e-Chautauqua 워크플로우 관리 시스템 엔진의 Workflow Message Queue 메커니즘 설계

안형진° 심성수 정재우 박민재 배성용 김광훈 백수기
경기대학교 전자계산학과 워크플로우 연구실
zzanggu1107@kyonggi.ac.kr

Workflow Message Queue:Internal Comm. Mechanism for the e-Chautauqua WfMS

Hyung-Jin Ahn° Sung-Su Sim Jae-Woo Jung Min-Jae Park Sung-Yong Bea Kwang-Hoon Kim Su-Ki Paik
Dept. of Computer Science, Kyonggi University

요 약

본 논문에서는 워크플로우 시스템의 엔진 컴포넌트들 간에 이루어지는 내부 통신 메커니즘에 대해 기술한다. 워크플로우 엔진의 내부 통신 메커니즘을 구현하는 방법은 동기적 매커니즘과 비동기적 메커니즘으로 나뉘어진다. 본 논문에서 제안하고 있는 e-Chautauqua 워크플로우 관리 시스템에서는 비동기적 메커니즘을 토대로 하여 엔진 컴포넌트들의 내부 통신 처리를 구현한다. 이러한 엔진 내부 통신의 성능 향상을 위해 비동기 메시징 방식을 기반으로 하는 EJB 메시지 빈(Message-Driven Bean)을 사용하여 설계한다. 또한 엔진 컴포넌트들 간의 내부 통신시에 사용되는 송수신 메시지를 구조적인 유연성과 확장성을 지닌 XML로 설계함으로서 엔진 컴포넌트들간의 내부 통신을 보다 효율적으로 이루어지도록 설계한다.

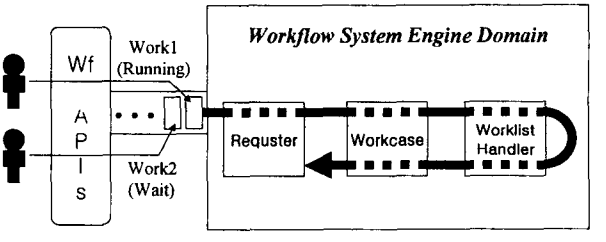
1. 서 론

인터넷과 네트워크의 발전은 기업 내의 업무 처리 환경에 많은 변화를 주었다. 갈수록 거대화되는 비즈니스 프로세스들을 처리하는 데 있어서 기존의 수동적인 업무 처리 방식이나 레거시(legacy) 애플리케이션들만 가지고는 감당할 수 없게 되었다. 기업 측에서는 급속한 업무 환경의 변화에 신속히 대처하기 위해서 우선적으로 기업 내 업무 환경을 통합적으로 관리할 수 있는 기술이 도입되야 한다고 인식하게 되었고, 이러한 기업 조직들의 요구사항을 만족시킬 수 있는 시스템으로서 대규모 기업 환경의 비즈니스 프로세스들을 자동화하여 처리해주는 워크플로우가 이슈화되었다. 다양한 비즈니스 프로세스들을 처리하기 위해서 분산 작업 환경 하에 여러 작업들의 실행을 지원할 수 있도록 개발된 시스템을 워크플로우 관리 시스템이라 한다. 이러한 워크플로우 관리 시스템이 제공하는 서비스들에 대한 관리 및 제어를 담당하는 심장부의 역할을 하는 것이 워크플로우 엔진이다. 엔진은 워크플로우 시스템의 핵심이라 할 수 있으며, 워크플로우 환경을 구성하는 컴포넌트들을 위한 런타임 실행 환경을 제공하고 사용자와의 상호작용을 통하여 작업을 처리하는 중요한 역할을 한다. 엔진과 런타임 클라이언트와의 상호 작용은 WfMC의 워크플로우 레퍼런스 모델 정의로부터 제안된 인터페이스들이나 객체 표준화 집단인 OMG의 JointFlow모델에서 제시한 인터페이스들과 같은 API들을 통해서 이루어진다. 이러한 사전 정의된 API들을 통해 엔진에게 해당 작업에 대한 요청이나 명령을 전달하게 되면 엔진 내부에서는 엔진 구성 컴포넌트들 사이의 내부적인 통신을 통해서 연계된 작업들을 수행하게 된다. 엔진의 성능을 향상시키기 위해서는 엔진 컴포넌트들 사이의 내부적으로 통신되는 메시지들이 병목현상(Bottleneck)을 최소화하며 빠르게 처리될 수 있어야 한다. 본 논문에서는 워크플로우 시스템의 중추 부분인 엔진의 성능 향상을 위한 방안으로서 엔진 컴포넌트들 간의 내부적인 메시지 통신 처리에 대해서 기존의 동기적 메커니즘보다 개선된 비동기적 메커니즘을 제안하고자 한다. 2장에서는 워크플로우 엔진의 내부 통신 메커니즘에 대한 연구로서 엔진 컴포넌트들의 내부 통신에 대해서 기존의 동기적 메커니즘과 개선 방안으로서 제시되고 있는 비동기적 메커니즘에 대한 개념적인 내용들에 대해서 알아보고 비동기적 메커니즘이 어떠한 이유로 인해 워크플로우 시스템에 효율적인지에 대해서 기술하며, 비동기적 내부 통신의 통신 매개체로서 사용되는 메시지를 XML을 통해 설계하는 것에 대한 이점에 대해서 간략히 이야기한다. 3장에서는 본 논문의 기반 시스템으로서 설명되고 있는 e-Chautauqua 워크플로우 시스템에 대해 간단하게 기술한다. 4장에서는 2장에서 설명한 비동기적 메커니즘을 기반으로 e-Chautauqua 워크플로우 관리 시스템에서의 엔진 내부 통신 메커니즘에 대해 기술하고, XML을 이용해 통신 메시지를 설계한다. 마지막으로 5장에서는 결론 및 향후 연구과제를 기술한다.

2. 워크플로우 엔진의 내부 통신 메커니즘에 대한 연구

2.1 동기적 메커니즘을 이용한 엔진의 내부 통신

런타임 클라이언트의 요청 작업은 엔진과의 표준 인터페이스들을 통해 엔진 컴포넌트로 전달되어 엔진 내의 메시지 통신을 통해서 적절한 워크플로우 구성 컴포넌트들과 연결되어 처리하게 된다. 기존의 엔진 컴포넌트들간의 내부 메시지 통신은 메시지를 보낸 근원지 컴포넌트가 목적지 컴포넌트로부터 클라이언트의 업무 요청에 대한 수행 결과에 대한 응답 메시지를 수신할 때까지 대기하는 동기적 메커니즘에 의해서 수행되어져 왔다. 그림 1은 엔진의 동기적 내부 통신에 대한 개념적인 구조도로서 작업 요청을 받은 근원지 컴포넌트인 Requester가 Worklist Handler에게 메시지를 전송하여 해당 작업의 처리가 완료될 때까지 대기하는 동기적 메커니즘에 대한 내용을 나타내고 있다.

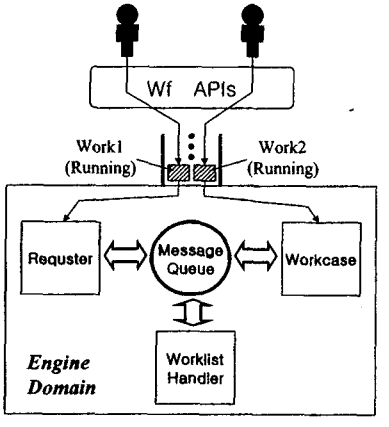


(그림 1) 워크플로우 엔진 컴포넌트들의 동기적 내부 통신 구조도

이러한 동기적 메커니즘의 문제점은 만약 작업 요청에 대한 호출이 성공적이지 못할 경우 문제가 해결될 때까지 지연되고 계속 요청이 손실됨으로써 엔진의 성능을 저하시키고 나아가서 시스템의 심각한 부하를 가져올 수 있기 때문에 시스템 운영의 유연성을 떨어뜨린다는 것이다.

2.2 비동기적 메커니즘을 이용한 엔진의 내부 통신

비동기적 메커니즘은 동기적 방식과는 달리 근원지 컴포넌트가 수행 완료 메시지를 수신할 때까지 지연없이 연속해서 다른 작업을 받아들여 처리해나간다. 또한 메시지 전송에 실패할 경우에 Message Queue에 저장해 놓았다가 후에 재처리해주는 유연성을 제공한다.



(그림 2) 워크플로우 엔진 컴포넌트들의 비동기적 내부 통신 구조도

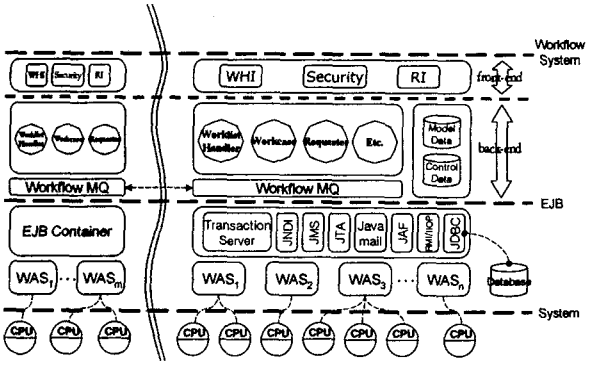
그림 2는 비동기적 메커니즘을 통한 워크플로우 엔진의 내부 통신을 나타내고 있다. Workflow API를 통해 외부로부터 업무를 전달받은 엔진의 근원지 컴포넌트는 해당 작업을 처리하기 위해서 메시지를 전송한다. 전송된 메시지는 메시지 큐에 임시 저장했다가 목적지 컴포넌트에게 전달하게 된다. 근원지 컴포넌트는 작업 수행 완료 메시지를 수신할 때까지 대기하지 않고 다음 작업을 처리한다. 이러한 비동기적 통신은 동기적 방식에서 발생될 수 있는 종속 현상이나 병목 현상을 최소화 시킴으로서 성능의 향상을 기대할 수 있다.

2.3 XML 메시지

본 논문에서 기술하는 e-Chautauqua 워크플로우 시스템에서는 엔진의 내부 통신 메시지를 XML을 이용해서 설계한다. 메시지의 일반적인 전송 방식이었던 클래스 또는 직렬화 데이터 송수신에 비해서 XML메시지는 데이터 계층 구조의 유연성을 보장해주며 구조적인 확장성이 용이하고 네트워크의 부하를 감소시켜주는 등의 여러 가지 이점들을 제공한다. e-Chautauqua 워크플로우 엔진의 내부 통신에 사용되는 메시지에 대한 설계는 4.2장에서 자세하게 기술한다.

3. e-Chautauqua 워크플로우 시스템

본 논문은 e-Chautauqua 워크플로우 시스템을 기반으로 엔진의 내부 통신 메커니즘을 기술하려 한다. 구체적인 설계에 앞서 우선 e-Chautauqua 워크플로우 시스템에 대해서 간단히 살펴보도록 한다. e-Chautauqua 워크플로우 시스템은 기존의 워크플로우 시스템이 관리할 수 있는 수 이상의 거대량의 작업들을 처리할 수 있는 확장성(Scalability)에 목적을 두고 현재 개발 중인 초대형 워크플로우 시스템이다. e-Chautauqua 워크플로우 시스템의 주요 구성 컴포넌트는 엔진 영역의 Requester, Workcase, Worklist Handler이다. Requester컴포넌트가 런타임 클라이언트로부터 Requester Interface를 통해 작업 요청을 받으면 그에 해당하는 Workcase를 생성하거나 생성된 작업을 실행시킨다. Workcase인스턴스의 액티비티 작업이 수동 작업일 경우엔 Worklist Handler에게 작업을 위한 Workitem을 전달한다. e-Chautauqua 워크플로우 시스템은 메시지 핸들러를 통해 비동기적으로 작업들을 처리하여 작업 객체 간의 종속 현상을 최소화시켜 주고 엔진 시스템에 추가적으로 인스턴스가 생성될 경우에 메시지 핸들러에 연결하여 인스턴스들의 추가를 용이하게 해주는 구조를 갖고 있다. 그림 3은 e-Chautauqua 워크플로우 시스템의 아키텍쳐를 나타내고 있다.

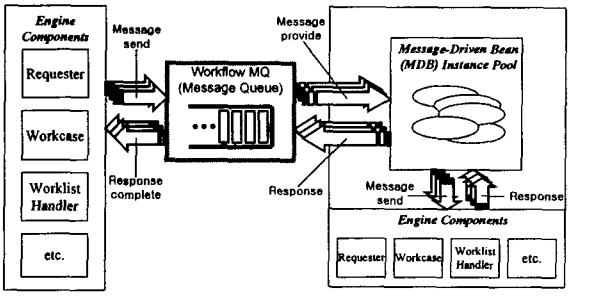


(그림 3) e-Chautauqua 워크플로우 시스템 아키텍쳐

4. e-Chautauqua 워크플로우 엔진의 내부 통신 설계

4.1 e-Chautauqua 엔진의 내부 통신 메커니즘

e-Chautauqua 엔진 내부 통신에 사용되는 비동기 메시지 핸들링에 대한 설계는 자바에서 제공하는 EJB 메시지 빈(EJB Message-Driven Bean)을 기반으로 한다.

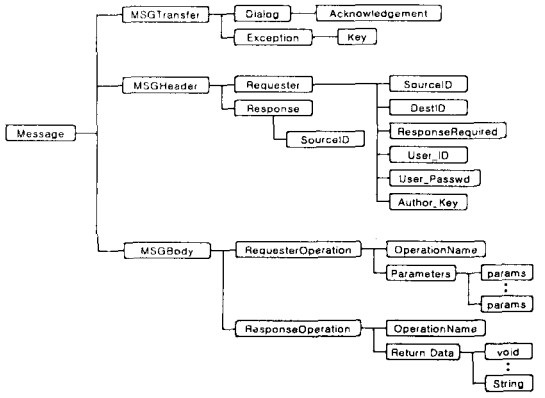


(그림 4) e-Chautauqua 엔진 내부 통신에 대한 구조도

위의 그림 4는 e-Chautauqua 엔진의 내부 메시지 통신 설계의 개략적인 구조도를 나타낸다. e-Chautauqua 엔진의 핵심은 컴포넌트들 간의 메시지 핸들링 및 저장소 역할을 하는 Workflow MQ이다. Workflow MQ는 비동기적으로 메시지를 처리하며 MQ Provider를 통해 EJB 컨테이너의 메시지 빈들이 저장되어 있는 인스턴스 풀에 메시지를 제공하게 된다. 제공된 메시지는 MDB 인스턴스 풀에서 메시지에 해당되는 빈을 생성시키거나 찾아서 해당

작업을 처리한 후에 다시 근원지 컴포넌트로 응답 메시지를 발송하게 된다.

4.2 XML을 이용한 엔진 내부 통신 메시지의 설계



(그림 5) 엔진의 내부 통신 메시지 기본 구조 모델

위의 그림 5는 e-Chautauqua 워크플로우 엔진의 내부 통신 메시지 설계에 대한 기본 구조를 정의하고 있다. 이 모델에서는 비동기적 통신에서 발생될 수 있는 메시지 미수신 또는 손실 방지를 위해 Acknowledgement 응답 필드를 두어 그 데이터를 통해 메시지 재전송 여부를 판단하게 된다. 그림 5의 메시지 기본 구조를 토대로 그림 4에서의 엔진 내부 통신의 시나리오를 가지고 메시지에 대한 설계를 살펴보도록 한다. 다음의 그림 6은 Requester컴포넌트와 Workcase컴포넌트 간의 내부 통신 메시지의 DTD를 나타낸다.

```
<!-- Entities -->
<!ENTITY % RequestOperation "create.Request|start.Request|assignAttribute.Request|
        getState.Request">
<!ENTITY % ResponseOperation "create.Response|start.Response|assignAttribute.Response|
        getState.Response">
<!ENTITY % Return "void|String|WorkcaseID">
<!ENTITY % States "open.notrunning|open.notrunning.suspended|open.running|
                   closed.completed|closed.abnormalCompleted|
                   closed.abnormalCompleted.terminated|closed.abnormalCompleted.aborted">

<!-- Root Element -->
<!ELEMENT Message ((MSGTransfer, (MSGHeader, MSGBody)*)|(MSGHeader, MSGBody))>

<!-- Message Transfer -->
<!ELEMENT MSGTransfer (Dialog?, Exception?)>
<!ELEMENT Dialog ((Acknowledgement, Key)|(ReplyToKey, Key?)|Key)?>
<!ELEMENT Acknowledgement EMPTY>
<!ATTLIST Acknowledgement ReceivedAt CDATA #REQUIRED>
<!ELEMENT Key (#PCDATA)>
<!ELEMENT ReplyToKey (#PCDATA)>

<!-- Message Header -->
<!ELEMENT MSGHeader ((Request|Response), Key)>
<!ELEMENT Request EMPTY>
<!ATTLIST Request SourceID CDATA #REQUIRED
                  DestID CDATA #REQUIRED
                  ResponseRequired (Yes|No|Error) #REQUIRED
                  UserID CDATA #IMPLIED
                  Passwd CDATA #IMPLIED
                  AuthorKey CDATA #IMPLIED>
<!ELEMENT Response EMPTY>
<!ATTLIST Response SourceID CDATA #IMPLIED>

<!-- Message Body -->
<!ELEMENT MSGBody (%RequestOperation;|%ResponseOperation;)>

<!-- Request Operations -->
<!ELEMENT create.Request (MsgID)>
<!ELEMENT MsgID (#PCDATA)>
<!ELEMENT start.Request (MsgID)>
<!ELEMENT getState.Request (MsgID)>

<!-- Responsse Operations -->
<!ELEMENT ReturnData (%Return;)>
<!ELEMENT StateData (%States;)>
<!ELEMENT create.Response (ReturnData|Exception)>
<!ELEMENT start.Response (ReturnData|Exception)>
<!ELEMENT getState.Response (StateData|Exception)>

<!-- Exception -->
<!ELEMENT Exception (Code, Type, Description?)>
<!ELEMENT Code (#PCDATA)>
<!ELEMENT Type (#PCDATA)>
<!ELEMENT Description (#PCDATA)>
```

(그림 6) Requester 컴포넌트와 Workcase 컴포넌트 간의 XML메시지에 대한 DTD

런타임 클라이언트로부터 작업 요청을 받은 Requester컴포넌트가 적합한 Workcase인스턴스를 생성하기 위해 요청하는 XML메시지 내용은 다음과 같다.

```
<?xml version="1.0" ?>
<!DOCTYPE MSG SYSTEM "message.dtd">
<Message>
   <MSGTransfer>
      <Dialog>
            <ReplyToKey>http://ctrl.kyonggi.ac.kr/e-Chautauqua/xml_messages/771107</ReplyToKey>
      </Dialog>
   </MSGTransfer>

   <MSGHeader>
      <Request SourceID="4003a10b-100d-1p99-9037-888s7a2001m3"
               DestID="3333c66n-100d-1p99-9037-888s7a2001m3"
               ResponseRequired="Yes"
               UserID="superman"
               Passwd="abc3434"
               AuthorKey="uibjs0011123rt" />
      <Key>http://super702.ac.ro/e-Chautauqua/processes/61380468</Key>
   </MSGHeader>

   <MSGBody>
      <create.Request>
            <MsgID>1212d-0909-3178ff-5026q3p</MsgID>
      </create.Request>
   </MSGBody>
</Message>
```

(그림 7) Requester컴포넌트로부터 Workcase컴포넌트로 전송되는 요청 XML메시지

그림 8에서 보는 바와 같이 Workcase컴포넌트가 성공적으로 해당 Workcase인스턴스를 생성하게 되면 Acknowledgement의 ReceivedAt에 적절한 값을 설정한 후에 응답 메시지를 Requester 컴포넌트에게 전송하게 된다.

```
<?xml version="1.0" ?>
<!DOCTYPE MSG SYSTEM "message.dtd">
<Message>
   <MSGTransfer>
      <Dialog>
            <Acknowledgement ReceivedAt="2003-2-20T16:33:212" />
            <Key>http://ctrl.kyonggi.ac.kr/e-Chautauqua/xml_messages/771107</Key>
      </Dialog>
   </MSGTransfer>
</Message>
```

(그림 8) Requester컴포넌트의 요청에 대한 Workcase컴포넌트의 응답 XML메시지

5. 결론 및 향후 연구과제

본 논문에서는 엔진의 성능 향상을 위한 방안으로서 워크플로우 엔진의 내부 통신을 비동기적 메시징 방식을 통해 개발하는 내용에 대해서 제안하였다. 이러한 내부 통신의 비동기적 메커니즘을 구현하기 위한 기술로서 EJB 메시지 빈을 사용하고 엔진의 내부 통신에 사용되는 메시지를 XML로 설계함으로서 메시지의 구조적인 확장성과 유연성, 유효성에 대해서도 고려하였다. 현재 e-Chautauqua 워크플로우 관리 시스템은 개발 진행 중인 단계이므로 본 논문에서는 개발에 대한 설계 내용에 대해서만 다루고 있다. 향후 e-Chautauqua 워크플로우 관리 시스템에서는 본 논문에서 제안한 내용을 바탕으로 엔진의 내부 통신 처리 구현을 완성할 것이다.

Acknowledgement

본 연구는 한국과학재단 목적기초연구(R05-2002-000-01431-0) 지원으로 수행되었음.

참 고 문 헌

[1] Object Management Group, "Workflow Management Facility Specification, V1.2", April 2000

[2] 심성수, 김광훈, "워크케이스 기반의 초대형 워크플로우 시스템 아키텍쳐", 한국데이터베이스학회 추계컨퍼런스 학회, 2002년 10월

[3] Wrox Press, "Professional J2EE EAI", 2001