워크플로우 관리시스템에서 실패 처리의 효율적인 방법

권순덕[1] 김병욱 고재진
울산대학교 컴퓨터 정보통신공학부
{trueguy ,gom, jjkoh}@cic.ulsan.ac.kr

Effective Failure Handling in WorkFlow Management System

Sun-Deok Kwon Byung-Wook Kim Jae-Jin Ko
School of Computer Engineering · Information Technology, University of Ulsan

요 약

기업의 거대화와 전산 환경과의 접목으로 비즈니스 규칙이 프로세스로 재구성되어 지고 있다. 하나의 트랜잭션이 수행되기 위해서는 여러 상관관계를 가진 여러 개의 프로세스가 상호 작용하면서 수행된다. 각 프로세스간의 상호작용 관계를 규정하고 멱등적인 결과와 데이터의 일관성을 유지하면서 보다 효율적인 워크플로우 관리시스템을 구성하는 방법을 제시한다.

1. 서 론.

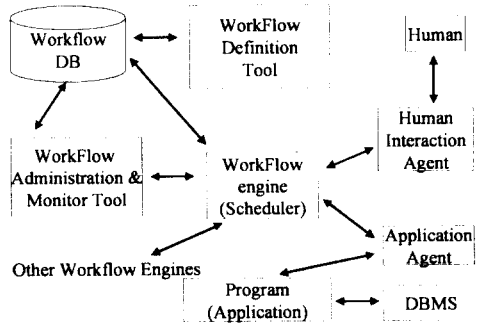
수많은 기업의 보편적인 목적은 이윤의 추구라는 것을 누구나 주지하고 있는 사실이다. 이런 기본적인 목표를 이루기 위해서는 여러 가지의 방법이 지금까지 동원되어 왔고 지금도 산업체에서 적용되고 있다. 하지만 전 근대적인 방법으로 생산성을 높이고 생산단가를 낮추는데는 한계점에 도달하게 되었다. 이런 한계에 대한 새로운 대안으로 컴퓨터 환경과의 접목이다. 비즈니스 리엔지니어링은 기존의 기업 업무를 컴퓨팅 환경의 프로세스 관점에서 재구성하여 생산성을 향상시키는 혁신 기법이다.

워크플로우 관리 시스템(Workflow Management System)은 비즈니스 프로세스의 자동화를 의미하며, 이때 문서, 정보, 업무 등이 한 사용자에서 다른 사용자로 미리 정해진 규칙에 의하여 전달된다.[3] 이러한 워크플로우 관리 시스템은 비즈니스의 요구와 정보 시스템의 변화에 따른 프로세스의 빠른 디자인과 구현을 가능하게 하며, 현재의 정보 기술이 추구하고 있는 클라이언트/서버 컴퓨팅, 다운사이징, 네트워크 컴퓨팅의 개념을 포함하고 있다.[3]

워크플로우 관리 시스템은 기업 전체를 대상으로 하는 전체 시스템으로서 기업 내의 모든 단계에서 발생하는 업무와 기업이 운영할 수 있는 자원을 프로세스 관점에서 통합하여 관리하게 된다. 워크플로우 관리 시스템의 통합 대상으로는 기업 전체의 인적, 물적 자원에 대한 관리 모델이 포함된다. 따라서, 기존에 작성된 기업에 대한 정보 시스템과 워크플로우 관리 시스템을 통합하려는 노력이 시도되고 있다.

특히, 워크플로우 관리 시스템을 현존하는 기업 내부 데이터와 연계시키고, 프로세스 데이터를 안정된 트랜잭션의 관점에서 다루기 위하여 데이터베이스 상에서 워크플로우 관리 시스템을 구축하는 연구가 시도되고 있다.

2.워크플로우 관리 시스템과 연구 목표



(그림 1)

2.1 워크플로우 관리 시스템의 구성 요소

그림1은 전체의 워크플로우 시스템이 가지고 있는 기본적인 구성을 보여주고 있다. Build-time에 비즈니스 프로세스의 분석, 모델링 및 각 단계에 대한 정의가 이루어지고, 런 타임에 프로세스 진행 및 통제를 수행한다. 프로세스 정의기(Process Definition)를 통해 Build-time에 모델링 한 프로세스는 런 타임에 인스턴스화되어 프로세스를 이루는 여러 태스크들이 수행되는데, 다양한 유형의 프로세스의 다양한 인스턴스들의 진행을 관리 감독하는 것이 워크플로우 엔진이다.

워크플로우 엔진에서는 각 프로세스의 상호 작용과 의존성을 담당하여 여러 프로세스의 흐름을 제어하는 역할을 한다. 실제로 태스크 수행을 담당하는 사용자들은 클라이언트 애플리케이션을 통해서 업무를 처리한다.

워크플로우 관리 시스템을 사용함으로써 태스크들을 조직화하고, 스케줄, 통제, 모니터링을 할 수 있으며, 프로세스에 대한 이해를 증진시키고 개선 가능성을 높일 수 있다.[5]

2.2 연구 목표 및 내용

일반 사용자는 응용 프로그램이나 툴(Tool)을 사용해서 자신이 원하는 작업을 인스턴스화 하여 워크플로우 엔진에서 수행된다. 이런 여러 가지의 작업은 각기 자신의 흐름과 순서를 기다리면서 처리된다.

하나의 프로세스가 독립적으로 실행되어서 모든 작업을 마치는 경우는 거의 없다. 하나의 프로세스는 다른 프로세스의 입력 값으로 사용되거나 다른 프로세스의 결과 값이 있어야 자신이 실행되는 것이 일반적이다.
이런 실질적인 프로세스의 상호작용에서 어떤 프로세스의 잘못된 연산이나 실패는 전체 워크플로우 관리 시스템의 구성에 예상치 못한 결과 값을 수반한다. 이번 연구에서는 워크플로우 관리 시스템의 구성에서 프로세스의 실패 처리(Failure Handling)에 대한 효율적인 방법을 제시함으로써 워크플로우 관리 시스템의 구성 전체 성능을 향상시킬 것이다.

3. 실패 처리

프로세스의 효율적인 실패 처리를 위해서는 아래에 기술한 내용을 고려해서 워크플로우 관리 시스템의 구성을 설계해야 한다.

(1). 상호 작용하는 프로세스

각기 다른 프로세스가 서로에게 영향을 주는 상관 관계를 가지고 있을 경우 프로세스는 비즈니스 룰에서 정해진 순서를 따르고 임계 영역에서는 하나만이 수행되어야 한다. 임계 영역에서 각기 다른 프로세스가 서로 같은 부분의 작업을 진행하고 있다면 데이터의 일관성, 작업의 일관성을 잃어버리게 된다.[2]

프로세스의 진행이나 상호관계는 워크플로우 관리 시스템 디자인 단계에서 정한 순서와 상호관계를 따른다. 워크플로우 관리 시스템에서는 작업 흐름, 데이터 흐름, 제어 흐름을 명시해야 하고 정해진 순서를 유지하면서 전체의 시스템이 가동되어진다[4].

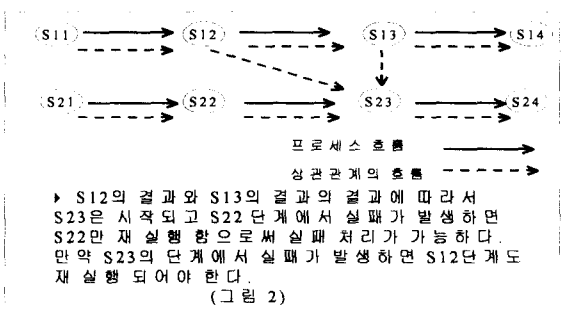
(2). 멱등적인(Idempotency) 작업 결과

같은 프로세스가 실패 처리에 의해서 여러 번 수행이 된다고 해도 한 번만 수행했을 때와 같은 결과의 유지해야 한다.

3.1 제시된 문제에 대한 해결방향

(1). 상호 작용하는 프로세스

상호 관계가 있는 작업이 같은 데이터나 자원을 공유해서 실행된다면 자원(데이터베이스, 산출되는 중간 결과 값)의 일관성을 유지하지 못하게 된다. 이런 문제에 대한 해결점으론 우선 상호 배제 실행(mutually exclusion execution)기법을 사용할 수 있다. 임계 영역을 설정한 후 하나의 프로세스만이 그 영역에서 작업을 할 수 있게 함으로써 데이터/자원의 일관성을 유지할 수 있다. 그리고 주어진 비즈니스 규칙에 의해서 전체의 작업의 흐름의 순서를 정함으로써 문제점을 해결 할 수 있다. 마지막으론 각 단계의 재실행(RollBack)의 의존성을 명시함으로써 효율적인 실패 처리를 할 수 있다.

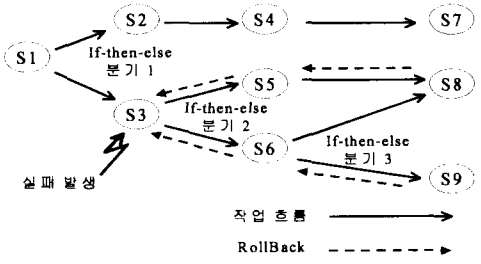


(그림 2)

어느 프로세스의 실패로 인해서 수행 중인 프로세스가 재실행되어야 한다. 그림 2에서는 재시행의 과정을 보여주고 있다. 이런 재실행에는 두 가지 형태의 재실행이 있다. 순차적인 형태로 이전에 수행된 모든 프로세스를 재실행함으로써 정확한 수행 결과를 얻어내는 방법과 필요한 부분만 재실행하는 두 가지로 분류가 된다. 전자의 방법은 알고리즘은 간단하다는 이점을 가지고 있지만, 정확하게 수행된 프로세스까지 재실행해야 하기 때문에 효율성이 떨어진다. 그리고 후자의 방법은 연관된

여러 프로세스 중에서 재실행을 필요로 하는 프로세스만을 재실행해서 전체적으로 효율성을 높이는 방법이 있다. 이 논문에서는 부분적으로 재실행함으로써 워크플로우 관리 시스템의 성능을 높이는 방법을 기술한다.

일반적으로 워크플로우 엔진은 각 단계의 인스턴트의 정보와 워크플로우 상태에 대한 정보를 가지고 있다.



(그림 3)

각 분기 단계에 해당되는 작업에 대한 정보를 워크플로우 엔진이 가지고 있다. 그림3)에서 S1에서 첫 번째의 분기가 발생하고 참이 발생하면 S2로 분기를 시작하고 거짓이 발생하면 S3이 수행하게된다.

위의 예에서

S1 = True 일 때 { S2, S4, S7 }

S1 = False 일 때

{ S3 {T, S5, S8 },{ F, S6, {T, S8}, {F, S9 }}

S3 = True 일 때{S5, S8 } ,

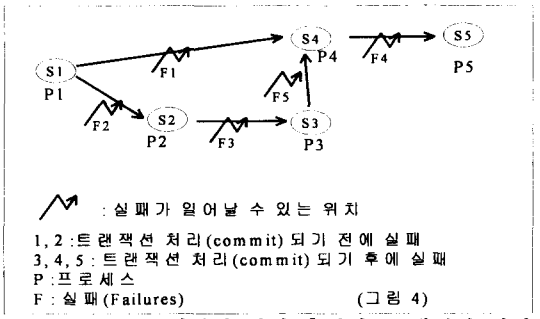
S3 = False 일 때 { S6, {T, S8}, {F, S9 }}

이라는 정보와 현재의 인스턴트의 상태를 워크플로우 엔진이 가지고 있으므로 부분적인 재 수행기능을 가능하게 한다.

(2). 멱등적인 작업 처리

어떤 단계에서 프로세스가 실패해서 다시 그 단계에서 프로세스가 재실행된다면 워크플로우 엔진은 에이젠트가 실행하지 않는 것으로 인식함으로써 에이젠트는 두 번 실행하는 결과를 가진다. 프로세스가 실패를 한다면 실행했던 모든 중간 결과는 초기화되어야 한다. 이와 같은 특성은 트랜잭션 시스템이 가지고 있는 원자적(atomic) 특성과 동일하다고 생각할 수 있다.[1]

멱등적인 문제를 해결하기 위해서는 각각의 프로세스마다 고유한 번호(ID)를 부여한다. 실패가 발생하면 워크플로우 엔진은 각 단계에서 수행한 프로세스의 ID을 체크해서 그 프로세스가 한 모든 작업을 초기화시킬 수 있다. 초기화 과정에선 두 가지의 경우로 분류가 될 수가 있다. 그림 4에서 1, 2의 경우는 트랜잭션 처리 전이고



(그림 4)

3, 4, 5경우는 트랜잭션 처리 후이다. 트랜잭션 처리 이전이면 롤백(RollBack)없이 다시 수행하고 트랜잭션 처리 이후이면 롤백을 수행한 후에 초기화를 수행한다. F5에서 실패가 발생하면 S3과 S2단계에서의 트랜잭션은 롤백 되어지고 발생한 중간 값은 초기화되어야 한다.

4. 결론 및 추후 연구 과제

본 연구에서는 워크플로우 관리 시스템에서 문제시되고 있는 일반 사용자나 경영자에 의해서 작업을 취소하거나 잘못된 작업에 대한 처리 방법으로 워크플로우 엔진이 자신의 상태와 인스턴트들의 정보를 보관, 관리함으로써 성능의 증가와 효율적인 워크플로우를 구축할 수 있는 방법론을 제시하였다.

설계 시점에서 정해진 비즈니스 규칙은 동적으로 변할 수 있다. 사용자나 경영자가 정의한 규칙이 워크플로우 관리 시스템이 운용중인 상태에서도 변경이 가능하고 변경된 규칙에 의해서 각 단계의 부분적인 재실행이나 프로세스의 상호 작용이 가능한 워크플로우 관리 시스템에 대한 연구가 필요하다.

[참고문헌]

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

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

[3] Amit Sheth , An Overview of Workflow Management From Process Modeling to Workflow Automation Infrastructure, 1995.

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

[5] SeongJoo Kim, InJun Choi, Rule Management in Workflow, 1996.