JDBC 3-tier 모델에 의한 지리 정보 검색

이 혜 진* 이 태 경
동 국 대 학 교 전 자 계 산 학 과

Geographical Infomation Retrieval by JDBC 3-tier Model

Hye Jin Lee* Tae Kyung Lee
Dept. of Computer Science, Dongguk University
msjh@wonhyo.dongguk.ac.kr tklee@wonhyo.dongguk.ac.kr

요 약

지리정보시스템(GIS)과 같은 다양한 데이터 접근과 처리 그리고 다수 사용자의 자료의 공유를 필요로하는 시스템에서는 분산환경 시스템이 매우 큰 가치를 가진다. 본 논문에서는 JDBC 3-tier 모델에 의한 시스템을 구현하였다. 기존의 JDBC 모델은 자바의 보안구조의 제한성에 대한 문제와 클라이언트와 미들웨어에 대한 표준문제등이 있다. 이러한 문제점을 개선하기 위하여 미들웨어로 OMG의 CORBA를 사용하였다. CORBA를 사용한 JDBC 3-tier 모델에서 GIS를 구현할 수 있는 환경을 구축하였으며, 이러한 환경에서 Java를 구현언어로 사용하여 플랫폼에 대한 독립성과 웹 클라이언트를 인터페이스로 사용하는 시스템에서도 가능함을 제공하며, 데이터의 접근면이나 공유면에서 보다 유용성을 제공하였다.

1. 서 론

지리정보시스템(GIS)은 과거에 단순한 지도의 처리에 대한 작업을 다루었으나, 현재에 와서는 처리하는 지리정보의 양도 많아 졌을 뿐만 아니라 거의 모든 부분을 컴퓨터로 판단, 제작, 처리하는 관계로 많은 분야에서 분산환경에서의 GIS에 대한 필요성을 느끼고 있다. 분산 환경에서는 분산적이고 이질적인 환경에 존재하는 많은 데이터들을 공유하면 시스템의 유용성 또한 추구할수 있다.

네트워크를 통해 여러 곳으로 분산되어 있는 다양한 데이터를 접근하기 위해 본 논문에서는 SQL 문장을 실행시키기 위한 방법으로 Java API인 JDBC 모델을 사용하였다. 2-tier의 JDBC(Java DataBase Connectivity)모델은 서버에 의존적이라서 성능저하와 병목현상을 일으키며 데이터 접근과 시스템 확장성에서 유용하다고 할 수 없다. 이러한 모델을 개선하여 나온 것이 바로 JDBC 3-tier 이다[4]. 이러한 기존의 JDBC 3-tier 모델은 자바의 보안구조의 제한성에 대한 문제와 클라이언트와 미들웨어에 대한 표준문제등이 존재하는 이유로 또다시 이 문제의 해결책을 찾아야만 한다. 그 해답으로 제시된 것이 바로 미들웨어인 CORBA를 이용한 JDBC 3-tier 모델이다. 자바의 플랫폼 독립적인 특성과 CORBA의 분산 객체에 대한 지원을 통합할 수 있게 하여 이질적인 시스템이 함께 존재하는 분산 네트워크 환경에서 보다 생산성 높은 응용 시스템 구축을 위한 해결방안이 가능해진 것이다. 또한 기존의 모델보다 데이터의 공유면이나 시스템 확장면에서 보다 유용한 시스템을 구현할 수 있도록 한다[1][3].

논문의 구성을 위하여 2장에서는 관련 연구로서 JDBC의 개요와 CORBA를 이용한 3-tier 모델에 대하여 기술하였으며 뿐만 아니라 이 모델과 지리정보 시스템의 접목에 대하여 그 전반적인 배경에 대하여 기술하였다. 3장에서는 본 논문에서 구현한 시스템에 대하여 전체적인 설계와 구현 그리고 인터페이스에 대하여 설명하며, 실험과 평가에 대하여 기술하였으며, 마지막으로 4장에서는 결론과 향후 연구 방향을 제시하고 있다.

2. JDBC 모델과 GIS

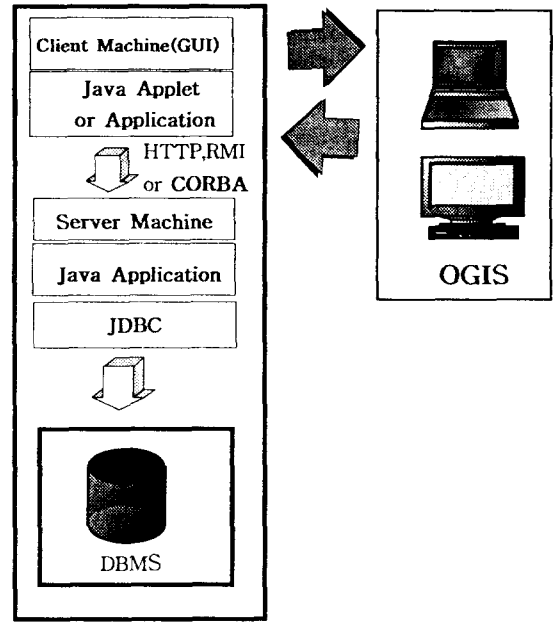
2.1 JDBC 개요

JDBC는 SQL 문장을 실행시키기 위한 자바 API이며, 자바로 쓰여진 클래스와 인터페이스의 집합으로 구성된다. JDBC는 응용프로그램과 데이터베이스 개발자들을 위한 표준 API를 제공하고, 순수 자바 API를 사용하여 데이터베이스 애플리케이션을 작성할 수 있게 해 준다. JDBC를 사용하면 SQL 문장을 데이터베이스로의 전달이 가능하다. 그러므로 자바 애플리케이션으로 다양한 다른 데이터베이스와 통신할 수 있도록 하는 방법이 JDBC 기술이다[4].

2.2 CORBA를 이용한 JDBC 모델

기존의 JDBC 2-tier 모델은 시스템의 성능저하 및 병목 현상을 초래함은 물론이고 응용프로그램 작성시 프로그램 로직과 구조가 서버 시스템의 구조에 의존함으로서 시스템 확장시 많은 문제를 초래했다. 대규모의 클라이언트를 요구하는 환경에서는 성능저하 등 많은 문제를 유발할 수도 있다. 이러한 기존 모델의 문제를 개선하고자 나온 것이 미들웨어를 사용한 3-tier 모델이다. 3-tier 모델은 그림1에서와 같이 클라이언트 응용 프로그램이 미들웨어 역할을 하는 응용 서버를 통하여 데이터 베이스에 연결되는 구조이다. 여기서 미들웨어의 역할은 클라이언트와 자체적으로 약속된 규약을 이용하여 통신을 하고 다른 한편으로는 클라이언트의 요구를 JDBC 관리자 계층을 통하여 데이터 베이스에 전달하고 수행된 결과를 받아 클라이언트에로 다시 전달하는 일을 한다. 미들웨어는 데이터베이스에서 제공하는 규약으로의 빠르고 직접적인 연결을 하기 위해서는 주로 'Native API partly-Java driver'형식의 JDBC 하위규약을 사용하는 쪽이 우수하다. 그러나 이러한 JDBC 하위규약은 클라이언트/서버의 2-tier 모델을 기본방식으로 하고 있어 미들웨어를 이용하는 3-tier 모델의 경우에 이용하려면 다른 네트워크 규약을 이용해서 구성해야 한다. 일반적으로는 미들웨어에서 클라이언트와의 통신을 하기 위해 HTTP, 자바 RMI(Remote Method Invocation), 또는 CORBA를 이용하는 경우가 많다.

다양한 미들웨어의 사용으로 기존의 JDBC 3-tier 모델을 실제로 적용하는데 다음과 같은 두 가지 문제점을 가지고 있다. 첫째는 클라이언트와 미들웨어 사이에 통신을 위한 표준 규약이 없다는 것이다. 둘째는 자바 애플릿을 이용한 응용 시스템에 적용하는 경우에 보안상의 제약을 받기 때문에 미들웨어는 항상 웹 서버와 동일한 시스템에 존재하여야 한다는 것이다. JDBC 3-tier 모델이 데이터베이스의 위치에 대한 투명성(transparency)은 보장해 주지만 반대 급부로 미들웨어의 위치가 서버에 종속적이고 실제 자료는 미들웨어가 함께 존재하는 시스템의 조건에서는 웹 서버에의 부하 집중에 대한 위험성은 본질적으로 내재되어 있는 것이다[9].



JDBC 3-tier 모델

그림 1. JDBC 3-tier 모델

과거 웹 상에서의 클라이언트와 서버는 계속 상태를 유지하면서 정보를 교환할 수 없으며, 브라우저는 단순히 HTML 문서를 표현해 주기 때문에 동적인

사용자 인터페이스와 지속적인 상태정보를 요구하는 정보 시스템을 구축할 때 많은 문제가 발생한다.
이 문제를 해결하기 위하여 새롭게 등장한 기술이 Sun Microsystems의 Java이다. 자바는 객체지향 개념을 지원하며 바이너리 코드가 아니라 중간 코드 형태의 바이트 코드를 생성해 내기 때문에 해당 바이트 코드를 해석하는 컴파일러가 있는 곳이면 어디서나 실행될 수 있는 구조를 가지고 있다. C++를 모델로 만들어진 자바 언어는 원시 파일뿐만 아니라, 실행 파일의 차원에서 작고, 간편하고, 플랫폼과 운영체제에 대하여 이식성이 강하게 만들어졌다[2].

그러나 이러한 자바가 제공하는 많은 장점들에도 불구하고 아직 해결해야 될 많은 문제점들이 있다. 먼저 자바가 진정한 개발 언어가 되기 위해서는 높은 성능과 빠른 수행 속도를 보장해야 한다. 또한 기존의 리거시 시스템을 손쉽게 연동하기 위한 기능을 제공해야 한다. 대부분의 엔터프라이즈 시스템들은 빠른 성능을 요구하는 한편 기존의 개발된 리거시 시스템들을 적극 활용해야 하기 때문이다. 정리하자면, 자바는 객체지향 개발 언어로서 출발했으나 개발 언어 그 이상은 아니다. 따라서 엔터프라이즈 솔루션을 제공하기 위해서는 보다 고도의 시스템 통합 기능을 지원해야 한다. 이러한 점이 원인이 되어 객체지향적 인터프리터 언어인 자바와 CORBA의 결합이 부각되고 있다. 이유는 자바와 CORBA가 서로간의 약점을 보완시키는 작용을 하기 때문이다. 자바는 플랫폼에 독립적이지만 보안상 분산처리에 약점이 있다. 반면 CORBA는 분산처리를 하기 위해서는 프로그램 개발면에서는 빈약하다. 또 캐시 버퍼 크기를 조절할 수 없고 TCP/IP 기반 위에 또 다른 층을 형성해 속도에서 문제가 발생했다.
CORBA를 자바로 매핑하면 이같은 약점을 서로 보완할 수 있다. 본 논문에서 굳이 구현 언어로서 자바를 선택한 이유는 자바가 앞에 언급한 많은 이점을 가진 언어지만 클라이언트의 형태나 환경에 많은 제약을 받지 않는다는데 그 중점을 둔다.
자바의 플랫폼 독립적인 특성과 CORBA의 분산 객체에 대한 지원을 통합할 수 있게 하여 이질적 시스템이 함께 존재하는 분산 네트워크 환경에서 보다 생산성 높은 응용 시스템 구축을 위한 해결방안이 가능해진 것이다.

2.3 JDBC 모델에 의한 GIS 접목

컴퓨터로 지도를 작성하는 데서 시작된 지리정보 시스템 모델이 현재는 매시간 마다 변화하는 지리정보를 그때 그때 수정하며, 거대한 네트워크 상에서 많은 클라이언트들이 그 정보를 공유할 수 있는 수준까지 와 있는 것이 현실이다.

이러한 지리정보 시스템 네트워크 구성은 OGC(Open GIS Consortium)의 멤버들이 국가 및 세계의 정보기반에 대한 긍정적인 비전을 가지고 있는데서 비롯되었다고 할 수 있다.
OGIS 인터페이스에 기초하여 시스템을 구축하는 개발자들은 모든 종류의 지리정보와 지리정보 처리기능을 다룰 수 있는 미들웨어와 componentware 및 어플리케이션을 개발하고 있다. 다른 목적을 가지고 다른 시스템을 사용하는 서로 관계없는 그룹에 의해 각각 서로 다른 시간대에 데이터가 만들어지고, 각 그룹이 여전히 자신의 시스템에 의해 일차적으로 지배된다 하더라도, OGIS에 기초한 시스템의 이용자는 잠재적으로 거대한 네트워크 데이터 공간을 공유할 수 있을 것이고, 이 공간에서는 모든 지리정보가 기본적인 데이터 모델에 적합하게 작동할 수 있을 것이다.

OGIS의 여러 분야들은 자료공유와 데이터 범위의 확산을 위하여 분산처리 컴퓨팅 환경에 관심을 갖고 있으며 이들 환경은 분산처리 컴퓨팅 플랫폼(DCPs)에 기초하고 있다. 이 중 하나가 OMG(Object Management Group)의 CORBA이다.
CORBA는 이러한 지리정보 시스템 네트워크를 구현하였을 때 누구나 그 네트워크 상의 데이터에 쉽게 접근할 수 있는 환경을 구축한다.

3. JDBC 3-tier에 의한 GIS 설계 및 구현

3.1 미들웨어 해결안 모색

서버와 클라이언트의 개방성과 상호 운용성을 높이기 위하여 출현하게 된 분산환경하의 분산 객체 미들웨어는 사용자들과 프로그래머에게 객체지향 기술이 제공하는 다양한 장점들을 제공하며 네트워크 프로그램에서 발생하는 수고를 덜어줌으로서 가장 각광받는 시스템이 되고 있다. 또한 분산 객체 미들웨어는 네트워크 투명성 이외에 다양한 플랫폼 상에 존재하는 다양한 객체들을 하나로 묶을 수 있는 기능을 제공하기 때문에 다른 종류의 플랫폼과 데이터 모델을 통합하는 시스템으로 자리잡고 있다.
Visigenic의 Vigibroker가 본 논문에서 사용된 분산 객체 미들웨어인 CORBA의 제품 이름이다.

```
module Display
{
struct Capital{
string C_name;
string Country_name;
short C_people;
string C_location;
short C_X;
short C_Y;
};
struct Countries{
string countries_NAME;
string capital;
string river;
string load;
string mountain;
string people;
short width;
};
typedef sequence<Capital> CapitalSeq;
typedef sequence<Countries> CountriesSeq;
interface DataBase{
Capital Reservations(in string C_name);
Countries getData(in string countries_NAME);
};
interface Dispenser{
void CreditObject(in string name);
};
};
```

그림 2. IDL 정의 모듈

그림 2는 본 논문의 IDL파일의 모듈이며, 다음과 같이 크게 세부분으로 나누어진다.

-클라이언트로 전달할 데이터 Struct를 정의 한부분

-DataBase Interface: Struct로 정의한 부분의 데이터들을 전달하는 함수를 선언하는 기능을 가진다.

-Dispenser Interface: 입력한 name으로 원하는 이미지 파일을 load하여 클라이언트로 전달해주는 함수(void CreditObject())를 선언하는 기능을 가진다.

이렇게 만든 IDL 파일을 Visigenic의 VisiBroker for 자바의 IDL 컴파일러를 이용하여 컴파일한 후 JDBC API 표준에 맞추어 구현된 클라이언트측의 IDL Stub 부분과 미들웨어를 구성하는 객체구현 부분을 함께 JDK의 자바 컴파일러를 통하여 컴파일함으로써 최종적으로 구현된다.

3.2 GIS 구현을 위한 JDBC 3-tier 구조

GIS 사용자들은 생산성 있는 응용 프로그램을 원할 뿐만 아니라 파일이나 데이터베이스처럼 다양한 데이터 저장공간으로 부터 정보를 획득하길 원한다. 이러한 시스템을 구성하기 위해서 선택하게 된 것이 그림 3의 JDBC 3-tier 구조이다. 본 논문에서는 CORBA 구현 VisiBroker for 자바를 이용이며, 클라이언트 미들웨어의 구현 도구는 Sun Microsystems Inc.의 JDK(Java Development Kit) 1.1.x를 이용하였다. Visigenic의 VisiBroker라는 제품은 Netscape Communicator에서 직접 지원하는 CORBA 구현 제품이다[8]. 클라이언트 부분은 플랫폼 독립적인 특성을 살려 자바 언어로 application과 웹상에서 applet으로 구현해 보였으며 클라이언트에서 질의한 데이터에 대하여 미들웨어를 통하여 오라클 데이타베이스 서버와 파일을 저장한 파일 서버에 접근하여 질에 결과에 맞는 데이터를 검색하여 화면에 출력하게 해 줌으로써 최종적으로 구현이 된다.

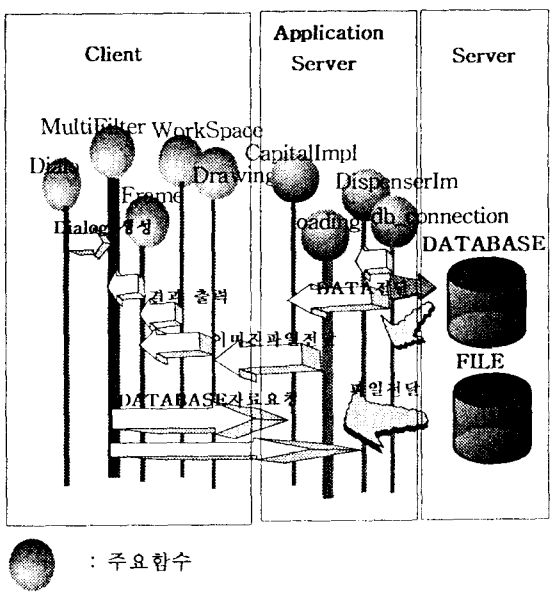


그림 3. 3-tier 구조와 함수

3.3 사용자 인터페이스 설계

본 논문에서 이용하고 있는 데이터는 데이타베이스 와 파일 서버에 존재한다. 먼저 데이타베이스에는 여러 텍스트 자료와 화면에 지리적인 Display를

할 수 있는 좌표데이터가 들어있으며, 파일 시스템에는 지도가 저장되어 있다. 사용자가 원하는 국가의 지도를 보여 줄 수 있도록 되어 있으며, 지도상에 수도, 강, 산, 도로 등을 보여줄 수 있도록 되어 있다.

그림 4와 그림 5에서 보인 전체적인 인터페이스는 크게 세 부분으로 나누어 볼수 있는데 원하는 국가 이름을 입력하여 질의하는 부분, 검색한 Text 데이터를 보여주는 부분, 그리고 지도와 함께 포함된 자료를 보여주는 부분이 그것이다. 자바 application에서 버튼이나 사용자가 입력한 데이터를 미들웨어를 통하여 JDBC API를 통하여 오라클 데이타베이스에서 찾은 자료와 검색한 지도를 같이 보여주는 과정으로 반복된다.

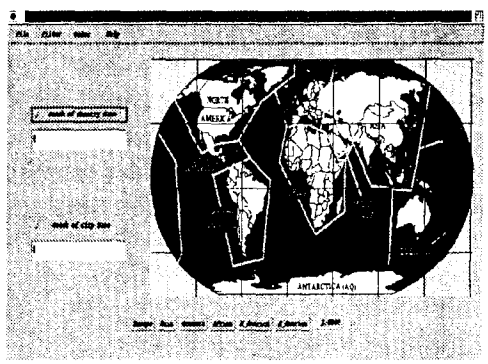


그림 4. 질의하는 부분의 Interface 구성(예)

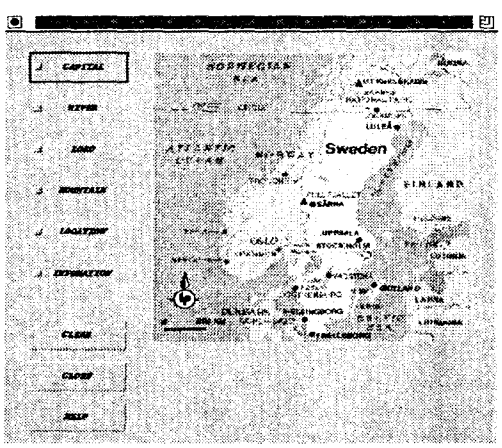


그림 5. 이미지 출력에 대한 Interface 구성(예)

3.4 질의에 의한 검색 방법

원하는 정보에 대한 검색방법으로 사용자가 입력한

```
public Display.Capital getReservations(String sev){
Display.Capital capital_name = new Display.Capital();
try{
 String select =
   "select * from t_capital where  C_name='"+sev+"'";
  ResultSet rs = stmt.executeQuery(select);
while(rs.next()){
  capital_name.C_name            =    rs.getString(1);
  capital_name.Country_name      =    rs.getString(2);
  capital_name.C_people          =    rs.getShort(3);
  capital_name.C_location        =    rs.getString(4);
  capital_name.C_X               =    rs.getShort(5);
  capital_name.C_Y               =    rs.getShort(6);
}
        stmt.close();
        connection.commit();
   return capital_name;
}catch (Exception e) {
    System.out.println("line 97 : error");
   return null;
  }
}
```

그림 6. 질의에 관련된 클래스의 함수 부분

국가이름이나 도시이름으로 JDBC에 의해 데이터베이스에 접근하기 위해 그림 6에서 보는 바와 같이 SQL문을 형성하고 있다. SQL문을 형성하기 전에 JDBC 드라이브를 설정하여 데이터베이스에 연결하고 계정을 얻어온다. 형성된 SQL문을 실행하여 결과를 만들어 놓은 Struct(capital_name)로 얻어온다. 얻고자 하는 데이터의 종류에 따라 각각 다른 질의문을 갖도록 구성되어 있다.

3.5 실험 및 평가

서로 다른 환경에서 데이터베이스와 연동 시스템을 구현하고자 한다면 자바를 사용하는 것은 좋은 방법이 된다. 본 논문에서는 자바 응용 프로그램에도 적용이 가능하며, 웹에서 자바 applet을 사용하는 경우에도 적용이 가능하도록 하였다. 이미 작성된 자바 응용 시스템이 존재 하는 경우에도 적용이 가능하다. applet에 대한 보안구조에 대한 제한사항도 CORBA를 응한 JDBC 3-tier 모델에서 극복되므로 웹 클라이언트를 사용자 인터페이스로 하는 경우에도 적용이 가능하다.

분산 객체의 환경에서 자바와 CORBA의 단점을 보완하여 장점을 살린 시스템이라는 점에서 의의를 둘만하나, 지리정보 시스템 부분에서 처리하는 데이

터의 한계와 많은 기능을 지원하지 못하는 부분과 미들웨어의 Service 기능을 다양하게 제공하지 못해 아쉬움이 남는다. 그러나 이러한 시스템을 시작으로 연구를 거듭하면 분산환경에서 거대한 지리정보 시스템 네트워크를 구성할 수 있다.

4.결론 및 향후 연구

본 논문에서는 분산 환경에서 자바 응용과 JDBC API를 이용한 오라클 DB 연동 그리고 미들웨어인 CORBA를 이용하여 간단한 지리정보 시스템을 구성해 보았다. 지리정보 시스템 자체가 데이터 요구량도 많고, 처리량도 많기 때문에 단순한 시스템에서는 구현에 제약이 따르거나, 비용이 많이 드는 부분이 있다. 본 논문은 이러한 현존 시스템의 제약점을 개선하고자 이질적인 분산환경에서 많은 사용자가 자료를 공유할 수 있는 대규모 시스템을 구현하고자 하는 노력에서 시작된 것이다.

그러나 아직은 개발 단계에 있어서 실제로 이러한 시스템을 구현하여 웹에 연동을 시킨다 하더라도 웹상에서 처리할 수 있는 기능이 매우 제한적이다. 특히 지리정보 시스템처럼 다양한 처리 방법을 요구하는 분야에서는 웹상에서 응용할 수 있는 기능 도구의 개발이 요구된다.

현재 CORBA의 경우에는 자바와 웹 그리고 다른 미들웨어와의 통합이 이루어지고 있다. 언젠가는 미들웨어를 통하여 아무런 제약도 받지 않고 원하는 데이터를 획득할 수 있으리라 기대된다.

참고문헌

[1] Orfali , Harkey , Client/Server Programming with Java and Corba , Vol I,II, Willy , 1997

[2] 박철우 역 , JAVA Networking & AWT API , 도서출판 대림, 1997

[3] 박재현 저 , Core CORBA , 영한 출판사 , 1998,

[4] 박재진, 손진국 역, Java로 처리하는 JDBC 데이터베이스 , 1997

[5] 유근배 , 지리정보론 , 상조사 , 1989

[6] 엄기현, Client/Server 구조 , 이한 출판사 , 1995

[7] J.Siegel , CORBA fundamentals and Programmin, Jone Willey and Sons , 1996

[8] Visigenic Software Inc , VisiBroker for Java Programmer's Guide version 1.0 , 1996

[9] 이진용 , 전순미 , 분산 객체 환경에서의 JDBC 3-tier 모델의 성능 확대와 CORBA의 이용 , 한국정보처리학회논문지, 제5권, 9호 , 1998

http://www.visigenic.com/techubs/#VBJL.ink