아니기 때문에 소켓 접속이 끊어진 것을 감지하고 새로운 소켓을 생성하여 재접속할 필요가 있다. 이러한 과정은 <그림 6>과 같이 나타낼 수 있다.



<그림 6> 고가용성 시스템의 장애 극복 흐름도

<그림 6>은 Active로 동작 중이던 서버 노드가 중단되었을 때 일어나는 상황을 나타낸다. 클라이언트는 소켓에 더 이상 데이터가 전송되지 않는 다는 것을 인지하고 기존에 연결되었던 소켓을 제거한다. Heartbeat는 Active 노드가 응답이 없는 것을 확인하고 IP takeover를 위한 작업을 수행한다. 그 후, haresource에 등록한 서비스인 mon을 실행하고, 클라이언트로부터 데이터를 전달받을 server 프로그램을 실행한다. server 프로그램은 MySQL에 로컬 연결하여 데이터를 쓰기 위한 작업을 하고 클라이언트 연결을 대기하였다가 클라이언트의 접속이 확인 되면, 클라이언트가 보내는 데이터를 전달 받아 데이터베이스에 저장한다.

4. 실험 및 결과

시스템 테스트를 위해 VxWorks에 클라이언트 프로그램을 배치하였고, Linux에 고가용성 시스템을 구축한 후 Server 프로그램을 동작시켰다. 클라이언트 프로그램은 임의의 풍력발전기 센싱 데이터를 생성하여 서버로 전송시키는 역할을 수행한다. Server 프로그램은 전송 받은 데이터를 데이터베이스에 저장하며, 데이터베이스는 Replication을 통해 Standby 서버의 데이터베이스에 데이터를 복제한다.

```
mysql> show slave status\G;
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 10.0.0.90
                  Master_User: root
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000001
          Read_Master_Log_Pos: 106
               Relay_Log_File: node2-relay-bin.000002
                Relay_Log_Pos: 251
        Relay_Master_Log_File: mysql-bin.000001
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
```

<그림 7> Active 서버 slave 정상 동작 확인

```
mysql> show slave status\G;
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 10.0.0.91
                  Master_User: root
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000001
          Read_Master_Log_Pos: 106
               Relay_Log_File: node1-relay-bin.000002
                Relay_Log_Pos: 251
        Relay_Master_Log_File: mysql-bin.000001
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
```

<그림 8> Standby 서버 slave 정상 동작 확인

5. 결론

본 논문에서 구축한 시스템은 풍력발전기에서 모니터링 하는 모든 센서 데이터들을 계획되지 않은 중단이 발생하는 경우에도 극단적인 데이터 누수 현상 없이 안정적으로 유지 및 관리 할 수 있다. 이러한 높은 안정성을 보장하는 센서 데이터 수집은 데이터 분석을 통해 풍력발전기의 고장을 미리 예측 가능하게 한다. 또한, 풍향 및 풍력 정도에 대한 지속적인 분석이 가능하게 하여, 보다 효율적인 풍력에너지 활용 기반이 될 것이다.

참고문헌

[1] Harald Milz, "LINUX High Availability HOWTO", http://halinux.rug.ac.be/High-Availability-HOWTO-5.html, Dec. 1998.