워크플로우 시스템에서 작업할당 정책을 위한 메타 모델링

A Meta-model Approach for Work Assignment Policy in a Workflow System

이승진(Seungjin Lee)*, 우치수(Chisu Wu)**, 이형원(Hyungwon Lee)***

초 록

워크플로우 시스템은 긴 주기의 프로세스의 실행을 통제하는 소프트웨어 시스템이다. 조직의 구성원들이 프로세스의 실행을 담당하는데, 프로세스의 특정 부분을 누가 수행하는가를 결정하는 작업할당 정책이 워크플로우 시스템에 설계 및 구현되어 있어야 한다. 여러 조직들의 작업할당에는 매우 다양한 문제들이 있을 수 있어서, 다양한 여러 문제들을 다 포괄하는 하나의 모델을 고안하는 것은 가능하지 않을 것이다. 본 논문에서는 워크플로우 시스템에서의 작업할당 정책에 대한 보편적인 설계 방법으로서, 각 조직에 주어진 임의의 고려할 사항들에서부터 작업할당 정책들을 정의하기 위한 메타 모델 기반 접근 방법을 제시한다.

ABSTRACT

Workflow systems are software systems that control the execution of long-term processes. Members of an organization are in charge of executing processes. A work assignment policy, i.e. who should perform a certain piece of process, has to be modeled and implemented in workflow systems. Organizations may have a large variety of problems in work assignment, and it may not be feasible to devise a single model to cover all problems. In this paper, we introduce generality to work assignment design problems in workflow systems. We provide a meta-model based approach which enables us to define arbitrary problem oriented work assignment policies.

키워드 : 메타 모델링, 워크플로우
Meta-modeling, ConceptBase, Workflow

* 성공회대학교 소프트웨어공학과 초빙교수
** 서울대학교 컴퓨터공학부 교수
*** 강릉대학교 정보전자공학부 교수

1. 서 론

워크플로우 시스템은 긴 주기의 프로세스의 실행을 통제하는 소프트웨어 시스템이다. 워크플로우 시스템은 조직의 업무 프로세스를 모델링하고, IT와 결합시켜 업무 처리를 효율적으로 진행시키며, 그 처리 결과에 대한 사후 평가는 물론이고 프로세스 자체의 개선까지도 가능하게 하는 기업 업무 프로세스 도구이다.

워크플로우 시스템에 설계 및 구현되어 있어야할 주요 요소 중 하나는 작업할당 정책이다. 작업할당 정책은 프로세스의 각 부분이 조직 구성원 누구에 의해서 수행되는가를 결정한다. 조직의 구성원은 한정된 자원이기 때문에 작업에 구성원을 어떻게 할당하는가는 프로세스 효율에 중요한 요인이다.

조직의 작업할당에는 그 조직에 독특한 고려할 문제들이 있을 수 있다. 예를 들어 보잉사의 비행기 설계 프로세스에는 여러 그룹으로 나뉘어진 수백명의 설계 엔지니어가 참여한다. 각 그룹은 별도의 워크플로우 시스템을 사용한다. 비행기 설계 프로세스는 이런 여러 워크플로우 시스템에 분산 구현되어 있어서, 벤더가 다를 수도 있는 이들 워크플로우 시스템들은 서로 연동되어야 하고, 이런 분산된 워크플로우 시스템 상에서 일관되게 작업할당이 이루어져야 한다[5].

여러 조직들의 다양한 문제들을 다 포괄하는 하나의 작업할당 모델을 고안하는 것은 가능하지 않을 것이다. 각 조직의 작업할당 정책은 그 조직에 주어진 문제와 요구사항에서부터 출발하여 설계되고 구현되어야 한다. 본 논문에서는 워크플로우 시스템에서의 작업할당 정책에 대한 보편적인 설계 및 구현 방법으로서, 주어진 임의의 요구사항들에서부터 작업할당 정책들을 정의하기 위한 메타 모델 접근 방법을 보인다. 그리고 ConceptBase 지식베이스를 이용한 메타 모델접근 방법의 구현을 보인다[9].

논문의 구성은 다음과 같다. 1.1절의 워크플로우 시나리오에서 추출한 작업할당 정책의 예제를 1.2절에 나열한다. 작업할당 정책을 설계 구현하는 메타 모델 접근 방법을 3장에서 설명한다. 제시된 접근 방법에 의해 추가 요구사항이 용이하게 반영될 수 있음을 제4장에서 보인다. 제 5장에서는 모델의 무결성을 유지하기 위해 강제되는 제약에 대해 설명한다. 제6장에서는 구현된 도구들을 설명한다.

1.1 워크플로우 시나리오의 예

작업할당 정책을 설명하기 위한 예제 워크플로우로서, OMG의 워크플로우 평가 그룹에 제출된 보잉사의 항공기 설계 프로세스의 일부를 발췌한다[5].

1.1.1 기본 구조

보잉사의 비행기 설계에는 수 백명의 엔지니어가 함께 일한다. 이들은 크게 두 그룹으로 나뉜다.

비행기를 설계하는 설계 엔지니어 그룹과, 표준 부품을 담당하는 표준 엔지니어 그룹이다. 비행기를 설계할 때 설계 엔지니어는 표준 부품의 재사용을 먼저 그리고 최대한

노력하여야 한다.

두 그룹은 서로 다른 별도의 워크플로우 시스템들을 운영한다. 이 워크플로우 시스템들은 서로 벤더가 다를 수 있다. 이 워크플로우 시스템들은 표준 인터페이스를 통해서 서로 연동될 수 있어야 한다. 하나의 워크플로우가 여러 워크플로우 시스템에 분산되어 실행될 수 있다.

1.1.2 표준 부품 선택 워크플로우 시나리오

필요한 부품을 찾기 위한 설계 엔지니어의 표준 부품 검색은 설계 워크플로우의 한 스텝으로 실행된다. 설계 엔지니어는 표준 부품을 검색하기 위한 워크플로우 스텝을 표준 부품 그룹의 워크플로우 시스템에 임의로 생성할 수 있다. 이렇게 생성된 검색 스텝에 의해 표준 부품의 검색이 수행된다. 검색 결과 보고서가 설계 엔지니어의 워크플로우에 전달된다.

1.1.3 표준 부품 (재)설계 워크플로우 시나리오

표준 부품 검색에서 필요한 부품을 찾지 못했다면, 설계 엔지니어는 기존 표준 부품을 재설계하거나 아니면 새로운 표준 부품을 설계하도록 표준 그룹에 요청하게 될 것이다. 요청과 함께 설계 엔지니어로부터 필요한 부품에 대한 요구사항이 전달되어야 한다.

표준 부품 재설계 워크플로우의 선두에서, 설계 엔지니어로부터 전달된 요구사항으로 표준 엔지니어가 재 검색을 수행한다. 엄청난 수량의 표준 부품들 속에서 설계 엔지니어가 적당한 표준 부품을 놓쳤을 수 있기 때문이다. 설계 엔지니어가 제안한 표준 부품 설계 혹은 재설계 의견과는 별도로, 표준 그룹은 상황을 자체적으로 평가한다.

1.1.4 표준 부품 수정 시나리오

정확히 말하면, 표준 부품 수정이 아니고 기존 표준 부품을 기초로하여 새 부품을 설계하는 것이다. 이들 표준 부품들 간의 관계는 데이타베이스에 기록되어야 한다.

표준 엔지니어는 설계 엔지니어의 요구사항을 검토하고 필요에 따라 추가 정보를 요구한다. 이 정보 교환은 데이타베이스에 기록되어야 한다. 표준 부품의 종류에 따라 담당하는 표준 엔지니어 그룹이 세분된다. 워크플로우의 스텝에 작업자를 할당 할 때 이 관계가 고려되어야 한다.

표준 부품 수정 워크플로우는 사전에 전체가 정의되어 있지 않다. 표준 부품 수정을 위한 정보가 충분하면, 표준 엔지니어는 표준 부품을 수정하기 위한 나머지 워크플로우 스텝들을 구성한다.

표준 엔지니어가 나머지 워크플로우 스텝들을 구성하자마자, 이들은 프로젝트 관리 도구에서 일정이 부여된다. 프로젝트 관리 도구는 참여한 엔지니어들의 작업 부하를 고려하여 일정을 정한다.

일정이 부여되자 마자, 먼저 일정이 설계 엔지니어에게 통보된다. 만약 일정이 너무

늦다면 설계 엔지니어는 워크플로우를 취소할 수 있다. 또는 요구사항을 완화하고 표준 엔지니어에게 다시 전달할 수 있다.

1.2 작업할당 정책의 예

주어진 문제로부터 출발하여 작업할당 정책을 구현하는 과정을 보이기 위해, 위 예제 시나리오로부터 다음과 같은 작업할당 정책을 추출한다.

① 엔지니어들은 모두 표준 부품을 검색할 수 있다.
② 설계 엔지니어가 표준 부품에 대한 수정을 요청한다.
③ 표준 부품의 종류에 따라 담당하는 표준 엔지니어 그룹이 세분화된다. 워크플로우의 스텝에 작업자를 할당 할 때 이 관계가 고려되어야 한다.
④ 표준 부품 수정 요청에 대한 보완은 그 수정 요청을 제출한 엔지니어가 담당해야 한다.
⑤ 여러 워크플로우 시스템에 분산된 작업들을 모두 고려하여 작업자의 작업부하를 산정하고 이를 바탕으로 작업을 할당한다.

2. 관련 연구

워크플로우 시스템에서 작업할당 정책의 설계 및 구현은 프로젝트의 성공을 위해서 매우 중요한 요소이지만 이에 대한 연구는 상대적으로 매우 적으며[12], 최근에 워크플로우 시스템에서의 웹 서비스의 비중이 커지면서 작업 할당 측면에 대한 관심은 더 적어지고 있다[13]. 예를 들어 작업 담당자를 표현하기 위한 표기법이 WSCI[3], WSCL[4], BPEL4WS[7] 같은 표준에 포함되어 있지 않다.

워크플로우 시스템에서 작업할당 정책의 구현을[6]에서 볼 수 있다. 이 연구에서는 작업할당정책을 표현하기 위한 어휘로서, 객체와 관계를 먼저 정의하고, 이를 바탕으로 작업할당 정책을 표현하는 메타 모델 접근 방법을 제시하고 있다. 표현된 작업할당 정책은 SQL 명령으로 변환되어 관계형 데이타베이스에서 실행된다. 확장 가능한 어휘로 작업할당 정책을 표현하는 것은 우리의 연구와 같으나, 이 연구에서 제시된 텍스트 기반 언어는 모델의 무결성 유지를 위한 무결성 제약을 표현하지 못한다.

관계형 데이타베이스를 이용한 작업할당 정책의 구현을 [8]에서도 볼 수 있다. 작업할당 정책을 자격 정책, 요구사항 정책, 대리 정책으로 구별하여 SQL 과 유사한 문법의 텍스트 기반 언어로 표현한다. [6]와 마찬가지로 이 연구에서 제시된 언어도 모델의 무결성 유지를 위한 무결성 제약을 표현하지 못한다.

[2]에서는 작업할당을 위한 메타 모델과 그 UML 클래스 다이어그램이 제시되었다. 이 연구의 목표는 작업할당을 위한 조직 모델을 여러 워크플로우 시스템간에 서로 전달할 수 있기 위한 메타 모델과 그 XML 표현을 제시하는 것이며, 우리의 연구와는

달리 확장 가능한 작업할당 모델링 및 그 구현 방법을 제시하고 있지 않다.

[15]은 UML을 사용한 프로세스 모델링을 보여준다. 이 연구에서는 모델의 무결성 유지를 위한 무결성 제약을 정의하기 위한 메타 모델 접근 방법을 제시하고 있다. 그러나 이 연구는 사례 데이타의 관리 및 질의 방법에 대한 내용을 포함하고 있지 않기 때문에 작업할당 정책의 구현에 대한 연구는 아니다.

우리의 연구에서는 논리 데이타베이스 시스템인 ConceptBase[9]에 메타 모델, 개념모델, 사례 데이타를 저장하고 관리한다. ConceptBase의 지식 표현 언어인 Telos는 메타 모델과 개념 모델을 표현하기에 적당한 단순하면서도 확장 가능한 표현력을 제공하며, Telos의 제한 규칙에 의해 모델의 무결성 제약이 구현된다[14].

3. 메타 모델링

메타 모델링은 다음의 3계층으로 구성된다.

- 메타 모델 계층
- 개념 생성 계층
- 개념 모델 계층

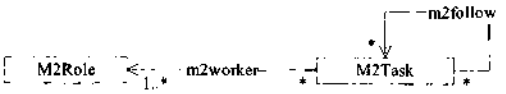
메타 모델링 접근 방법의 절차는 다음과 같다.

1) 작업할당 정책을 표현하기 위한 어휘를 메타 모델 계층과 개념 생성 계층에서 생성한다.
2) 생성된 어휘를 사용하여 개념 모델 계층에서 작업할당 정책을 표현한다.
3) 념 모델에서 표현된 작업할당 정책은 ConceptBase 코드로 자동 변환된다. 생성된 ConceptBase 코드와, 미리 구현된 ConceptBase 규칙과, 추가되는 ConceptBase 질의로 작업할당 정책이 구현된다.

3.1 어휘 생성

메타 모델은 개념 모델에 대한 스키마(schema)이다. 메타 모델은 개념 모델에 등장하는 요소들의 스테레오타입(stereotype)을 정의한다. 〈그림 1〉의 M2Role, M2Task는 개념 모델에 등장하는 개념들에 대한 스테레오타입이다. 예를 들어 〈그림 2〉의 Project Manager, Engineer, Standard Engineer는 M2Role 스테레오타입의 사례이다.

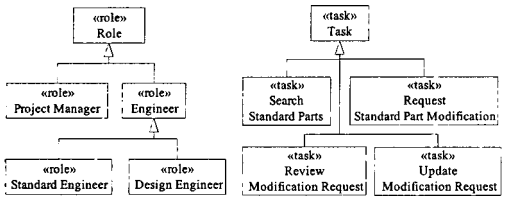
개념에 대한 스테레오타입뿐만 아니라, 연



〈그림 1〉 메타 모델

관들에 대한 스테레오타입도 메타 모델에 정의된다. 〈그림 1〉에서 m2worker는 작업을 담당할 작업자를 표현하기 위한 스테레오타입이고, m2follow는 작업들간의 실행 순서를 표현하기 위한 스테레오타입이다.

이름이 시사해 주듯이, 개념 생성 계층은 개념을 생성한다. 생성되는 개념은, 메타 모



〈그림 2〉 개념 생성

델에 정의된 스테레오타입의 사례들이다. 예를 들어, 〈그림 2〉의 Search Standard Parts, Request Standard Part Modification은 〈그림 1〉의 M2Task의 사례이다.

메타 모델 계층과 개념 생성 계층의 목적은, 개념 모델에서 사용될 어휘를 생성하는 것이다. 〈그림 1〉과 〈그림 2〉에서 생성된 어휘들을 사용하여, 다음 절에서 작업할당 정책을 표현한다.

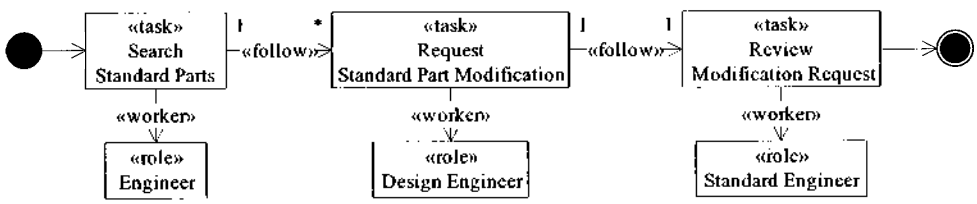
3.2 개념 모델

〈그림 3〉은 3.1절에서 생성된 어휘를 사용하여 1.2절의 작업할당 정책 ①, ②를 표현한다. Search Standard Parts 와 Engineer 사이의 〈〈worker〉〉 연관은 〈그림 1〉의 m2worker 스테레오타입의 사례이다. Search Standard Parts는 M2Task의 사례이고, Engineer는 M2Role의 사례이므로, 이들 사이의 〈〈worker〉〉 연관은 〈그림 1〉에 정의된 m2worker 스테레오타입의 제약을 만족한다(5.1절 참고).

〈〈worker〉〉 연관에 의해서 작업할당 정책 "① 엔지니어들은 모두 표준 부품을 검색할 수 있다"와 "② 설계 엔지니어가 표준 부품에 대한 수정을 요청한다"가 표현된다. Search Standard Parts와 Engineer 사이의 〈〈worker〉〉 연관은 작업할당 정책 ①을 표현하고, Request Standard Part Modification과 Design Engineer 사이의 〈〈worker〉〉 연관은 작업할당 정책 ②를 표현한다.

메타 모델의 m2worker 연관이 개념 모델의 〈〈worker〉〉 연관에 대한 제약을 정의하듯이, 개념모델의 연관도 사례 데이타에 대한 제약을 정의한다. 예를 들어, Request Standard Part Modification의 사례에서 출발하는 〈〈worker〉〉 연관의 사례는 반드시 Standard Engineer나 그 하위 클래스의 사례를 연결하여 한다(5.1절 참고).

〈그림 3〉의 〈〈follow〉〉 연관은 작업들 간의 실행 순서를 표현하고 있다. 예를 들어 Search Standard Parts 작업 뒤에는 Request Standard Part Modification 작업이 없거나 하나 이상 연결될 수 있다. 필요한 표준 부품의 검색에 성공했으면 워크플로우는 종료된다. 검색에 실패했을 경우에는 필요한 부품



〈그림 3〉 개념 모델

을 만들기 위해, 표준 부품들에 대한 수정 요청이 여러개 만들어질 수 있다.

메타 모델 계층과 개념 생성 계층은, 단지 개념 모델에서 사용될 어휘를 생성하는 것 뿐만 아니라, 개념 모델에서 그 어휘의 사용에 대한 제약 즉 그 어휘의 문법도 규정한다. 예를 들어 〈그림 3〉의 〈〈worker〉〉 연관은 m2worker 스테레오타입의 사례이므로 〈그림 1〉에 표현된 다중성의 제약을 받는다. 즉 M2Task의 사례인 Search Standard Parts, Request Standard Part Modification, Review Modification Request는 적어도 하나 이상의 〈〈worker〉〉 연관으로 M2Role의 사례와 연결되어야 한다(5.3절 참고).

〈그림 1〉 메타 모델의 m2worker, m2follow은 개념 모델에 대한 제약을 정의한다. m2worker 스테레오타입의 사례들은, 〈그림 1〉의 메타 모델에 정의된 제약을 만족해야 한다. 즉 m2worker의 사례들은 M2Task의 사례에 M2Role의 사례를 연결하여야 한다(5.1절 참고).

이런 제약들은 ConceptBase에 의해서 강제되어, 개념 모델의 일관성을 유지하는데 도움을 준다. 〈그림 3〉은 이 제약들을 모두 만족하고 있다.

3.3 개념 모델의 구현

UML 다이어그램으로 표현된 메타 모델, 개념 생성, 개념 모델은 모델링 도구에서 ConceptBase 코드로 자동 변환되어 ConceptBase 데이타베이스에 저장된다. 아래는 〈그림 1〉의 M2Task에 대해 생성되는 ConceptBase 코드이다. M2Task 메타 클래스는 ConceptBase의 클래스로 구현되고, m2worker, m2follow 연관은 ConceptBase의 링크로 구현된다.

```
Class M2Task with
  attribute
    m2worker : M2Role;
    m2follow : M2Task
end
```

아래는 〈그림 3〉의 RequestStandardPartModification에 대한 ConceptBase 코드이다.

```
SimpleClass RequestStandardPartModification
in M2Task with
  M2follow, "from(1)", "to(*)"
    f1 : ReviewModificationRequest;
    w1 : DesignEngineer
end
```

위 코드에서 "from(1)"과 "to(*)"은 다중성 제약을 정의한다. Request Standard Part Modification과 Review Modification Request 사이의 연관에 1대 다의 다중성 제약이 있음을 의미한다. 다중성 제약을 ConceptBase 규칙으로 구현하였다(5.3 절 참고).

4. 작업할당 정책의 확장

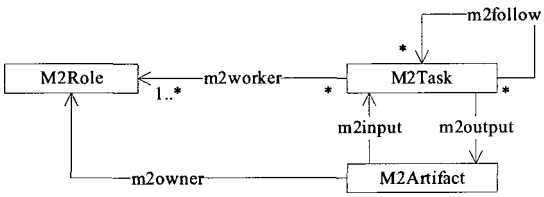
2절의 메타 모델에서 생성된 어휘로는 작

업할당 정책 ③, ④를 표현할 수 없다. 4절에서는 새로운 요구사항을 반영하기 위해, 메타 모델을 확장하여 새 어휘를 생성하고, 이를 사용하여 추가된 작업할당 정책을 구현하는 과정을 보인다.

4.1 새 어휘의 추가 생성

작업할당 정책 ③을 표현하려면, 작업의 산출물을 언급할 수 있어야 한다. 작업의 산출물을 언급하기 위한 어휘를 메타 모델에 추가한다. 〈그림 4〉의 M2Artifact는 작업의 산출물들에 대한 스테레오타입이다. 〈그림 5〉의 Standard Part, Standard Part A, Standard Part B는 M2Artifact 스테레오타입의 사례이며, 작업의 산출물을 표현한다.

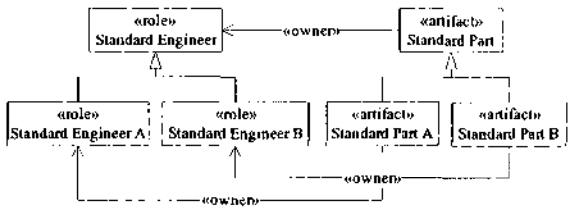
〈그림 4〉의 m2input, m2output 연관은 작업과 산출물의 입출력 관계를 표현하기 위한 연결의 스테레오타입이고, m2owner 연관은



〈그림 4〉 확장된 메타 모델

산출물과 담당자를 연결하기 위한 연결의 스테레오타입이다. 〈그림 5〉의 〈〈owner〉〉 연관은 m2owner 스테레오타입의 사례이다. 예를 들어 산출물 Standard Part B는 Standard Engineer B가 담당해야 함을 표현한다. 〈그림 5〉는 작업할당 정책 "③ 표준 부품의 종류에 따라 담당하는 표준 엔지니어 그룹이 세분화된다." 를 표현한다.

4.2 개념 모델



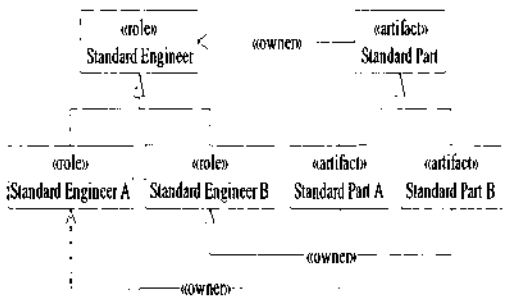
〈그림 5〉 확장된 개념 생성

〈그림 6〉는 4.1절에서 추가된 어휘를 사용하여 작업할당 정책 "③ 표준 부품의 종류에 따라 담당하는 표준 엔지니어 그룹이 세분된다. 워크플로우의 스텝에 작업자를 할당할 때 이 관계가 고려되어야 한다"를 표현한다. 작업 Review Modification Request 는 그 입력 산출물인 Standard Part의 담당자인 Standard Engineer 에게 할당되어야 한다.

또한 〈그림 6〉는 작업할당 정책 "④ 표준 부품 수정 요청에 대한 보완은 그 수정 요청을 제출한 엔지니어가 담당해야 한다" 를 표현 한다. 작업 Request Standard Part Modification와 작업 Update Modification Request는 동일한 Design Engineer에게 할당되어야 한다.

4.3 개념 모델의 구현

〈그림 6〉 개념 모델에서 작업할당 정책



〈그림 6〉 확장된 개념 모델

③은 아래의 ConceptBase 질의로 구현된다. ConceptBase에서 질의는 클래스이며, 질의의 결과가 질의 클래스의 사례이다. Review Modification Request 작업에 할당할 작업자를 조회하여 리턴하는 ReviewModification Request_worker 질의 클래스는 Standard-Engineer 클래스의 하위 클래스이다. 즉 질의 결과는 StandardEngineer 의 사례 집합의 부분 집합이다.

```
GenericQueryClassReviewModificationRequest
_Worker isA StandardEngineer with
parameter
    rmr : ReviewModificationRequest
constraint
    r : $ exists sp/StandardPart (sp input
    rmr) and (sp owner this) $
end
```

ReviewModificationRequest_Worker 질의 클래스의 파라미터는 Review Modification Request 의 사례 rmr 이다. 파라미터로 주어진 작업 rmr 의 작업자를 선택하기 위한 제약조건이 질의 클래스의 constraint 속성으로 표현된다.

작업자를 구하기 위한 ReviewModification Request의 사례 rmr이 파라미터로 주어졌을 때, StandardPart의 어떤 임의의 사례 sp가 존재하여, sp는 rmr의 입력 산출물이고, 이 sp의 소유자인 StandardEngineer의 사례들이, 이 질의의 결과이다. 즉 Review Modification Request의 입력 산출물인 Standard Part의 소유주인 Standard Engineer에게 이 작업이 할당되어야 함이 ConceptBase 질의로 구현되었다.

5. 무결성 제약

모델의 무결성을 유지하기 위해서 다음 무결성 제약이 강제된다.

5.1 연관의 사례화 제약

연관의 사례화는 다음 제약을 만족하여야 한다.

개념 x, y와 연관 a가 있어서, [x] —a→ [y] 이고,
개념 s, t와 연관 b가 있어서, [x] —b→ [y] 이고,
b가 a의 사례이라면,

s는 x혹은 상속구조에 의한 x의 하위 개념의 사례이어야 하고, t는 y혹은 y의 하위 개념의 사례이어야 한다.

연관의 사례화 제약은 ConceptBase 엔진에 의해서 강제된다.

5.2 연관의 상속 제약

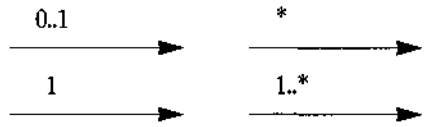
연관의 상속은 다음 제약을 만족하여야 한다.

개념 x, y와 연관 a가 있어서, [x] -a-> [y] 이고,
개념 s, t, u, v와 연관 b, c가 있어서,
[s] -b-> [t] [u] -c-> [v]
v 는 t의 하위 개념이어야만 한다.

연관의 상속 제약은 ConceptBase 엔진에 의해서 강제된다.

5.3 연관의 다중성 제약

연관에 부여 가능한 다중성 제약의 종류는 아래 그림과 같다.



이들은 ConceptBase 코드에서, 연관에 부여 가능한 속성들인 "from(0.1)", "from(1)", "from(1..*)", "to(0..1)", "to(1)", "to(1..*)"로 표현된다. "from(...)"는 연관의 시작 쪽 다중성을 정의하고, "to(...)"는 연관의 끝 쪽 다중성을 정의한다. "from(*)"과 "to(*)" 아무런 제약이 없는 것이므로 의미가 없다.

다중성 제약의 검사는 ConceptBase 규칙으로 구현되었다. 아래 코드는, "to(1)" 다중성을 검사하여 제약을 만족하지 않는 사례를 Inconsistent 클래스의 사례로 분류하는 ConceptBase 규칙이다. Inconsistent 클래스의 사례들은 프로세스 모델러 UI에서 붉은 색으로 표시되기 때문에, 제약을 위반하는 요소를 쉽게 발견할 수 있다.

```
Class Inconsistent with rule, attribute
 to1:
   $ forall p/Class!"to(1)"
   (exists c,m,d/VAR x/VAR P(p,c,m,d)
and In(x,c) and
   (exists y1, y2/VAR In(y1,d) and
In(y2,d) and
   A(x,m,y1) and A(x,m,y2) and
   not IDENTICAL(y1, y2))) ==>
   In(p, Inconsistent) $
end
```

6. 도구의 구현

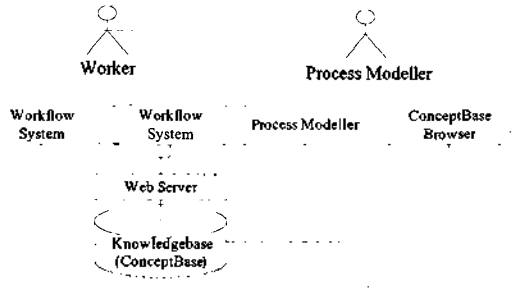
본 연구의 작업할당 정책 모듈은, 소프트웨어 프로세스에 대한 워크플로우 시스템 개발 연구의 한 모듈로서 개발되었다[1]. ConceptBase로 구현된 작업할당 모듈은, 여러 워크플로우 시스템들이 공유하는 중앙 정보 저장소 형태이다.

워크플로우 시스템과 ConceptBase의 대화는 단방향의 요청과 응답 구조이다. 워크플로우 시스템이 작업할당을 요청하면 ConceptBase는 질의를 실행하고 결과를 리턴한다. 워크플로우 실행과 관련된 개념 모델의 사례 데이타는 워크플로우 시스템들로부터 모아져 ConceptBase에 전달되어 저장된다.

작업자의 작업 부하를 고려하여 작업할당

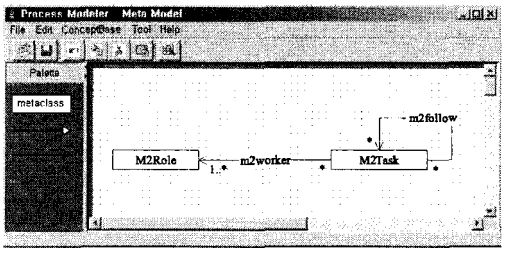
을 하려면, 일정과 관련된 어휘가 추가되어 작업의 일정 정보를 포함하는 개념 모델을 정의하여야 한다. 여러 워크플로우 시스템으로부터 모아져 전달되는 개념 모델의 사례 데이타를 조회함으로써 여러 워크플로우 시스템에서 진행중인 작업들과 작업자의 작업 부하를 알 수 있다. 1.2절의 작업할당 정책 "**⑤ 여러 워크플로우 시스템에 분산된 작업들을 모두 고려하여 작업자의 작업 부하를 산정하고 이를 바탕으로 작업을 할당한다.**"는 여러 워크플로우로 시스템으로부터 전달된 개념 모델의 일정 관련 사례 데이타를 조회함으로써 구현될 수 있다.

여러 워크플로우 시스템과들과의 인터페이스를 단순화할 수 있도록, 이 요청과 응답을 XML로 표현하여 HTTP 프로토콜을 통해 전달하도록 구현하였다. HTTP 프로토콜 통신은 Apache Tomcat웹 서버를 사용함으로써 쉽게 구현된다[16]. ConceptBase가 제공하는 Java 인터페이스는 단순하면서도 강력하기 때문에 워크플로우 시스템과의 인터페이스는 비교적 쉽게 구현된다[10].



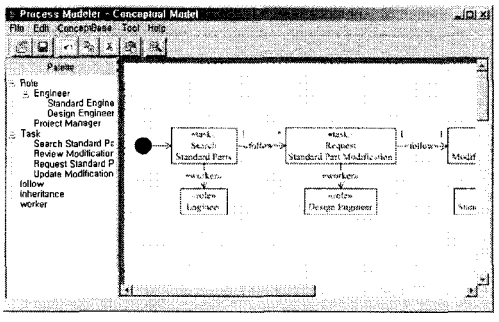
〈그림 7〉 시스템 구조

제 3장과 제 4장에서 보인 작업할당 모델은 프로세스 모델러의 UI를 사용하여 작성된다. 〈그림 8〉은 메타 모델을 작성하는 프로세스 모델러의 UI이다. 왼쪽 팔레트 창에 표시된 요소를 드래그 드롭하여 모델을 작성한다. 프로세스 모델러에서 작성된 모델은 ConceptBase 코드로 자동 변환되어 ConceptBase 에 저장된다.



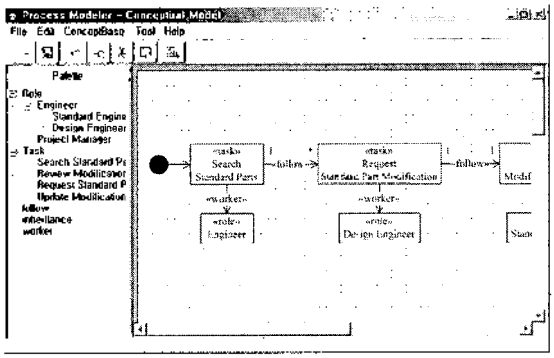
〈그림 8〉 Process Modeler - Meta Model

〈그림 9〉는 개념 생성을 작성하는 프로세스 모델러의 UI이다. 메타 모델에서 정의된 스테레오타입의 목록이 왼쪽 팔레트 창에 표시된다. 이 스테레오타입들을 드래그 드롭하여 필요한 개념을 생성하고 그들간의 관계를 표시한다.



〈그림 9〉 Process Modeler - Concept Creation

〈그림 10〉은 개념 모델을 작성하는 프로세스 모델러의 UI이다. 개념 모델에서 작성된 개념들과 메타 모델에서 작성된 연관 스테레오타입이 왼쪽 팔레트 창에 표시된다. 이들 개념과 연관을 드래그 드롭하여 개념 모델을 작성한다.



〈그림 10〉 Process Modeler - Conceptual Model

7. 결 론

본 연구의 공헌은 다음과 같다.

첫째, 각 조직에 주어진 임의의 고려할 사항들에서부터 작업할당 정책들을 정의하기 위한 확장 가능한 접근 방법을 제시하였다. 미리 정의된 한정된 어휘와 문법으로 작업할당 정책을 표현하는 것은 융통성이 부족하다. 작업할당 정책을 표현하기 위한 어휘를 개념과 연관으로 생성하고 개념과 연관 사이의 제약으로 어휘의 문법을 정의하고, 이들을 사용하여 작업 할당 정책을 표현하는 메타 모델 접근 방법을 제시하였다.

둘째, ConceptBase 지식 베이스를 이용하여 메타 모델 접근 방법을 효과적으로 구현할 수 있음을 보였다. 메타 모델, 개념 모델의 개념과 연관은 각각 ConceptBase의 개념과 연관으로 표현된다. 개념과 연관 사이의 제약은 ConceptBase의 제약과 규칙으로 구현될 수 있음을 보였다.

셋째, ConceptBase 코드 대신 UML 다이어그램으로 메타 모델과 개념 모델을 작성할 수 있음을 보였다. 작성된 모델은 ConceptBase 코드로 저장된다.

향후 연구 과제는 다음과 같다.

첫째, 안정성 확보. 구현에 ConceptBase를 활용하여 높은 생산성을 얻을 수 있었으나, 많은 사례 데이타를 저장하고 관리하기에 ConceptBase의 안정성이 충분하지 못했다. 안정성을 확보하기 위해, 사례 데이타를 관계형 데이타베이스에 저장하고, 이를 ConceptBase와 연동하는 방안에 대한 연구가 필요하다.

둘째, 메타 모델과 개념 모델은 UML 다이어그램으로 표현될 수 있으나, 규칙과 질의는 ConceptBase 코드로 직접 구현하여야 한다. 이전 연구에서 UML 다이어그램으로 질의를 표현하는 기법도 포함되었으나[11], 표현된 다이어그램의 높은 복잡도로 인하여, 본 연구에서 규칙과 질의는 ConceptBase 코드로 직접 구현된다. 규칙과 질의의 코딩을 도와주는 일종의 마법사 기능에 대한 연구가 필요하다.

참 고 문 헌

[1] 이형원, 이승진, " PRAiSE : 규칙 기반 프로세스 중심 소프트웨어 공학 환경", 정보과학회논문지 : 컴퓨팅의 실제, 제 11권, 제3호, pp. 246-256, 2005.

[2] W. M. P. van der Aalst, , A. Kumar, and H. M. W. Verbeek, "Organizational Modeling in UML and XML in the context of Workflow Systems", Proc. of the 18th ACM Symposium on Applied Computing (SAC 2003), ACM, 2003.

[3] A. Arkin, S. Askary, S. Fordin, W. Jekeli, K. Kawaguchi, D. Orchard, S. Pogliani, K. Riemer, S. Struble, P. Takacsi-Nagy, I. Trickovic, and S. Zimek, "Web Services Chorcography Interface (WSCI) 1.0", World Wide Web Consortium (W3C), 2002, p. 121.

[4] A. Banerji, C. Bartolini, D. Beringer, V. Chopella, K. Govindarajan, A. Karp, H. Kuno, M. Lemon, G. Pogossiants, S. Sharma, and S. Williams, "Web Services Conversation Language (WSCL) 1.0", World Wide Web Consortium (W3C), 2002, p. 22.

[5] C. Bussler, The Boeing Company, "Airplanc Design Process Standard Part Selection", OMG Document bom/98-02-14.

[6] C. Bussler and S. Jablonski, "Policy resolution for workflow management systems", Proc. of the 28th Hawaii International Conference on System Sciences, IEEE Computer Society, 1995.

[7] F. Curbera, Y. Goland, J. Klein, F. Leymann, D. Roller, S. Thatte, and S. Weerawarana, "Business Process Execution Language for Web Services, Version 1.0", BEA, IBM, Microsoft, 2002.

[8] Y.-N. Huang and M.-C. Shan, "Policy-Based Resource Management", Proc. of the 11th International Conference on Advanced Information Systems Engineering, Springer, 1999.

[9] M. Jarke, R. Gallersorfer, M.A. Jeusfeld and M. Staudt, "ConceptBase-A Deductive Object Base for Meta Data Management", Journal of Intelligent Information Systems, Vol. 4, No. 1, 1995.

[10] M. Jarke, M. Jeusfeld and M. Staudt, ConceptBase V5.2 Programmer's Manual.

[11] S. Lee, J. Shim, C. Wu, "A Unified Approach for Software Policy Modeling: Incorporating Implementation into a Modeling Methodology", Proc. of the 22nd International Conference on Conceptual Modeling (ER2003), Springer-Verlag, 2003.

[12] C. Moore, "Common Mistakes in Workflow Implementations", Giga Information

Group, Cambridge, MA, 2002.

[13] M. zur Muehlen, "Organizational Management in Workflow Applications", Information Technology and Management Journal, Kluwer Academic Publishers, Vol. 5, No. 3, pp. 271-291, 2004.

[14] J. Mylopoulos, A. Borgida, M. Jarke and M. Koubarakis, "Telos: a language for representing knowledge about information systems", ACM Transactions on Information Systems, Vol. 8, No. 4, ACM Press, 1990.

[15] A. Schleicher and B. Westfechtel, "Beyond stereotyping: metamodeling approaches for the UML", Proc. of the 34th Annual Hawaii International Conference on System Sciences, IEEE Computer Society, 2001.

[16] Apache Tomcat User Guide, http://tomcat.apache.org.

저 자 소 개

이승진 (E-mail : lsj@skhu.ac.kr)
서울대학교 해양학과 (학사)
서울대학교 계산통계학과 전산과학전공 (석사)
서울대학교 전산과학과 (박사수료)
성공회대학교 소프트웨어공학과 초빙교수
관심분야 소프트웨어 공학, 워크플로우, 지식 표현

우치수 (E-mail : wuchisu@selab.snu.ac.kr)
서울대학교 컴퓨터정보공학부 교수

이형원 (E-mail : lhw@kangnung.ac.kr)
서울대학교 계산통계학과 (학사)
서울대학교 계산통계학과 전산과학전공 (이학석사)
서울대학교 계산통계학과 전산과학전공 (이학박사)
University of Massachusetts at Amherst 방문연구원
현재 강릉대학교 정보전자공학부 컴퓨터공학전공 교수
관심분야 소프트웨어 프로세스, 형상 관리, 웹 컨텐츠 관리 등