대한산업공학회/한국경영과학회 2000 춘계공동학술대회 논문집. 2000년4월21-22일. 경남대학교
Session OB1.4

3-Tier 기반으로의 시스템 변경에 다른 이기종간의 인터페이스 및 개발 방법론에 대한 사례 연구

백태현[1]*, 정귀훈[2], 박주철[3]
[1, 2]현대중공업 산업기술연구소, [3]울산대학교 산업공학과

어플리케이션이 조직 전체로 대형화되고, 대규모의 분산 시스템이 적용됨에 따라 예상치 못한 새로운 문제에 직면하게 되는데, 분산환경에 있어서 어플리케이션의 관리, 보안 관리, 어플리케이션의 부하 증대로 인한 성능 저하, 이기종 DBMS간의 연계 문제 등이 그 것이다. 이러한 문제 해결 방법의 하나로 3계층(3-Tier) 아키텍처를 들 수 있는데, 초기 시스템 투자 비용이 2-Tier에 비해 많이 들고 구현이 상대적으로 어려움에도 불구하고 이기종 DBMS 연결, 다양한 하드웨어 및 개발툴을 선택할 수 있는 유연성, 전사 규모로의 확장성, 동시 병행 개발과 소프트웨어 재활용을 통한 개발 생산성 향상 측면에서의 장점을 제공한다.

본 연구에서는 부서 단위의 관리시스템을 개발하기 위해 초기 2-Tier 개념의 클라이언트/서버 시스템 컨셉에서 출발한 뒤, 사업부 차원의 상위 시스템 아키텍처가 이기종 자원(하드웨어, DBMS, 개발 툴 등)을 통합해 주는 패러다임인 3-Tier로 변경됨에 따라 발생하게된 관련 시스템과의 인터페이스 및 개발 구축 방법론에 대한 사례를 소개한다.

1. 서론

초기의 메인프레임에 의존했던 정보기술 환경에서 '90년대 이후 PC, 네트워크, RDBMS, GUI툴 등의 발달로 클라이언트/서버 환경의 오픈 시스템 개념이 도입되었고 이로 인해 시스템 상호 접속의 용이성, 유리한 가격대 성능비, 프로그램 이식의 편리성 등이 제공되었다. 이를 1세대 클라이언트/서버 컴퓨팅 모델 또는 2 계층(2-Tier) 어플리케이션 모델이라고 한다. 이와 같은 장점에도 불구하고 어플리케이션이 조직 전체로 대형화되고, 대규모의 분산 시스템이 적용됨에 따라 예상치 못한 새로운 문제에 직면하게 되었는데, 분산환경에 있어서 어플리케이션의 관리, 보안 관리, 어플리케이션의 부하 증대로 인한 성능 저하, 이기종 DBMS간의 연계문제 등이 그것이다. 또한, 여러 서버 중 특정 서버에 트랜잭션이 집중됨으로써 서버간에 부하의 차이가 발생하며, 서버 장애시에 타 서버로 백업이 안되고, 비 관계형 DBMS 연결 등과 같은 기본적으로 해결되어야 할 문제들이 제기되었다. 이러한 상황에서 '90년 중반이후 이기종 자원(하드웨어, DBMS, 개발툴 등)을 통합해 주는 새로운 패러다임이 등장하였으며 이것이 바로 3 계층(3-Tier) 또는 n 계층(Multi-tier) 아키텍처이다. 3계층 아키텍처는 초기 시스템 투자 비용이 2계층에 비해 많이 들고 구현이 상대적으로 어려움에도 불구하고 이기종 DBMS 연결, 다양한 하드웨어 및 개발툴을 선택할 수 있는 유연성, 전사 규모로의 확장성, 동시 병행 개발과 소프트웨어 재활용을 통한 개발 생산성 향상 측면에서의 장점을 제공하는 3계층 기반의 미들웨어(Middle-ware)라고 하는 새로운 형태의 소프트웨어 장르를 형성

하게 되었다.

본 연구에서는 부서 단위의 관리시스템을 개발하기 위해 초기 2-Tier 개념의 클라이언트/서버 시스템 컨셉에서 출발한 뒤, 사업부 차원의 상위 시스템 아키텍처가 이기종 자원을 통합해주는 패러다임인 3-Tier로 변경됨에 따라 발생하게된 관련 시스템과의 인터페이스 및 개발 구축 방법론에 대한 사례를 소개한다.

2. 본론

'H'사의 조선사업부는 보다 경쟁력있는 선박의 건조를 위해, 부서 단위로 개발 및 운영되어 오던 관리시스템들을 하나의 관리 형태로 묶는 사업부 단위의 통합화 작업을 수년전부터 시행하고 있다. 하지만 이러한 작업은 많은 인적 자원 투입과 구축 비용이 동반되어야 하며, 통합 운영이 안정화되기 까지는 상당한 시간이 소요된다. 또한 각 단위 부서에서는 효율적인 관리 업무를 위해 부서 단위의 관리시스템이 먼저 개발되어 운영되기를 희망한다. 이러한 경우, 사업부 차원의 통합화 작업이 2년 이후에나 가능하다면, 먼저 단위 부서용 운영 시스템을 개발 및 적용한 뒤 향후 사업부 단위의 통합 작업을 하는 것이 바람직할 것이다. 'H'사의 경우도 먼저 2-Tier 개념에서 출발하여 향후 타 시스템과 3- Tier로 통합화될 수 시스템 개발 방향을 설정하였으며, 시스템 설계를 위해 AS-IS(현재)와 TO-BE(향후) 두 부분으로 나누어 분석하였다. 그림 1은 개발된 시스템(HYPOS; 도장공장 운영시스템)의 구성 모듈을 나타내고 있다.

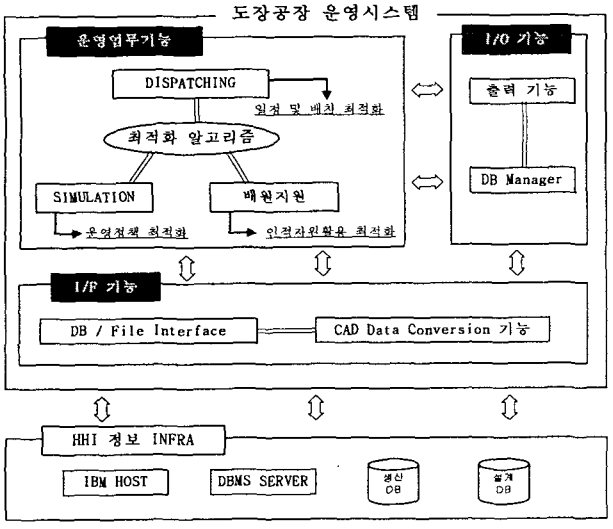

그림 1. HYPOS 구성 모듈

그림 1과 같이 HYPOS 시스템의 연계성과 관련 시스템과의 인터페이스 등을 파악한 뒤, 그림 2와 같이 3-Tier 아키텍처로 개발하기 위한 시스템 설계 작업을 시행하였다.

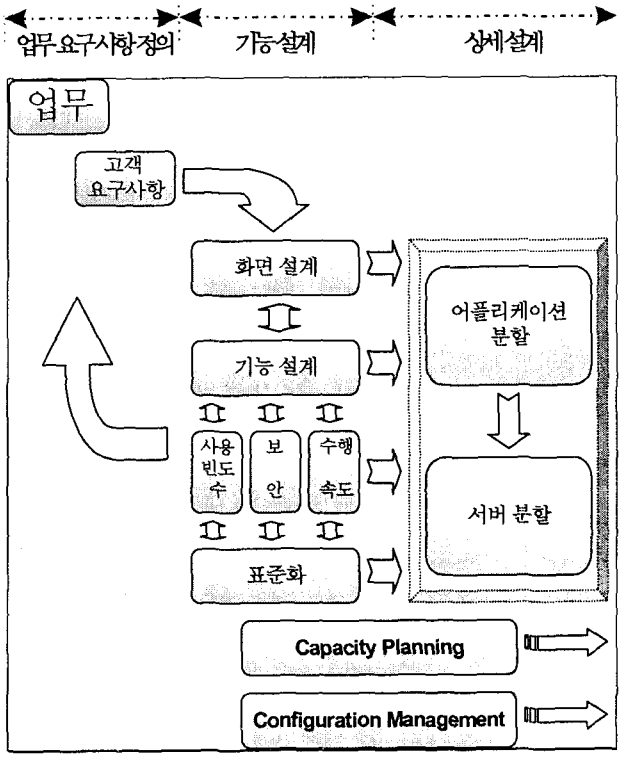

그림 2. 3-Tier 아키텍처 구축을 위한 어플리케이션 설계 작업

또한 분산된 시스템의 어플리케이션간에 통신 서비스를 기반으로 하는 표준기술 하부구조(Infrastructure)를 활용하여 고성능(High performance), 확장성(Scalabil-ity), 무장애(Fault tolerance), 위치 투명성(Location transparency) 및 통합 보안(Security) 기능등을 제공하며, 통합된 분산 서비스들의 집합 분산 컴퓨팅 환경에서 어플리케이션들간에 정보를 교환할 수 있도록 해주는 미들웨어인 Entera를 이용하였는데, 다음과 같은 장점을 가지고 있기 때문이다.

① 다양한 이기종 하드웨어 환경에서 개방형 분산 시스템 구현 가능

② 하드웨어 및 데이터베이스의 위치에 무관한 어플리케이션 접근 가능

③ 네트워크상의 프로토콜에 대한 독립성 유지

④ 기존 시스템 및 외부 시스템과의 연계 및 통합이 용이한 개방형 구조

⑤ 클라이언트와 서버간의 일관되고 표준화된 인터페이스로 정보시스템 표준화 용이

⑥ 시스템 유지 보수 용이

이러한 절차를 거쳐 그림 3과 같이 관련 시스템과의 인터페이스를 고려한 시스템 개발 구성도가 작성되었다.

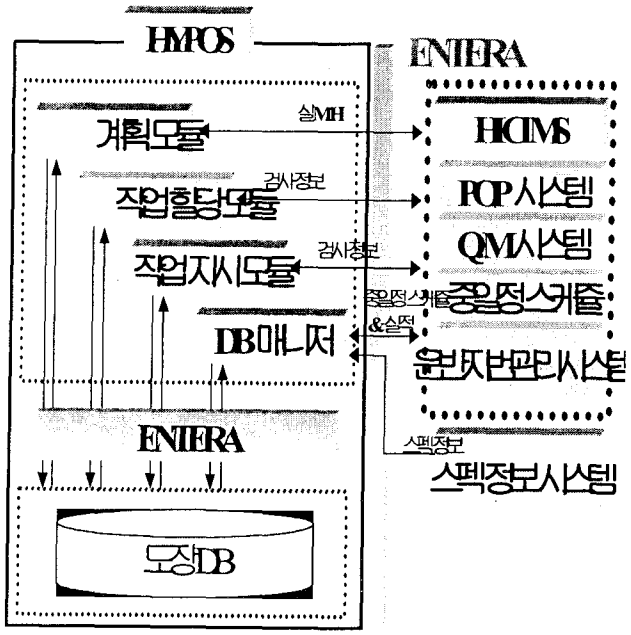

그림 3. 관련 시스템과의 인터페이스를 고려한 HYPOS 전체 개략도

3. 결론

Entera를 이용한 3-Tier 아키텍처 기반의 HYPOS 시스템 개발 사례는 6개월이라는 개발 기간의 제약으로 인해, 초기 시스템 설계 단계와 어플리케이션 개발을 동시공학(Concurrent Engineering)적으로 병행하는 기법을 적용하였다. 이번 개발 사례와 같이 향후 변경될 시스템을 단기간에 개발 하는 경우, 시행착오를 최소화하기 위해서는 초기 설계 단계에서 얼마나 많은 노력과 시간이 투입되어야 하는 가를 절감할 수 있었다. 또한 3-Tier 아키텍처의 성공적인 프로젝트 수행을 위해서는 클라이언트와 서버 구성 요소들을 개별적으로 테스트하며 개발해야 하며, 심지어 다른 구성 요소들이 개발되기 이전이라도 각 구성요소에 대한 개별 테스트가 시행되어야 한다. 이것은 GUI 개발자들은 사용자 인터페이스에, 업무 개발자들은 업무 로직에, 그리고 데이터베이스 개발자들은 데이터베이스에 집중할 수 있도록 많은 도움을 줄 것이다.

참고문헌

1. 이화식, 대용량 데이터베이스 솔루션, 대청, 1996.
2. 조 규익, 데이터베이스 설계, 홍릉과학출판사, 1995.
3. 한더정보시스템, 데이터 모델링, 한더정보시스템(주), 1999.
4. Rajkumar, T. M. and Dawley, Donald L., "Designing and Managing Client/Server DBMSs," Information System Management, Vol. 24, No. 6., pp.11-19, 1994
5. Umar, Distributed Computing & Client/Server Systens, Prentice Hall, 1993.
6. W. H. Inmon and R. Hackathorn, Using the Data Warahouse, John Wiely & sons, Inc., 1994.