# 워크플로우 시스템에서 XML을 이용한 실패 처리

권순덕 이단영 이원조 고재진
울산대학교 컴퓨터 정보통신공학부
{trueguy ,danyoung64, wonjo, jjkoh}@cic.ulsan.ac.kr

# Using XML For A Failure Handling in WorkFlow Management System

Sun-Deok Kwon Dan-Young Lee Won-Jo Lee Jae-Jin Ko
School of Computer Engineering · Information Technology, University of Ulsan

**요 약**

워크플로우 시스템의 전체적인 성능을 향상시키기 위해서 실패 처리의 방법이 무척이나 중요하다. 이런 문제를 최근에 각광을 받고 있는 XML를 이용해서 해결하고자 한다. XML이 가지고 있는 여러 가지의 특징 중에서 표준화와 일관성을 유지하면서 전체 워크플로우 시스템의 실패 처리를 수행하고자 한다.

## 1. 서 론

워크플로우 관리 시스템(WorkFlow Managerment System)은 비즈니스 프로세스의 자동화를 의미하며, 이때 문서, 정보, 업무 등이 한 사용자에서 다른 사용자로 미리 정해진 규칙에 의하여 전달된다 [1]. 워크플로우 관리 시스템은 기업 전체를 대상으로 하는 전체 시스템으로서 개업 내의 모든 단계에서 발생하는 업무와 기업이 운영할 수 있는 자원을 프로세스의 관점에서 통합하여 관리하게 된다. 워크플로우 관리 시스템의 통합 대상으로는 기업 전체의 인적, 물적 자원에 대한 관리 모델이 포함된다. 또한 예전에는 많은 기업이 기업 내의 전산환경에서 이루어지던 업무를 인터넷을 이용해서 많은 작업을 수행하고자 한다. 이런 흐름속에서 새로운 각광을 받고 있는 분야가 바로 XML(eXtensible Markup Language)이다. 문서의 표준화를 제시함으로써 업무의 통일성과 일관성을 제공 해 준다. 기업이나 프로세스에서 정한 문서의 구조를 가지고 작업을 처리함으로써 각 단계에서 발생되는 결과는 하나의 결과물로 사용되는 문서의 형태로 변환될 수 있다. 따라서 XML과 WorkFlow를 연계함으로써 프로세스의 흐름에서 발생되는 결과물을 쉽게 생성할 수 있다. 하지만 WorkFlow 시스템에서 발생될 수 있는 오류(Failure)에 대한 대응이 부족하여 정확성과 신뢰성 떨어진다는 점이다. 오류가 발생한 경우에 대한 대응은 필수적인 부분임에도 불구하고 대부분의 시스템들이 기능에 대한 대비가 부족하다. 이 논문에서는 XML를 이용해서 이런 오류에 따른 실패 처리 해결 방법을 제시하고자 한다.

## 2. 연구 목표 및 내용

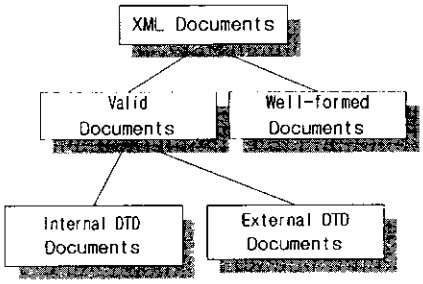
위크플로우 시스템에서 예상치 못한 에러가 발생할 경우 실패에 대한 처리는 WorkFlow System의 성능과 아주 밀접한 관계를 가지고 있다. 더욱 효율적인 실패 처리를 하기 위해서 새로운 표준을 제시하고 있는 XML를 이용해서 이런 문제를 해결하고자 한다. XML이 가지고 있는 표준화와 정형화를 이용해서 이런 문제점을 해결하고자 한다.

### 2.1 XML의 특징

XML은 문서의 컨텐트로서 데이터를 기술하는, 객체사이의 정보교환을 위한 교환포맷으로서 데이터를 기술하는, 그리고 정보의 저장과 관리를 위한 데이터를 기술하는 표준을 만들 수 있는 언어이다. 따라서 구현범위가 광범위하고 배치가 용이한 데이터의 구조적 표현을 제공한다.

- 인터넷상에서 사용할 수 있다.

- 응용프로그램의 폭 넓은 다양성을 지원한다.
- XML 문서를 처리하는 프로그램을 작성하기 쉽다.
- 문서는 인간이 읽을 수 있어야 하며, 합리적이고 명확하다.
- 문서는 만들기 쉬워야 한다.




< XML 문서의 구조 그림 1 >

### 2.1.1 DTD의 구성 요소

DTD는 다음과 같은 요소(element), 속성(attribute), 개체(entity) 등의 구성요소를 가진다.

1) 요소(Element) : 문서의 논리적 단위로 나타내며, 태그를 통하여 문서의 실제 내용부분을 마크업 한다.

| < | ELEMENT | S | 요소형 | S | 내용 | > |
|---|---|---|---|---|---|---|
| a | b | c | d | | e | s |

[ 그림 2 ] 요소(Element)

a. 마크업 선언 개방 구분자
b. DTD 요소 선언에서 사용하는 예약어
c. 구분자(seperator)
d. 요소의 공통 식별자
e. 요소의 내용
f. 마크업 선언 폐쇄 구분자

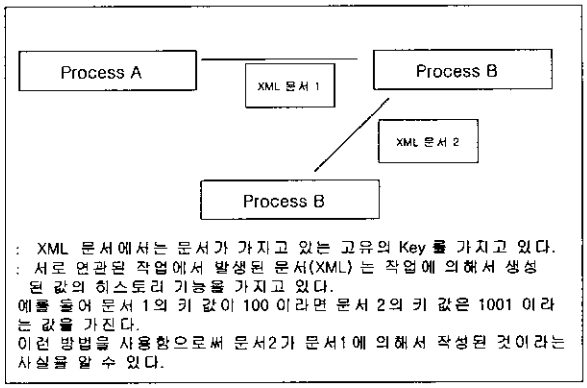
### 2.2 XML문서와 WorkFlow 시스템과의 연관성

XML의 데이터 교환 포맷은 매우 융통성이 있다. XML이 없다면 두 통신용 애플리케이션은 그들 간에 전송될 문서의 포맷, 전달될 데이터 요소 및 데이터 요소가 정렬될 순서를 사전에 정의해야 한다. 그러나 XML이 메시지 포맷인 경우에는 두 애플리케이션은 XML 구문 분석기를 사용하여 능동적으로 메시지 포맷을 해석한다. 그리고 XML 메시지 포맷은 확장이 가능 한다. XML 문서를 만들어 내는 같은 애플리케이션을 가지고 다른 애플리케이션을 지원하기 위한 추가적인 데이터 요소를 만들 수 있다. 그 문서를 사용한 원래의 애플리케이션은 영향을 받지 않는다. 기업 간 e-commerce 애플리케이션에서 XML 문서는 일반적으로 진정한 텍스트 문서라기보다는 (구매 주문서와 같은) 양식이다. XML이 특정한 교환 포맷을 정의하는데 사용되면, 전통적인 데이터 요소(즉, 고객이름, 주소, 부품 번호, 설명, 가격, 주문날짜 등)를 지시하는 XML 태그는 상대적인 데이터베이스에 저장된다. XML 문서는 일시적인 양식이다. 영원한 보존을 위해서 XML 문서는 상대적인 데이터베이스에 저장되어 있는 다양한 요소로 분리되어야 한다.

첫 단계는 비즈니스 프로세스에 관련된 정보를 설명하는 공통된 문법(어휘)을 확립하는 것이다. 어떤 데이터 요소가 필요하고 어떻게 그것들이 DTD에서 나타나는 지를 결정한 필요가 있다. 일치된 DTD의 요소들은 시스템 내에서 그런 요소들을 나타내는 DB2 데이터베이스 내의 기존 테이블과열로 매핑 되어야 한다.

일단 DTD가 정의되면 요구된 문서를 처리하는데 필요한 프로그래밍을 간단하게 하는 DB2 XML Extender의 저장된 절차를 애플리케이션 내에서 사용할 수 있다. 그 결과로 나오는 XML 문서는 Web Application Server로 인터넷을 통해서 고객의 사이트에 응답 애플리케이션 서버로 보내어진다.

XML는 기존의 WorkFlow에서 필요로 하는 값을 엘리먼트로 추출하여 XML DTD로 정의하고, 인터넷을 기반으로 정의된 태그를 사용하여 다음 단계의 새로운 값으로 사용될 수 있다 .이런 과정에서 발생하는 값들은 문서화로 표현되어서 WorFlow의 실행 결과를 알 수 있다.




< WorkFlow 와 XML의 연동 >

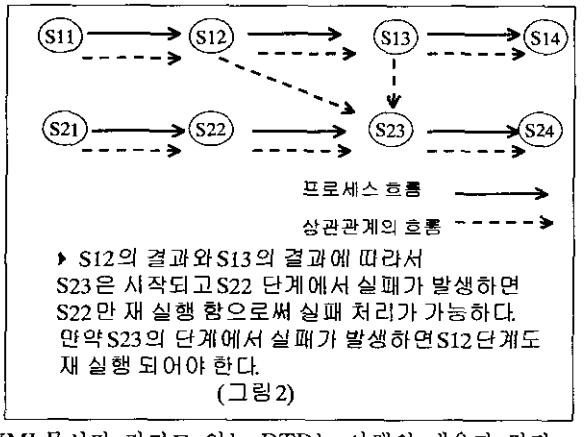
각 단계에서 필요한 문서의 DTD를 작성해서 각 프로세스에서 작성된 결과물인 XML문서를 다른 프로세스의

입력 값으로 사용함으로써 프로세스의 연관성과 프로세스의 결과를 쉽게 알 수 있다.

### 3. XML를 이용한 실패 처리 해결

(1) 상호 작용하는 프로세스

하나의 완전한 작업을 수행하기 위해서는 서로 다른 프로세스간의 상호 작용이 필요하다. S12의 작업이 완료되면 S13 작업이 수행된다. 또한 S12작업의 결과는 S23의 입력으로 사용된다. 이런 상호 보완적인 여러 작업이 완전히 완료되었을 때 비로소 하나의 작업이 끝나게 된다. 하지만 이런 작업의 흐름에서 하나의 작업이라도 에러가 발생하면 이와 연관되는 모든 작업이 수행한 결과를 원래의 상태로 돌려가야 한다. 즉 작업의 결과에서 생성된 내용의 데이터베이스를 이전의 값으로 다시 RollBack 해야 한다. 이런 연관성 있는 작업의 결과를 알기 위해서 하나의 작업의 결과로 생기는 문서(XML)를 이용할 수 있다.



(그림2)

XML문서가 가지고 있는 DTD는 아래의 내용과 같다.

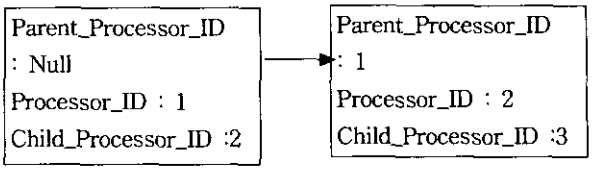
<!ELEMENT Document(Processor_ID
| Parent_Processor_ID | Child_Processor_ID) >
<!ELEMENT Processor_ID (#PCDATA) >
<!ELEMENT Parent_Processor_ID (#PCDATA) >
<!ELEMENT Child_Processor_ID (#PCDATA) >

①Processor_ID : 이 XML문서를 생성한 현재의 프로세스의 고유의 값

②Processor_ID : 이 XML문서를 생성한 현재의 프로세스 의 고유의 값

③Child_Processor_ID : 현재 프로세스와 연관이 있는 이전에 실행된 프로세스의 고유한 값

현재의 프로세스에서 에러가 발생하게 되면 XML문서의 엘리먼트의 값(Parent_Processor_ID)을 추출해서 이전 프로세스에서 생성된 모든 결과를 원래 상태로 되돌린다.




Processor_ID1에 의해서 XML1문서가 생성되고 이 때 필요한 DataBase작업이 이루어진다. 또한 Processor_ID 2 에 의해서 새로운 작업이 실행된다.

하지만 Processor_ID2를 진행하는 시점에서 실패가 발생하게 되면, Processor_ID2에 의해서 생성된 XML문서를 이용해서 상위 레벨의 작업과 현재의 작업이 어떤 일을 했는지 알 수 있다. Processor_ID1 에 의해서 변경된 DataBase값은 원래의 값을 복구한다. 부가적으로 DataBase는 이런 작업의 흐름을 알고 있어야한다. 이런 문제는 DTD에서 해결할 수 있다.

### 4. 결론 및 추후 연구 과제

XML를 이용한 WorkFlow Manager System은 기업에서 업무의 흐름을 파악하기 위해서 사용하는 문서를 자동적으로 생성하고, 사용자별로(최고 경영자, 각 부서 팀장, 혹은 사원) 각기 다른 양식의 문서로 변환이 용이하다. XML이 제시하고 있는 표준화를 각 업체에서 수용함으로써 동일 업종간의 프로세스를 공유할 수 있다. 현재로는 BtoB분야에서도 적용하고 있는 부분이다.

### 5. 참고 문헌

[1] Amit Sheth, An OverView of WorkFloe Managermanet From Process Modeling to WorkFlow Automation Infrastructure, 1995.

[2] Mohan Umesh Kamath, Improving Correctness and Failures Handing In Workflow Management Systmes , May. 1998.

[3] C. Mohan, Recent Trends in Workflow Management Products, Standards and Research

[4]Rusinkiewics, A. sheth, "Specification and Execution of Transactional Workflow," in the Modern Database Systems, 1994.