분산 환경에서 XMDR을 이용한 예약 정보 시스템

정계동* · 황치곤* · 최영근*

A Reservation Information System using XMDR on Distributed Environments

Kye dong Jung* · Chi gon Hwang* · Young-keun Choi*

요 약

사용자는 인터넷을 통해 다양한 정보를 제공 받을 수 있고 예약도 가능하다. 그러나 이러한 작업은 사용자의 요구조건을 만족하기가 어렵다. 그 이유는 사용자의 요구조건을 스스로 조정하여 대안을 마련해야 하고, 정보를 제공하는 곳들이 분산되어 있고, 데이터의 형태 및 저장형태가 다를 수 있다. 그리고 이러한 경우는 최근 기업들의 변화에서도 나타난다. 기업 간의 M&A 및 약속상, 협력 등의 과정에서 각 기업들이 사용하는 애플리케이션 및 데이터베이스에 대한 상호운용성의 보장과 통합된 정보관리를 위한 새로운 전략의 필요성이 대두된다. 이에 따라 분산 데이터의 상호운용은 기존 시스템들의 협업을 위한 기본적인 조건으로서, 데이터의 가능성, 투명성과 협업 시스템들의 문제점을 해결해야 한다.

본 논문에서는 분산된 시스템들을 통합하고, 상호운용성을 보장하기 위해 XMDR을 이용한다. 이 XMDR을 바탕으로 여행정보 시스템의 통합하고, 여행 정보 교환을 위한 통합적 관리 기법을 제안함으로써 상호운용성의 문제와 분산된 데이터를 통합하는데 있어 발생하는 데이터의 이질적인 문제들을 해결하고자 한다.

ABSTRACT

User can be offered various information from internet and reserve. However this operation is hard to be satisfied with user’s requirement. Because it offer the alternative by self-mediation and the place where offer the information is distributed and a form or stored form of data would be different. These cases occurred in change of enterprise recently. In a process of M&A, outsourcing or cooperation, necessity of new strategy become a pressing issue for ensuring interoperability on application, database of each enterprise and integration information management. Thus, interoperability of distributed data is a base condition for cooperation of legacy system, it will solve a problem such as available or clarity of data, problem of cooperation system.

This paper use a XMDR for integration of distributed system and ensuring interoperability. By using XMDR, integrate a traveling information system and it will be solve the problem of interoperability and variety data problem that be occurred in integration of data by suggesting a technique of transaction management.

키워드

XMDR(exTended Meta-Data Registry), MDR(MetaData Registry), Ontology, XML, Distributed DataBase

I. 서 론

이온화정보를 얻을 수 있다. 예를 들어 여행자들은 인터넷을 통해 교통수단, 여행경로, 좌석예약, 숙박정보 및 예약, 차량임대 정보 및 예약, 여행목적에 대한 예약과 같은

e-Business와 인터넷의 발달을 통해 사용자들은 광범위한 정보를 얻을 수 있다. 예를 들어 여행자들은 인터넷을 통해 교통수단, 여행경로, 좌석예약, 숙박정보 및 예약, 차량임대 정보 및 예약, 여행목적에 대한 예약과 같은

· 광운대학교

접수일자 : 2007. 5. 16
온 다양한 정보를 얻어 예약 한다. 이러한 작업들은 여행 자의 요구조건을 모두 만족시키기에는 어렵다. 이는 여행 자의 요구조건을 스스로 조정하여 만족을 마련해야 하고, 여행 정보를 제공하는 여행사들이 각 분야, 같은 분야 내에서도 여러 업체로 분산되어 있고, 데이터의 표 현과 구조가 서로 일치적인 문제점이 있다.

이러한 문제점들은 최근 여행업의 환경변화에 따른 요구자의 증가로 새로운 전략의 필요성이 대두되어 있다. 이에 의해 이러한 기업환경에서 업무 단순화와 프로세스

설계의 의연 최소화를 위한 최선의 정책에 대한 중요성이 강조되고 있다[1]. 기업 내의 완전한 비즈니스

을 운영하기 위한 애플리케이션으로서 메인프레임, 테

이터베이스, CRM(Customer Relationship Management) 시스템, ERP(Enterprise Resource Planning) 시스템 등에 많 은 비용과 시간을 투자하였다[2]. 그러나 IT 환경이 더욱 복잡해짐에 따라 기업 내의 분산된 애플리케이션의 정보 활용이나 인터넷 기반 애플리케이션의 연계가 필요하게 되었으며, 기업간의 M&A 및 아웃소싱, 협력 등의 과정에서 각 기업들이 사용하던 애플리케이션 및

t 데이터베이스에 대한 상호운용 요구가 대두되고 있다.

이에 따라 최근 기업들은 비즈니스 업무의 통합과, 비즈니스 트랜잭션을 기업내부적으로 한정하지 않고 기업 간으로 확장하기 위해 많은 투자를 하고 있다. 그러나 시스템 통합을 가능하도록 하기 위해서는 신뢰성과 상호 운용성을 제공할 수 있는 전자상거래 프레임워크가 필요하다. 이를 위해 국 내 기업, 비영리 단체, 기업 등에서 전자상거래 프레임워크 표준화 작업을 하고 있다. 이를 위해 이 기존의 시스템의 통합과 데이터 이동에 대한 국 제 표준화 동향으로 Microsoft에서 개발한 BizTalk, 국제 EDI(Electronic Data Interchange) 표준 개발 기구인 UN/CEFACT와 OASIS에서 개발한 ebXML, IT에 종사하 는 기업들의 비영리 컨소시엄이 결성하는 e-Business 표 준 프레임워크인 RosettaNet 등의 프레임워크가 있다

[3][5][6]. 이들은 시스템 통합을 제공하고 있고 전철적 데이터의 교환을 위해 XML로 이루어지고 있지만 지식 공유와 트랜잭션 관리와 같은 한계가 있다.

따라서 분산 데이터의 상호운용은 기존의 시스템들의 협업을 위한 기본조건으로서, 데이터의 가용성과 투명성 문제, 협업에 참여하는 시스템의 이질성에 따른 문제들을 해결해야 한다. 또한 기존의 시스템의 변경을 최소화하고 전체 시스템에 통합적이며 자율적인 환경을 제공해야 한다[4].

본 논문에서는 분산된 시스템들을 통합하고, 상호운 용성을 보장하기 위해 XMDR(eXtended Meta-Data Registry)을 이용한다. 이 XMDR은 여행 예약 시스템에 도입하여 투명한 여행사들의 정보를 통합하기 위한 표 준으로 제공하여 상호운용성의 문제와 분산된 데이터

를 통합하는데 있어 발생하는 데이터의 이질적인 문제 들을 해결하고 관리관제를 통합하는 기본 요구 조건에

따른 예약 정보 없이도 관리관제를 분산하여 원리

목적을 수행할 수 있는 정보를 제공한다. 이를 위한 트랜잭션 관리 기법들을 제안한다.

본 논문의구성은 1장에서 서론, 2장에서 관련연구, 3

장에서 시스템 설계, 4장에서 적용사례와 비교분석, 5장

에서 결론을 기술한다.

Ⅱ. 관련연구

최근 메타데이터에 대한 연구 및 개발은 메타데이터 동등기를 기반으로 XML 관련 기술을 적용한 해결 방법

이 주류를 이루고 있다. 따라서 이러한 한계를 XML 이

라는 기술과 ISO/IEC 11179라는 메타데이터 생성, 관리

방법론을 통하여 해결하려는 시도가 있다. [8][9]

데이터 통합에 따른 데이터 이질성을 해결하기 위해,

XML 기반의 관계형 데이터베이스 메타데이터를 객체

지향 데이터베이스에 저장하는 기술과, 분산된 데이터

의 이질성을 해결하고자 MDR(Metadata Registry)과

온돌로지의 통합형을 XMDR이라 한다. XMDR의

구성 표준은 온돌로지, 로케이션 온돌로지, 카페고리, 지식

베이스로 구성된다[10][11][12].

표준 온돌로지(Standard Ontology)는 적응하고자 하

는 영역이 개별정보를 추출하고 추출된 개념들의 관계

를 표현한다.

그림 1. 표준과 로케이션 온돌로지 간의 매핑

Fig. 1. Mapping between Standard and Location Ontology
레거시시스템의 MDR와 통합하기 위한 XMDR의 표준 온톨로지는 그림 1의 Ontology로 표현하고 속성들의 정의와 관계는 다음과 같다.

식별속성: 데이터 식별과 키네고리 분류 속성 (ST_ID, ST_CLASSID)
정의속성: 데이터 표준요소와 각 레거시시스템의 요소와의 관계성을 위한 속성(ST_NAME, LEGACY_NAME)
표현속성: 데이터 표현을 위한 속성(ST_TYPE, ST_SIZE, ST_FORMAT)
관계속성: 표준 온톨로지와 로케이션 온톨로지를 연결하기 위한 속성(LOC_ID)
로케이션 온톨로지(Location Ontology)는 각 레거시시스템의 위치정보, 접근권한 및 표준 온톨로지와 관계를 표현한다. 속성은 다음과 같다.

관계속성: 표준 온톨로지와 연결하기 위한 속성(LOC_ID)
지역속성: 참여한 레거시시스템에 접근하기 위한 권한, 이름, 위치 정보 속성(URL, Auth_ID, Auth_Pass, DB_NAME, TBL_NAME)
작용속성: 우선순위 부여를 통한 적응형 검색을 지원하기 위한 속성으로 검색결과와 사용자의 선택에 의해 부여되는 가중치 속성(LOC_Weight)

MDR의 생성은 상품 분류 기준의 키네고리에 의거한 각 상품의 데이터 표현은 표준 온톨로지 영역, 접근 정보에 관한 것은 로케이션 온톨로지 영역, 그리고 유사성이나 대체할 수 있는 상품의 관계성들은 지식 베이스(Knowledge Base) 영역의 결합으로 XMDR은 생성된다. 

그림 3. 시스템 구성도
Fig. 3. The structure diagram of System

3.1 통합 서버(Integration Server)

협업을 위한 레거시시스템들의 통합을 위해 XMDR이 구성하고 이를 최적의 상태로 유지한다. 접근 관리자, 에이전트 관리자, 데이터 관리자로 구성되며 그림 4와 같다.

그림 4. 통합 서버의 구성
Fig. 4. The structure to Integration Server

접근 관리자(Access Manager)는 로그 정보, 접근권한, 병행 수행 제어, 트랜잭션으로 구성된다.

• 로그정보(Log Info)는 비트위크에서 발생하는 모든
작업에 대한 정보를 관리하는 역할을 한다.
  - 접근권한(Authorization)은 각 레거시에 접근하기 위한 접근 권한 정보를 관리하고 이를 제공하는 역할을 한다.
  - 병행 수행 제어(Concurrency)는 두 이상의 작업이 같은 레거시에 있는 자원을 접근할 때 작업이 중복되어 발생할 수 있는 문제를 처리한다.
  - 트랜잭션(Transaction)은 각 레거시에서 발생하는 작업을 하나의 트랜잭션으로 보고 데이터의 일관된 상태를 유지하도록 하고 처리기법은 3결에서 다룬다.
  - 에이전트 관리자(Agent Manager)는 변환 에이전트, 검색 에이전트, 운행 에이전트, 구매 에이전트로 구성된다.
  - 변환 에이전트(Convert Agent)는 질의를 글로벌 XML_QUERY로 변환, 결과를 XSLT 변환을 통해 사용자에게 제공하는 기능을 수행한다.
  - 검색 에이전트(Retrieve Agent)는 인터페이스 생성하기 위한 항목들을 가져오기 위해 XMDR에 접근하여 필요한 항목과 적용적 검색을 위한 로케이션 온도로지의 가중치항목(LOC_Weight)을 가져오는 역할을 수행하는 에이전트이다.
  - 운행 에이전트(Navigation Agent)는 변환 에이전트에 의해 생성된 글로벌 XML_QUERY를 레거시시스템의 데이터베이스의 XMDR 레퍼에 표준 질의를 전송하는 역할을 수행한다.
  - 구매 에이전트(Purchase Agent)는 검색 에이전트를 통해 검색된 결과를 사용자가 확인 후 구매의사 결정에 의한 구매를 수행하는 에이전트이다. 트랜잭션 관리 기법은 3장 3절의 트랜잭션 관리 알고리즘을 이용한다.
  - 데이터관리자(Data Manager)는 XMDR 중개자와 XMDR로 구성된다.
  - XMDR 중개자(XMDR Mediator)는 레거시에서 변 경된 데이터베이스 정보를 XMDR Server에 적용시키 업그레이드된 XMDR을 각 레거시의 XMDR에 제공하여 레거시의 XMDR이 최신의 정보를 유지하도록 하는 역할을 한다.
  - XMDR은 XMDR Manager에서 생성되고 개선된 XMDR을 유지하여 레거시의 요구나 XMDR 중개자의 요구에 의해 각 레거시에 제공하는 역할을 수행한다. 예행 예약 시스템에서 데이터의 의미적 이질성이나 구조적 이질성 문제를 해결하기 위한 핵심적인 부분이 된다.

3.2 XMDR 레퍼(XMDR Wrapper)
레거시스템에 있는 실제 데이터를 접근하기 위해 XMDR 표준으로 전송된 질의문을 변환하는 XMDR 레퍼를 레거시스템에 덮는다.

표 1 XMDR 레퍼의 질의 변환 알고리즘
Table 1. The Query convert Algorithm of XMDR Wrapper

```
procedure ConvertQuery(char enterprise_id, char input_date, char return_sql)
{
    //XML 레퍼에 대입된 결과정보 저장
    XMDR_LOG_stack = FindDOMNode(enterprise_id)
    return_sql = SELECT * + mapped_ST_NAME, XMDR_LOG_stack
    WHERE * + mapped_ST_NAME, XMDR_LOG_stack
    WHERE * + input_date
    //XML의 표준 CNT_ID에 해당하는 항목정보를 이용한 질의문 생성
    procedure mapping(char st_id, char legacy_id, char st_attribute)
    {
        //표준 온도로지에 대입되는 항목의 성장 정보
        XMDR1_ID = FindDOMNode(legacy_id, st_id)
        st_attribute = XMDR1_ID, st_attribute
    }
}
```

XMDR 레퍼는 레거시스템의 데이터베이스 정보를 통합서비스에서 제공되는 XMDR 표준과의 매핑정보를 유지하고, 표준 질의를 해당 실제 데이터베이스에 적합한 질의로 변환하는 역할을 담당한다.
매핑 정보를 XMDR 레퍼에서 관리함으로서 웹 서버에서 전송되는 표준 질의를 레거시스템에 적합하게 변환하여 질의를 수행하고, 수행된 결과는 다시 표준항목에 적합하게 변환한다. XMDR 레퍼는 표 1에서 표현된 알고리즘에 의해 수행된다.

3.3 데이터 상호운영을 위한 트랜잭션 처리
사용자의 예약 요구 발생 즉시 결과를 반환하지 않고 예약대기 또는 웹 서비스와 같은 유연한 접근 방식에서 사용하는 트랜잭션과 같이 처리를 수행하며, 사용자의 요구조건을 충족시키기 위해 표준 온도로지의 정보뿐만 아니라 지식베이스를 통한 연관정보를 검색하기 위한 시간이 필요하다.
통합 서버에서는 하나의 트랜잭션과 같지만 각 레거시스템에서는 개별 데이터베이스에 의해 개별적인 트
랜체시션이 발생한다. 랜체션의 처리는 기존의 2단계 트랜잭션 기법을 각 레거시시스템에서 그대로 적용하며, 통합서비스에는 지식베이스 접근하여 연관관계 정보를 검색과 수정가능 여부를 확인하는 대기기계를 추가한 방안을 제안한다.

3.3.1 연관정보 적용
검색조건은 필수조건과 부가조건으로 나누어 선택하도록 한다. 검색조건은 필요한 검색조건에 대한 결과가 검색되지 않거나 빈약할 수 있으므로 필수조건을 검색에 따른 연관관계에 대한 부가정보로 같이 검색할 수 있는 유연성을 부여함으로써 사용자는 목적과 기간 만으로 그 부가정보를 교통, 숙박, 관광정보를 검색할 수 있도록 연관정보를 제공한다.

표 2에서 보면 조건은 검색결과의 여부에 따라 8단계의 검색단계를 둔다. 적용방법은 필수조건은 "O"로 표현하여 입력된 조건을 그대로 적용하고, 부가조건은 여 행을 위한 세부조건으로 트랜잭션 정보 테이블의 Flag를 위한 비트로 저장되어 기본 요구 항목 정보와 지식베이스에서 정의된 연관 관계에 따른 연관 정보를 저장한다. 이 Flag는 "10"인 기본정보를 표현하고, "01"이면 연관정보를 표현한다. 각 단계에서는 입력한 요구 조건을 그대로 적용하는 검색과 연관관계에 따른 처리가 가능하도록 한다.

표 2 연관정보 적용을 위한 조건 단계
Table 2. Condition step for Apply Relational Information

<table>
<thead>
<tr>
<th>단계</th>
<th>필수조건</th>
<th>부가조건</th>
<th>내용</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>O</td>
<td>O</td>
<td>10 10 10 10 기본 요구 충족 단계</td>
</tr>
<tr>
<td>2-1</td>
<td>O</td>
<td>O</td>
<td>10 10 01 관행에 대한 연관관계 (ex:공항=경소투어)</td>
</tr>
<tr>
<td>2-2</td>
<td>O</td>
<td>O</td>
<td>10 01 10 숙박에 대한 연관관계 (ex:별성=오피, 리조트)</td>
</tr>
<tr>
<td>2-3</td>
<td>O</td>
<td>O</td>
<td>01 10 10 교통에 대한 연관관계 (ex:비행기=기차, 편도기)</td>
</tr>
<tr>
<td>3-1</td>
<td>O</td>
<td>O</td>
<td>10 01 01 숙박, 관광에 대한 연관관계</td>
</tr>
<tr>
<td>3-2</td>
<td>O</td>
<td>O</td>
<td>01 01 01 교통, 관광에 대한 연관관계</td>
</tr>
<tr>
<td>3-3</td>
<td>O</td>
<td>O</td>
<td>01 10 01 교통, 숙박에 대한 연관관계</td>
</tr>
<tr>
<td>4-1</td>
<td>O</td>
<td>O</td>
<td>01 01 01 전체 연관관계</td>
</tr>
</tbody>
</table>

3.3.2 상호운영을 위한 트랜잭션 정보 테이블의 구성
상호운영을 위한 정보를 관리하는 상호운영 정보 테이블은 운영 구분 식별자로 TrID, 검색정보의 기본정보 를 표준항목으로 표현한 XMDR_Info, 연관 정보 KB_Info, 진행 정보를 저장하는 Flag로 구성되고 그림 5 와 같다.

그림 5. Flag의 처리 과정
Fig. 5. Process Course of Flag

Flag는 처리 순서 0, 완료를 1로 표현된 2진수 비트로 구성하는 상품종류에 따라 생성한다. 각 Flag는 두 개의 비트로 하고, first bit는 XMDR를 통한 기본항목 처리 결과를 반영하고, second bit는 지식베이스를 통한 연관관계 정보를 처리한 결과를 반영시킨다. 이러한 Flag의 상태는 표 3과 같다.

표 3. Flag 상태
Table 3. Status of Flags

<table>
<thead>
<tr>
<th>Flag</th>
<th>First Bit</th>
<th>Second Bit</th>
<th>Work</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>start</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>commit(연관관계)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>commit(기본항목)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Cancel &amp; rollback</td>
</tr>
</tbody>
</table>

Flag 처리 과정에서 "00"는 시작상태를 지시하고 해당 레거시시스템을 접근하여 수행을 시작한다. 수행 완료하고 요구조건을 만족하는 정보가 사용자가 제시한 기본정보에 묻 "10"로 전환하고, 요구조건에 만족하지 못한 연관정보를 이용하여 수행하고 연관정보가 존재하면 "01"로 전환을 시켜준다. 모든 Flag가 "10" 또는 "01"을 가지면 각 레거시시스템을 commit 시킴으로서 예약 작업을 완료 시킨다. 그러나 수행 후 해당 레거시시스템의 상태나 요구조건이 부적합한 것은 "11"로 전
그림 6. 트랜잭션 정보 테이블
Fig. 6. Transaction Information Table
이렇게 생성된 Flag는 First bit가 0인 즉, 기본 항목을 포기하여 선택한 트랜잭션들에 대한 리스트들에 대해서 지역별로 연결된 리스트를 생성한다. 생성된 리스트의 도식은 그림 7과 같이 기본항목과 연관항목의 선택을 나타내는 Flag 비트 정보, 지역정보 그리고 예약한 날짜 정보 등을 가지고 있다. 사례로 “서울” 지역을 포기한 항목들에 대한 리스트를 나타내고 있다.

이미 예약된 사항 중 사용자의 요구에 의해 포기된 내용이 존재할 경우, 즉 그림 7과 같이 “서울”을 선택한 항목이 취소되었을 경우 포기된 기본 항목이 “서울”이었던 연결리스트를 검색하여 가장 최근에 예약한 사용자에게 보상처리 원래의 요구사항을 반영시킬 수 있도록 한다. 예약 변경은 사용자의 변경여부를 확인하여 수행한다. 현재 예약사항을 그대로 유지하고 수정하지 않는 경우는 다음으로 최근 날짜에 예약한 사항에 대해 적용한다.

이러한 우선순위의 적용은 가장 최근에 선택한 예약을 상위의 우선순위를 적용하며, 예약 완료된 시간이 짧은 순으로 여정에 대한 세부계획이 세워져 있지 않아 여정 계획 수정에 유리하다고 가정한다. 텍스트로 리스트를 생성하는 것보다 비트로 처리하는 것이 효율적이다.

3.3.3 트랜잭션 수행 알고리즘
앞에서 제시한 트랜잭션은 맞춤형 페키지 상품에 적합하다. 맞춤형 페키지의 형태는 고정된 것이 아니라 사용자의 취향에 따라 선택하여 페키지를 구성하는 상품으로 페키지와 단일 상품의 구분이 없다. 즉, 서버에서는 페키지로 구성되지만 레거시시스템에서는 개별 상품에 대한 주문이 되고, 레거시시스템은 한곳으로 한정되지 않는다. 알고리즘의 구성은 서버 알고리즘과 레거시 알고리즘으로 구성한다.

표 4. 트랜잭션 정보 테이블 검사 알고리즘
Check Algorithm of Transaction Information Table

서비스 알고리즘은 트랜잭션의 동시성 제어를 위해 트랜잭션 정보 테이블 검사 알고리즘과 트랜잭션 관리 알고리즘으로 구성된다.

트랜잭션 정보 검사 알고리즘은 표 4에서와 같이 트랜잭션과 레거시시스템의 상대 파악으로 Flag를 갖도록 하여 트랜잭션의 완료 또는 취소시키는 역할을 수행한다.

표 5의 트랜잭션 관리 알고리즘은 검색 작업, 예약 작업, 취소 작업으로 분할하고, 각 작업은 현재의 상태를 트랜잭션 상태 비트를 변화시키며 트랜잭션 정보 검사 알고리즘을 통해 레거시시스템의 작업을 지시한다.
검색 작업은 레거시시스템에 데이터를 확인하는 메시지를 전달하고, 레거시시스템은 거래를 위한 수량을 확인하여 거래가 가능하면 'ACK' 신호를, 수량의 문제나 시스템의 문제로 인한 거래가 불가능하면 'NAK' 신호를 전송한다. 'ACK' 신호가 전송되면 거래가 가능한 상태이므로 사용자에게 검색된 상품에 대한 결과를 제공한다. 그리고 서버는 레거시시스템의 데이터베이스를 직접적으로 접근을 사용하지 않고 레거시시스템의 관리 알고리즘에 의해 해당 데이터베이스에 대해 참가를 설정함으로써 서버에서 발생하는 중복된 참가를 발생하지 않도록 한다. 메시지 전달은 기본정보를 먼저 전달하고, 기본정보로 검색한 결과가 없을 경우 연관정보를 전달하여 검색할 수 있도록 한다.

예약 작업은 레거시시스템에 확인된 메시지에 의해 실제적인 경신작업을 수행하며, 트랜잭션 정보 검사 모듈을 호출하여 레거시시스템에 실패한 경신 작업이 이루어진다. 이때 해당 레거시시스템은 정보 갱신을 위한 참가가 발생한다. 반면 거래 불가능 신호인 'NAK' 신호가 전송되면 거래 작업 취소가 발생되며, 트랜잭션 상태변을 갱신하며 트랜잭션 정보 검사 알고리즘에 의해 해당 트랜잭션이 수행한 작업을 보상처리 함으로써 데이터의 일관성을 유지한다.

표 5 서버의 트랜잭션 관리 알고리즘
Table 5. Transaction Control Algorithm of Server

<table>
<thead>
<tr>
<th>Transaction Request new type</th>
<th>generate TrId</th>
</tr>
</thead>
<tbody>
<tr>
<td>case request type search:</td>
<td>Flag=00</td>
</tr>
<tr>
<td>// 기본정보 전달</td>
<td></td>
</tr>
<tr>
<td>StateCheck(Flag, XMOR_info, Loc_info)</td>
<td></td>
</tr>
<tr>
<td>if the message from the Legacy System is &quot;ACK&quot; then</td>
<td></td>
</tr>
<tr>
<td>Flag=10</td>
<td></td>
</tr>
<tr>
<td>service result to user</td>
<td></td>
</tr>
<tr>
<td>reserve</td>
<td></td>
</tr>
<tr>
<td>else</td>
<td></td>
</tr>
<tr>
<td>// 일반정보 전달</td>
<td></td>
</tr>
<tr>
<td>StateCheck(Flag, KB_info, Loc_info)</td>
<td></td>
</tr>
<tr>
<td>if the message from the Legacy System is &quot;ACK&quot; then</td>
<td></td>
</tr>
<tr>
<td>Flag=01</td>
<td></td>
</tr>
<tr>
<td>service result to user</td>
<td></td>
</tr>
<tr>
<td>reserve</td>
<td></td>
</tr>
<tr>
<td>else</td>
<td></td>
</tr>
<tr>
<td>Flag=11</td>
<td></td>
</tr>
<tr>
<td>cancel</td>
<td></td>
</tr>
<tr>
<td>end if</td>
<td></td>
</tr>
<tr>
<td>end if</td>
<td></td>
</tr>
<tr>
<td>reserve</td>
<td></td>
</tr>
<tr>
<td>// 요청되는 정보가 기본정보가거나 연관정보므로</td>
<td></td>
</tr>
<tr>
<td>// 상태정보로 Info로 한다.</td>
<td></td>
</tr>
<tr>
<td>StateCheck(Flag, Info, Loc_info)</td>
<td></td>
</tr>
<tr>
<td>if Flag=01 or Flag=10 then</td>
<td></td>
</tr>
<tr>
<td>if the message from the Legacy System is &quot;ACK&quot; then</td>
<td></td>
</tr>
<tr>
<td>update to legacy system</td>
<td></td>
</tr>
<tr>
<td>commit</td>
<td></td>
</tr>
<tr>
<td>else</td>
<td></td>
</tr>
<tr>
<td>Flag=11</td>
<td></td>
</tr>
<tr>
<td>cancel</td>
<td></td>
</tr>
<tr>
<td>end if</td>
<td></td>
</tr>
<tr>
<td>end if</td>
<td></td>
</tr>
<tr>
<td>cancel:</td>
<td></td>
</tr>
<tr>
<td>if the message from the Legacy System is &quot;complete&quot; and</td>
<td></td>
</tr>
<tr>
<td>Flag=11 then</td>
<td></td>
</tr>
<tr>
<td>StateCheck(Flag, Loc_info)</td>
<td></td>
</tr>
<tr>
<td>end if</td>
<td></td>
</tr>
</tbody>
</table>

레거시시스템에서 수행하는 단위 트랜잭션의 동시성 제어를 위한 알고리즘은 표 6과 같다. 요구된 작업의 형태에 따라 검색, 주문, 취소, 완료로 구분된다. 각 작업들은 크게 두 가지 역할을 수행한다. 하나는 서버의 요구가 가능할지 불가능한지에 대한 응답으로 'ACK'와 'NAK' 신호를 반송하는 것이고, 다른 하나는 처리된 결과에 대한 값을 반환한다.

검색 모듈은 서버에서 전송된 클로로 XML_QUERY를 XMOR 데이터 질의로 변환하고, 변환된 질의문을 수행하여 그 결과를 서버에 반환한다. 그에 대한 참가는 발생하지 않는다.

예약 모듈은 서버에서 사용자의 주문결제의 의도에 의해 발생하는 것으로 실제의 수량의 변화가 일어나다. 그러므로 요구된 수량의 충족과 충돌 발생 유무에 대한 검사를 수행해야 한다. 충돌이 발생하지 않으면 매탕적 참가를 반영하는 것으로, 가능한 수량을 검사하고 수량이 가능하면 경신
작업과 갱신된 작업 사항을 해퍼의 임시 데이터베이스에 저장한다. 수량이 만족하지 않으면 가능한 수량을 서버로 전송하여 구매여부를 재확인한다. 충돌이 발생한 다면 대기상태로 전환되어 대기한다.

취소 모듈은 주문수량 부족과 사용자의 요구에 의한 취소 작업으로 충돌이 발생하지 않는다면, 해당 작업 숫자로 트랜잭션의 수행내용을 임시 데이터베이스에서 호출하여 작업들을 ROLLBACK시키고 잡음을 해제한다.

완료 모듈은 거래의 완료 메시지를 서버에 전송하고, 임시 데이터베이스에 저장된 해당 트랜잭션의 내용을 제거하고 트랜잭션을 COMMIT 시켜 완료한다.

그림 8. 알고리즘의 실행 과정
Fig. 8. Execution Process of Algorithm
이와 같은 알고리즘을 수행하는 전체 과정을 도식화

크리 9. 시스템 작업 수행 과정
Fig. 9. System Work Flow

3.4 시스템 적용결과

사용자가 통합서버에 접근하여 여행예약 정보를 검색하고 선택한 결과에 따라 예약을 수행할 수 있다. 그럼 9는 사용자의 접근으로부터 발생하는 원래의 작업 과정이다. 사용자의 접근으로 통합서버의 에이전트 관리자에 의해 XMDR에서 카테고리 정보를 사용자에게 제공하고, 카테고리를 선택함으로써 XMDR의 지식베이스와 표준/스케일 온라인을 접근하여 인터페이스를 생성한다. 생성된 인터페이스를 통해 검색을 수행한다.

각 작업을 보면 다음과 같다.
SelCat() : XMDR의 카테고리 정보 항목 선택
RetrieveAgent() : XMDR의 기본 정보와 지식베이스를 통한 연관정보 검색 지시
FindXmdr() : XMDR의 표준 항목 정보를 검색하여 인터페이스 생성을 위한 정보 제공
FindKnowledge() : 지식베이스를 이용한 연관정보 검색
StateInitialize() : 트랜잭션 상태 정보 초기화
NavigationAgent() : 레거시시스템을 검색하기 위한 예 이전 생성
ConvQry() : 사용자의 요구 정보를 이용하여 클로벌 XML_QUERY 생성
NavigationRequest() : 생성된 질의를 XMDR 레퍼에 전송시켜 검색 수행을 요구
Mapping() : 클로벌 XML_QUERY를 XMDR 레퍼로 레거시시스템에 적합한 로컬 질의 변환
ExecQuery() : 변환된 질의로 실제데이터 접근
Return(ARQ or NAK) : 검색 결과 전송
StateCheck() : 검색된 결과에 의해 기본 정보를 통한 결과인지, 연관 정보를 통한 결과인지 기록
PurchaseAgent() : 예약 수행
RequestMassage() : 예약 수행 메시지 전송
UpdateLegacy() : 레거시시스템의 개선
Return2Result() : 결과 반환
ConXSLT() : 수집된 XML 결과를 사용자에게 제공하기 위해 XSLT변환
Result() : 반환된 결과 인터페이스 적용으로 사용자에게 알림

IV. 적용사례 및 분석

4.1 적용 사례

본 시스템은 예약 정보 시스템에 적용한다. 그림 10, 11, 12, 13은 XMDR을 적용한 시스템의 사용자 인터페이스 요구에 의한 XMDR을 이용한 부분이다. 지식베이스에 의한 연관성 분석에서의 지식과 표준 엔터프라이즈의 상품 카테고리에 의한 상품 선택을 할 수 있도록 지원하고, 선택된 상품에 대한 표준 항목을 지원하도록 표현한다.

그림 10은 본 시스템의 메인 화면으로 XMDR에서 제공되는 카테고리 정보를 선택하고, 선택된 카테고리에 의한 표준 엔터프라이즈 정보를 제공됨에 따라 사용자가 원하는 요구조건을 입력받을 수 있는 인터페이스가 생성된다. 

그림 11은 그림 10에서 생성된 검색조건에 따른 기본 항목 검색 결과로 원래 검색 대상이었던 "원주시"에 대해 검색 결과와 지식베이스를 통한 "원주시"와 연관 지역에 대한 정보를 제공한다.

![그림 10. 검색 인터페이스 예시](image1.png)

Fig. 10. Example of Search Interface

![그림 11. 기본 검색 결과](image2.png)

Fig. 11. Result for Standard Search

그림 12는 그림 11에 대한 기본 검색결과에 대한 연관 관계 검색 결과로 "원주시"와 연관관계인 "평창군"에 대한 검색 정보를 나타낸 인터페이스 화면이다.

![그림 12. 연관관계에 의한 검색결과](image3.png)

Fig. 12. Result for Relational Search
그림 13는 그림 11의 기본관계에서 선택한 레지스트로 결과와 행렬의 정보와 그림 12에서 선택한 연관지역 인 “평창군”의 막락에 대한 예약결과 화면이다.

![그림 13 완료 예시](image)

Fig. 13. Example of Completed

4.2 비교분석

서론에서 언급한 기존의 전자상거래 표준 프레임워크들을 대상으로 표 7과 같이 비교 분석하고, 비교 항목들로는 ISO/IEC 1179에서 논의 중인 XMDR 지원 여부, 데이터 교환 자동화 여부, 범용 명세 스키마 지원 여부, 저장소 구조, 계층적 구조, 데이터의 관계성 지원, 상호 운용성과 호환성 문제 해결 여부, 프로세스에 관련된 모델 복잡성의 경의 여부에 관하여 비교한다[7].

<table>
<thead>
<tr>
<th>항목</th>
<th>EDI</th>
<th>ebXML</th>
<th>RosettaNet</th>
<th>BizTalk</th>
<th>본 시스템</th>
</tr>
</thead>
<tbody>
<tr>
<td>XMDR 지원</td>
<td>부분 지원</td>
<td>부분 지원</td>
<td>부분 지원</td>
<td>부분 지원</td>
<td>지원</td>
</tr>
<tr>
<td>데이터 교환 자동화</td>
<td>부분 지원</td>
<td>부분 지원</td>
<td>지원</td>
<td>지원</td>
<td></td>
</tr>
<tr>
<td>범용 명세 스키마</td>
<td>지원</td>
<td>지원</td>
<td>지원</td>
<td>지원</td>
<td></td>
</tr>
<tr>
<td>저장소 구조</td>
<td>지속적</td>
<td>일반적</td>
<td>일반적</td>
<td>지속적</td>
<td>지속적</td>
</tr>
<tr>
<td>계층적 구조</td>
<td>수평적</td>
<td>수평적</td>
<td>수평적</td>
<td>수평적</td>
<td>수평적</td>
</tr>
<tr>
<td>데이터 관계성</td>
<td>부분 지원</td>
<td>부분 지원</td>
<td>부분 지원</td>
<td>지원</td>
<td></td>
</tr>
<tr>
<td>상호 운용</td>
<td>지원</td>
<td>지원</td>
<td>부분</td>
<td>지원</td>
<td></td>
</tr>
<tr>
<td>프로세스 모델</td>
<td>정의</td>
<td>정의</td>
<td>정의</td>
<td>정의</td>
<td></td>
</tr>
</tbody>
</table>

Table 7. Compare to system for data integration

V. 결 론

본 시스템에서는 이질적 환경을 가진 레지스트시스템을 통합하기 위해 표준 명세인 MDR를 바탕으로 XMDR을 설계 및 구축하고, 본산 시스템 프레임워크의 이질성 극복을 위해 XML기반의 메시지 교환 시스템 구축하였다. 특히 XMDR를 통한 표준 채용을 사용자에게 제시함으로써 실제 데이터들의 이질적 환경과 관계없이 통합된 환경을 제시할 수 있고, 이를 위한 데이터 교환 방법을 구축하는데 중점을 두고 있다.

검색뿐만 아니라 데이터를 위한 데이터의 교환과 수정이 가능하게 하기 위해 데이터의 상호 운용을 위한 트랜잭션 관리 기법에 제안했고, 기업 간의 정보공유를 위한 정보 통합을 위해서는 XML을 이용한 XMDR 기반으로 한다. 따라서 본 논문 XMDR은 웹 서비스 및 기업의 데이터베이스에 적용이 가능할 것으로 기대된다. 테이블과 기술적 특성이 이를 통해 기업의 기술 가치의 향상과 새로운 비즈니스 모델의 제시를 통한 시장 창출을 기대할 수 있으며, 기업 간의 정보를 공유함으로써 생산성을 향상 시킬 수 있다.

이후는 P2P환경에서 Grid기술을 도입하여 시스템을 확장하고, XMDR을 이용한 표준 프레임워크를 제시할 수 있는 지식 표현 기술과 더 효율적인 동시성 관리 기법에 대한 연구가 필요하다.

참고문헌


저자소개

정계동(Kye-dong Jung)
1985년 광운대학교 전자계산학
(이학사) 1992년 광운대학교 산업정보학
(이학석사) 2000년 광운대학교 컴퓨터과학(이학 박사) 1993년 ~ 2004년 광운대학교 정보과학원 교수 2005년 ~ 현재 광운대학교 교양학부 교수 ※ 관심분야: XML, 분산시스템, 분산컴퓨팅기술, 이동메이트

황치곤(Chi-gon Hwang)
1995년 창원대학교 경영학과(학사) 2004년 광운대학교 정보통신학과(공학석사) 2006년 ~ 현재 전자넷 연구원 ※ 관심분야: 웹서비스, XMDR, 그라드컴퓨팅, 이동메이트, 상호운용성

최영근(Young-keun Choi)