분산 워크플로우 운용 관리 도구의 메시지 구조와 흐름

이봉석[0] 강태규 김광훈 백수기
경기대학교 전자계산학과
{leanbun, tgkang, kwang, skpaik}@kuic.kyonggi.ac.kr

Message Structure and its flow of Administration Tool in Transactional Workflow Management System

Bong-Seok Lee[0] Tae-Gu Kang Kwang-Hoong Kim Su-Ki Paik
Dept. of Computer Science, Kyonggi University

요 약

워크플로우 시스템은 비즈니스 OS 라고 불리어질 만큼 복잡하고 많은 기능을 수행하고 있는 커다란 시스템이다. 최근엔 시스템의 주요 컴포넌트들이 객체 단위로 설계되며 코바와 같은 미들웨어를 통해 통신을 하는 수행구조를 갖는다. 적용 분야에 따라서 수행을 위해 생성되어진 객체들이 수만 수백만개가 동시에 존재하기도 한다. 운용 서버에서는 운용자의 요구에 따라 이러한 객체들과 통신을 해야 한다. 그러기 위해서는 운용 서버 내에서 이러한 객체들에 대한 레퍼런스들을 유지, 관리를 해야 한다. 또한 빈번한 통신으로 인해서 각각의 컴포넌트 간에 부하가 걸리는 것을 방지하거나 올바른 실행을 보장하기 위해서 메시지의 구조를 최대한 효율적으로 구성하고, 흐름을 최적화 할 필요가 절실하다. 본 논문에서는 그러한 객체들이 구성되어지는 시점부터 유지 관리되며 요구 발생시 처리되는 시점까지의 효율적인 운영을 위한 방안들을 추출하여 제안하고 구현하였다

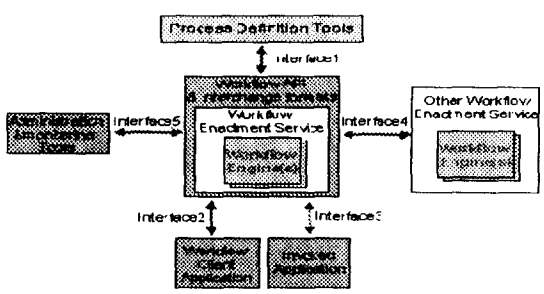
1. 서론

컴퓨터 기술과 전자통신 기술의 급진적인 발전과 이들 간의 기술적 수렴은 전자적인 작업환경이라고 하는 새롭고도 효율적인 상호 작용 지원 수단 및 방법을 잉태하게 되었다. 워크플로우 기술이란 바로 이러한 전자적 작업환경을 구현하기 위한 종합적인 연구 분야로서 궁극적으로 기존의 업무를 컴퓨터 및 네트워크를 통하여 자동화하려는 시도이다. 즉, 워크플로우 관리 시스템을 중심으로 한 조직내의 모든 업무 처리 절차의 통합을 구현함으로써 워크플로우 또는 업무 처리 절차가 주요 인프라 구조를 구성한다. 그렇지만 조직의 업무가 사람이 개입되는 문서관리 중심에서 시스템 내 프로그램이 처리하는 트랜잭션 중심으로 워크플로우 시스템이 변화하고 있다. 이를 위해 운용적인 측면에서는 업무 수행을 담당하고 있는 분산 객체와 시스템 및 자원이 효과적으로 관리가 되어야 한다. 또한 시스템의 처리가 많은 부분을 차지하기 때문에 객체간에 통신은 복잡한 구조를 형성하며 이루어지게 된다. 워크플로우에서 엔진과 함께 주요 요소로 자리잡은 운용 도구는 대부분이 엔진의 수행 객체와의 통신을 통해서 제어 및 시스템 전반적인 관리를 한다. 업무가 복잡하고 다양하며 일처리가 빈번한 업무의 형태를 지닌 조직일수록 운용자의 개입이 절실하며 이는 엔진의 수많은 수행 객체들과의 통신량이 늘어남을 의미한다. 이러한 이유로 네트워 트래픽에 영향이 미칠 수도 있고, 또한 불필요한 호출을 야기할 수도 있으며, 처리율을 저하시키는 결정적 원인이 될 수도 있다. 이러한 문제들은 효율적인 객체 레퍼런스 관리와 통신간에 사용되는 메시지의 구조 및 흐름을 최적화 함으로써 개선될 수 있다. 시스템의 전반적인 수행도 향상에 중요한 요소이다. 본 논문에서는 객체의 효과적인 관리와 객체간 통신에 사용되는 메시지의 구조 및 흐름을 정의하고 구현하였다.

2. 운용 관리 도구

2.1 정의 및 위치

운용 관리 도구는 정의된 프로세스나 실시간에 진행중인 프로세스, 또는 수행이 완료된 프로세스의 상황을 모니터링을 통해 추출된 정보를 바탕으로 사용자 또는 역할을 관리하거나 자원 관리, 프로세스의 진행에 관계하여 감독 명령하는 기능을 운용자에게 제공하는 도구를 의미한다.



<그림 1> WfMC 참조 모델

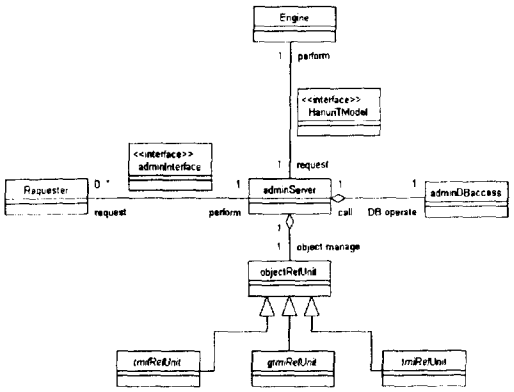
<그림 1> 과 같이 인터페이스 5 로 정의된다. 그렇지만 최근에는 운용자가 개입되는 부분이 확장됨에 따라서

엔진 뿐만 아니라 다른 컴포넌트들과의 인터페이스에 대해서도 많은 부분 연구가 진행중이다.

2.2 구성도

운용 관리 도구에는 서버부분과 실제 운용자와의 인터페이스 역할을 하는 GUI 기반의 클라이언트 부분이 있다. 서버부분에서는 클라이언트로부터의 요청을 접수하고 수행한다. 엔진부분의 여러 컴포넌트와 통신을 하며 수행에 관여를 하는 핵심 역할을 담당한다..

<그림 2> 클래스간의 관계를 나타내고 있다.



<그림 2> 운용 관리 부분의 클래스 다이어그램

- ♦ adminServer : 클라이언트 요청과 수행을 담당
- ♦ objectRefUnit 수행 객체 레퍼런스 유지, 관리 담당
- ♦ adminDBaccess : 이벤트 히스토리를 제어하거나 운용자 권한을 인증받기 위한 운용자 로그인을 위해 데이터베이스로의 접근, 연산을 담당

엔진의 수행을 담당하는 객체들은 크게 5 가지로 되어 있다. 이들 중 항상 동작하고 있는 상위 2 가지 레벨의 객체—생성의뢰 및 제어담당, 생성담당을 위한 컴포넌트—와 서버 구동시 연결을 시작하여 유지하고 하위 3 가지 객체와는 다음과 같은 과정을 통하여 유지, 관리한다.

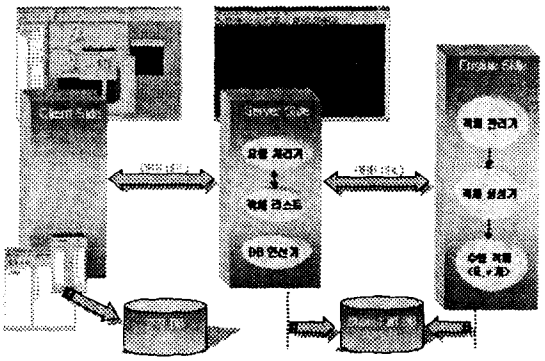
- ♦ 생성 → 객체트리에 삽입(하위 객체 포함)
- ♦ 소멸 → 객체트리에 삭제(하위 객체 포함)

객체리스트에 있는 동안은 엔진과의 통신을 위해 조회되고 참조의 대상으로 사용된다. 운용부분에서 엔진 객체를 레벨별로 유지, 관리함으로써 엔진과의 통신 시 해당 객체와의 연결을 위한 불필요한 작업을 없앨 수가 있다. 객체간에는 계층이 존재한다. 이러한 계층은 트리라는 자료구조와 매핑이 잘 이루어지고 있다. 또한 새로이 생성되고 완료 시 소멸되는 객체들에 대한 업데이트는 LTT(Long -Time-Term)을 이용한 Pull 방식과 사용자 요청시 정보갱신의 병행적인 방법을 사용한다.

3. 메시지의 구조와 흐름

분산 객체 관리의 표준인 CORBA 를 객체 간 통신 메커니즘으로 사용한다. 객체 호출은 CORBA 의 이름 서비스를 이용한다.

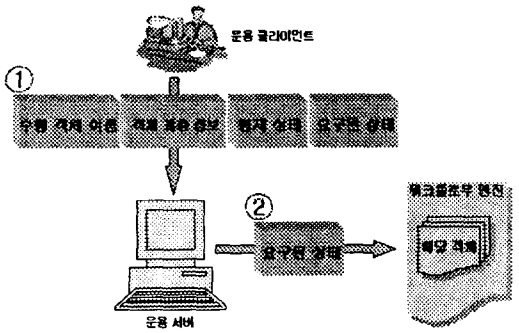
운용 서버는 클라이언트와 엔진 사이에서 일종의 중개 역할을 담당하고 있다. 따라서 <그림 2> 와 같이 클라이언트와 운용서버 간, 운용서버와 엔진간의 통신을 위해 정의된 IDL 에 따라서 메시지가 이동한다.



<그림 3> 실행 환경의 Overview

3.3.1 메시지 구조

GUI 기반의 클라이언트에서 유지하는 정보는 이름, 이름에 관련된 ID, 그룹, Propertys, 상태 정보뿐만 아니라 다른 정보들까지 유지된다. 운용서버에서는 객체관리기가 위치하기 때문에 객체의 레퍼런스를 얻기에 필요한 정보와 처리에 필요한 정보만 필터링할 수 있으면 엔진과의 통신을 위한 메시지의 양을 대폭적으로 줄일 수 있다. 또한 객체 관리기를 거치지 않고 해당 객체와 직접 연결됨으로써 정확성도 높이는 부수적 효과를 얻는다.



<그림 4> 메시지 구조의 예

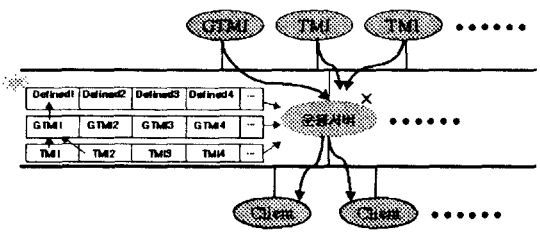
운용 서버에는 클라이언트로부터 메시지가 도착되면 그것을 분류하여 엔진으로 전달하기 전에 통신에 꼭 필요한 속성만을 추출하기 위한 처리를 한다. 이름으로부터 ID 를 찾고, 페어로 묶여 있는 레퍼런스를 키로 해당 객체와 통신이 이룬다. 이때 상태를 점검하고 무효하면 상태 무효 예외 메시지를 발생시키고, 유효하다면 해당 객체에게 요구된 상태만을 전달한다.

다음으로, 구현 되어진 운용 서버의 메시지 스위칭 예를 들어본다.

◆ 업무 인스턴스 객체 생성 기능
  ◆ 클라이언트 : 업무이름, ID, enable/disable flag
  ◆ 엔진 : 업무 ID
◆ 단위업무 인스턴스 객체의 참여자 재할당 기능
  ◆ 클라이언트 : 단위업무이름, ID, 현재 참여자클래스, 새로운 참여자클래스
  ◆ 엔진 : 새로운 참여자 클래스
◆ 특정 서버의 최대 객체 생성 수의 제어 기능
  ◆ 클라이언트 : 서버이름, ID, 환경정보, 현재 객체 수, 할당된 객체 수, enable/disable flag.
  ◆ 엔진 : 할당된 객체 수

운용서버와 엔진간에 메시지 크기가 많은 부분 줄어드는 것을 볼 수 있다.

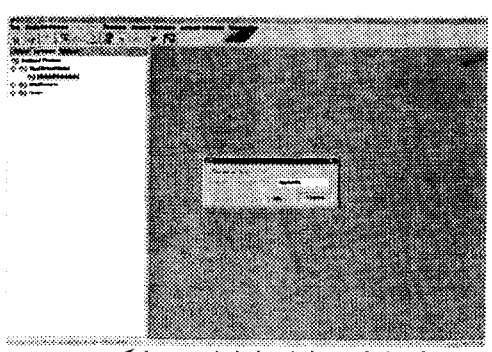
3.3.2 메시지 흐름



<그림 5> 객체간의 메시지 흐름

메시지는 클라이언트에서 운용 서버로 이동되면 자동으로 대응되는 구체적 메시지를 엔진으로 보내게 된다. 이는 요청과 즉각 처리라는 개념에 상응하는 것이다. 운용자의 감독 명령은 일반 참여자와는 구별된다. 시스템이나 업무에 적용될 때에는 우선 순위가 가장 높다. 그러한 처리 루틴은 엔진과 같은 곳에 존재하는 운용 서버에 있기 때문에 클라이언트는 엔진과 직접 연결되어서는 안되고 운용 서버를 통해야만 한다. 이는 보안에도 필요한 흐름이다.

◆ 클라이언트(요구) → 운용 서버 → 엔진 (수행) → 운용 서버 (응답) → 클라이언트(통보)



<그림 6> 메시지 생성 구현의 예

운용 서버에서 엔진의 객체들이 트리 형태로 구성이 되고, 객체간의 계층까지 표현하고 있으며 이는 클라이언트 GUI 에 트리로 비슷하게 보여지고 있다. 두 부분은 동일한 데이터를 유지하게 되어 있다. 다만 레퍼런스 값은 서버에서만 유지된다. 클라이언트에서 요구 발생시 이름 파라메터로 서버에서 레퍼런스를 찾아서 해당 엔진 수행 객체와 적지만 충분한 정보를 내포한 메시지를 주고 받으며 통신을 이룬다. 메뉴나 툴바를 통해 메시지 생성을 위한 다이어로그를 생성하고 결과가 리턴되는 경우에 한하여 메시지상자를 통하여 통보하게 된다.

4. 결론 및 향후과제

객체 중심으로 설계되어진 워크플로우 시스템은 특성상 수행 객체가 짧은 시간동안 많은 수가 생성되고 또 소멸되며 활동 기간동안은 그들 간에, 또는 서버를 구성하고 있는 컴포넌트 간에 통신이 빈번하게 일어나고 있다. 이러한 것들은 외부적으로는 보이지 않을 수도 있지만 빈약한 설계로 인해 시스템 내부적으로 비효율성을 일으켜 시스템 전반적인 성능을 저하시키는 요소가 된다. 따라서 시스템과 실행에 개입되어 관리, 감독하고 있는 역할을 담당하는 운용 부분의 통신 방법이 제시되어야 한다. 본 논문에서는 그러한 방법중의 하나로 객체간에 통신 방법으로는 메시지의 구조와 흐름에 대한 방법을 제시하였다. 아직 워크플로우 분야에 이러한 객체 통신에 대한 언급은 거의 고려되고 있지 않지만, 앞으로 거대한 조직의 Large Scale 워크플로우를 지향한다면 더욱더 이러한 사항은 설계 단계부터 고려가 되어야 한다.

향후엔 제시된 방식과 다른 방식들을 시뮬레이션이나 테스트를 통해 성능을 비교할 것이며, 결과에 대한 하부 원인과 그것의 분석을 통한 개선책을 발전시켜 나갈 것이다.

[참고 문헌]

[1] A. Elmagarmid, editor. 'Transaction Models for Advanced Database Applications'. Morgan-Kaufmann, 1992.
[2] A. Sheth and M. Rusinkiewicz, 'On Transactional Workflows', in Data Engineering Bulletin ,16(2), June,1993.
[3] A. Silbercharz, H. F. Korth and S. Sudarshan, 'Database System Concepts', McGraw Hill, 1997.
[4] A. Zhang, M. Nodine, B. Bhargava and O. Bukhes, 'Ensuring Relaxed Atomicity for Flexible Transaction in Multidatabase Systems' , ACM SIGMOD RECORD, Vol. 23, No. 2, 1994
[5] C. Mohan, D. Agrawal, G. Alonso, A. El Abbadi, R. Günthör, and M.Kamath. 'Exotica: A Project on Advanced Transaction Management and Workflow System'. ACM SIGOIS Bulletin, 16(1), 1995.
[6] Clarence A. Ellis and Gary J. Nutt, 'Office Information Systems and Computer Science', Computing Surveys, Vol. 12, No. 1, March 1980.
[7] Clarence A. Ellis and Gary J. Nutt, 'Modeling and Enactment of Workflow Systems', in Proceedings of the 1993, June 1993.
[8] D. Geogakopoulos and M. Hornick, 'An Overview of Workflow Management : From Process Modeling to Workflow Automation Infrastructure', International Journal of Parallel and Distributed Databases, Vol. 3, No. 2, pp. 119-153, 1995