CASE 자료 형식으로부터 CDIF 형식으로 변환하는 프로그램 자동 생성기의 구현

배 상 현† · 남 영 광†† · 신 규 상†††

요 약

미국의 전자산업협회 (Electronic Industries Association)에서는 1991년부터 CASE 도구간의 정보교환을 용이하게 하기 위하여 CDIF (CASE Data Interchange Format) 이라는 표준 형식을 제정해 오고 있다. 각 CASE 도구에서 사용된 데이터를 CDIF 형식으로 변환만 해주면 다른 CASE 도구에서 이를 자신에 맞는 형식으로 변환하여 사용할 수 있다. 이때 각 CASE 도구 공급자는 CDIF 형식으로 변환 혹은 CDIF 형식을 자신에 적합한 형식으로 변환하는 프로그램을 작성하여야 한다.

본 논문에서는 이러한 불편을 해소하기 위해서 CASE 도구 공급자가 각 CASE 도구에서 사용된 자료의 저장형식을 지정하여 주면 CDIF 형식을 지원하는 변환기의 소스코드를 자동으로 생성하는 생성기을 개발하였다. 이 자동 생성기는 CASE 도구에서 사용된 자료의 저장형식, 저장형식에 대한 메타정보, CDIF 메타정보를 입력받아서 CDIF 문장을 생성한다. 본 연구에서 개발한 자동 생성기는 정보저장소에 관계없이 단일화된 인터페이스를 제공하는 JDBC 및 Oracle과 Sun O/S 환경하에서 구현되었다.

Implementation of An Automatic Program Generator for Transforming CASE Data Format into CDIF Format

Sang-Hyun Bae† · Young-Kwang Nam†† · Gyu-Sang Shin†††

ABSTRACT

Electronic Industries Association has set a standard format called CDIF(CASE Data Interchange Format) for exchanging information between CASE tools from 1991. If data used on a CASE tool is represented in CDIF format, then any other CASE tools can use the data by translating CDIF format into their own format. In order to do so, each CASE vendor must provide a translator program from CDIF to its own data format or from its own data to CDIF format.

In this thesis, we have implemented a source code generator for translating CASE data format into CDIF format only if they provide how data is stored. This automatic program generator generates CDIF statements using data format used in CASE tools, meta information about stored data and CDIF meta information. In the thesis, the automatic program generator has been implemented with Java and Oracle DBMS with JDBC interface in Sun O/S environment.

1. 서 론

현재의 CASE(Computer Aided Software Engineering) 도구는 공급자만의 고유한 저장형식으로 정보를 저장하기 때문에, 한 CASE 도구에서 작성한 정보를 다른 CASE 도구 사용자는 직접 사용할 수가 없다. 다른 CASE 도구에서 작성한 정보를 사용하기 위해서는 CASE 도구 공급자가 다른 CASE 도구의 저장형식을 지원해야 한다. 이것은 CASE 도구 공급자의 입장에서 볼 때, 다른 CASE 도구 저장형식을 지원하기 위해서

※ 본 논문은 1998년도 연세대학교 교내학술연구비의 지원에 의하여 이루어 졌음.
† 정 회 원 : 핸디소프트 시스템 엔지니어
†† 정 회 원 : 연세대학교 전산학과 교수
††† 정 회 원 : 한국전자통신연구원 책임연구원
논문접수 : 2000년 6월 19일, 심사완료 : 2000년 12월 21일

매번 새로운 모듈을 추가해야 하는 부담을 가지게 된다.

이런 문제점을 해결하기 위해서 미국 전자산업협회(Electronic Industries Association)에서는 1991년부터 CASE 도구 사이에 정보 공유를 원활하게 하기 위한 방법의 일환으로서 CDIF(Case Data Interchange Format)라는 표준 문서 양식을 제정해오고 있다. CDIF는 두 개의 CASE 도구간에 데이터를 교환하기 위해서 사용하는 변환의 구조와 내용을 정의한다. 만약 각 CASE 도구들이 CDIF 형식을 지원하면, 서로 공유하고자 하는 정보를 CDIF 형식으로 변환하여 상대방의 CASE 도구의 자료 저장 형식을 모르더라도 각 CASE 도구들이 CDIF 형식의 정보를 읽어서 정보를 공유할 수 있다. 그러면 각 CASE 도구 공급자들은 각 공급자가 만든 CASE 도구의 정보를 공유하고자 하는 CASE 도구 저장형식에 맞게 데이터 저장 방법을 유지하지 않고도, CDIF 형식만 지원하면 모두 원하는 데이터를 사용할 수 있으므로 새로운 모듈의 개발하여야 하는 추가적인 부담을 줄일 수 있다.

본 논문에서는 자료 저장형식이 바뀔 때마다 모듈을 재작성 해야 하는 번거로움을 없애기 위해서, 사용자가 임의로 자료 저장형식을 지정하기만 하면, CDIF 형식을 지원하는 모듈이 자동적으로 생성되도록 하는 프로그램에 대해 기술한다. 그 기능은 C 소스 코드를 자동으로 생성해주는 Lex&Yacc[1], JAVA 소스 코드를 자동으로 생성해주는 javacc[2]와 같은 역할을 한다.

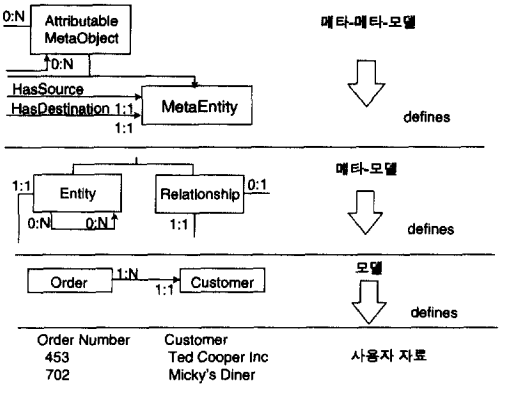
2. CDIF의 구조

2.1 CDIF의 정보체계

CDIF는 두 개의 CASE 도구간에 데이터를 교환하기 위해서 사용하는 변환(transfer)을 사용한다. 이 때 변환정보 내용을 정의하기 위해서 모델을 사용하였다.

(그림 1)은 CDIF에서 사용하는 메타계층(meta-layer)을 보여준다[3]. 두 번째는 첫 번째의 추상개념이고, 세 번째는 두 번째의 추상개념 등이다. 메타계층이라 불리는 각각의 추상개념은 그것이 어느 계층으로부터 추상화되었는가에 대한 규칙을 정의한다. CDIF를 통해 두 개의 CASE 도구간에 교환되는 정보는 "모델(model)"로 정의하였다. 모델은 사용자 데이터에 관한 기술이다. 모델은 변환에 저장되므로 해서 정보를 CASE 도구사이에서 공유된다.

이와 같은 데이터 구조의 추상적 표현은 ERA(Entity-Relationship-Attribute) 모델링을 사용해서 정의한다. 이 표현은 모델에서 발견된 데이터 항목에 대한 타입 정의인 데이터 항목을 포함한다. 예를 들면 개체, 데이터 형, 프로세스와 같은 것들이다. 이것을 "메타-모델(meta-model)"이라 한다. 왜냐하면 데이터에 대한 데이터를 포함하고 있기 때문이다. 메타-모델을 설명하기 위해, 메타-데이터 구조(다이어그램 형, 정의, 표기법, 기술과 구문)에 대한 합의된 세트(set)를 정의한다. 이 세트를 "메타-메타-모델(meta-meta-model)"이라 한다.



(그림 1) 메타 계층

CDIF 구조의 기본 개념은 어떤 정보가 변환될 것인가에 대한 정의와 어떻게 변환 될 것인가에 대한 정의를 분리하여 구체화하였다.

CDIF 변환의 정보내용은 메타모델에서 정의한다. CDIF는 'CDIF 통합 메타모델'이라 불리는 표준화된 메타 모델을 제공한다. 변환 형식은 정보가 변환을 위해서 어떻게 표준화된 형태로 삽입되는지에 대한 정의를 제공한다. CDIF에서 변환 내에 변환되는 정보의 정의와 CDIF의 다중변환 형식에 대한 지원과 실제 변환 형식 사이에 명확한 분리 때문에, 변환의 의미(semantic)정보에 대한 구체적 정의가 필요하다. 정의는 구문단계에서 할 수 없다. CDIF 통합 메타-모델에 포함된 의미내용은 ERA 모델링을 일부 변화시켜서 정의한다[4].

CDIF 통합 메타-모델의 목적은 CASE 도구에 있어서 필요한 모든 정보의 정의를 제공하는 것이다. 그러나, 이것은 크기와 복잡성, 그리고 소프트웨어 공학이 끊임없이 발전하고 변하고 있기 때문에 한 개의 표준

으로 만들어질 수 없다.

CDIF 통합 메타-모델에서 취한 방법은 각 CASE도구에 대해 각각의 정의를 제공하는 것이다. 각각의 영역은 표준 CDIF 조직에 포함되는 다른 모든 영역에 대하여 모르더라도 사용이 가능하도록 하였다. 이것을 "주제영역(subject areas)"이라 부른다[3]. 각각의 주제영역은 시스템 개발 생명주기 중의 특정한 측면에 관련되거나 혹은 비슷한 기능을 제공하는 컬렉터블 메타-오브젝트(collectable meta-object)들을 그룹으로 묶는다[3].

여러 주제 영역들은 동일한 메타-오브젝트를 공유할 수 있고, 전체 CDIF 통합 메타-모델을 정의하는 표준의 부분 세트를 구성하는 분리된 표준으로 정의된다. 현재 약 11개의 주제영역들이 표준으로 정의되어있고, 새로운 영역들에 대해서 표준화가 진행중이다.

각 주제영역에서 포함된 메타-엔티티, 메타-속성, 메타-관계는 포함된 영역에서 사용된 방법에 의해 필요한 일반적인 개념을 지원한다. 확장 메커니즘은 도구 구축자가 비슷하거나 다른 도구사이에 추가적 정보가 필요할 때 정보에 대한 정의를 확장하는 방식을 제공한다.

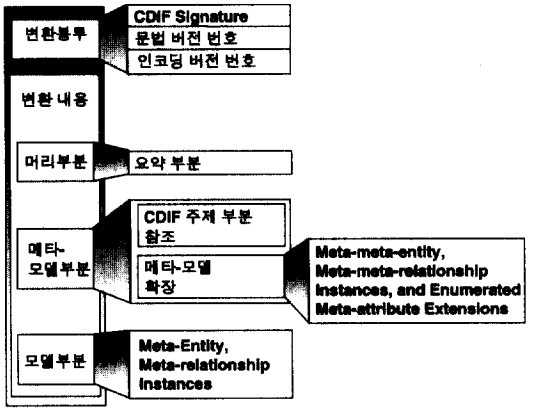
메타-모델은 모델에서 유지되는 정보에 대한 정의이다. 이 경우에 모델은 도구의 사용자가 기술하는 시스템을 표현하기 위해 CASE 도구에 의해서 사용된다.

2.2 CDIF 변환의 구조

EIA-CDIF 위원회에서는 CASE 도구 사이에 정보를 전달하기 위한 정보의 단위를 CDIF 변환(transfer)이라고 정의하였다[5].

(그림 2)에서와 같이 하나의 CDIF 변환은 변환 봉투(transfer envelope)와 변환 내용(transfer contents)으로 구성된다. 변환 봉투는 CDIF signature, CDIF 문법 버전 번호, CDIF 인코딩 버전 번호 등을 표시한다. CDIF 변환 내용은 머리 부분, 메타-모델 부분, 모델 부분으로 구분된다.

머리 부분의 CASE 도구 이름, 사용한 사람, 날짜 등의 정보가 수록된다. 메타-모델 부분은 CDIF 위원회에서 정한 표준 주제 영역 외에 추가로 사용된 정보를 의미한다. CDIF 표준의 특성은 어떠한 형태의 정보라도 ER 개념을 이용하여 표시할 수 있다는 확장성의 특징을 가진다. 따라서 이 부분에서 표준 이외의 추가 부분을 정의하여 유연성을 제공할 수 있도록 한다.

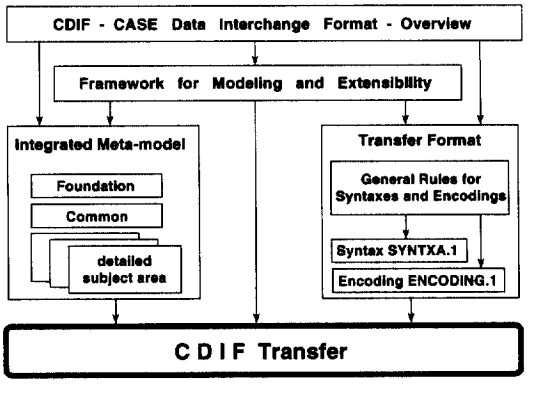


(그림 2) CDIF 변환(transfer)의 내용

모델 부분은 실제로 CASE 도구에 의해서 생성된 데이터가 들어가는 부분인데 이 데이터는 CDIF에서 정한 주제 영역의 메타-모델에 의해서 만들어진다. 따라서 본 논문에서는 모델 부분의 데이터만 고려한다.

2.3 표준 CDIF 조직

정보내용과 변환형식 표현을 정의하기 의해 필요한 정보의 구성은 (그림 3)과 같다. 이것은 표준 CDIF 조직의 내용에 대한 개괄과 표준 CDIF 조직에서 하는 역할을 보여준다[3].



(그림 3) 표준 CDIF 조직의 구조

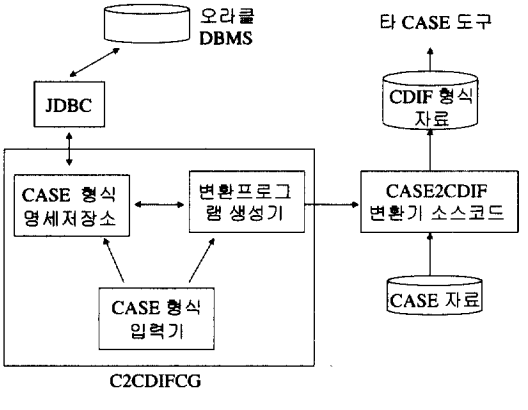
CDIF 전체적 구조(Framework)와 확장성은 CDIF 메타-메타-모델에 대한 정의를 포함하고, 표준 CDIF 조직을 통해서 사용되는 그래픽 표기와 규칙을 기술한다. 또한 전체적 구조는 CDIF 통합 메타-모델을 확장하기 위한 규칙을 정의한다. CDIF 통합 메타-모델은

개개의 표준의 연속에 의해서 정의된다. 각각의 구별되는 요소들은 분리된 주제영역을 구성한다.

표준 CDIF 변환형식은 세 가지 표준 요소로 정의된다. 첫째는 구문과 인코딩에 대한 일반적인 규칙, 둘째는 변환 구문 정의(SYNTAX.1), 셋째는 구문 정의를 사용하는 변환형식 인코딩(ENCODING.1)이다[5-7]].

3. 자동생성기의 전체 구조

CASE 형식으로부터 CDIF 형태로의 변환을 위한 자동코드 생성기(이하 C2CDIFCG(Case to CDIF Code Generator)로 칭함)의 구성은 (그림 4)와 같다. 먼저 사용자는 CASE형식입력기에 해당 CASE가 속하는 주제영역을 파악한 후 해당되는 주제영역에 맞추어서 그 CASE 도구의 저장형식을 입력한다. 이를 입력받으면 변환프로그램생성기는 이 자료와 이미 입력되어 있는 주제영역에 관한 정보를 바탕으로 하여 각각의 개체와 속성을 이용하여 프로그램을 생성한다. 변환프로그램 생성기에 의하여 생성된 CASE2CDIF 변환기는 사용하고자하는 CASE 도구의 자료를 읽어서 이를 CDIF 형식으로 변환시켜주고 이렇게 CDIF 형식으로 변환된 자료를 CDIF 형식을 읽을 수 있는 CASE 도구에서 사용할 수 있게 되어 CASE 도구간의 자료교환이 이루어진다. C2CDIFCG는 크게 세 부분으로 나뉘어진다



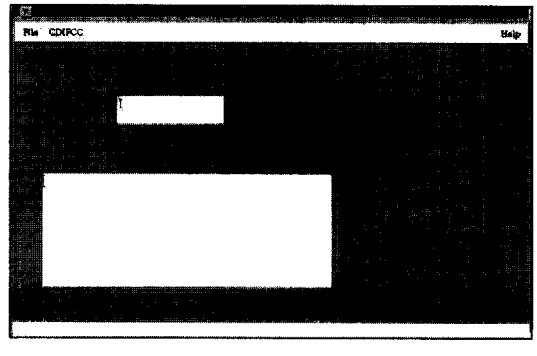
(그림 4) C2CDIFCG 구성도

3.1 CASE형식입력기

사용하고자 하는 CASE도구와 관련된 모든 입력(정보저장소에 저장된 정보의 메타정보, CDIF 형에서의 메타-오브젝트등), 출력(정보저장소에 저장된 정보의 CDIF 형식의 관계들), 이벤트 핸들링, 에러 핸들링(전송에러, 테이블이 존재하지 않는 경우)등을 입력하여 데이터베이스에 저장하는 기능을 담당한다. 이 부분을 CASE 개발자가 입력하는 부분이다.

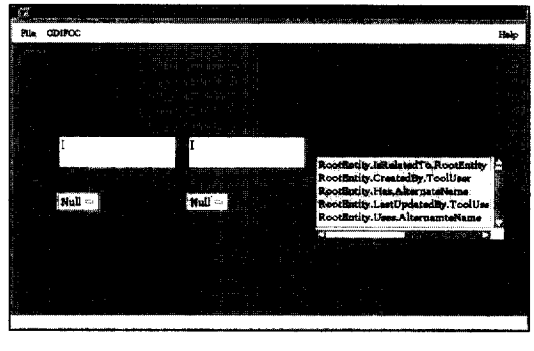
먼저 정보저장소에 저장된 데이터는 테이블 단위로 저장이 되는데, 각각의 테이블 단위로 저장된 데이터를 CDIF 형식으로 변환을 시킬 때 크게 테이블내의 데이터와 테이블과 테이블 사이의 관계를 CDIF 형식으로 변환시키기 위한 입력 창 두 개가 있다.

(그림 5)는 테이블 내에서 데이터 사이에 관계가 있을 경우 CDIF 형식으로 변환시키기 위한 입력 창으로서, 크게 정보저장소내의 테이블의 스키마를 출력해주는 부분, 정보저장소내의 테이블 이름을 입력받는 부분, 입력된 테이블이 생성되는 CASE2CDIF 변환기에서 메타-엔티티 이름을 명시적으로 선택하는 부분으로 구성된다.



(그림 5) CASE형식입력부분

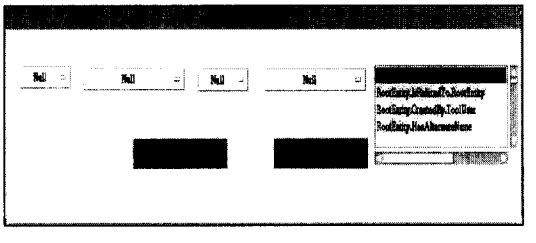
(그림 6)은 CASE 도구에서 테이블과 테이블 사이에 관계가 있을 경우 CDIF 형식으로 변환시키기 위한 입



(그림 6) 테이블 간의 관계를 위한 부분

력 부분이다. 첫 번째 테이블 이름부분은 입력된 테이블이 CDIF 형식으로 변환되었을 때 근원지 부분을 나타내고, 두 번째 테이블 이름부분은 입력된 테이블이 CDIF 형식으로 변환되었을 때 목적지 부분을 나타낸다. 각각의 선택 버튼은 각 테이블내의 속성을 나타낸다. 리스트 박스에 나와있는 관계이름은 첫 번째 테이블과 두 번째 테이블의 관계가 표준 CDIF에서 사용되는 관계이름들이다.

(그림 7)은 테이블 내의 칼럼이 CDIF에서 가지는 메타-속성을 지정하는 경우와 만약 각각의 속성이 서로 관계가 있을 경우, CDIF에서 정의하는 표준 메타-관계이름을 지정할 수가 있다. 4 개의 선택 버튼이 있고, 크게 첫 번째와 두 번째 선택버튼이 한 부분을 이루고 세 번째와 네 번째 선택버튼이 한 부분을 이루어서 두 부분으로 구분된다. 첫 번째와 세 번째 선택 버튼은 테이블내의 속성을 선택할 수 있게 한다. 두 번째와 네 번째 선택 버튼은 테이블내의 속성이 CDIF에서 속성으로서의 역할을 할 경우에 관련되는 메타-속성을 선택할 수 있게 된다. 다섯 번째 리스트 박스는 둘 사이의 관계가 CDIF에서 정해지는 관계이름을 선택할 수 있는 리스트 박스이다. 메타-관계이름이 처음에는 Null로 설정되어있지만, 다른 이름을 선택을 하면 반드시 크게 두 부분으로 나눠져 있는 것이 관계가 있음을 암시적으로 정의하는 것이므로 각각의 선택버튼에 대해서 선택을 하여야 한다.



(그림 7) 테이블의 속성들의 관계를 정의하기 위한 부분

3.2 변환프로그램 생성기

데이터베이스에 저장된 값들을 이용해서, CDIF 문장을 생성시키는 변환기인 CASE2CDIF의 소스코드를 생성하기 위한 파일 입출력을 담당한다. (그림 8)은 CASE2CDIF 소스코드에 대한 자세한 구성도이다.

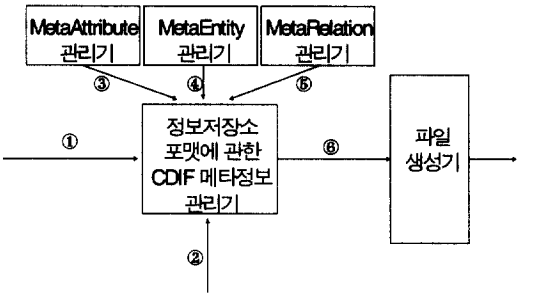
① DB 액세스를 통해 정보저장소에 있는 데이터 형식에 관한 메타 데이터를 넘겨준다 (테이블의 칼럼의 위치 등등)

② 사용자가 정보저장소에 있는 데이터 형식의 CDIF 형식의 관계를 넘겨준다 :

- 테이블 이름
- 테이블 이름에 대한 CDIF MetaEntity
- 테이블 칼럼에 대한 CDIF MetaAttribute
- 테이블과 테이블 사이의 관계에 대한 CDIF MetaRelation

③, ④, ⑤ 넘겨받은 CDIF 형식의 관계에 대한 정보를 가지고 메타-오브젝트(메타-엔티티, 메타-속성, 메타-관계)관리기에서 지정된 자료구조에 맞게끔 데이터를 가공한다.

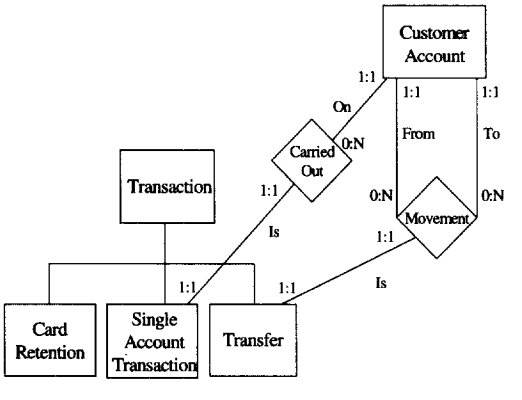
⑥ 지정된 자료구조에 맞게끔 가공된 데이터는 파일 생성기에 전달된다.



(그림 8) 소스코드 생성기

4. 변환프로그램 생성 예제

만약 개체관계도를 작성하는 것을 지원하는 CASE 도구에서 (그림 9)과 같이 개체관계도를 작성하였다고



(그림 9) 개체관계도를 그려주는 CASE 도구 예제

가정하고 이에 대한 자료의 일부분이 <표 1>과 같이 저장되었다고 가정하고 이 개체관계도의 데이터를 CDIF 형식으로 변환하는 과정을 C2CDIFCG에서 실행되는 방법을 알아보자.

위의 개체관계도에는 개체, 관계, 부개체(subentity), 사상수, 역할, 속성이 있다. "Single Account Transaction" 개체는 "Customer Account" 개체와 "Carried Out"이란 관계를 맺고 있다. "Is", "On", "To", "From"은 각각의 역할자 이름이다[13, 14]. 이런 각각의 개체, 관계, 사상수, 속성등의 정보가 정보저장소에 테이블형식으로 저장이 된다.

저장된 테이블 형식이 다음과 같을 경우에 사용자는 각각의 테이블에 대한 CDIF관계를 명시를 해야한다. <표 1>은 개체관계도에서 사용된 자료를 저장하기 위한 테이블의 일부분이다.

〈표 1〉 개체관계도에 대한 스키마

entity 테이블

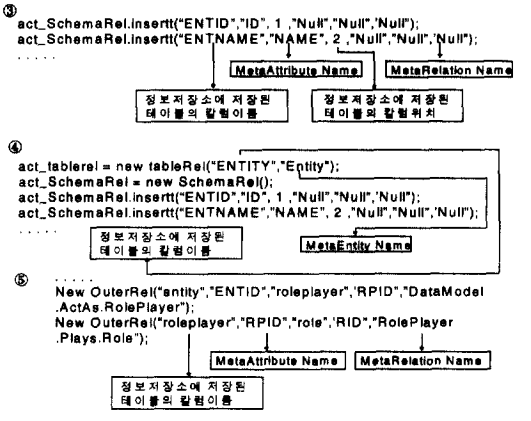
칼 럼	자 료 형
entid	NUMBER(10)
entanme	VARCHAR2(40)

roleplayer 테이블

칼 럼	자 료 형
entid	NUMBER(10)
entanme	VARCHAR2(40)

(그림 10)은 위의 예제에 대한 C2CIDFCG로 생성된 CASE2CDIF 변환기의 일부분이다. Entity 테이블의 정보를 CDIF로 변환하기 위해서 해야 할 일은 entity 테이블의 CDIF에서의 MetaEntity 이름을 부여해야 한다. Entid와 entname은 entity 테이블의 localattribute 이므로 각각의 CDIF에서 정의하는 MetaAttribute를 지정한다. 이것은 자료모델링 주제영역[12]에서 정의된 속성의 값으로 이것은 이미 생성기에 저장이 되어있다. (그림 10)의 ③, ④는 생성된 소스코드 중에서 entity 테이블에 대한 CDIF 변환하기 위한 과정을 나타낸 것이다. 같은 방법으로 roleplayer 테이블에 대해서도 위와 같은 방법을 이용한다.

다음으로 entity 테이블과 roleplayer 테이블이 서로 관계가 있을 경우, CDIF에서 정의하는 "DataModel. ActAs.RolePlayer"라는 관계를 이 둘 사이에 지정한다. (그림 10)의 ⑤가 이것을 나타낸다.



(그림 10) 메타-오브젝트 관리기로부터 생성된 소스코드

생성된 소스코드를 컴파일시키면 CASE2CDIF 변환기가 생성되고 이를 실행시키면 위의 테이블에 대한 (그림 11)과 같은 CDIF변환 파일이 생성된다. 변환된 파일 내에 숫자들은 CDIF에서 정의하는 CDIF 식별자로서 개체 자체를 식별하는 유일한 식별자이다.

```
( ENTITY 0
  (Name "Customer Account")
)
( ENTITY 1
  (Name "Transaction")
)
( ENTITY 2
  (Name "Card Retention")
)
( ENTITY 3
  (Name "Single Account Transaction")
)
( ENTITY 4
  (Name "Transfer")
)
( ROLEPLAYER 20
  (Name "Actor")
)
( ROLEPLAYER 21
  (Name "Object")
)
( ROLEPLAYER 22
  (Name "Actor")
)
( ROLEPLAYER 23
  (Name "Sender")
)
( ROLEPLAYER 24
  (Name "Receiver")
)
```

```
(DataModel.ActAs.RolePlayer R1 0 20)
(DataModel.ActAs.RolePlayer R2 1 21)
(DataModel.ActAs.RolePlayer R3 2 22)
(DataModel.ActAs.RolePlayer R4 3 23)
(DataModel.ActAs.RolePlayer R5 4 24)
```

(그림 11) CASE2CDIF변환기에서 생성된 결과

5. 결론 및 향후 연구

본 논문에서는 CASE 자료 형식으로부터 CDIF 형식으로 변환하는 프로그램의 자동 생성기(C2CDIFCG)에 대해서 기술하였다. 현재의 CASE 도구들은 데이터 저장 형식의 불일치로 인하여 데이터 교환을 하는데 있어서 상대방 CASE 도구의 저장 형식을 지원하는 모듈을 직접 개발하지 않는 한 데이터의 호환은 불가능하였다. 상대 CASE 도구의 저장 형식을 지원하는 모듈을 개발하는 것보다는 CDIF 형식을 지원하는 모듈을 개발하는 것이 상대적으로 부담이 적다. 그러나 정보저장소의 데이터 저장형태가 조금이라도 바뀌면 CDIF 형식을 지원하는 모듈을 새로 개발해야만 한다. 본 논문에서 제시하는 C2CDIFCG를 이용하면 이러한 근본적인 문제점을 해결할 수 있게 되고, CDIF 형식을 지원하는 모듈을 개발하여야 하는 부담을 없애준다.

현재는 C2CDIFCG 내에 내재된 주제영역은 기본적인 주제영역만을 포함하고 있으나 향후 새로운 주제영역이 표준화가 되면 C2CDIFCG 내에 추가하는 일과 CASE 자료 형식으로부터 CDIF 형식으로 변환하는 프로그램의 자동생성기를 개발하였지만, 반대방향인 CDIF 형식으로부터 CASE 자료 형식으로 변환하는 프로그램의 자동 생성기의 개발 등이 향후 연구되어야 할 과제이다.

참 고 문 헌

[1] 강동헌, 박재현 역, "코아자바", 영한출판사, 1997.

[2] 배상현, 남영광, 정효택, "개체관계도 CASE 도구의 정보를 공유하기 위한 CDIF 형태로의 변환", 제 1회 소프트웨어 품질관리 심포지엄 논문집, pp.231-234, October 1997.

[3] Peter Chen. "The Entity-Relationship Model-Toward a Unified View of Data," ACM TODS, No. 1, March 1976.

[4] EIA/IS-106 CDIF-CASE Data Interchange Format-Overview, January 1994.

[5] EIA/IS-107 CDIF-Framework for Modeling and Extensibility, January 1994.

[6] EIA/IS-108 CDIF-Transfer Format-General Rules for Syntaxes and Encodings, January 1994.

[7] EIA/IS-109 CDIF-Transfer Format-Sytax SYNTAX.1, January 1994.

[8] EIA/IS-110 CDIF-Transfer Format-Encoding ENCODING.1, January 1994.

[9] EIA/IS-111 CDIF-Integrated Meta-model-Foundation Subject Area, January 1994.

[10] EIA/IS-112 CDIF-Common Subject Area, January 1994.

[11] EIA/IS-113 CDIF-Data Definition Subject Area, January 1994.

[12] EIA/IS-114 CDIF-Data Modeling Subject Area, January 1994.

[13] J. Ernst, "Data Interoperatability between CACSD and CASE tools using the CDIF family of Standards," in Proceedings of 1996 IEEE International Symposium on Computer-Aided Control System Design, 1996, pp.346-351.

[14] J. Gray, B. Ryan, "TGE Approach for Constructing Software Engineering tools," Software Engineering Conference, 1998, Australia, pp.144-150.

[15] J. Gray, B. Ryan, "Technologies and techniques for rapid CASE tool development," in Proceedings of Database Engineering and Application Symposium, 1998, pp.188-201.

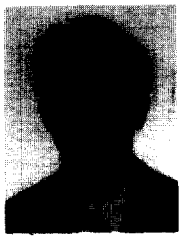
[16] J. Gray, B. Ryan, "Applying the CDIF standard in the constructions of CASE design tools," in Proceedings of Software Engineering Conference, 1997, Australia, pp.88-97.

[17] John R. Levine, Tony Mason & Doug Brown, "lex & yacc," 2nd Ed., O'Reilly & Associates, Inc., October 1992.

[18] M. Raingruber, W. Gregory, "The Data Modelling Handbook," John Wiley & Sons, 1994

[19] George Reese, "Database Programming with JDBC and JAVA," O'REILLY.

[20] Sriram Sankar, Rob Duncan, Sreenivasa Viswanadha, "http://www.suntest.com/JavaCC," April 1996.



배 상 현

email : cherrie@hanmail.net

1996년 연세대학교 전산학과 졸업(석사)

현재 핸디소프트 시스템 엔지니어

관심분야 : 소프트웨어공학

남 영 광

email : yknam@dragon.yonsei.ac.kr

1982년 연세대학교 수학과 졸업(학사)

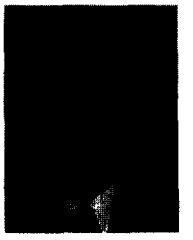
1985년 한국과학기술원 전산학과 졸업(석사)

1992년 노스웨스턴대학교 전산학과 졸업(박사)

1992년~1994년 시스템공학연구소 선임연구원

1995년~현재 연세대학교 전산학과 부교수

관심분야 : 소프트웨어공학, 프로그래밍언어, 정보검색



신 규 상

email : gsshin@etri.re.kr

1981년 성균관대학교 통계학과 졸업(학사)

1983년 서울대학교 대학원 계산통계학과 졸업(석사)

1983년 시스템공학연구소 연구원

1987년 시스템공학연구소 선임연구원

1997년 시스템공학연구소 책임연구원

1999년 충남대학교 대학원 컴퓨터과학과 박사과정 수료

현재 한국전자통신연구원 컴퓨터·소프트웨어기술연구소 소프트웨어공학연구부 컴포넌트공학연구팀 팀장

관심분야 : 컴포넌트 기반 개발방법론, 컴포넌트 개발 지원도구, S/W 재공학, CASE 도구, 객체지향 기법