분산환경에서의 비즈니스 정보 시스템 아키텍처 분류

이혜선[O] 이은배 고현희 박재년
숙명여자대학교
{hslee[O], eunbae, hhkoh, jnpark}@sookmyung.ac.kr

Distributed Business Information System Architectures

HyeSeon Lee[O] Eunbae Lee, HyonHee Koh, JaeNyon Park
Sookmyung Womens' University

요 약

소프트웨어 아키텍처는 소프트웨어 시스템 구축시 설계단계의 첫번째 부분으로 소프트웨어 개발시 중요시되고 있다. 아키텍처 설계시는 비즈니스 목표나 품질요구사항, 도메인의 특징과 개발 환경등 여러가지 사항을 고려해야하고 설계된 아키텍처를 검증할 수 있어야 한다. 그러나 성숙한 아키텍트가 아닌 경우 개발하고자 하는 시스템의 아키텍처 설계나 검증이 어렵다. 따라서 본 논문에서는 비즈니스 정보시스템에서 많이 사용되고 있는 아키텍처를 도출, 분류해보고, 품질 속성 만족 여부를 분석함으로써 비즈니스 정보 시스템 구축시 아키텍트들에게 아키텍처 참조 모델을 제공하고자 한다.

1. 서 론

하나의 시스템을 설계하기위해서는 여러 아키텍처 스타일들이 복합적으로 결합하여 시스템의 전체적인 아키텍처를 구성하게 된다.
이때 다양한 아키텍처중 어떤 아키텍처를 선택 할 것인가는 완성될 시스템이 어떤 품질속성을 더 우선적으로 만족시켜야 하는지에 따라 달라지게 된다. 즉 아키텍처 선정 시는 그 시스템의 요구사항에 따라 아키텍처를 검증하고 품질요구사항을 포함하는 요구사항을 만족할 수 있는 아키텍처를 선정해야 한다.
그러나 성숙한 아키텍트가 아닌 경우 선정한 아키텍처가 품질요구사항을 만족시키는지 여부를 평가하기가 모호하고, 평가를 한다고 하더라도, 그 평가 결과에 대한 객관적인 증명을 하기가 어렵다. 따라서 현재 많이 사용되고 있는 아키텍처들을 도출, 분류해보고, 사용되고 있는 과정에 어떤 품질 속성을 만족하고 만족하지 못했는지를 조사해 보는 것이 향후 유사한 시스템 개발시 아키텍처 선정에 많은 도움을 줄 수 있을 것이다.
본 논문에서는 분산 환경에서의 비즈니스 정보 시스템의 일반적인 아키텍처 패턴들을 도출, 분류해보고, 요구되는 품질 속성 만족 여부를 조사해 봄으로써 향후 아키텍트 들이 유사시스템 설계시 참조 모델을 제공하고자 한다.

2. 비즈니스 정보시스템의 특성

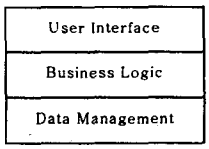
비즈니스 정보시스템(Business Information System)이란 자원, 규칙, 비즈니스 프로세스가 반영되는 비즈니스에서의 실질적인 활동을 지원하는 비즈니스 시스템에 대용량의 복잡한 구성관계를 가진 데이터를 핸들링하여 정보를 저장, 조회, 전달, 변형, 통지 기능을 하는 정보시스템을 결합한 시스템을 말한다.
이 시스템은 분산된 비즈니스 프로세스들을 지원하며, 기업에서의 정보공유와 불필요한 통신, 중복된 데이터의 제거에 의해서 효율적으로 증진될 수 있다. 또한 비즈니스 정보시스템은 다수의 사용자가 대량의 데이터를 동시에 작업할 수 있으며, OLAP, 오프라인 프로세싱, 배치처리 같은 다양한 형태의 데이터 처리기능을 지원해 준다[1].
대부분의 비즈니스 정보시스템은 비즈니스 목적을 수행하고 데이터베이스를 처리할 수 있는 3-레이어드 아키텍처 구조를 따르고 있다.

3. 정보시스템의 레이어드 아키텍처

분산환경에서의 정보시스템은 성능 향상을 위해 기존의 2-레이어드 아키텍처의 중간에 서버 계층 레이어가 추가된 3-레이어드 아키텍처가 일반적인 클라이언트/서버 아키텍처의 형태로 구현되고 있다. 이 중간계층 레이어는 비즈니스 로직과 규칙이 실행되는 프로세스 관리를 갖고 있다[2][3].
다음에서 비즈니스 정보시스템에서 일반적으로 사용되는 3-레이어드 아키텍처에 대해 설명한다.

3.1 3-레이어드 아키텍처

[그림1]에서 보는 바와 같이 3-레이어드 아키텍처는 세개의 레이어로 구성된다.
User Interface Layer는 가벼운 기능을 갖는 클라이언트로써 화면 디스플레이와 데이터 엔트리를 위한 프리젠테이션 작업을 처리하는 레이어다. Business Logic Layer는 서버에서 실행되는 어플리케이션 로직으로 특정한 도메인 비즈니스 작업을 수행한다. 이 레이어의 컴포넌트들은 잘 알려진 컴포넌트 인터페이스를 제공해야 하며, 비즈니스 로직은 재사용되어질 수 있다.
Data Management Layer는 데이터베이스 액세스 레이어로 어플리케이션 로직을 데이터베이스로 연결하는 레이어이다. 이 3-레이어드 아키텍처는 J2EE나 .NET같은 상용 개발 플랫폼에서 사용되고 있는 아키텍처에 기반이 되고 있다.



[그림 1] 3-레이어드 아키텍처

3.2 J2EE에서의 레이어드 아키텍처

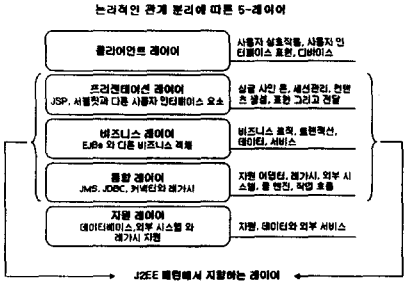
[그림 2]에서는 각각의 레이어에 대한 사항을 보여 주고 있다. 그리고 각 레이어에 대한 설명은 다음과 같다[4][5].

(1) 클라이언트 레이어

시스템이나 응용프로그램에 접근하는 모든 디바이스나 시스템 클라이언트를 나타낸다.

(2) 프리젠테이션 레이어

시스템에 접근하는 클라이언트들에게 서비스하기 위해 요구되는 프리젠테이션 로직이 캡슐화 되어있다.



[그림 2] 레이어 접근법

(3) 비즈니스 레이어

응용프로그램 클라이언트에 의해 요구되는 비즈니스 서비스를 제공하는 계층이다.

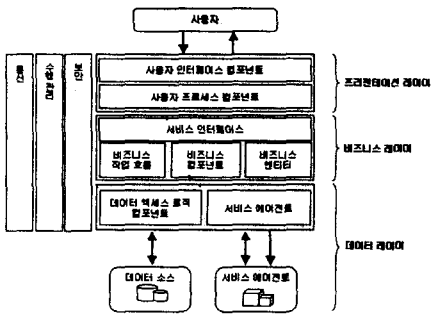
(4) 통합 레이어

데이터 저장 장치나 레가시 응용프로그램 같은 시스템과 외부 자원과 함께 통신할 수 있도록 하는 책임을 갖는 레이어이다.

(5) 자원 레이어

이 계층에서는 비즈니스 데이터와 메인프레임과 레가시 시스템들, B2B 통합시스템, 그리고 신용 카드 인증과 같은 외부 자원들을 포함한다.

3.3 .NET에서의 레이어드 아키텍처



[그림 3] .NET 응용 아키텍처

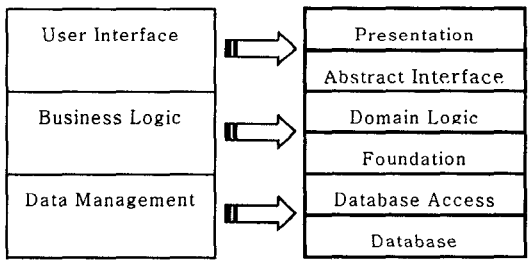
.NET에서의 응용 아키텍쳐 안내에서는 다음 [그림 3] 같이 제시하고 있다[6][7][8].

이는 사용자 인터페이스 컴포넌트, 사용자 프로세스 컴포넌트, 비즈니스 작업 흐름 컴포넌트, 비즈니스 컴포넌트, 서비스 에이전트, 서비스 인터페이스, 데이터 엑세스 로직 컴포넌트, 비즈니스 엔티티 컴포넌트, 보안, 프리젠테이션, 비즈니스, 데이터 레이어 전반에 걸쳐 적용이 되어져야 하는 컴포넌트들인 수행 관리 및 통신 컴포넌트 구성되어 있다.

3.4 세분화된 레이어드 아키텍처

앞에서 정보시스템에서 일반적으로 사용하는 3-레이어드 아키텍처와 이를 기반으로 한 J2EE와 .NET의 레이어드 아키텍처를 살펴보았다. 비즈니스 정보시스템은 기본적으로 레이어드 아키텍처를 채택하고 있지만, 비즈니스 요구사항이나, 전략, 그리고 품질 속성 요구사항에 따라 각 계층의 분류나 세분화 정도가 달라질 수 있다. 여러 세분화된 레이어드 아키텍처를 정리하여 보면 기본적인 3-레이어드 아키텍처를 확장한 비즈니스 정보시스템의 아키텍처는 [그림 4]와 같이 여섯개의 레이어로 세분화할 수 있다.

먼저 Presentation Layer는 클라이언트에서 작업할 수 있는 화면 프리젠테이션을 포함하고 있으며, Abstract Interface Layer는 Presentation Layer와 Domain Logic Layer 사이의 인터페이스 기능을 한다. Domain Logic Layer는 특정 도메인의 비즈니스 로직을 처리하며, Foundation Layer는 공통적으로 재사용될 수 있는 비즈니스 로직을 처리한다. Database Access Layer는 미들웨어를 통해 데이터베이스를 엑세스할 수 있으며, Database Layer는 물리적으로 저장된 데이터베이스를 포함하고 있다.



[그림 4] 세분화된 레이어드 아키텍처

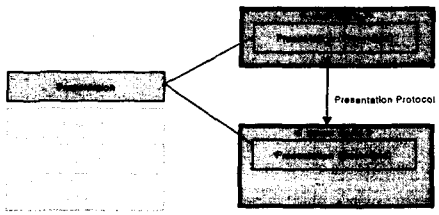
4. 분산 유형별 아키텍쳐

분산환경에서의 비즈니스 정보시스템은 수많은 사용자가 다량의 데이터를 가지고 동시에 작업을 할 수 있기 때문에 비즈니스 프로세스를 어떻게 분산시키냐에 따라 시스템의 성능과 효율성에 영향을 미칠 수 있다[9].

위에서 살펴본 6개의 계층에 대한 분산유형에 따라 다양한 형태의 시스템 분산 패턴으로 구분할 수 있다. 이 분산 패턴은 6계층의 레이어드 아키텍처를 구성하는 각각의 레이어드를 어떻게 분리하여 분산시키는 가에 따라 다음의 다섯가지 패턴으로 구분된다.

(1) 분산 프리젠테이션 패턴

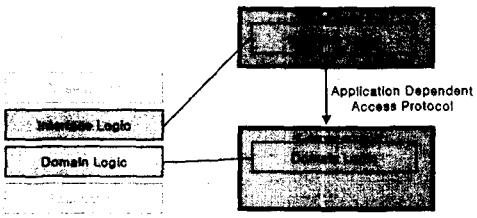
프리젠테이션 컴포넌트내에서 시스템을 분할한 패턴으로, 프리젠테이션 컴포넌트는 분산 단위로 패키지되고 다른 응용계층에서 패키지된 다른 프리젠테이션과 구분되어 처리된다. 호스트-터미널 스타일이라고도 한다.



[그림 5] 분산 프리젠테이션 패턴

(2) 원격 사용자 인터페이스 패턴

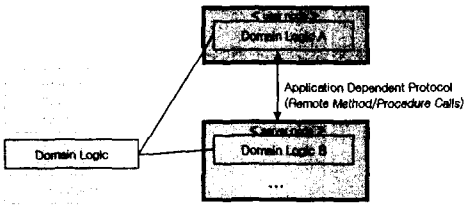
모든 사용자 인터페이스는 분산의 단위가 되어 서버상의 도메인 로직의 클라이언트로서 행동한다. 이 패턴에서 도메인 로직 액세스 프로토콜은 미들웨어로 구현된다.



[그림 6] 원격 사용자 인터페이스 패턴

(3) 분산 도메인 로직 패턴

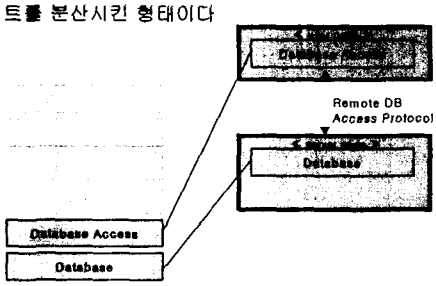
도메인 로직을 규칙에 의해 클라이언트와 서버 노드에 자유롭게 분산시킨 형태로 도메인 로직간에는 RPC로 연결된다



[그림 7] 분산 도메인 로직 패턴

(4) 원격 데이터베이스 패턴

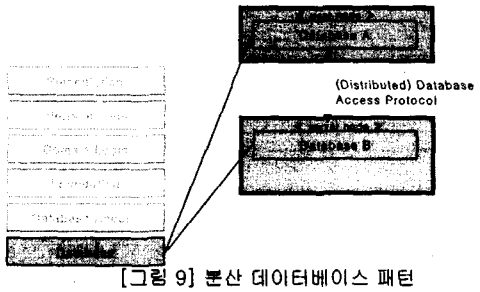
시스템 네트워크내의 분리된 노드상에 데이터베이스 컴포넌트를 분산시킨 형태이다



[그림 8] 원격 데이터베이스 패턴

(5) 분산 데이터베이스 패턴

데이터베이스 컴포넌트를 서로 다른 노드 상에 위치한 여러 데이터베이스에 분산시킨 형태이다.



[그림 9] 분산 데이터베이스 패턴

각 패턴은 어플리케이션 아키텍처의 부분적인 분산 관점을 보여주며, 이것을 물리적인 시스템 아키텍처로 매핑한다. 또한 전체적인 분산 모델은 한 개 이상의 분산 패턴이 적용된 결과이다.

5. 결론 및 향후 연구과제

본 논문에서는 비즈니스 정보시스템에서 현재 광범위하게 사용되고 있는 여러 스타일의 아키텍처를 레이어드 아키텍처, 분산 유형별 아키텍처, 그리고 JE22와 .NET에서 채택하고 있는 아키텍처 등으로 분류하여 보았다.

비즈니스 정보시스템은 기본적으로 레이어드 아키텍처를 채택하고 있지만, 비즈니스 요구사항이나, 전략, 그리고 품질 속성 요구사항에 따라 각 계층의 분류나 세분화 정도나 분산유형의 선택이 달라질 수 있다. 또한 각 아키텍처들은 품질 속성에 대한 상호 절충 요소(Tradeoff)를 지지고 있기 때문에 아키텍처 선정시, 비즈니스 요구사항이나, 전략 그리고 정보 시스템의 분산환경에서 요구되는 품질속성 우선순위 등을 고려하여야 한다.

따라서 향후 정보시스템과 분산환경에서 요구되는 품질속성 도출과 각 아키텍처별로 품질속성의 상호 절충 요소를 분석하여 아키텍처 선정시 지침이 될 수 있도록 참조 모델을 정립하고자 한다.

6. 참고문헌

[1] Moisés Daniel Díaz Toledano , ' The Architecture of Enterprise Information Systems. A view based on patterns' , Url: http://www.moisesdaniel.com/wri/eisa.htm

[2] http://www.neurauter.at/Diplomarbeit/html/node35.html

[3] George Schussel, ' Client/Server:Past, Present and Future' , Url: http://www.sei.cmu.edu/str/descriptions/clientserver_body.html

[4] Sun Microsystems, JavaOne에서 Core J2EE Patterns 2판 발표, http://www.suntraining.co.kr/jsp/webzine/TechArticleView.jsp?wz_code=477&category=0

[5] Deepak Alur, John Crupi, Dan MalksCore " J2EE Patterns, Best Practices and Design Strategies" , Sun Microsystems, 2001

[6] Microsoft, Enterprise Solution Patterns Using Microsoft .NET Verson2.0

[7] Microsoft, Application Architecture for .NET : Designing Application and Services

[8] Microsoft, Guidelines for Application Integration

[9] Klaus Renzel, WolfgangKeller, ' Client/Server Architectures for Business Information Systems. A Pattern Language' , PloP' 97 Conference