2004년도 대한전기학회 하계학술대회 논문집 2004.7.14-16

객체지향기법을 적용한 배전계통의 조류계산

박 지 호
경북대학교 전자전기컴퓨터공학부

Load Flow in Distribution Systems Using Object-oriented Approach

Ji-Ho Park
Kyungpook National University

Abstract - 전력조류계산은 배전계통관리시스템의 기본적인 기능이다. 객체지향기법은 기존의 절차식 프로그램방식보다 많은 장점이 있다는 것이 증명되었다. 객체지향기법을 적용한 전력조류계산 모델링을 배전계통에 적용한다. 뉴턴-랍손방법에 기초한 객체지향 알고리즘을 구현한다. 뉴턴-랍손법에 기반한 방법은 방사상 배전계통의 전압강하와 전력조류를 계산하기 위하여 배전전력조류방정식을 사용한다.

1. 서 론

객체지향기법이 전력계통의 해석에 적용된 것은 1980년대 후반부터이다. Hakavik과 Holen은 전력계통의 모델링과 스파스행렬의 구성에 객체지향적인 방법을 제시하였고, Gaing과 Lu등은 전력계통의 고장복구에 객체지향적인 접근 방법을 제시하였다. 전력계통의 해석과 그래픽인터페이스를 위한 객체지향기법의 적용 예도 많다. 또한 국내에서도 객체지향기법을 전력계통에 적용한 논문이 계속 발표되고 있다. 본 기법을 샘플시스템에 적용하여 본 결과 기존해법보다 여러 면에서 유용성이 있음이 나타나고 있다.

첫째 프로그램의 유지 보수의 용이함이다. 기존의 프로그램은 일단 완성된 후 시스템의 변화와 같은 외부 조건의 변화에 따라 프로그램의 수정 보완을 해야할 경우 어려움이 있다. 즉 전체적인 기법의 이해를 전제로 해야하므로 수정 보수에 따른 비용 및 시간적인 낭비요소가 있다. 그러나 본 기법은 해당되는 부분만의 수정 교체만으로 가능하다. 둘째로 프로그램 사용의 편리함이다. 시뮬레이션에 객체생성작업으로 그대로 사용 가능하므로 프로그램의 확장성이 뛰어나다. 셋째 해법의 유연성이다. 본 기법은 어떤 시스템이라도 모델링만 가능하면 해를 얻을 수 있다. 넷째 다른 프로그램과의 연결성 및 재사용 가능성이다. 기존 프로그램은 다른 프로그램과의 연결시 프로그램을 전면 수정하는 등의 노력이 필요하나, 본 기법은 쉽게 연결이 가능할 뿐 아니라 기존 개발된 프로그램을 필요부분만 떼어서 사용도 가능하므로 개발의 효율을 높일 수 있다. 최근 국내에서도 전력계통 해석용 소프트웨어의 효율적인 설계와 유지보수 비용을 줄이기 위한 객체지향적인 소프트웨어의 설계가 많이 도입되고 있는 실정이다.

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)의 등장이다. 객체지향데이터베이스의 장점은 첫째, 관계형 데이터베이스의 단점을 극복할 수 있다. 즉 가변길이의 데이터를 저장할 수 있고, 사용자 정의의 데이터를 사용가능하다. 둘째, 객체지향프로그램의 장점을 그대로 적용할 수 있다. 즉 다형성, 상속성, 캡슐화등의 장점을 이용할 수 있고, 객체지향방법으로 구성된 응용프로그램, GUI등과 쉽게 연결이 가능하다. 관계형 데이터베이스의 장점을 그대로 구현 가능하다. 본 논문에서는 객체지향 데이터베이스를 설계하고, 전력계통의 단선도를 그리고 해석할 수 있는 GUI를 구축하고, 전력조류계산을 OODB을 이용하여 수행하였다.

2. 본 론

2.1 전력조류계산을 위한 전력계통의 객체지향적 표현

그림 1은 모선객체의 구성을 나타낸 것이다. 모선객체는 고유의 아이디를 가지고 있으면서 모선에 연결된 가능성이 있는 발전기, 부하, 차단기 그리고 선로객체에 대한 포인터를 가진다. 모선 객체에서는 발전기객체, 부하객체 그리고 차단기 객체를 양방향 포인터 변수로 관리한다. 이런 구조의 장점은 각각의 객체가 독립성을 유지하면서도 상호관계를 유지한다는 것이다. 특정 아이디를 갖는 모선객체를 통하면 현재의 모선에 연결된 모든 정보를 한번에 파악할 수 있다. 모선객체가 갖고 있는 선로객체의 정보는 현 모선에 연결된 선로를 포인터를 이용하여 하나의 연결리스트 첫 객체의 포인터이다. 변수 Next는 시스템에 존재한 모든 모선 객체들을 연결리스트로 구성하기 위한 포인터변수이다.

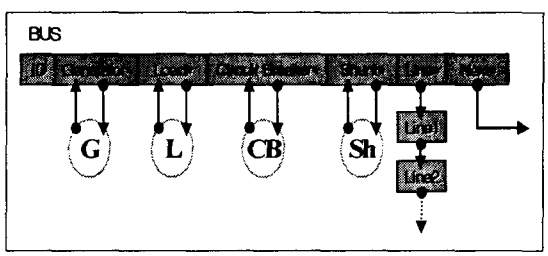

그림 1 모선객체의 구성

그림 2는 선로객체의 구성을 나타낸 것이다. 선로객체는 아이와 모선객체, 변압기 객체를 변수로 가진다. 변수 Bus는 현재의 선로가 연결된 두 개의 모선정보를 포인터로 가진다. 모선과 선로 객체는 양방향 연결구조이다. 변압기가 존재하지 않을 때는 변압기 변수는 널 포인터가 된다. 선로객체도 시스템에 존재하는 다른 선로와 포인터로 연결되기 위한 Next 포인터를 가지고 있다.

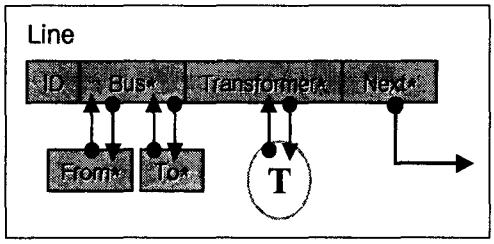

그림 2 선로객체의 구성

그림 3은 발전기, 변압기 그리고 Shunt객체의 구성을 나타낸 것이다. 이들 객체는 모선객체와 선로객체와의 양방향 연결구조를 가진다.

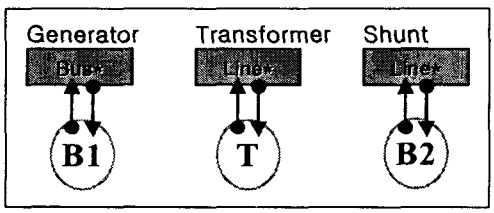

그림 3 나머지 객체들의 구성

전력계통을 해석하기 위한 객체지향 구조의 모델링은 그림 1~3과 같다. 이들 구성은 전력계통을 구성하는 중요 요소들을 모델링하고 이들의 연결구조를 설계한 것이다. 이와 같은 구조는 시스템의 변화에 유연하게 대처할 수 있는 구조이다. 시스템에 새로운 요소를 추가하거나 삭제할 필요가 발생하면 객체연결 구조에만 변화를 주면 변동된 상황에 대처할 수 있다.

시스템의 전체구조를 파악하기 위해서는 그림 4와 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 1의 구조를 지니고 있다. 마찬가지로 선로객체의 경우도 선로객체 연결리스트의 첫 번째 객체를 선로헤어 포인터에 연결하면 순차적으로 연결된 모든 선로객체에 접근하는 것이 가능하다. 이제 두 개의 헤더 포인터만 참조하면 시스템을 구성하는 모든 구성요소에 접근이 가능한 것이다.

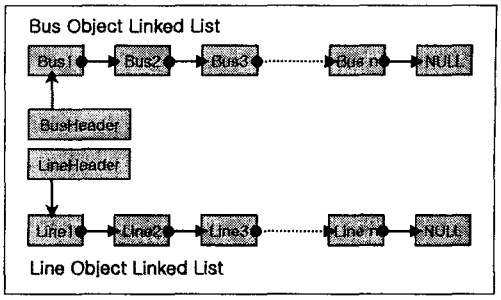

그림 4 시스템구성 객체의 연결구조

그림 5는 앞에서 설명한 객체들을 모두 연결하여 구성되는 시스템의 객체구성도이다. 그림에서 알 수 있듯이 시스템의 모든 구성요소는 개별적으로 단독적인 모델링 구조를 가지고, 이들을 포인터 연결구조를 가지게 함으로써 시스템의 구성을 완성시키는 방식이다. 이러한 방법 자체가 시스템해석을 위한 소프트웨어적인 방법의 유연성을 부여하는 것이다.

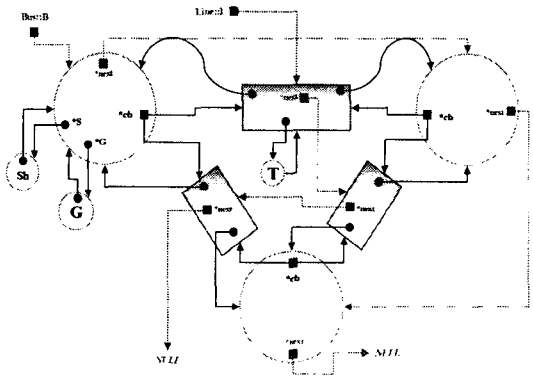

그림 5 시스템 객체의 구성

2.2 전력조류계산을 위한 데이터베이스 설계

전력계통해석 프로그램에 사용되는 데이터베이스는 고주파해석, 과도현상해석, 전력조류계산 그리고 고장계산과 같은 다양한 응용프로그램을 지원할 수 있어야한다. 이를 위하여 데이터베이스에 저장되는 데이터는 보다 기본적이고 저수준의 데이터로 모델링 되어야한다. 고주파해석과 같은 복잡한 응용프로그램에 있어서는 많은 데이터의 형태가 필요하고, 정상, 역상, 영상시퀀스의 단위값으로 단순하게 표현하는 것은 적당하지 않다. 즉 어떤 응용프로그램에서는 단위값 데이터가 필요하고 어떤 응용프로그램은 더 낮은 레벨의 데이터가 필요하다. 따라서 데이터는 많은 응용프로그램에서 다양한 레벨로 유도될 수 있는 최저 레벨의 데이터로 표현되어야 한다. 응용프로그램의 사용자의 관점에서 본 데이터는 하나의 데이터가 하나 이상의 응용프로그램 사용되어야하고, 하나의 프로그램의 결과가 다른 프로그램에서 이용될 수 있어야 한다. 따라서 각각의 응용프로그램은 데이터를 읽고 계산결과를 일관성 있는 포맷으로 저장할 수 있어야 한다.

만약 이러한 계통의 데이터를 텍스트 파일로 저장하여 사용한다면 데이터의 양이 매우 큰 경우는 데이터를 읽는 시간이 상당히 길고, 중복된 데이터가 많아진다. 계통해석의 초창기에는 응용프로그램마다 해석에 필요한 데이터가 특별히 고안된 포맷으로 정의되었고, 초기의 전력계통해석의 기초가 되었다. 이 포맷의 장점은 데이터 접근속도가 빠르다는 것이고 단점은 프로그램에서 사용될 때 하나의 포맷에서 배열형태의 데이터로 사용되기 위해서는 데이터 변환이 있어야 하고, 같은 데이터 파일을 이용하여 다른 응용프로그램을 수행할 수 없다.

설계된 객체가 OODB에 자신의 데이터를 저장할 수 있는 기능을 가지기 위해서는 각 객체가 객체의 지속성을 가져야한다. 사용자의 클래스가 지속성를 가지기 위해서는 사용자 정의 클래스가 지속기반클래스 Persist에서 파생되는 클래스 구조가 되도록 하고, 지속기반 클래스에서 가상함수로 정의되어 있는 ObjRead()와 ObjWrite() 함수에 대한 처리루틴을 만들면 된다. 그림 6은 모선객체와 선로객체에 지속성을 부여하는 예이다.

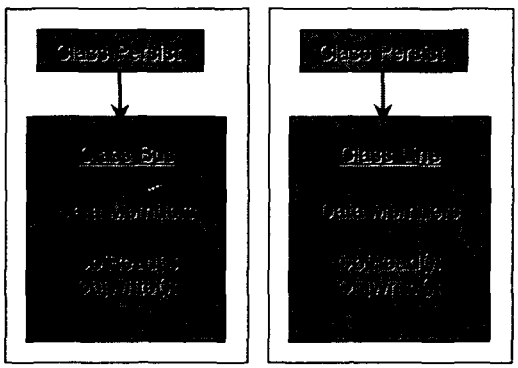

그림 6 객체에 지속성 부여하기

마이크로소프트(MS)사에서 제공하는 MFC(Microsoft Foundation Class)는 윈도우 운영체제를 위해 MS사 자체에서 개발된 OOP를 근간으로 하는 윈도우용 C++ 라이브러리로서 윈도우용 프로그램을 개발하기 위한 편리한 클래스들을 제공한다. 본 논문에서는 MFC를 이용하여 전력계통의 온라인 안정도 해석을 위한 MMI를 구현한다. 주요 기능은 주화면에 전력계통을 직접 그릴 수 있는 기능과 그려진 시스템에 마우스를 이용한 사건처리로 시스템 데이터를 직접 입력하고 전력조류 계산을 행하는 기능이다. 메인 메뉴는 크게 파일, 편집, 보기, 그리기, 해석 그리고 창의 메뉴로 구성된다. 파일 메뉴는 새 파일, 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. 그림7은 프로그램의 전체적인 구성도이다.

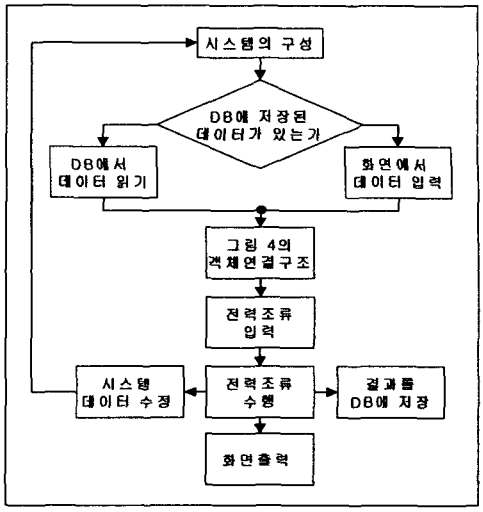

그림 7 프로그램의 기능도

3. 결 론

본 논문에서는 객체지향기법을 적용하여 배전계통의 전력조류계산을 위한 소프트웨어의 설계와 객체지향 데이터베이스의 설계 및 전력조류프로그램과의 연계에 관하여 새로운 방법을 제시하였다. 실제로 객체지향적인 해석방법이 최근에 전력계통분야의 해석에 활발하게 이용되고 있고, 그 유용성이 입증되고 있다. 따라서 본 논문에서는 이러한 경향을 맞추어 전력조류계산을 보다 유연하게 수행할 수 있는 독창적인 방법을 객체지향 데이터베이스와 연계하여 제시하였다.

[참 고 문 헌]

[1] B. Hakavik and A.T. Holen, "Power System Modelling and Sparse Matrix Operations Using Object-oriented Programming," IEEE Trans. on Power Sytems, vol. 9, No. 2, May 1994.

[2] Z. L. Gaing and C. N. Lu, "An Object-Oriented Approach for Implmenting Power System Restoration Package", IEEE Trans. on Power Sytems, vol. 11, No. 1, Feb. 1996.

[3] S. Liu, S.M. Shahidehpour, "An Object-Oriented Power System Graphical Package for Personal Computer Environment", IEEE Summer Power meeting, Sealttle, WA, July, 1992.

[4] M. Foley, A. Bose, W. Michell and A. Faustini, "An Object Based Graphical user Interface for Power System," IEEE Trans. on Power Systems. Vol. 8 No. 1, Feb. 1993, pp. 97-104.

[5] B. Venkatesh and R. Ranjan," Data Strucure for radial distribution system load flow analysis", IEE Power Gener. Transm. Distrib. Vol 150, No. 1, Jan. 2003, pp101-106.

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