

분산 JavaBeans 컴포넌트 통합을 위한 연결자 모델링

정성옥⁰, 정찬주^{*}, 김재석^{**}

광주여자대학교 컴퓨터과학과, 동강대학 인터넷정보통신과, 조선대학교 컴퓨터공학과
sojung@mail.kwu.ac.kr⁰, cjjung@dongkang.ac.kr, freeone@daum.net

Modeling of Connector for Distributed JavaBeans Component Intergration

Sung-Ok Jung⁰, Chan-Ju Jung^{*}, Jae-Seog Kim^{**}

Dept. of Computer Science, Kwangju Women's University⁰
Dept. of Internet Communication, DongKang College^{*}
Dept. of Computer Engineering, Chosun University^{**}

요 약

현재의 소프트웨어 아키텍처에 관한 연구는 컴포넌트 집합과 같은 소프트웨어를 구성하는 객체 또는 컴포넌트의 상호동작 및 관련성을 보다 효과적으로 연결할 수 있는 다양한 기법이 제시되고 있다. 본 논문에서는 JavaBeans에 기반을 둔 분산 시스템 환경에서 객체와 객체간에 관련성을 모델링하기 위해 컴포넌트, 연결자(connector) 및 컴포넌트 스키마로 구성된 구조화된 모델을 제시한다

1. 서 론

객체지향 소프트웨어 시스템은 상호 연관되어 있는 수많은 객체의 집합으로 구성되며 객체지향 분석 및 설계 단계에서 객체에 대한 정보는 일반적으로 클래스 및 클래스간의 관계를 정형화된 기법을 적용하여 명확하게 기술한다[1]. OMG의 CORBA에 기반을 둔 분산 객체 시스템에서도 서로 연관되고 상호 동작을 수행하는 수많은 객체의 집합으로 구성되어 있다. 그러나 CORBA에 기반을 둔 시스템에서는 객체간에 모델을 설정하고 설계를 하는데 객체지향 기술에서 사용하였던 클래스에 기반을 둔 모델링 기술을 적용하는데 제약을 가진다. 따라서 추상화와 CORBA 객체간의 관련성을 일반적인 클래스 관련성을 이용하는 새로운 기법이 적용되어야 한다.

본 논문에서는 JavaBeans에 기반을 둔 분산시스템 환경에서 객체와 객체간의 관련성을 모델링 하기 위해 컴포넌트, 연결자(connector) 및 컴포넌트 스키마로 구성된 구조화된 모델을 제시한다.

2. 관련 연구

2.1 분산컴포넌트 기술

최근까지 객체지향 프로그래밍 기술이 인기를 누리왔지만 재사용 가능한 소프트웨어 컴포넌트를 조합하여 새

로운 어플리케이션을 만드는 방법이 사용자의 요구에 따른 어플리케이션을 개발하는데 생산성이 높은 것으로 인식되고 있다 마이크로 소프트의 Visual Basic은 윈도우 기반에서 어플리케이션을 개발하는데 사용되었으며, Borland의 Delphi는 강력한 데이터 액세스 컴포넌트와 확장 기능을 추가함으로써 여러 가지 방법을 보완하였다. SUN의 JavaBeans는 컴포넌트 기반의 소프트웨어 패러다임을 플랫폼이 독립적이고 네트워크를 인식할 수 있는 컴퓨팅으로 확장시키고 있다. 컴포넌트 기반의 소프트웨어는 특정 아키텍처와 관련된 API의 집합으로 구성되는 소프트웨어 모델을 기술하는데 사용된다[1][5].

2.2 CORBA

CORBA는 OMG에서 정의한 분산객체 컴퓨팅 환경의 표준으로서 이질적인 분산 환경에서 어플리케이션을 구현하는데 필요한 프레임워크, 객체서비스, 객체클래스 라이브러리등이 정의되어 있다. OMG는 CORBA외에 객체의 기본적인 조작과 복합문서, 시스템 관리등 일반적인 어플리케이션의 공통적인 기능인 인터페이스, 비즈니스 객체, 특정분야의 고유 객체등을 가지고 있으며 또 다른 어플리케이션을 포함한 OMA로서 이루어졌다. ORB의 기능은 클라이언트로부터 요구를 전달받은 후 클라이언트의 요구를 실질적으로 구현해줄 객체를 찾아서 매개변수를 전달하고 해당되는 메소드를 호출하며 다시 결과를

반환해 주는 역할을 한다. CORBA는 ORB 구성 요소에 대한 인터페이스를 제공하고 인터페이스에 대한 구현은 실제로 ORB를 제작하는 개발자에 맡긴다[2][3]. CORBA ORB는 그림 1과 같은 구조를 가지며 ORB 구조에서 클라이언트는 연산을 수행하고자하는 요소이고 객체의 구현은 실제로 객체를 구현하는 코드와 데이터로 구성된다[3]. 클라이언트 IDL 스타트는 서버 객체에 대해 정적인 인터페이스를 제공하며 클라이언트가 서버상의 서비스를 호출하는 방법을 정의한다[4].

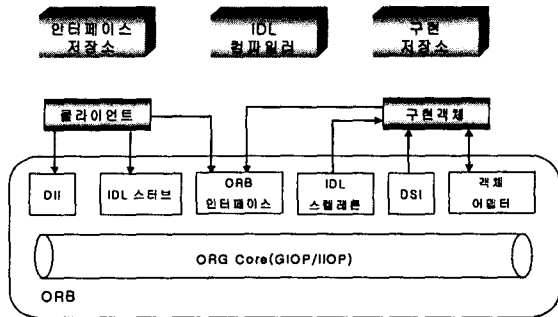


그림 1 CORBA ORB 구조

2.3 JavaBeans

JavaBeans는 소프트웨어 컴포넌트 개발을 위한 표준 API이다. JavaBeans 구조의 가장 큰 목적은 플랫폼이 독립적인 컴포넌트 구조를 제공하는데 있으며 JavaBeans 컴포넌트 모델은 Java에서 지원하는 다양한 클래스에 기반을 두고 있으며 JavaBeans 컴포넌트 모델을 사용하기 위해서는 클래스가 도구화되어야 하고 재사용 등의 특성을 활용하기 위해서는 몇 가지 규칙을 따라야 한다. 사용자 입장에서 보면 JavaBeans는 그림 2와 같이 외부에 자신의 Method, Properties, 그리고 생성하는 Event만을 보여주며 BeanInfo 클래스를 통해서 자기 자신에 대한 정보를 보여준다.

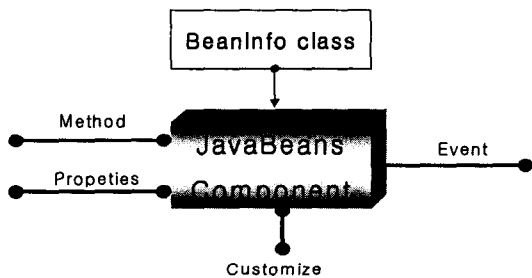


그림 2 JavaBeans 컴포넌트

3. 컴포넌트 통합을 위한 연결자 모델링

본 논문에서는 JavaBeans 환경에서 컴포넌트 통합을 위한 연결자 모델링은 소프트웨어 아키텍처, 아키텍처 기술언어, 객체지향 모델링기술, 분산객체의 처리를 위한 모델에 기반을 둔다. 이를 위해 JavaBeans에 기반을 둔 소프트웨어 아키텍처를 모델링하며 그림 3과 같이 구성된 컴포넌트, 연결자 및 컴포넌트 스키마로 구성된 프레임워크로 구성된다. 컴포넌트는 JavaBeans의 환경에 존재하는 분산 객체의 집약적인 기술로서 인터페이스의 집합인 명세부분, 내부적인 객체지향 스키마인 표현부분, 외부적인 객체지향 스키마인 표현-지도도를 포함하고 있다. 연결자는 컴포넌트의 논리적인 관련성을 갖는 통합과 객체의 동적인 관계를 나타내는 컴포넌트의 상호작용을 추상화 하며 연결자는 Roles, Role Interface, Interaction protocol의 기술을 포함하고 있다.

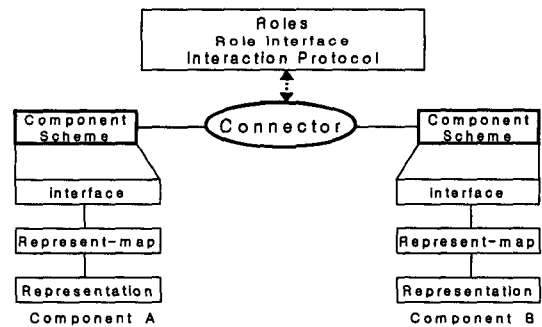


그림 3 컴포넌트 통합 연결자

4. 컴포넌트 통합을 위한 연결자 구현

4.1 이벤트통보 연결자 구현

객체의 동작이 동기화 된 수행의 결과로 메소드 호출이 발행되는 표준화된 CORBA 객체 통신 모델링 방법에 반대로, 이벤트는 다양한 객체사이에서 비동기화 된 방법으로 객체간에 통신을 한다. OMG의 이벤트 서비스는 표준화된 인터페이스와 publish/subscribe에 기반을 둔 이벤트-참여를 위한 객체 상호 작용 모델을 지정한다.

하나 이상의 객체들은 이벤트 데이터를 발생시키고 이벤트 제공자(supplier)로서 동작한다. 또한 수많은 다른 객체들은 이벤트 데이터를 받고 이벤트 소비자로서 동작한다. 본 논문에서 일반적인 이벤트 통신 방법과 반대가 되는 모든 이벤트 공급자는 동적으로 이벤트를 전송하는 push 형태 또는 동적으로 이벤트 발생에 대한 질의

를 하는 pull 형태를 갖는 이벤트 채널(channel)로서 포괄 이벤트(generic event)통신 방법으로 모델링된다.

연결자 EventNotification의 명세서에는 이벤트 통보를 집약적으로 수행하기 위해 객체사이의 상호 의존성을 기술하며 객체의 기능, 기능 인터페이스, 상호 작용 프로토콜에 의해 구조화된 형태를 갖게 된다. 기능 부분은 컴포넌트를 구성하기 위해 협력하는 참여자를 말한다.

이러한 객체들은 확장된 수준에서 특정 컴포넌트에 의해 동작되어지며 각각의 컴포넌트는 서로 같거나 다른 연결자에 의해 하나 이상의 기능이 수행될 수 있으며 각각의 기능에 대해 하나 또는 그 이상의 기능 인터페이스가 기술될 수 있다.

4.2 기능 인터페이스

기능 인터페이스 부분에서는 서로 협력관계를 유지하는 컴포넌트가 지원해야할 서비스 명세를 기술한다. 기능 인터페이스 부분은 OMG에서 지원하는 일반적인 이벤트 명세서 정의된 IDL 인터페이스 기준을 따른다.

4.3 상호작용 프로토콜

협력관계를 유지하는데 있어서 연결자 내부의 실제적인 동작은 상호작용 프로토콜을 통해 지정되어진다. 인터페이스 프로토콜은 선행조건(precondition) 및 후행조건(postcondition)을 고려하여 협력 관계에 대한 동작과 동작의 순서를 기술한다. 그림 9에서와 같이 이벤트 데이터 교환(exchanging event data)을 위해서는 4개의 동작으로 표현된다. 인터페이스 프로토콜을 기술할 때 오름차순 순서를 갖는 숫자는 수행되는 동작의 순서를 의미하며, 알파벳 문자의 순서는 동시에 수행되는 병행 동작을 의미한다.

V. 결 론

본 논문은 기존의 객체 및 컴포넌트를 위한 연결 기법을 기반으로 하여 연결자와 컴포넌트를 원활하게 연결하기 위한 컴포넌트 스키마를 도입하였으며, 이를 CORBA 환경에서 분산 JavaBeans 컴포넌트간에 연결자를 사용하여 기존의 동기화 방식에 추가적으로 비동기화 방식으로 컴포넌트 연결을 위한 정보를 주고받을 수 있도록 하였다.

향후에 연구 과제로는 보다 보완된 상용화된 컴포넌트의 통합 및 연결이 이루어질 수 있도록 하기 위하여, 다양한 응용 분야에 존재하는 객체 및 컴포넌트에 대한 연

결 및 통합이 진행되어야 한다.

참고문헌

- [1] Joao Pedro Sousa and David Garlan, "Formal modeling of the Enterprise JavabBeans™ Component Integration Framework", FM'99 : World Congress on Formal Method, Sep., 1999.
- [2] Robert Orfall, Dan Harkey, Client/Server programming with JAVA and CORBA, Guide, John Wiley & sons Inc., 1997.
- [3] James Rumbaugh, Michael Blaha William Premerlani, Frederick Eddy, William Logensen, Object-oriented Modeling and Design, Prentice-Hall International Edition, 1991.
- [4] Mary Campione, kathy Walrath, The Java Tutorial: Object-oriented Programming Addison Wesel, 1996.
- [5] Robert Orfall, Dan Harkey,, Jeri Edward, The Essential Distributed Object Survival Guide, John Wiley & Sons Inc., 1996.