DBC Editor: 데이터베이스 컴포넌트 에디터

서지원*
*고려대학교 소프트웨어공학과
e-mail : cozyplace@korea.ac.kr

DBC Editor: Database Component Editor

Ji-Won Seo*
*Dept. of Software Engineering, Korea University

요 약

다수의 개발자들이 참여하는 프로젝트에서 개발자들의 필요에 의해 동일한 기능의 유사한 코드가 생성된다. 이것은 시스템 전체적으로 중복 코드가 생성되어 시스템 성능에 영향을 준다. 또한 개발자의 능력에 따라 화면의 응답속도 및 데이터베이스 처리 속도에 차이가 나게 되어 결국 시스템에 악영향을 끼치게 된다. 이러한 문제는 개발자들의 의사소통을 위한 표준 개발 도구를 활용함으로써 해결 가능하다. DBC Editor 는 이클립스 플러그인 기반으로 기동되어 OS 에 영향을 받지 않고 기능별로 독립적인 플러그인을 구성하여 확장 가능하다. 또한 쿼리만을 관리하므로 시스템이 유연해지며 개발자들은 표준화 된 코드개발이 가능하여 개발 생산성을 향상시킬 수 있다.

1. 서론

최근 정보통신 기술의 획기적인 발달과 인터넷의 보급에 따라 이를 기반으로 하는 모든 산업 분야의 형태를 바꾸어 놓았다. 각각의 분야에 이르는 구성원들의 관계가 보다 유기적이고 긴밀한 관계로 만들어져 다양한 정보를 보다 빠르고 쉽게 공유 할 수 있게 되었다. 이러한 흐름은 인터넷을 기반으로 언제 어디서나 어떠한 Network Connection 상에서도 High Performance 를 제공하는 서버기반 컴퓨팅 기술이 산업의 여러 분야에 적용되면서 더욱 빨라지고 다양해지는 모습을 나타내게 되었는데, 주변 환경이 빠르게 변화하고 비즈니스의 요구사항이 복잡해지면서 사내 정보 시스템에도 새로운 비즈니스 요구사항을 신속하게 반영해 주길 원하고 있으며 심지어 기존에 운영되어 제공되는 업무 프로세스의 기능 조차 변경을 요구하는 것이 다반사다. 이로 인해 정보시스템을 개발 및 운영하는 측면에서 서버 어플리케이션 개발이나 유지보수에 소모되는 비용이 증가하게 되어 사내 시스템에 다양한 방법론과 개발 도구를 적용하여 비용절감을 위한 개발 및 운영 업무를 수행하려고 한다.
자사의 개발 방법론을 적용하여 다양한 함수를 라이브러리로 만들어 놓고, 성능을 고려한 Pro*C 프로그램과 C 프로그램을 결합하여 하나의 서버 어플리케이션으로 사용하는 시스템에서는 개발자의 능력에 따라 개발 화면의 응답속도 및 데이터베이스 처리 속도는 차이가 나게 되어 결국 시스템에 악영향을 끼치게 된다. 이렇게 Pro*C 를 기반으로 하는 시스템에서 개발자가 비표준 SQL 사용으로 데이터베이스 성능을 저하 시킬 수 있는 개발 환경에서 개발 생산성과 유지보수 향상을 위하여 이클립스 플러그인 기반의 DBC Editor 를 제시한다. 본 논문에서 제시한 DBC Editor 는 이클립스 플러그인 기반으로 기동되어 OS 에 영향을 받지 않고 기능별로 독립적인 플러그인을 구성하여 확장 가능하며 쿼리만을 관리하여 비즈니스 로직과 쿼리를 분리하여 시스템이 유연해지며 개발 생산성을 향상시킬 수 있다. 본 논문의 구성은 다음과 같다. 2 장에서는 본 논문에서 제시한 데이터베이스 접근 표준 컴포넌트에서 사용한 기반 기술인 이클립스 플러그인 기술과 Pro*c 기반의 어플리케이션 프로그램 그리고 DB I/O 모듈 생성 엔진에 관한 기존 연구를 살펴보고 3 장에서는 본 논문에서 제시한 DBC Editor 의 설계 및 구현에 대해 설명하였다. 4 장에서는 본 논문의 결론과 앞으로의 발전 방향에 대해 기술한다.

2. 관련연구

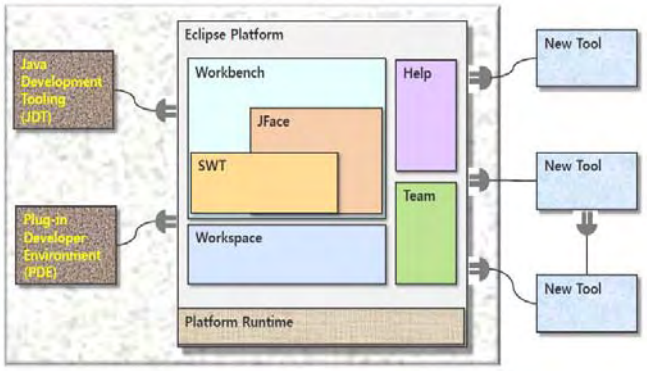
2.1 Pro*c 의 정의와 특징

SQL 은 절차형 언어가 아니다. SQL(Structur query language) 언어는 4 세대 언어로써, 3GL 언어가 절차식 언어인데 반해서, 4 세대 언어는 좀더 자연어에 근접한 언어적 특성을 가진다. 그래서 오라클을 포함한 많은 데이터베이스는 PL/SQL 이라는 절차형 언어를 제공한다. PL/SQL 은 오라클 내부에서 실행되는 프로그램으로서 오라클 내부라는 한정된 공간에서 실행되는 도구이다. 하지만 오라클 엔진에 부하가 가해진다면 이는 돌이킬 수 없는 장애로 이어질 수 있다. 이

렇듯 내부 PL/SQL 을 통한 프로그램의 경우 오라클이라는 내부 울타리에서만 수행되는 특징이 큰 제약으로 작용한다. 이런 고민을 해결하기 위해서 대부분의 DBMS 벤더는 외부 C 프로그램과 결합할 수 있는 선행 컴파일러를 제공하고 있으며 오라클에서는 이를 Pro*C 라고 한다. Pro*C 는 PL/SQL 과 같이 절차적 프로그래밍이 가능한 프로그램 도구로서 PL/SQL 처럼 오라클 내부에서 수행되는 프로그램이 아니라 실행 가능한 외부 프로그램으로 작성이 되어 관련 작업을 수행할 수 있게 해주는 도구이다. 좀 더 높은 레벨에서 C 언어와 SQL 언어를 결합해서 프로그래밍 할 수 있는 환경을 제공하고자 하는 것으로써, Pro*C 의 소스코드를 보면 알 수 있겠지만 복잡한 API 들 대신에 SQL 의 문장을 그대로 사용할 수 있다.[1]

2.2 이클립스 플러그인

이클립스(Eclipse)는 자바를 비롯한 다양한 언어를 지원하는 프로그래밍 통합 개발 환경을 목적으로 시작하였으나, 현재는 OSGi 를 도입하여, 범용 응용 소프트웨어 플랫폼으로 진화하였다. 이클립스 플랫폼은 이클립스 프로젝트에서 개발한 확장 가능한 개발 플랫폼으로, 개발 API 를 오픈하여 사용자들이 이클립스 플랫폼에 다른 기능들을 쉽게 추가할 수 있다. 즉, 이클립스 플랫폼은 런타임 커널을 제외한 모든 것을 플러그인의 모습으로 광범위한 기능의 확장을 지원함으로써 기존의 플러그인을 다른 플러그인이 쉽게 재사용할 수 있도록 설계되었다. 그림 1 은 이클립스 플랫폼 구조의 아키텍처이다. 플랫폼 런타임(Platform Runtime)은 플러그인을 동적으로 발견하고 플러그인 및 확장점에 대한 정보를 관리한다. 워크벤치는 UI 컴포넌트를 추가하기 위한 확장점을 정의하고 세부적으로는 Editor, View, Perspective 로 구성되어 있다. 또한 UI(user interface)를 빌드하기 위한 추가 툴킷인 SWT 및 Jface 을 제공한다. 그림 1 에서 보는 것처럼 이클립스는 외부의 플러그인 뿐만 아니라 내부의 이클립스 위크벤치 구동 시에도 각 기능의 플러그인들을 붙여서 실행한다. 각각의 플러그인이 내부적으로 완전히 독립성을 가지고 있어서 새로 추가된 기능들을 마치 플러그에 연결한 것처럼 플랫폼 전체를 다시 빌드할 필요없어 무한한 확장성과 유연성을 가지게 된다.[2][3]



(그림 1) 이클립스 플러그인 구조

2.3 DB I/O 모듈 생성 엔진에 관한 연구

[4]의 연구는 Application 의 개발 및 운영의 유연성, 효율성 확보에 중점을 두고 data 의 효율적인 처리를 위한 DB I/O 모듈 제작을 개발자 수작업으로 수행하기 보다는 생성엔진의 설계 및 구현을 통한 자동화 시스템을 보여주고 있고 생성엔진 구성도 및 이를 활용한 제작절차에 대한 세부적인 내용을 제시하였다. 이 연구는 대형 금융회사에서 DB/IO 모듈 제작에 실질적인 활용을 통하여 개발 및 운영 생산성 향상뿐만 아니라 SW 품질 향상에 기여한 것으로 분석되었다.
이처럼 DB I/O 생성엔진을 활용하면 개발언어의 종류에 따라 개발된 여러 도구들은 생산성 증대에 기여하기 때문에 개발비용을 줄일 수 있다. 본 논문에서는 이 연구에서 제시한 DB I/O 생성엔진을 활용하여 개발자의 역량차이로 인한 생겨날 수 있는 개발 생산성 저하를 극복하기 위한 방안으로 DBC Editor 개발 도구를 제시한다.

3. DBC Editor 설계 및 구현

3.1 플러그인 기반의 컴포넌트 에디터

본 논문에서 제시하는 DBC Editor 는 다음과 같은 기준을 만족하도록 설계 구현 하였다.

- 플러그인 개발을 위해 Text 기반 에디터를 사용
- XML 스키마를 사용하여 모델을 생성
- Template 엔진을 사용하여 소스 생성
- DBMS 에 독립적인 어플리케이션 구축
- 비 숙련된 개발자도 쉽게 DB 처리프로그램을 구현 가능
- 표준 SQL 사용으로 DB Access 안정성 및 성능 저하 방지

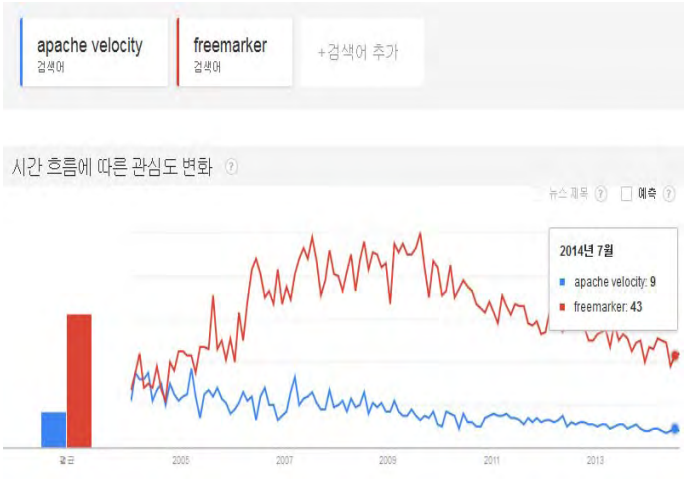
위와 같은 기준으로 컴포넌트 에디터를 구현하기 위해 먼저 XML 스키마를 사용하여 모델을 생성할 때는 JAXB 모델로 변환한 후에 소스코드를 생성한다. EMF 모델을 사용하지 않고 JAXB 모델을 선택한 것은 서버와 데이터 전달, 변환 등을 용이하게 하기 위해 고려하였다. [5][6]

```
<?xml version="1.0" encoding="UTF-8"?>
<schema targetNamespace="http://www.example.org/DBResourceModel"
    elementFormDefault="qualified" xmlns="http://www.w3.org/2001/XMLSchema"
    xmlns:tns="http://www.example.org/DBResourceModel" xmlns:Q1="http://www.exam

    <complexType name="DBResourceModel">
        <attribute name="name" type="string"></attribute>
        <attribute name="typeDetail" type="tns:DBModuleTypeDetail"></attribute>
        <attribute name="group" type="string"></attribute>
        <attribute name="description" type="string"></attribute>
        <attribute name="detailInformation" type="string"></attribute>
        <attribute name="creator" type="string"></attribute>
        <attribute name="owner" type="string"></attribute>
        <attribute name="lastModifier" type="string"></attribute>
        <attribute name="createdTime" type="string"></attribute>
        <attribute name="updatedTime" type="string"></attribute>
    </complexType>

    <simpleType name="DBModuleTypeDetail">
        <restriction base="string">
            <enumeration value="SELECT"></enumeration>
            <enumeration value="INSERT"></enumeration>
            <enumeration value="UPDATE"></enumeration>
            <enumeration value="DELETE"></enumeration>
        </restriction>
    </simpleType>
</schema>
```

(그림 2) XML 스키마 정의

이 소스코드는 스키마에 맞도록 디자인된 자바 클래스형태의 모델이라 이 클래스들을 컴포넌트 에디터에

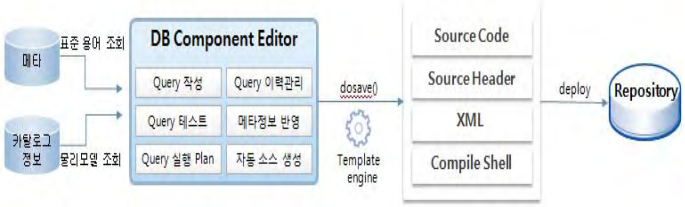
서 원하는 형태로 사용할 수 있다. XML 스키마 정보에 의해서 모델에 대한 소스가 자동 생성됨으로 스키마 메타 정의가 중요하다. 그림 2 는 XML 스키마를 사용하여 메타모델을 직접 정의 하는 방법이다. [7] 플랫폼을 구성하는 Editor Class 들 대부분이 텍스트를 기반으로 하는 Editor 로 구현되어 있어 DBC Editor 또한 텍스트를 기반으로 개발하였다. 플러그인 개발을 위한 Editor 는 기본적으로 소스를 기반으로 편집기의 동작을 정의하는 IEditorPart 인터페이스를 구현하는 클래스에서 찾을 수 있고 EditorPart 클래스는 일반적으로 Abstract Editor Class 를 사용한다. 또한 확장성을 고려해 MultiPageEditorpart 를 상속해 구현하였고 DBC Editor 에서 변환된 소스와 헤더파일 등을 확인할 수 있다. Eclipse 에서 에디터를 저장을 할 경우, doSave() 메서드가 호출되어 템플릿을 통한 소스변환, 네이게이터 연결, XML 포맷 저장 등의 기능들은 해당 메서드 안에서 원하는 동작을 구현하였다. 객체를 생성할 때에는 직접 인스턴스를 만들 수도 있고 다른 Object Factory 를 이용하여 생성할 수도 있는데 에디터에서는 팩토리를 사용하여 소스를 구현하였다. [8]



(그림 3) 실시간 구글 트랜드[9]

에디터에서 자동으로 소스 변환하기 위해서 템플릿 엔진을 사용한다. 템플릿 엔진은 우선 만들어 놓은 템플릿에 실행 시 Parameter 에 값을 던져주고, Data 를 생성해 주는 엔진이다. 현재 가정 널리 사용되는 템플릿 엔진으로는 Apache Velocity 와 FreeMarker 가 있다. 그림 3 은 실시간 구글 트랜드 분석 결과이다. 전반적으로 FreeMarker 의 인기가 Velocity 을 앞서고 있다. 트렌드가 품질을 보장하지는 않지만 많은 사람들이 FreeMarker 에 더 많은 관심을 가지고 있다는 점은 확실하다. DBC Editor 는 이 트렌드와 정렬 기능 등을 지원하는 FreeMarker 를 사용했다.[10] 템플릿 엔진을 사용하면 프로젝트에서 개발자가 직접 embedded SQL 프로그래밍을 하지 않고 DBC Editor 를 통해 SQL 을 저장하면 미리 표준으로 정의된 템플릿과 데이타 모델를 적절히 맵핑하여 소스코드와 컴파일 쉘을 생성한다. 소스코드는 C 언어와 SQL 이 결합된 Pro*c 소스코드이며 개발 표준에 맞게 자동 생성 및 관리가 가능하여 표준에 맞는 품질 높은 AP 프로그램 코드가 자동 생성된다.

에디터 설계를 통해 그림 4 와 같은 플러그인 기반의 DBC Editor 를 구현하였다. 이 에디터는 다음과 같은 특징을 가지고 있다. DBC Editor 를 이용하여 프로젝트를 할 경우 표준화된 개발방법을 통하여 개발생산성을 높일 수 있고 안정적인 DB 조작과 쿼리의 성능을 확보할 수 있는 특징이 있다. 개발자가 직접 embedded SQL 프로그래밍을 하지 않기 때문에, 특정 DBMS vendor 제품에 의존적이지 않게 개발할 수 있다.

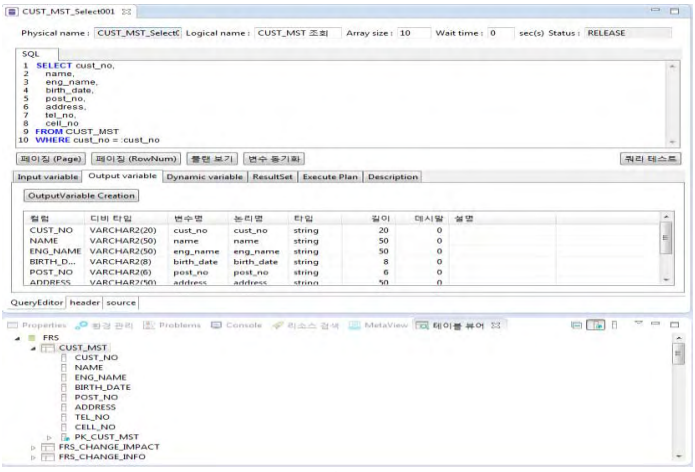


(그림 4) 에디터 아키텍처

3.2 DBC Editor 주요 기능

- DML Query 작성 및 등록
- Query 검색 및 테스트
- Query Plan 정보
- Query Lock에 대한 timeout 시간 설정 기능
- Query 이력 관리 및 복구
- DB 카탈로그 메타정보 반영
- SQL 소스 자동 생성
- Embedded와 Dynamic SQL을 자동 구분

DBC Editor 는 비 숙련된 개발자에게 편리한 SQL 편집을 도와준다. 개발자는 에디터에서 Query 를 생성하고 튜닝하고 저장하는 동작만으로 자동 생성 기능을 이용하여 프로그램을 완성할 수 있다. 또한 SQL 의 테스트 및 실행플랜보기 등의 SQL 작성 및 테스트에 필수적인 기능들을 지원한다. 그리고 개발된 Query 는 Repository 에서 관리되므로 Query 를 관리할 수 있는 기능을 제공한다. 또한 Embedded 와 Dynamic SQL 을 자동 구분하므로 개발자가 별도로 구분할 필요 없이 동일한 UI 를 사용하여 개발 편의성을 향상시킬 수 있다.



(그림 5) 에디터 화면

4. 결론

본 논문에서는 이클립스 플러그인 기반으로 하여 DBC Editor 를 설계 개발하였다. 에디터의 코드 자동 생성 기능을 활용함으로써 더 빠르게 개발할 수 있을 것이며 개발 생산성 향상을 기대할 수 있다. 또한 플러그인을 활용하면 기능의 확장이 용이하고 에디터에 필요한 다양한 기능들을 더 유연하게 추가할 수 있을 것이다. 이것은 개발자들이 표준화 된 코드 개발을 가능하게 하고, 나아가 유지보수성이 개선시킬 수 있다.

참고문헌

[1]주경호, "ORACLE PRO C 실무 프로젝트 활용서", 비팬북스, 2009
[2]Eclipse, http://www.eclipse.org
[3]Eclipse Platform Plug-in Developer Guide, http://help.eclipse.org/juno/index.jsp
[4]정재군 외 2 명 , “대형 금융정보 Data 처리를 위한 DB I/O 모듈 생성엔진 개발”, 한국정보과학회 추계학술발표대회, Vol.36, No.2(B) , 2009.
[5]http://www.eclipse.org/modeling/emf/
[6]JAXB Tutorial, https://jaxb.java.net/tutorial/
[7]XML Tutorial, http://www.w3schools.com/xml/
[8]양석호, “이클립스 실전 플러그인 개발”, 에이콘, 2010.
[9]구글 트렌드, http://www.google.co.kr/trends/explore#q=apache+velocity,freemarker
[10]FreeMarker Manual, http://freemarker.org/docs/index.html