論 文
53A-5-2

객체지향 데이터베이스를 이용한 전력계통 해석

Power System Analysis using OODB

朴 志 皓* · 白 榮 植**
(Ji-Ho Park · Young-Sik Baek)

Abstract - The complex documentation involved in power system analysis software require a well-defined and friendly database system. We have developed an object-oriented database management system for power system analysis, and have described load flow analysis and transient stability analysis using object-oriented database(OODB). Database management systems are widely used and achieve high reliability of data management in the engineering fields. However relational database system have shortcomings in application to power system analysis. In relational database, the data model is too simple for modeling complex data and database languages are very different from programming languages. Object-oriented techniques are sufficiently powerful to support data-modeling requirements of GUI applications. The GUI is implemented using C++ on a MS windows platform. The OODB supports data modeling requirements of GUI applications and the performance is well acceptable for GUI applications.

Key Words : object-oriented techniques, load flow, transient stability, OODB, GUI

1. 서 론

소프트웨어의 개발분야에 있어서 객체지향 프로그램방식(OOP)은 전통적인 프로그램방식의 비유연성을 극복하는 대안으로써 호평을 받고 있다. 왜냐하면 복잡한 문제의 해결에 유연성이 뛰어나고 유지보수가 쉽다는 장점을 지니기 때문이다. 시스템을 객체지향적인 방법으로 모델링하는 것은 객체지향적 해석, 객체지향적 설계 그리고 통합과 테스트 단계를 거친다. 전력계통은 매우 방대하고 이을 해석하기 위한 소프트웨어 또한 복잡하여 소프트웨어의 개발뿐 아니라 개발 후의 유지보수의 문제가 크다. 따라서 전력계통의 해석에 OOP가 적합하다고 할 수 있고, 전력계통의 해석에 객체지향적인 방식의 적용이 일반화되었다. 본 논문에서는 이러한 OOP의 장점을 이용하기 위하여 객체지향 데이터베이스를 연결하여 전력계통해석 소프트웨어를 설계한다.

전력계통해석 소프트웨어는 전력조류계산프로그램, 안정도해석프로그램, 고장해석프로그램과 같은 다양한 응용프로그램으로 구성되는데 이들 프로그램들은 해석에 필요한 데이터를 공유하거나 하나의 프로그램의 해석결과를 다른 프로그램이 이용하기도 한다. 따라서 우수한 자료관리 시스템은 프로그램의 성능을 평가하는데 중요한 요소가 된다. 따라서 자료관리를 원활하게 하기 위해서는 잘 정의된 친숙한 데이터베이스가 필요하다. 자료의 관리가 필요한 일반적인 프로그램에서는 관계형데이터베이스를 이용하고 있고, 전력계통의 경우도 마찬가지이다. 관계형데이터베이스(RDB)는 [1] 현실에 존재하는 개체를 정의하고 그들의 관계를 최소한의 의미를 가지는 테이블로 구성하고 그 테이블에 있는 열을 연결한 것이다. RDB의 장점은 첫째로 관계형의 스키마가 데이터베이스의 카탈로그에 저장되어있다. 둘째 범용의 질의 프로그램은 카탈로그를 사용할 수 있다. 셋째 SELECT, PROJECT, JOIN등의 연산자는 실행시간에 새로운 데이터베이스 뷰를 생성할 수 있다. 그리고 데이터베이스가 동일한 DBMS을 사용하는 응용 프로그램과 상호호환성이 있다는 점등을 들 수 있다. 반면 단점은[2] 첫째 영상이나 멀티미디어, 지형데이터 그리고 기상데이터와 같은 가변적인 길이의 데이터 멤버를 지원할 수 없다. 둘째 사용자 정의 데이터 형식을 사용할 수 없다. 셋째, 배열과 같은 반복되는 그룹의 표현이 불가능하다. 실제로 전력계통 해석에 필요한 데이터는 배열과 같은 데이터를 취급하는 경우가 많고, 관계형데이터베이스를 이용하기 위해서는 실제로 느끼는 것보다 많은 테이블이 필요하고, 최적의 접근속도를 얻기 위해서는 테이블이 고정되고 직접적으로 접근해야한다. 또한 테이블의 수가 많아지면 응용프로그램의 인터페이스가 매우 힘들다. 최근의 전력계통 해석프로그램 개발의 일반적인 경향은 객체지향 프로그램방식의 적용이다. 이것은 전력계통의 자체의 복잡성과 해석프로그램의 복잡함, 그리고 사후의 프로그램의 유지보수의 관점에서 객체지향프로그램이 탁월한 장점을 지니기 때문이다. 데이터베이스에도 이러한 소프트웨어적인 경향이 반영되고 있는데, 즉 객체지향데이터베이스(OODB)[3-6]의 등장이다. 객체지향데이터베이스의 장점은 첫째, 관계형 데이터베이스의 단점을 극복할 수 있

* 正 會 員 : 慶北大 工大 전자전기컴퓨터공학부 계약교수
** 正 會 員 : 慶北大 工大 전자전기컴퓨터공학부 교수
接受日字 : 2004年 1月 7日
最終完了 : 2004年 4月 8日

다. 즉 가변길이의 데이터를 저장할 수 있고, 사용자 정의의 데이터를 사용 가능하다. 둘째, 객체지향프로그램의 장점을 그대로 적용할 수 있다. 즉 다형성, 상속성, 캡슐화등의 장점을 이용할 수 있고, 객체지향방법으로 구성된 응용프로그램, GUI등과 쉽게 연결이 가능하다. 셋째, 관계형 데이터베이스의 장점을 그대로 구현 가능하다. 본 논문에서는 객체지향 데이터베이스를 설계하고, 전력계통의 단선도를 그리고 해석할 수 있는 GUI를 구축하고, 전력조류계산과 안정도 해석을 OODB을 이용하여 수행하였다.

2. 본 론

2-1 관계형데이터베이스와 객체지향형 데이터베이스

전력계통의 해석에 이용되고 있는 관계형 데이터베이스는 강력한 상용의 DBMS을 이용하므로 데이터의 관리가 용이하다. 하지만 관계형 데이터베이스는 데이터를 단순히 문자열이나 실수의 형태로 저장되므로 응용프로그램에서는 필요한 형태로 변환이 있어야하므로, 복잡한 공학용 데이터를 표현하기가 어렵다. 또한 데이터가 많은 테이블에 흩어져 있는 경우가 많아서 프로그램의 인터페이스가 용이하지 못하다. 관계형 데이터베이스는 일반적으로 데이터를 설계하기보다는 데이터를 레코드에 저장하도록 설계되었으므로 데이터의 변환이나 설계데이터의 관리에 있어서 속도가 느리다. 반면에 객체지향 데이터베이스는 데이터 구조가 메모리상의 C또는 C++의 데이터구조와 일치하고 대상데이터를 객체로 모델링하여 객체 자체는 물론 데이터 상호간의 복잡한 관계를 저장 가능하다. 따라서 데이터처리 속도가 빠르고 유연성이 뛰어나며 데이터 변환을 최소화할 수 있다. 또한 객체지향데이터베이스는 특별히 고안된 데이터포맷의 장점과 관계형 데이터 베이스의 장점을 모두 취할 수 있다. 데이터 베이스가 공학적인 문제에 응용될 때는 사용자는 디스크에 저장된 데이터를 메모리에 불러들이고, 그 데이터를 필요한 시간만큼 사용하고 수정된 데이터를 다시 디스크에 저장하게된다. 전력계통해석 응용프로그램이 주어진 시스템을 해석하기 위해서는 관계형데이터베이스에 저장된 레코드 형태의 데이터를 읽고 해석에 필요한 형태로 컴퓨터 메모리 구조에 맞게 데이터를 포인터 형태로 재구성한다. 응용프로그램이 GUI를 갖추고 있다면 GUI에서 사용자에 의한 시스템을 변화를 저장하기 위해서는 관계형데이터베이스는 데이터를 여러 개의 테이블에 레코드의 형태로 변환하여 저장하게 된다. 여러 개의 테이블을 넘나들며 데이터를 모아 데이터를 구성하는 것은 많은 처리시간을 요구하고 단일 프로그램에서 C++언어 자료구조를 이용하여 데이터를 표현하는 것보다 처리속도 면에서 매우 느리다. 객체지향 데이터베이스는 데이터의 저장구조를 컴퓨터 메모리상의 데이터 구조와 동일한 트리구조를 가진다. 각 객체들간의 포인터 연결구조에 데이터 베이스에서 유지하고, 메모리에 불러들일 때 이 구조의 포인터 연결만을 재구성하면 된다. 데이터베이스에서 C++언어 자료구조를 그대로 유지하고 있다. 따라서 응용프로그램을 작성하는 사람은 데이터베이스를 하나의 큰 가상메모리처럼 사용할 수 있다.

2-2 객체지향데이터베이스의 설계

2-2-1 객체지향데이터베이스 시스템의 구성

그림1은 객체지향데이터베이스 시스템의 환경을 단순화시켜 그린 것이다. 그림의 아래 부분은 데이터베이스로서 데이터파일과 인덱스파일들로 구성된다. 데이터파일은 저장데이터 접근소프트웨어에 의해 관리되고. 색인파일들은 색인파일 접근소프트웨어에 의해 관리된다. 이 두 개의 소프트웨어는 데이터베이스를 관리하는 목적을 가진 함수들의 라이브러리이다. 즉 데이터베이스 관리시스템(DBMS)을 구성한다. 데이터베이스와 DBMS을 합쳐서 데이터베이스 시스템이라 부른다. 데이터베이스가 데이터파일과 색인파일들의 집합이므로 데이터베이스 관리함수들은 데이터 파일과 색인파일에 직접 접근한다. 따라서 데이터 베이스를 이용하는 응용프로그램들은 DBMS을 통하여 데이터베이스에 접근할 수 있다. 색인파일은 데이터파일에 대한 색인을 가지고 있으므로 색인을 통하여 응용프로그램이 데이터 파일에 접근할 수 있게 해준다. 보조기억장치에 저장된 데이터에 검색하는데는 B트리구조가 적합하다. 따라서 본 논문에서는 B트리를 이용하여 색인을 구성한다.

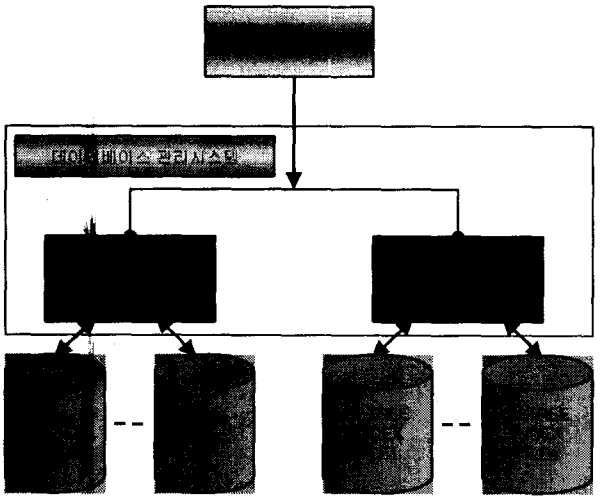

그림 1 객체지향데이터베이스 시스템 환경

Fig. 1 Object-oriented database system environment

2-2-2 객체의 지속성

객체지향데이터베이스에서 객체가 지속성을 가진다는 것은 객체를 디스크에 저장하고 다시 불러오는 것을 말하는데 객체는 생성자가 소멸된 후에도 존재해야되고 객체의 위치는 객체가 생성된 메모리의 주소공간으로부터 디스크로 이동한다. 따라서 객체에게 지속성을 부여하기 위해서는 지속기반클래스를 설계해야된다. 지속기반클래스의 기능은 객체의 데이터멤버를 디스크에 쓰기, 클래스가 디스크에 쓰여진 데이터 멤버를 읽어서 클래스의 데이터 멤버를 생성하기 그리고 클래스가 데이터 멤버의 크기와 위치를 알 수 있도록 하는 것 등이다. 하지만 이러한 지속기반클래스의 설계는

불가능하다. 왜냐하면 지속기반 클래스에서 파생되는 클래스 자체에서 결정되는 것 즉 파생클래스 자신의 크기, 실행시간에 결정되는 정보등이 대부분이기 때문이다. 따라서 지속기반클래스는 그것의 기능을 가상함수로 구현하여 실제의 기능이 파생클래스에서 실행되도록 해야한다.

```
class Persist
{
 virtual writeObj();
 virtual readObj();
    :
};
class Bus: public Persist
{
 data objects;
 void writeObj();
 void readObj();
  :
};
```

그림 2 지속기반 객체의 상속

Fig. 2 Inheritance of persistent base class

2-2-3 설계된 객체지향데이터베이스의 기능

객체지향데이터베이스에서 데이터 처리는 모두 객체단위로 이루어지므로, 설계된 객체지향데이터베이스는 객체의 존재검사, 객체의 삽입, 객체의 수정, 객체의 삭제등의 기능과 객체가 무결성을 유지할 수 있도록 하여야한다. 객체가 무결성을 유지한다는 것은 객체의 기능과 관련된 것으로, 이미 존재하는 객체는 다시 추가되어서는 안되고 또한 존재하지 않는 객체를 대상으로 하는 추가, 변경, 삭제의 기능은 수행 될 수 없다. 따라서 데이터베이스는 객체를 추가 가능한지, 변경가능한지 그리고 삭제 가능한지를 판단하는 함수를 제공하여야 한다. 데이터베이스는 객체단위로 부여되는 기본적인 기능과 무결성외에 저장된 모든 객체를 검색할 수 있는 기능을 가져야한다. 데이터베이스는 객체를 각각에 부여된 객체식별 아이디 순서로 트리구조로 이용하여 마치 디스크에서 가상메모리처럼 저장하므로 저장된 객체의 검색은 객체의 아이디를 이용하거나 첫 번째 객체 또는 마지막 객체만을 참조하면 검색이 가능하다. 그림 3은 지속기반클래스의 기능과 데이터베이스의 관리기능을 추가시킨 객체지향 데이터베이스 시스템의 기능구성이다. 본 논문에서는 그림 3과 같은 기능을 수행하는 객체지향데이터베이스를 설계하고 ODBPSA(Object-oriented DataBase for Power System Analysis)라 명명한다. 그림4는 그림3에서 구성된 객체지향 데이터베이스를 구현하기 위한 중요 클래스들의 계층구조를 나타낸 것이다. ODBPSA 클래스는 색인파일과 데이터파일을 관리하는 객체를 멤버로 가지고 있고, 지속기반 클래스 Persist는 객체의 아이디를 관리하는 ObjKey클래스를 멤버로 가지고 있다. ODBPSA와 Persist는 상호 프랜드 클래스이다.사용자는 ODBPSA를 이용하여 데이터파일과 색인파일명을 지정하고 생성하고, Persist를 상속하는 클래스를 설계

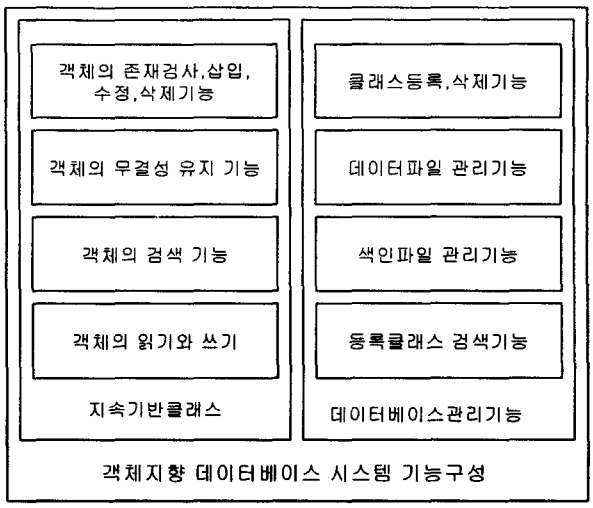

그림 3 객체지향데이터베이스의 기능구성

Fig. 3 The functional construction of OODB

한 다음 ODBPSA에서 생성된 색인파일과 데이터파일에 사용자 객체의 데이터를 저장하여 관리할 수 있는 구조이다.

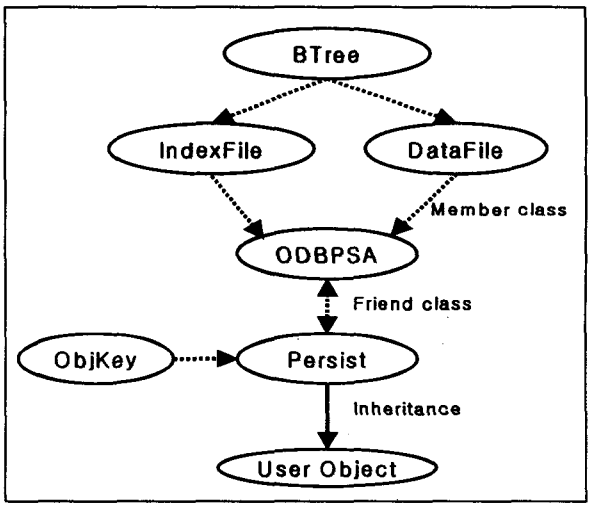

그림 4 설계된 클래스의 계층구조

Fig. 4 The hierarchy of designed classes

3 전력계통해석

3-1 전력계통의 객체지향적 표현

전력계통은 그림 5와 같이 모선, 선로, 모선과 모선사이의 선로에 설치된 장치들, 부하, 모선에 설치된 장치들의 집합으로 표현할 수 있다. 선로에 설치된 장치는 단로기, 차단기, 변압기등 이고 모선에는 Shunt 장치을 들 수 있다. 전력계통의 구성요소들을 객체로 모델링하여 이들의 연결관계를 계통의 구성에 맞게 맺어주면 전력계통의 표현이 완성되는 것이다. 객체지향적 모델링의 기본개념에 충실하게 시스템의 실 객체의 독립된 단위를 클래스로 모델링하여야 한다. 따라서 기본적으로 계통요소를 표현하는 데이터는 각 요소를 모델링 하는 객체단위로 분리된다.

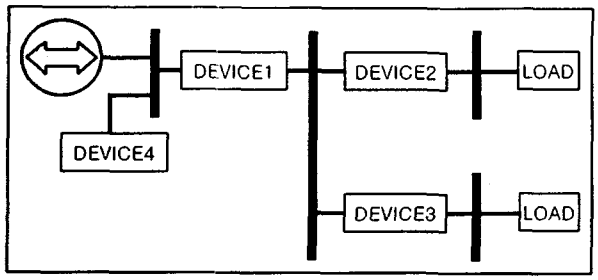

그림 5 전력계통의 구성
Fig. 5 The configuration of power system

3-2 전력조류계산

3-2-1 시스템의 객체지향적 구성.

시스템의 전체구조를 파악하기 위해서는 그림 6과 같이 모선객체의 포인터 연결리스트, 선로객체의 포인터 연결리스트를 사용하면 된다. 각각의 연결리스트의 첫 번째 연결객체를 모선헤더 포인터에 연결하면 순차적으로 연결된 모든 모선객체 즉 시스템에 존재하는 모든 모선에 접근 가능하다. 각각의 모선객체는 그림 7의 구조를 지니고 있다. 마찬가지로 선로객체의 경우도 선로객체 연결리스트의 첫 번째 객체를 선로헤더 포인터에 연결하면 순차적으로 연결된 모든 선로객체에 접근하는 것이 가능하다. 이제 두 개의 헤더 포인터만 참조하면 시스템을 구성하는 모든 구성요소에 접근이 가능한 것이다.

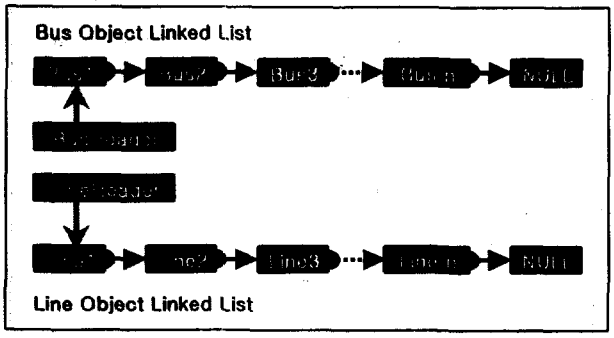

그림 6 전력계통구성 객체의 연결구조
Fig. 6 Object connecting structure for power system

그림 7은 모선객체의 구성을 나낸다. 모선객체는 자신의 식별 아이디, 모선에 연결되는 발전기, 부하, 차단기 그리고 선로객체에 대한 양방향 포인터를 가진다. 각각의 객체가 독립성을 유지하면서도 상호관계를 유지하기 위한 것이다. 변수 Next는 모선 객체들을 연결리스트로 구성하기 위한 포인터변수이다. 그림 8은 선로, 발전기, 변압기 그리고 Shunt 객체의 구성을 나타낸 것이다. 각 객체는 다른 객체와의 연결을 위하여 양방향 포인터를 가진다. 객체지향적인 전력조류계산의 장점은 그림 6-8에서 나타낸 것처럼 시스템의 모델링이 쉽고, 간단한 자료처리 기법을 이용하면 아무리 복잡한 시스템이라도 시스템의 모든 구성요소에 대한 접근은 매우 빠르고 쉽다. 또한 시스템의 변화에 대한 수정이 매우 용이하다.

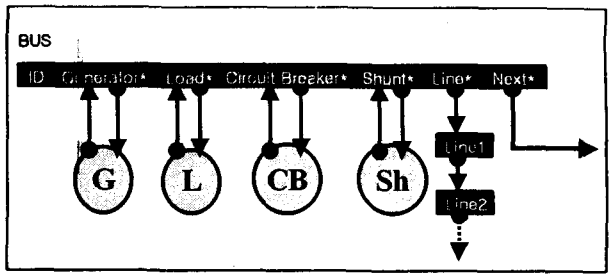

그림 7 모선객체의 구성
Fig. 7 The structure of bus object.

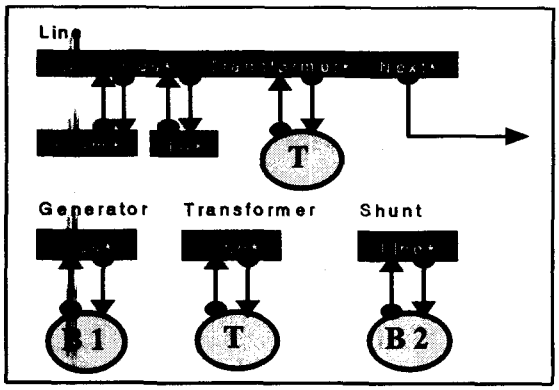

그림 8 선로, 발전기, 변압기 그리고 Shunt 객체
Fig. 8 Line, generator, transformer and shunt object

3-2-2 설계된 객체의 OODB 저장구조

설계된 모든 클래스는 OODB에 자신의 구조 및 데이터를 저장하는 기능을 가져야한다. 이를 위해서는 설계된 클래스가 그림 4와 같은 구조가 되도록 하면 되는데, 응용객체는 Class Persist에서 파생되는 구조를 가지면 된다. 파생클래스는 ObjRead()와 ObjWrite()함수에 대한 처리루틴을 만들어 객체자신을 저장한다. 그림 6,7,8의 구조, 즉 시스템의 구성 및 데이터를 OODB에 그대로 저장한다. 그림 9는 모선객체와 선로객체에 지속성을 부여하는 예이다.

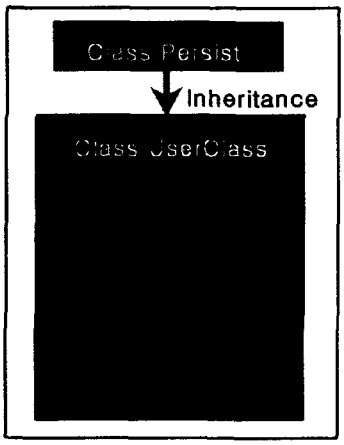

그림 9 지속객체
Fig. 9 Persistent object

3-3 안정도 해석

3-3-1 발전시스템의 객체지향적 구성

본 논문에서 수행한 전력계통의 안정도해석은 동기발전기와 발전기제어 장치들를 미분방정식으로 모델링하고 각각의 발전기들을 상호 연결하여 계통전체의 안정도를 모의하는 방법이다. 동기발전기의 이축 모델은 발전기의 차과도상태(subtransient state)는 고려하지 않고 과도상태만을 고려한 것으로 직축과 횡축의 과도 유기전압의 변화를 모두 고려한다. 본 논문에서는 발전기 이축 모델을 객체지향적으로 모델링한다. 이축 모델의 전기 및 운동방정식은 아래의 식들로 주어진다[7].

$$\dot{E}_q' = \frac{1}{T_{d0}'}(E_{FD} - E) \tag{1}$$

$$\dot{E}_d' = \frac{1}{1+sT_{q0}'}(x_q - x_q') \tag{2}$$

$$E = E_q' + (x_d' - x_d)I_d \tag{3}$$

$$T_e = E_d' I_d + E_q' I_q - (x_q' - x_d')I_d I_q \tag{4}$$

$$V_d = -rI_d - x_q' I_q + E_d' \tag{5}$$

$$V_q = -rI_q + x_d' I_d + E_q' \tag{6}$$

$$\dot{\omega} = \frac{1}{M}(T_M - D\omega - T_e) \tag{7}$$

$$\dot{\delta} = \omega - 1 \tag{8}$$

여기서 E_q'는 횡축 과도 유기 기전력, E_{FD}는 여자전압, E는 고정자 공극 실효치 전압, V_d는 단자전압의 직축 성분, V_q는 단자전압의 횡축 성분, I_d는 단자전류의 직축 성분, I_q는 단자전류의 횡축 성분, x_d는 직축 리액턴스, x_q는 횡축 리액턴스, x_d'는 직축 과도 리액턴스, x_q'는 횡축 과도 리액턴스, T_e는 전기적 토크, T_{d0}'는 직축 과도 개방회로 시상수, r은 고정자 저항, T_{q0}'는 횡축 과도 개방회로 시상수이다.

미분방정식으로 모델링되는 동적방정식을 객체지향적인 방법으로 해석하기 위하여 시스템의 기본 객체인 덧셈기, 곱셈기, 적분기, 이득기등을 클래스로 정의하고 이외에 시뮬레이션에 필요한 객체들 즉 제한기, 수학함수처리 객체등을 정의한다. 이들 기본 객체들은 한 번의 구성으로 모든 시뮬레이션에 사용할 수 있는 기본소자가 된다. 이 요소들을 사용하여 발전계통의 부시스템인 발전기, 조속기, 여자기, 안정기등이 각 기능별로 정의된다. 이들 각 부시스템들은 수많은 모델이 존재할 수 있으므로, 만약 모델의 변화가 있다면 변경된 부시스템만을 교체하면 되므로 시스템의 변화에 쉽게 대응할 수 있다. 그림 10의 발전기 안정도 해석을 위한 발전시스템의 객체지향적 모델링 단계를 나타낸 것이다. 시스템을 구성하는 가장 기본적인 요소를 객체로 모델링하고 이들을 이진트리구조로 상호연결하여 복잡한 시스템을 구성하는 방식이다. 따라서 객체지향적인 모델링 방식은 복잡한 시스템의 모델링에 있어 핵심 요소를 먼저 객체로 구현하고, 이를 조립하는 방식을 채택함으로써 시스템의 모델링이 보다 명확하고 복잡성에서 오는 모델링의 오류를 확실하게 줄여준다. 즉 동적인 시스템의 소프트웨어적인 모델링 방식의 새로운 기법을 본 논문에서 제안했다.

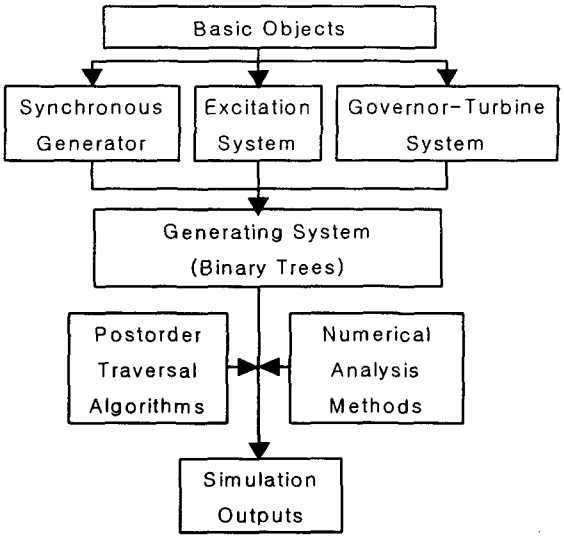

그림 10 발전시스템 모의를 위한 모델링 단계
Fig. 10 Modeling steps for generating system

그림 11은 설계된 객체들을 상호 연결하여 하나의 발전시스템을 완성한 객체연결구조이다. 실제 프로그램상에서는 이들의 연결구조를 이진트리를 이용하여 구현한다. 그림 11의 구조를 전력조류계산을 위하여 이미 설계된 발전기 객체가 이진트리 멤버데이터로 가지도록 만들어 주는 것은 OOP에서는 매우 쉬운 일이다. 이러한 방식의 가장 큰 장점은 시스템의 변화에 유연하고 모든 시스템을 모델링할 수 있다는 것이다. 이것은 다양한 제어장치로 구성되는 발전시스템의 모델링에 매우 큰 유연성을 부여한다.

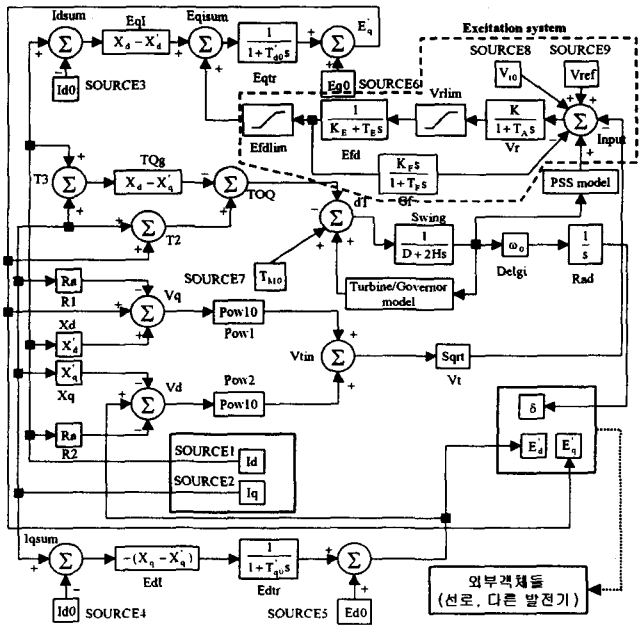

그림 11 발전기 이축 모델의 객체연결 구성
Fig. 11 Object connecting structure of two-axis model

3-3-2 지속성의 부여

그림 12는 트리구조로 연결된 객체들의 메모리상의 구조와 OODB상의 저장구조를 나타낸 것이다. 그림 11의 구조가 발전기의 동적인 모델링은 이진트리를 이용하여 구현한 것인데, 이 연결구조가 지속성을 가지기 위해서는 컴퓨터 메모리상에서의 연결구조가 그대로 디스크상에 저장되어야 한다. OODB는 B트리를 이용하여 객체들을 저장하므로 컴퓨터메모리상의 연결구조와 동일한 연결구조를 가지게 된다. 따라서 지속성을 위하여 연결구조를 재구성할 필요가 없고, 메모리상의 주소연결 구조를 디스크에서는 객체식별자를 이용하는 구조로만 바꾸어 주면 된다. 관계형 데이터베이스는 그림 11과 같은 시스템의 데이터를 저장하기 위해서 테이블을 만들고 데이터를 흩어서 저장하고, 다시 데이터를 불러올 때는 이들을 결합해야 하는데 이것은 데이터처리의 속도의 저하와 효율성을 감소시킨다.

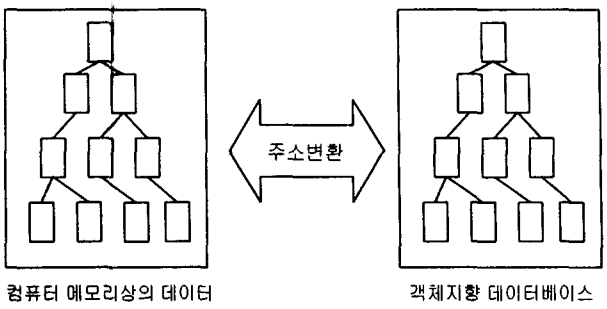

그림 12 OODB의 저장구조

Fig. 12 The storage structure of OODB

4. 전력계통 해석용 GUI의 구현

마이크로소프트(MS)사에서 제공하는 MFC(Microsoft Foundation Class)는 윈도우 운영체제를 위해 MS사 자체에

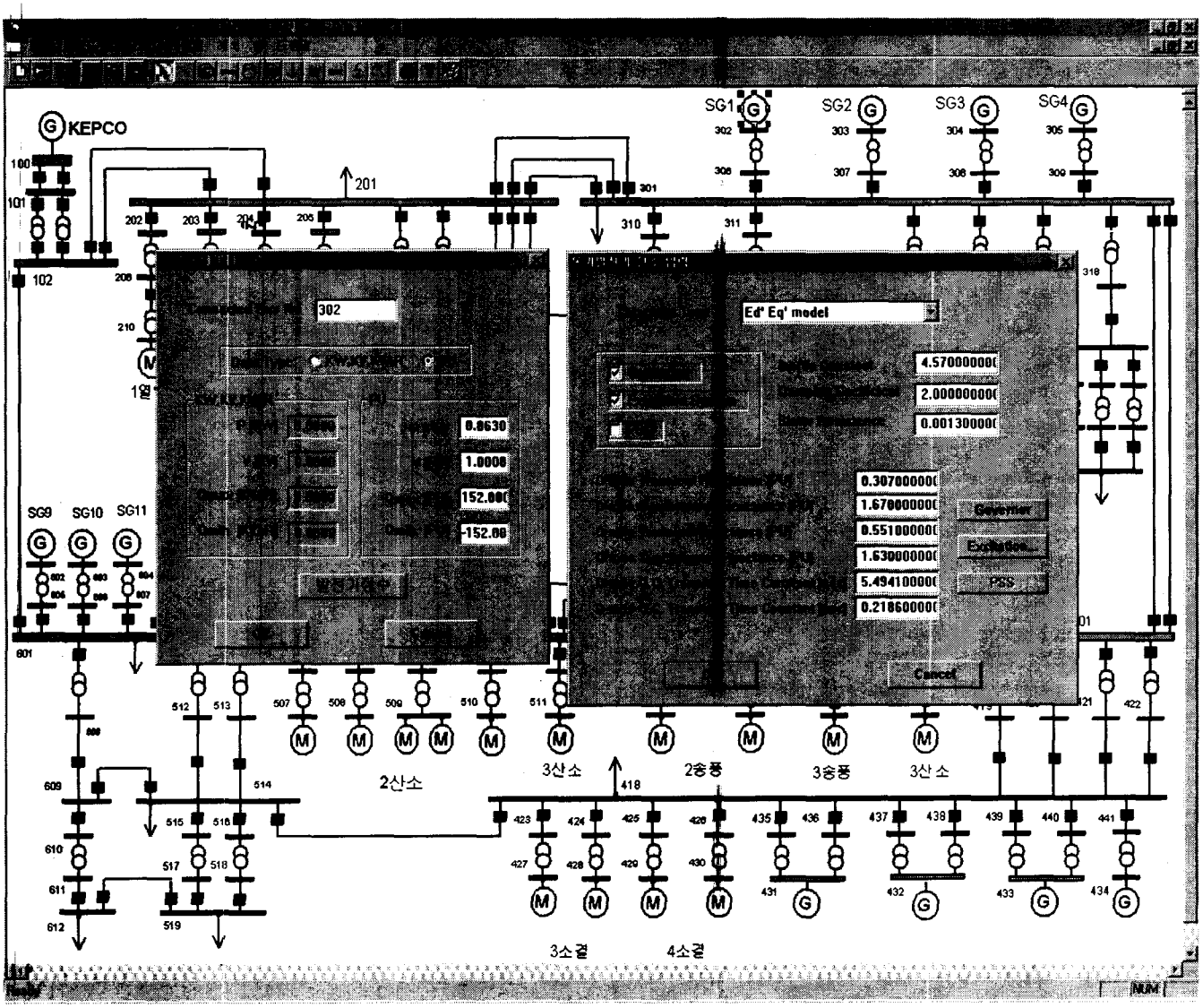

그림 13 광양제철소 전력계통의 단선도

Fig. 13 The power system diagram of Kwang Yang Steel Works.

서 개발된 OOP를 근간으로 하는 윈도우용 C++ 라이브러리로서 윈도우용 프로그램을 개발하기 위한 편리한 클래스들을 제공한다. 메인 메뉴는 크게 파일, 편집, 보기, 그리기, 해석 그리고 창의 메뉴로 구성된다. 파일 메뉴는 새파일, 열기, 닫기, 저장, 다른 이름으로 저장, 프린터 미리보기 그리고 프린터설정으로 구성된다. 편집 메뉴는 실행취소, 잘라내기, 복사, 삭제, 전체선택 그리고 새 객체삽입으로 구성된다. 보기 메뉴는 안내선, 바탕색, 객체들 보기, 툴바 그리고 상태바로 구성된다. 그림 13은 구현된 GUI에서의 광양제철소 전력계통의 단선도를 그린 예를 보여주고 있다. 객체지향기법을 적용한 소프트웨어는 개발자뿐만 아니라 사용자에게도 큰 이점을 주는데, 본 논문에서 구현한 GUI는 계통의 모든 구성요소를 객체단위로 모델링하고 데이터를 관리하도록 시각화 시켰다. 이것은 전력계통 해석용 소프트웨어의 사용자에게 매우 큰 편리성을 제공해 준다. 광양제철소 전력계통[8-9]은 모선의 총수는 123개이고 발전기는 17대이다. 부하모선에는 동기모터부하와 일반부하가 있는데, 변압기를 거쳐서 22kV, 6.6kV의 모선을 거쳐 분산되어 있다.

평상시 운전상태에서는 102번과 601번 모선사이의 선로, 213번과 314번 모선사이의 선로, 418번과 514번 모선사이의 선로 그리고 514번과 608번 모선사이의 선로는 개방되어있다. 화면의 각 부분을 마우스 이벤트 핸들링으로 계통 각 요소의 데이터를 직접 입력가능하고, 데이터 베이스를 통해서도 입력 가능하다. 그림 13에서는 전력조류계산과 안정도 해석을 위한 발전기 데이터 입력창을 보여주고 있다. GUI상에서 시스템의 단선도를 그리고 각 객체의 데이터를 입력하면 시스템의 전체구성과 데이터는 OODB에서 저장되고 관리된다. 예제시스템을 구성하는 많은 요소들을 객체로 구성하고 이들의 연결구조 또한 OODB에서 관리된다. 이들을 OODB에서 관리하면 관계형 DB를 이용하는 것 보다 많은 장점을 지닌다. 즉 데이터베이스의 구조가 시스템을 해석하기 위하여 설계된 클래스의 구조와 동일하여 데이터 저장을 위한 새로운 기능을 구현할 필요가 없어진다. 관계형 DB를 이용하여 그림 13과 같은 시스템을 해석하고 데이터를 관리한다면 구성되어야 될 테이블의 수는 매우 많아지고, 또한 데이터가 여러 장소로 분산된다.

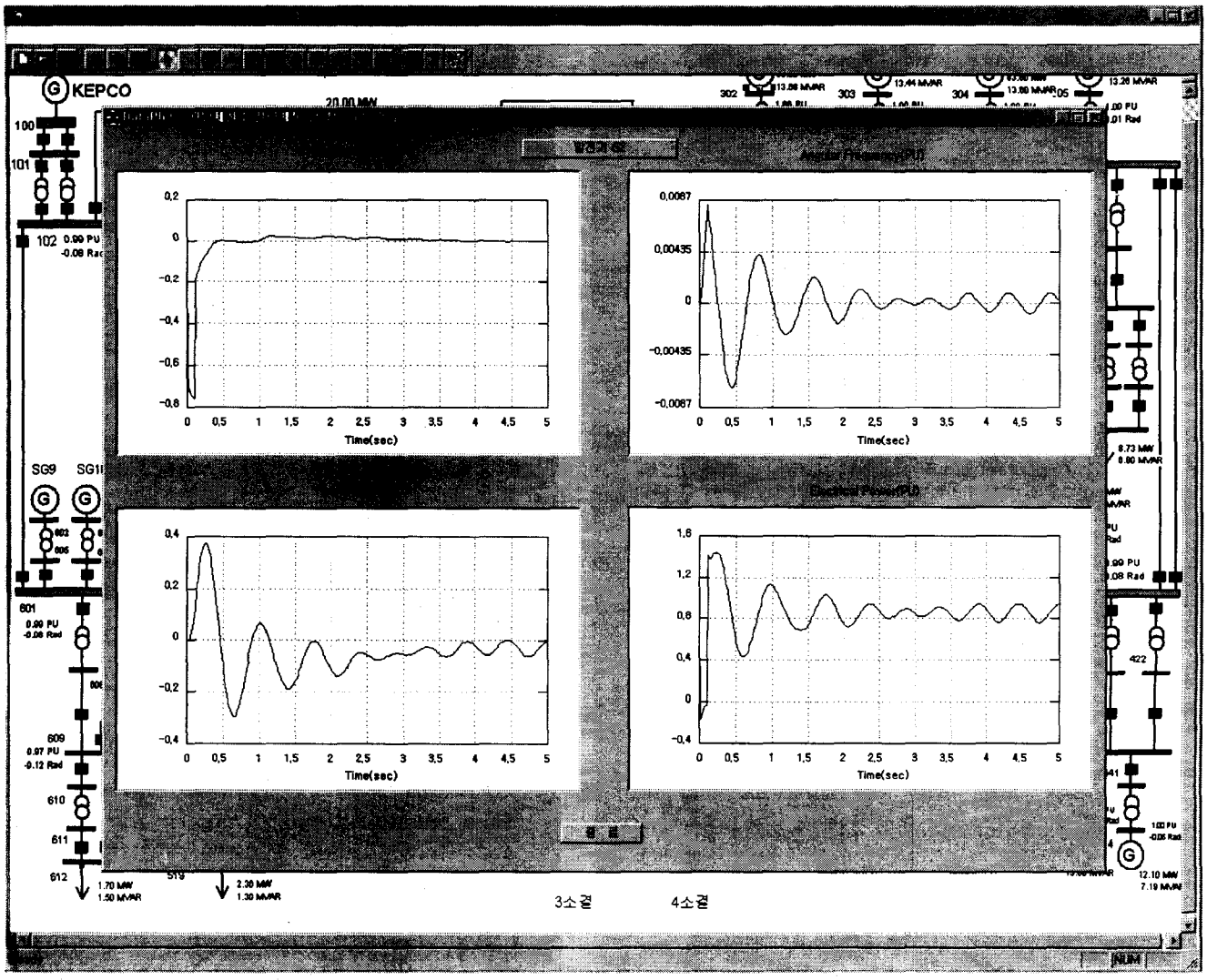

그림 14 광양제철소 전력계통의 시뮬레이션

Fig. 14 The simulation result of Kwang Yang Steel Works.

이는 데이터를 관리하는 효율성의 문제뿐 아니라 데이터 처리속도 면에서도 매우 불리하다. 여러 개의 테이블에 흩어져 있는 데이터를 모아서 하나의 새로운 뷰를 생성하기 위해서 관계형 DB에서는 쿼리을 사용하는데 이는 내부적으로 매우 복잡한 처리과정을 거치게 된다. 하지만 OODB는 데이터를 저장할 때 이미 관련된 모든 데이터가 하나로 집결된 형태로 저장되기 때문에 시스템의 구성요소에 관련된 데이터를 처리하기 위한 작업은 빠르고 쉽다. 그림 14는 전력조류를 수행하고 모선 201과 204사이의 3상 지락사고를 가정하고 안정도를 모의했을 때 사고전의 전력조류와 2번 발전기의 동적 응답을 보여준 것이다. 그림 15는 프로그램의 전체적인 흐름을 나타낸 것이다. GUI에서 사용자가 시스템을 그리고 이를 데이터베이스에 그림6~8의 구조가 그림 9의 지속객체의 형태로 저장된다. 전력조류 계산을 하기 위해서 OODB에 저장된 지속객체를 불러서 컴퓨터 메모리상에서 재구성하고, 사용자가 시스템의 데이터를 변화시키면 변경된 데이터를 DB에 다시저장하고 전력조류를 수행하고 그 결과를 역시 화면상 또는 DB에 저장할 수 있다. 발전기 안정도 해석을 위해서는 OODB에서 시스템 구성정보를 가져와서 각 발전기의 발전시스템을 구성하는 지속객체들을 재구성하여 하나의 발전시스템을 구성한 다음, Ybus를 이용하여 이들을 상호 연결시킨다. GUI에서 모의하고자 하는 선로의 사고를 지정하고, 사고전의 전력조류계산결과를 초기값으로 하여 발전기 안정도를 모의할 수 있다. 각 발전기의 동적인 응답을 편리한 그래픽환경으로 볼 수 있다.

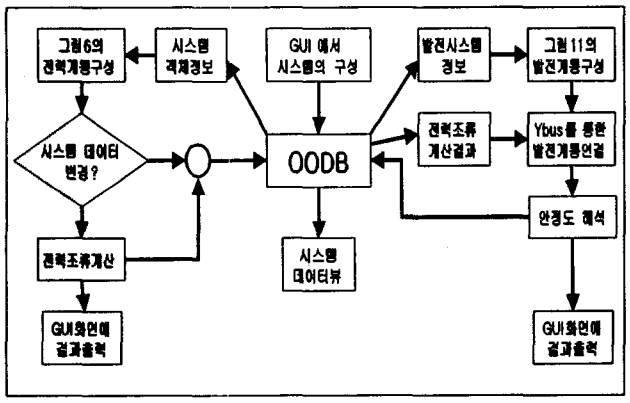

그림 15 프로그램 기능도

Fig. 15 The functional diagram of program.

3. 결 론

OODB는 전력계통과 같이 시스템 자체의 규모가 매우 방대하고 또한 취급하는 데이터의 분류가 매우 다양하고 취급하기 까다로운 시스템의 데이터 취급에 적합하다. 본 논문에서는 얻은 결과는 다음과 같다

- 전력계통 해석용 응용프로그램을 위한 객체지향 데이터베이스를 설계하였다.
- 편리한 그래픽환경의 구현을 통하여 데이터베이스의 접근을 보다 편리하게 하여 데이터베이스의 수정과 유지를 쉽게 만들었다.
- 전력계통의 모델링에 있어서 가장 기본적인 요소를 객체로 모델링하고 이를 연결하여 복잡한 시스템을 구성하는 방식을 구현하여 복잡한 전력계통의 모델링에 매우 큰 유연성을 구현하였다.
- 전력조류계산 프로그램과 안정도해석 프로그램이 OODB를 통하여 연결되어 있으므로 전력계통의 해석이 동적으로 수행된다.

객체지향기법으로 만들어지는 어떠한 응용프로그램이라도 본 논문에서 설명한 방식으로 지속성을 부여하여 OODB에 저장하는 것이 가능하다. GUI을 이용하여 전력계통해석 패키지를 구현할 때 많은 응용프로그램의 데이터를 관리하는 문제에 있어서 OODB는 유연성과 확장성이 매우 뛰어나므로 다른 응용프로그램을 결합하는 것은 매우 쉽다.

감사의 글

본 연구는 산업자원부의 지원에 의하여 기초전력공학공동연구소(02-전-01) 주관으로 수행된 과제임

참 고 문 헌

[1] Elmasri, Navathe, "Fundamentals of database systems", Addison-wesley, 2000

[2] Won Kim, "Introduction to object-oriented databases", MIT Press, 1990.

[3] T. Kawamura, R. Wakizono, T. Tsuchiya, T. Tanaka," Evaluation of Object-Oriented Database for distribution network monitoring system", Proceedings, ISAP '96., International Conference on , 28 Jan.-2 Feb. 1996

[4] Dennis G. Flinn, Roger C. Dugan," A Database for Diverse Power System Simulation Applications", Power Systems, IEEE Transactions on , Volume: 7 Issue: 2 , May 1992

[5] T.M. Atwood, "Object-Oriented Databases: A New Enabling Technology For Electronic Design Applications ", Electro International, 1991 , April 16-18, 1991

[6] Weiyi Meng, A. Kamada, Yu-Hsi Chang," Transformation of relational schemas to object-oriented schemas ",Computer Software and Applications Conference, 1995. COMPSAC 95. Proceedings

[7] Paul M. Anderson, Analysis of Faulted Power System, IEEE Press, 1995

[8] SIEMENS, "Power Transmission and Distribution Power System Planning", FINAL REPORT PART1:

REPORT, 1996

[9] 이희춘, "광양제철소 전력계통분석 모의실험 및 전산기 적용방안 연구", POSCO, 1999

[10] 오명희, "C로 만드는 데이터베이스", 도서출판 삼각형, 1996년

[11] 이상엽, "Visual C++ Programming Bible", 영진출판사, 2003년

저 자 소 개

박 지 호 (朴 志 皓)

1991년 2월 경북대학교 전기공학과 졸업 1996년 8월 동대학원 졸업(석사). 2001년 2월 동대학원 졸업(공박). 2000년 5월~2001년 8월 (주)네모메스 연구원. 2001년 9월 ~ 2003년 2월 강원대 BK21 계약교수. 2003년 3월 ~ 현재 경북대 BK21 계약교수.

Tel : (053) 950-7321
Fax : (053) 950-5505
E-mail : pjh@ee.knu.ac.kr

백 영 식 (白 榮 植)

1950년 7월 8일생. 1974년 서울대 전기공학과 졸업. 1977년 동 대학원 전기공학과 졸업(석사). 1984년 동 대학원 전기공학과 졸업(공박). 1977년 명지대 전기공학과 조교수. 현재 경북대 전자전기공학부 교수.

Tel : 053-940-8802
Fax : 053-950-5505
E-mail : ysbaek@bh.kyungpook.ac.kr