韓國精密工學會 2001年度 春季學術大會論文集 pp.703~706 KSPE 01S154

표준 인터페이스를 이용한 컴포넌트 기반의 설계 지원 시스템 통합에 관한 연구

이창근*(연세대 기계공학과), 이수홍(연세대 기계공학과), 방건동(㈜자이오넥스)

A Study for Component-based Integration Framework for Design System using Standard Interface

C. G. Lee(Mechanical Eng. Dept. YSU), S. H. Lee(Mechanical Eng. Dept., YSU), G. D. Pahng(Zionex, Inc)

ABSTRACT

This paper presents a component-based integration framework and its software component architecture for supporting the rapid integration of legacy design supporting systems in the distributed environment. Also, using standard interface, this software component architecture provides flexibility, extensibility and compatibility which ensure software components to be independent of the integration middleware and systems to be integrated.

Key Words : Component-based System Integration, Integrated Design, Distributed Design, Software Component, Standard Interface, XML

1. 서론

1.1 연구의 배경

최근의 제품 개발 환경은 컴퓨터 하드웨어뿐만 아니라 인터넷, 네트워크 기술의 발전으로 인해서, 소수의 서버에 모든 데이터와 작업을 집중시키던 기존의 Server/Client 방식에서 점차 PC 에 그것들을 분산시켜서 작업을 하게하는 분산 환경으로 바뀌고 있다. 즉, 하드웨어적으로는 서로 분산되어 있지만 소프트웨어적으로는 유기적으로 통합되고 있는 추세라고 볼 수 있다. 따라서 CORBA, DCOM, RMI(Remote Method Invocation)와 같이 이를 지원하는 다양한 S/W 기술이 개발되고 있고, 이것들을 적용한 시스템들도 계속 개발되고 있다. 특히 JAVA 와 XML 의 등장은 이러한 네트워크 기반의 분산 환경을 더욱더 가속화 시키고 있는데, 그 이유는 JAVA 는 "Cross-platform Code"를 표방하면서 어떤 시스템 플랫폼에서는 실행이 가능하며, XML 은 "Cross-platform Data"로써 어떤 시스템에서도 데이터 처리가 가능해질 것이기 때문이다.

1.2 연구의 필요성

현재의 상황에서 실질적이고 효율적으로 분산 환경을 구축하기 위해서는 기존의 시스템에 이러한 기능을 추가하는 방식을 적용해야 한다. 그래야지만, 기존의 설계 환경과 데이터를 최대한 유지하면서 설계자들을 자연스럽게 그러한 환경으로 유도시킬 수 있기 때문이다. 물론 이러한 환경을 구축하기위한 비용도 매우 절감시킬 수 있을 것이다. 따라서 본 연구는 이러한 설계 부분의 기존 시스템들을 어떻게 하면 쉽고 빠르게 분산 환경으로 통합할 수 있을까 하는 목적에서 시작되었다.

이것을 구현하기 위해서 본 연구에서는 표준 인터페이스를 이용한 컴포넌트 기반의 시스템 통합에 대한 방법을 제안한다. 즉, 통합 컴포넌트의 구조와 컴포넌트 기반의 통합 시스템 구조를 제시한다.

1.3 기존 연구

설계 지원 시스템을 분산된 환경에서 통합하기 위한 여러 가지 연구들이 최근에 많이 있어왔는데, 크게 분산 객체 표준인 CORBA 를 이용한 시스템 통합[1]과 제품 데이터 표준인 STEP 을 이용한 데이터 통합[2]으로 나눌 수가 있다. 하지만 재사용성을 고려한 컴포넌트 기반의 통합에 관한 체계적인 연구는 MIT CAD Lab.의 DOME 관련 연구[3,4]를 제외하고는 미흡한 실정이다.

2. 컴포넌트 기반 시스템 통합

2.1 개요

시스템 통합 작업은 현재의 복잡한 설계 환경에서 반드시 필요하지만, 현재 대부분의 시스템 통합 작업이 각각의 특정 시스템을 대상으로 해서 이루어지기 때문에 재사용성의 고려가 별로 이루어지지 않고 있는 실정이다. 그렇기 때문에 이제까지의 시스템 통합은 많은 비용과 시간, 노력이 들어가는 작업이었다.

하지만 본 논문에서 논의하고 있는 컴포넌트 기반의 시스템 통합은 한 번 구현하면 그 다음에는 간단한 설정만 해서 다시 사용할 수 있는 방식, 즉 재사용성이 매우 좋기 때문에 적은 비용, 빠른 시간, 적은 인력으로 시스템 통합을 효과적으로 이룰 수 있다. *Fig. 1* 은 기존의 시스템 통합 작업과 컴포넌트 기반 시스템 통합 작업을 비교한 그림이다

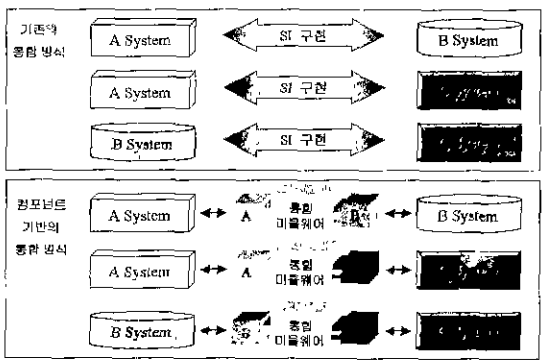

Fig. 1 Comparison of system integration method

컴포넌트 기반의 시스템 통합은 2 가지 측면에서 생각할 수 있는데, 하나는 컴포넌트를 이용해서 시스템 통합을 구현하는 측면이고, 다른 하나는 시스템 통합을 위한 컴포넌트를 개발하는 측면이다. 이 논문에서는 후자에 더 초점이 맞추어져 있다.

컴포넌트를 기반으로 한 시스템 통합을 위해서는 대상이 되는 시스템에 맞는 컴포넌트가 먼저 존재해야 한다. 즉, 기존의 새로운 시스템에 맞는 컴포넌트가 존재한다면 그 컴포넌트를 이용하면 되겠지만, 만약 해당하는 컴포넌트가 존재하지 않는 새로운 시스템을 통합하기 위해서는 새로운 컴포넌트를 개발해야 한다. 따라서 이 논문에서는 이런 상황에서 새로운 시스템을 효율적으로 통합할 수 있는 컴포넌트 개발 방법을 제안하려고 한다.

2.2 통합의 정도

CIMdata 에서는 대표적인 설계 지원 시스템인 CAD 와 PDM 을 통합하는 데 있어서 통합의 정도에 따라 "Integration", "Interface", "Encapsulation" 의 3 가지 수준으로 정의하고 있다. [5] "Integration" 은 시스템간의 양방향 연결이 가능하며, "Interface" 는 단방향으로만 연결되어 있는 상태를 말하고 있다. 마지막으로 "Encapsulation" 은 가장 낮은 단계로써 파일과 그에 해당하는 응용 프로그램 연결 정도의 통합만을 제공한다. 일반적으로 통합의 정도가 높을수록 사용이 편리하고, 작업 효율이 올라가겠지만, 그만큼 많은 노력과 비용이 들어가게 된다.

본 논문에서 구현하고자 하는 통합의 정도는 위의 3 가지 중에서 "Integration" 인데, 컴포넌트를 이용하면 충분히 통합 정도를 높이면서 노력과 비용을 줄일 수 있다

3. 통합 미들웨어 (Integration Middleware)

컴포넌트 기반의 분산 통합 환경을 구축하기 위해서는 컴포넌트가 plug-in 형식으로 연결될 수 있는 통합 미들웨어가 필수적인데, 이것은 다음의 조건을 만족해야 한다.

3.1 개방성과 상호 호환성

기본적으로 시스템의 사양을 충분히 공개하고 확장을 위한 API 를 제공함으로써 이 미들웨어에 장착될 수 있는 컴포넌트를 개발하는데 어려움이 없어야 한다. 그리고 서로 다른 시스템간의 호환성과 이식성이 보장될 필요가 있다.

3.2 독립성

특정 메이커나 플랫폼에 종속되지 않고 사용자 측의 사용 목적과 용도에 적합한 시스템 구축이 가능해야 한다

3.3 분산 환경 지원

기존의 서버/클라이언트 방식이 추구해온 중앙집중식이 아닌, 분산되어 있는 각 시스템에 설계 자원을 배분시켜 이것들을 통합시킬 수 있는 환경을 지원해야 한다.

3.4 보안성과 객체지향성

통합 환경 내에서는 참여자간의 정보교환이 자유로이 가능하나, 각 참여자가 갖고 있는 지적재산에 대한 보호가 필요하다. 이러한 문제는 객체 지향적인 시스템모델의 구현을 통하여 해결이 가능한데, 이러한 시스템 구현은 객체간의 정보교환을 위한 데이터 접근방식을 체계적으로 정의함으로써 각 개체가 갖는 내부정보의 접근을 차단시킬 수 있다.

4. 표준 인터페이스

앞에서 설명한데로 통합 컴포넌트를 이용해서 시스템을 통합한다면 재사용성을 매우 높일 수 있다. 하지만 통합의 대상이 되는 특정 시스템에 맞는 통합 컴포넌트가 존재한다면 이것이 가능하지만, 만약 그에 해당하는 컴포넌트가 존재하지 않다면 이것을 개발해야 하는 문제점이 존재한다. 따라서 본 논문에서는 표준 인터페이스를 이용한 컴포넌트 개발을 제안하는데, 이것은 컴포넌트 구조에서 표준 인터페이스를 이용해서 공통된 부분은 최대한 남겨두고, 최소한의 구현만으로 특정 시스템을 통합시킬 수 있는 방안이라고 할 수 있다. 본 연구에서는 표준 인터페이스로 차세대 데이터 표준으로 각광을 받고 있는 XML 을 이용하였다.

4.1 XML (extensible Markup Language)

XML 은 웹 표준 문서로 자리잡은 HTML 의 확장된 형태이며 W3C(World Wide Web Consortium)에서 1998 년 초반부터 제시되고 있는 자료 교환 표준으로써, 기본적으로 어플리케이션 사이에서 이동되는 메시지와 데이터를 정의하고 표현하는 방법을 제공하는 데이터 모델링 언어라고 할 수 있다. HTML 과 유사하게 태그를 기반으로 데이터나 문서의 내용을 구조화 할 수 있으며 정해진 태그를 통해 문서의 표시 형식을 정의하는 HTML 과 달리 각 태그를 사용자의 편의에 따라 정의하고 사용할 수 있기 때문에 다양한 분야의 정보를 다룰 수 있다. 더욱이 최근에는 XML 만을 이용해서 웹에서의 다양한 객체들간에 메시지 교환과 RPC(Remote Procedure Call)를 구현하기 위한 SOAP(Simple Object Access Protocol) 연구가 한창이다. [6]

4.2 XML 기반 메시지 교환 방식

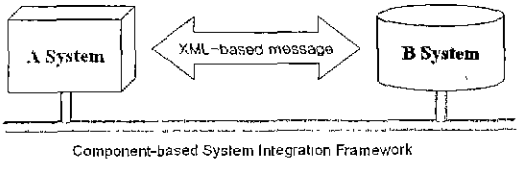

Fig. 2 XML-based message exchange

Fig. 2 에서는 컴포넌트 기반의 시스템 통합 프레임워크에서 XML 기반 메시지 교환 방식을 보여주는데, 이것은 데이터 교환을 위한 요청과 응답을 XML 형태로 전송하는 방식으로써, 원격으로 함수를 부르고 값을 리턴하는 대신, 데이터를 요청하는 시스템이 다른 시스템에게 Request-XML 메시지를 생성해서 전송하고, 이것을 받은 시스템은 파리미터를 받아서 처리한 다음, 결과를 Response-XML 메시지 형태로 돌려주는 방식이다.

이러한 XML 기반 메시지를 이용해서 데이터 교환을 한다면, 실제적으로 시스템간에는 String 값만 주고 받을 수 있으면 된다. 이 String 값은 XML 문서 자체일 수도 있고, 이것을 나타내는 URL 일 수도 있는데, 전자는 상대적으로 간단하고 적은 분량일 때, 후자는 상대적으로 복잡하고 큰 분량일 때 효율적이라고 볼 수 있다. 또, 이러한 메시지 교환 방식은 통합 미들웨어에 관계없이 쉽게 구현이 가능한데, 왜냐하면 미들웨어에서는 XML 문서 자체 또는 그것의 URL 정보를 가지는 String 값의 전달만을 담당하면 되기 때문이다. 따라서 이러한 방식은 어떠한 통합 미들웨이와 통합 대상 시스템과도 유연하게 연결될 수 있는 컴포넌트 개발을 가능하게 한다. 이러한 XML 기반 메시지에 대한 간단한 예가 Fig. 3 에 나타나있다.

시나리오 : PDM 시스템으로부터 Part ID 가 "ZP001"인 부품의 "name", "version", "cost"를 가져옴.

```
Request Message :
    <message type="request">
      <set>
        <part_id type="string">ZP001</part_id>
      </set>
      <get>
        <name type="string"/>
        <version type="string"/>
        <cost type="number"/>
      </get>
    </message>

Response Message :
    <message type="response">
      <set>
        <part_id type="string">ZP001</part_id>
      </set>
      <get>
        <name type="string">cylinder</name>
        <version type="string">0 8 1</version>
        <cost type="number" unit="$">5</cost>
      </get>
    </message>
```

Fig. 3 An example of XML-based message

5. 통합 컴포넌트

5.1 통합 컴포넌트 구조

Fig. 4 는 본 논문에서 제안하는 표준 인터페이스를 이용한 통합 컴포넌트의 전체적인 구조이다. 각 Layer 에 대한 설명은 아래와 같다.

- Adapter : 통합 대상 시스템이 제공하는 API를 이용해서 대상 시스템과 직접 연결되는 부분.
- Interface Layer : 다수의 Adapter 들로부터 넘어온 데이터 또는 넘겨줄 데이터를 동일한 방식으로 처리하기 위해서 표준화하는 부분.
- Application Layer : 각 컴포넌트가 담당하는 logic을 처리하는 부분.
- XML parsing Layer : 통합 미들웨어로부터 넘겨받은 또는 넘겨줄 XML 문서를 해석하고 생성하는 부분.
- Integration Layer : 통합 미들웨어에 직접 연결되는 부분.

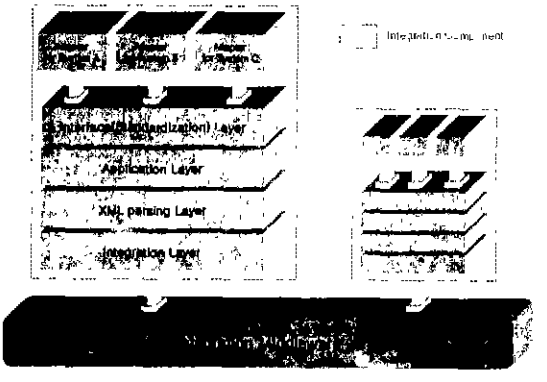

Fig. 4 Architecture of integration component

위의 구조에서 주목할 것은 표준 인터페이스를 이용함으로써 시스템들을 통합하기 위해서는 최소한의 구현인, 대상 시스템에 맞는 Adapter 만을 개발해서 통합을 구현할 수 있다는 것이다.

5.2 통합 컴포넌트 구현

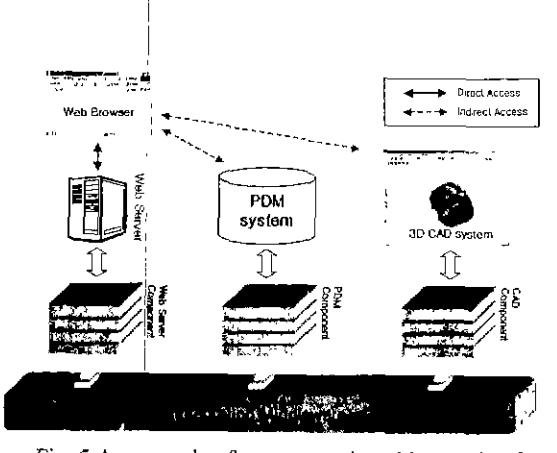

Fig. 5 An example of component-based integration for design systems (Web-based integrated design environment)

Fig. 5 는 컴포넌트 기반의 설계 시스템에 대한 통합의 예를 보여준다. 여기에서는 대표적인 설계 시스템인 PDM 시스템과 CAD 시스템을 웹 서버와 통합함으로써 웹 기반 통합 설계 환경을 구현하였다. 즉, 이 환경에서는 웹브라우저만으로 PDM 과 CAD 시스템에 쉽게 접근이 가능하다. 또 대상 시스템에 맞는 컴포넌트 또는 Adapter 만 존재한다면 어떤 시스템도 쉽고 빠르게 이 환경에 통합될 수 있고, 웹브라우저만으로 그 시스템에 접근이 가능하다.

6. 결론

시스템을 통합하기 위해서 왜 소프트웨어 컴포넌트를 이용해야 하고 그것의 개발에 관한 몇 가지 중요한 사항, 그리고 그것의 구조와 구현 예에 대해서 살펴보았다. 특히 XML 기술의 표준 인터페이스를 이용함으로써 유연성과 확장성을 부여했다는 데 주목할 필요가 있다.

본 연구에서 제안한 컴포넌트 기반의 시스템 통합 프레임워크를 통해서 쉽고 빠르게 기존의 설계 시스템을 통합함으로써, 빠르게 변하는 앞으로의 기업 환경에 대응하면서 통합 설계 환경을 더욱 효율적이고 경제적으로 구축할 수 있을 것으로 기대된다.

참고문헌

1. Jinmin Z., Junjun W., Jihong L, Guodong J., Jibin W., "Corba based System Integration of Product Development Process Management for Concurrent Engineering", Proceedings of DETC'00 ASME 2000 Design Engineering Technical Conference and *Information in Engineering Conference Baltimore*, Maryland, CIE-14612, 2000
2. 오유천, 한순흥 "CAD 와 PDM 시스템 간에 STEP 제품 구조 정보의 교환" 한국 CAD/CAM 학회 논문집, 제 5 권, 제 3 호, pp.215-223, 2000
3. Pahng, G.F. "Modeling and Evaluation of Design Problems in a Network-Centric Environment", 박사 학위논문, MIT, 1998
4. Abrahamson, S.,Wallace, D., Senin, N. and Sferro, P., "Integrated Design in a Service Marketplace", Computer-aided Design, volume 32, no 2, pp. 97-107, 2000
5. *CIMData*, "Product Data Management: The Definition", http://www.cimdata.com. 1997
6. "Apache XML Project", http://xml.apache.org