메모리 캐싱 저장소를 이용한 대규모 워크플로우 모델 데이터 처리 메커니즘

박민재° 심성수 정재우 안형진 김민홍 김광훈
경기대학교 전자계산학과 워크플로우 연구실
mean222@kyonggi.ac.kr

The Large Scale Workflow Model Data Process Mechanism Using Memory Cashing Repository

Min-Jae Park° Sung-Soo Sim Jae-Woo Jung Hyung-Jin Ahn
Min-Hong Kim Kwang-Hoon Kim
Dept. of Computer Science, Kyonggi University

요 약

워크플로우 시스템의 핵심을 이루고 있는 엔진의 효율성을 극대화 시키기 위하여, 워크플로우 시스템 엔진 안에서 운용되는 데이터의 관리는 매우 중요하다. 본 논문에서는 워크플로우 시스템에서 운용되는 각 시스템 데이터의 특징을 고찰 한 후, 일반적으로 사용하는 데이터 베이스 시스템 호출을 통해 데이터를 관리하는 방법을 보완 할 수 있는 방법을 제시한다. 그 방법으로 각 시스템 데이터가 가지고 있는 특성에 맞추어 기존의 데이터 베이스 호출을 통한 방법에 메모리에 데이터를 로드시켜 공용적으로 사용하는 방법을 더해 시스템 데이터를 관리하는 방법을 기술한다.

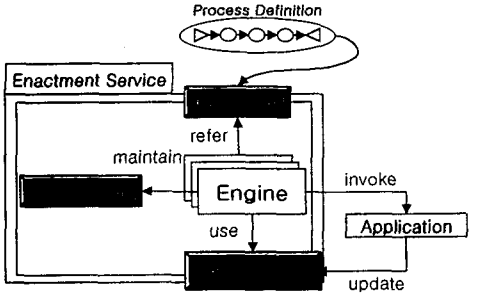
1. 서론

워크플로우는 비즈니스 프로세스 자동화를 의미하며, 워크플로우 관리 시스템은 비즈니스 업무를 컴퓨터에 의해 자동으로 실행하고 관리하는 소프트웨어 시스템을 말한다. 워크플로우 관리 시스템의 비즈니스 프로세스 자동화는 기업의 효율적인 관리와 처리에 있어 많은 효과가 있다. 최근 워크플로우를 적용한 많은 기업들의 성공 사례가 보고 되면서, 워크플로우 기술에 대한 관심이 커지게 되었다. 현대의 기업은 네트워크의 발달과 인터넷 환경의 발달을 기반으로 대량의 사용자와 서버, 데이터 그리고 작업을 처리할 수 있어야 하며, 처리해야 하는 시스템도 거대해 져야 한다. 시스템이 거대해 짐에 따라, 시스템이 처리할 데이터의 양과 구조는 거대해지고, 복잡해진다. 이러한 개념을 바탕으로 현대의 워크플로우 관리 시스템은 대량의 업무를 처리 할 수 있도록 설계된다. 시스템이 거대해 짐에 따라, 워크플로우 관리 시스템에서 관리될 시스템 데이터들을 관리하는 것 또한 이슈화 되어진다. 시스템 데이터들은 각각의 특징적인 요소를 가지고 있고, 그에 따라 워크플로우 관리 시스템에 많은 영향을 줄 수 있다. 따라서 본 논문에서는 워크플로우 관리 시스템에서 사용되는 시스템 데이터의 종류와 특성을 연구하여, 데이터 베이스를 이용한 데이터 관리 방법과 메모리를 이용한 데이터 관리 방법에 있어서, 두 가지 각각의 데이터 처리 방법의 장단점을 살펴보고, 각 데이터 특성에 맞는 데이터관리 방법과 각 데이터를 처리하는데 있어 필요한 운용을 제시한다.

2. 워크플로우 데이터 처리

2.1 워크플로우 데이터

워크플로우 시스템에서 사용되는 데이터의 종류는 각 특성에 따라 크게 워크플로우 모델 데이터(Workflow Model Data), 워크플로우 관련 데이터(Workflow Relevant Data), 그리고 워크플로우 제어 데이터(Workflow Control Data) 이 세 가지 종류로 분류할 수 있다.



(그림 1) 워크플로우 데이터

① 워크플로우 모델 데이터(Workflow Model Data)
워크플로우 프로세스를 정의할 때 사용하며, 미리 정의되어 있어, 워크플로우 관리 시스템 또는 엔진에서 프로세스 인스턴스를 생성할 때, 참조하여 사용한다.

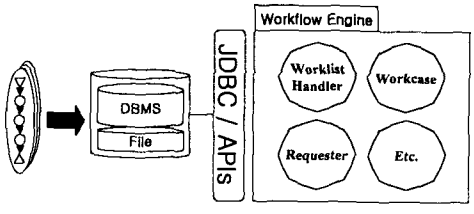
② 워크플로우 관련 데이터(Workflow Relevant Data)

워크플로우 인스턴스 상태 전이를 결정 하기 위해 워크플로우 관리 시스템이 사용한다.

③ 워크플로우 제어 데이터(Workflow Control Data) : 워크플로우 관리 시스템 또는 워크플로우 엔진에서 관장하는 데이터로써, 워크플로우 시스템과 그 시스템의 프로세스 인스턴스의 동적인 상태를 나타낸다.

2.2 데이터 베이스 호출을 통한 워크플로우 데이터 처리와 문제점

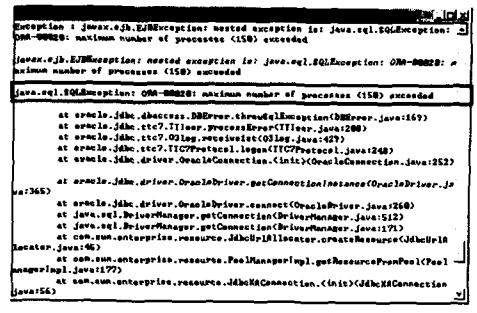
워크플로우 엔진이 데이터를 처리하는데 고려해야 할 각 데이터의 특성을 비교하면 다음과 같다. 첫째로 워크플로우 모델 데이터는 데이터가 사전에 정의되어 있어 변하지 않는다는 가장 큰 특성이 있고, 데이터가 변하지 않기 때문에 단 방향적인 질의를 한다. 그리고 동일한 모델 데이터는 프로세스 인스턴스가 생성될 때마다 참조되어 사용되기 때문에 매우 많은 질의를 받을 수 있다. 둘째로 워크플로우 관련 데이터는 프로세스 인스턴스가 처리 되면서 발생하는 프로세스 흐름이 흘러감에 따라 계속적으로 데이터 갱신이 발생하는 특성이 있다. 셋째로 워크플로우 제어 데이터는 프로세스 인스턴스의 상태를 관장하기 때문에 인스턴스의 상태에 따라 적절한 데이터 갱신이 필요하다.



(그림 2) 데이터 베이스 호출을 통한 데이터 처리

위 그림 2와 같이 일반적인 데이터 처리 방법에 있어서 데이터 베이스 관리 시스템 호출을 통해 워크플로우 데이터를 처리하는 과정은 다음과 같다. 워크플로우 엔진의 데이터를 사용하기 위한 각 컴포넌트들은 적절한 시기에 JDBC와 같은 API들을 통해 데이터 베이스 관리 시스템을 호출하여 데이터가 저장되어 있는 데이터에 접근하여 사용한다. 이 방법은 데이터 베이스 관리 시스템(DBMS)이 존재하기 때문에, 프로세스 인스턴스의 전이나 상태 변화와 같은 동적인 데이터의 변화에 따라 쉽게 적응 할 수 있는 장점이 있고, 지속적인 데이터 저장에 따라 데이터를 잃어버릴 염려 또한 적다. 따라서 데이터 베이스 호출을 이용한 방법은 시스템에서 프로세스 인스턴스의 전이를 관장하는 워크플로우 관련 데이터나 인스턴스의 상태를 관장하는 워크플로우 제어 데이터를 관리할 때에 적절하다.

그러나, 모델 데이터는 프로세스 인스턴스가 생성될 때마다 반복적으로 같은 데이터를 호출하여 사용해야 하기 때문에 대규모의 작업을 처리하기 위해 많은 인스턴스를 처리해야 할 때에는 계속적인 데이터 베이스 시스템으로의 접근이 필요함으로 다음 그림 3의 SQL 예외 상황과 같은 문제점을 야기할 수 있다.



(그림 3) SQL 예외 상황

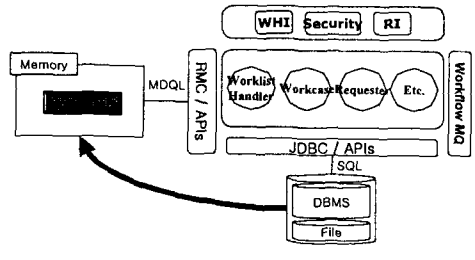
이와 같이 각 데이터의 특징을 비교해 보았을 때, 일반적인 데이터 베이스 호출을 통한 방식에서 가장 문제가 되는 것은 워크플로우 모델 데이터를 처리하는 것이다. 따라서 다음 절에서는 워크플로우 모델 데이터를 처리하는 데 있어 가장 효율적인 방안을 제시하여 워크플로우 엔진이 워크플로우 데이터를 처리하는데 가장 효율적인 상태가 되도록 하게 한다.

3. 워크플로우 모델 데이터 처리

워크플로우 시스템에서 데이터를 관리하는데 있어, 기존의 데이터 베이스 시스템을 통한 관리는 동적인 상태 변화에 적절히 반응하고, 데이터의 안정성을 유지하는 데는 좋다. 그러나, 변화가 없는 데이터를 반복적으로 사용할 때에는 불필요한 호출이 많고, 그에 따른 문제점도 발생한다. 이러한 문제점을 해결하기 위해 이번 장에서는 워크플로우 모델 데이터의 특징을 알아보고, 워크플로우 모델 데이터를 처리하는데 있어 새로운 메커니즘을 제시하고자 한다.

3.1 메모리 상주를 이용한 워크플로우 데이터 처리 메커니즘

데이터 베이스 호출을 통해 야기할 수 있는 문제점을 해결하는 방안으로 다음 그림과 같이 메모리 상주를 이용한 워크플로우 데이터 처리 메커니즘을 제시하고자 한다.



(그림 4) 워크플로우 데이터 처리 메커니즘

기존 워크플로우 데이터를 처리할 때와는 달리 이 방법에서는 메모리가 존재하고, 메모리에 데이터를 상주시키는 기능과 메모리에 존재하는 데이터를 처리하는 기능이

필요하다. 따라서 데이터 베이스 호출 방식에서 JDBC와 같은 API들을 사용하여 SQL 질의를 한 것처럼 메모리 캐싱 방식에서도 RMC(Repository Memory Connectivity)라는 API를 정의하여, MDQL(Model Data Query Language) 질의를 하는 방안을 제시하고자 한다.

처리과정을 알아보면, 처음 프로세스 인스턴스가 발생할 때, 워크플로우 모델 데이터를 사용하게 되는데, 워크플로우 모델 데이터가 메모리에 상주하고 있다면 메모리에 상주하고 있는 메모리를 호출하여 사용한다. 하지만 메모리에 상주하고 있지 않다면, 데이터 베이스를 호출하여 메모리에 상주시키도록 하여, 메모리에 상주하고 있는 메모리를 사용한다. 인스턴스의 사용이 끝난 후에도 워크플로우 모델 데이터는 메모리에 상주해 있으며, 메모리에 상주되어 있는 메모리의 변화가 발생 하거나, 메모리에 상주하고 있는 모델 데이터의 사용 빈도가 일정 수준 이하로 떨어졌을 경우에는 삭제하도록 한다.

3.2 메모리에 캐싱을 이용하여 데이터를 처리하기 위한 인터페이스 기능 설계

데이터를 메모리에 상주시켜 모델 데이터를 처리하기 위해 모델 데이터를 처리하는 엔진 컴포넌트는 메모리를 다음 정의된 기능들 통해 데이터 베이스 관리 시스템 저장된 워크플로우 모델 데이터를 메모리에 상주시키고, 메모리에 상주하고 있는 모델 데이터 호출하여 사용할 수 있어야 하며, 메모리에 상주한 모델 데이터를 삭제할 수 있어야 한다.

- **모델을 메모리에 삽입**

사용하고자 하는 워크플로우 모델 데이터가 메모리에 상주하고 있지 않을 시에 메모리에 모델 데이터를 삽입하는 기능이다.

- **메모리에 상주하는 모델 데이터를 호출**

사용하고자 하는 워크플로우 모델 데이터가 메모리에 상주하고 있을 때, 데이터를 호출하여 사용하는 기능이다.

- **모델 데이터를 메모리에서 삭제**

메모리에 상주하고 있는 모델 데이터의 사용빈도가 일정 수준 이하로 떨어졌을 경우, 메모리에 상주하는 필요성이 없다고 판단하여 메모리에서 삭제한다.

3.3 메모리 상주를 위한 모델 데이터 구조

메모리에 모델 데이터를 상주 시키기 위해서 XML 형식의 구조를 사용한다. 다음은 WfMC에서 정의한 표준에 따른 XPDL 스키마의 일부를 나타낸 것이다.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema
targetNamespace="http://www.wfmc.org/2002/XPDL1.0"
xmlns:xpdl="http://www.wfmc.org/2002/XPDL1.0"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified"
attributeFormDefault="unqualified">
...
```

```
...
<xsd:element name="Xpression">
    <xsd:complexType mixed="true">
      ...
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

(그림 5) XPDL Schema

앞에서 제시한 워크플로우 모델 데이터를 처리하는데 있어 메모리에 데이터를 상주시키는 방식은 메모리의 특성에 따라 다음과 같은 여러 가지 장점이 있다. 첫째로는 데이터 베이스 관리 시스템으로의 반복적인 접근이 필요 없어 SQL 예외 상황과 같은 문제점의 발생을 줄일 수 있고, 둘째로 엔진으로 접근하는 속도가 데이터 베이스를 호출하여 접근 하는 것보다 월등히 빠르다. 결론적으로 이러한 방식을 사용하면 워크플로우 데이터를 처리하는데 있어 엔진의 효율을 극대화 시킬 수 있다.

4. 결론 및 향후 연구방안

본 논문에서는 데이터를 관리하는데 있어, 데이터 베이스 시스템에서 필요한 데이터를 시스템이 적절한 시간에 호출하여 사용하는 방법의 장단점을 살펴보았고, 데이터 베이스 시스템을 호출하여 워크플로우 시스템 데이터를 처리하면서 발생하는 문제점을 살펴보았다. 따라서 기존의 데이터 베이스 방식의 단점을 보완할 수 있는 방법으로 데이터 베이스 호출을 통한 방식에 메모리에 데이터를 상주시켜 공용적으로 사용하는 방식을 더해 새로운 메커니즘을 제시하였다. 향후 연구에서는 본 논문에서 제시한 각 데이터들을 관리하는 엔진 컴포넌트들의 기능 따라 워크플로우 엔진을 개발하고, 데이터 베이스를 이용한 방법과 데이터 베이스를 이용한 방식에 메모리 상주 방식을 더한 방법을 비교하여 성능을 시험할 예정이다.

Acknowledgement

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

참고문헌

[1] David Holingsworrth, Workflow Management Coalition Specifition: " Workflow Reference Model" , WfMC, Jan, 1995

[2] Workflow Management Facility Specification, V1.2, OMG, April 2000

[3] Workflow Management Coalition Specifition " Workflow Process Definition Interface -- XML Process Definition Language" October 25, 2002

[4] Frank Leymann, Dieter Roller " Production Workflow Concepts and Techniques"

[5] 심성수, 김광훈. 워크케이스 기반의 초대형 워크플로우 아키텍쳐. 한국데이터베이스학회 2002년 추계학술발표논문집, 2002.10