IE Interfaces
Vol. 17, No. 2, pp. 190-199, June 2004.

경량 컴포넌트 구조의 XPDL 기반 워크플로 관리 시스템 개발

한관희[1†] · 김강용[2]

[1]경상대학교 산업시스템공학부 · 공학연구원 / [2]이놉스(주)

Development of an XPDL-Based Workflow Management System Using the Light-Weight Component Structure

Kwan-Hee Han[1] · Kang-Yong Kim[2]

[1]Department of Industrial & Systems Engineering, Gyeongsang National University, Jinju, 660-701
[2]Institute of Information Technology, INOPS Co. Ltd., Seoul, 157-030

Recently, many enterprises are introducing a workflow management system for the successful implementation of BPR(Business Process Reengineering). Proposed in this study is the workflow management system which has a light-weight component structure and an XPDL(XML Process Definition Language) file interpretation facility. The XPDL is the standard process definition exchange format developed by WfMC(Workflow Management Coalition). The major causes of inefficiency at current implementations of workflow management systems are the centralized workflow engine structure and the use of proprietary workflow definition format among most solutions. The proposed light-weight component structure in this study is the intermediate structure that takes the strength of both centralized and distributed workflow engines. And a prototype workflow system which uses an XPDL process definition file as input is developed through the thorough analysis of functional requirements.

Keywords: light-weight component, XPDL, workflow management system, WfMC

1. 서 론

최근의 급속한 기업 환경의 변화를 요약하면 공급자에서 고객으로의 힘의 이동, 국경을 초월한 기업 간의 경쟁, 기술과 시장의 급격한 변화로 특징지을 수 있다(Hammer and Champy, 1993). 이러한 극심한 환경 변화 아래서 생존하기 위해 각 기업들은 프로세스 혁신이나 BPR(Business Process Reengineering) 등을 통해 경쟁력을 확보하려 총력을 기울이고 있다. 특히, 최근 들어 프로세스 혁신이나 BPR의 구현을 성공적으로 수행하기 위해 많은 기업에서 워크플로 관리 시스템의 도입이 확산되고 있는 추세인데, 워크플로란 "전체적인 조직의 목표를 달성하기 위해 정해진 규칙들에 의거하여 참여자들 사이의 정보 및 업무가 전달되는 절차와 과정들을 자동화하는 것(OMG, 2000)"이라고 정의되고 워크플로 관리 시스템은 "소프트웨어를 이용하여 컴퓨터로 표현된 업무 규칙에 의해 실행 순서가 제어되는 업무 흐름을 정의하고 관리 및 실행하는 시스템(Hollingsworth, 1995)"이라고 정의되어 용어 자체에 이미 프로세스 자동화의 개념이 내포되어 있어서 Hammer and Champy(1993)에서는 워

본 연구는 산업자원부 전자상거래기술개발 사업(주문적응형 부품산업을 위한 웹기반 통합 제조 정보시스템 개발)의 지원으로 수행되었음.
†연락저자 : 한관희 교수, 660-701 경남 진주시 가좌동 900번지 경상대학교 산업시스템공학부, Fax : 055-762-6599,
E-mail : hankh@nongae.gsnu.ac.kr
2004년 1월 26일 접수, 1회 수정 후 2004년 4월 13일 게재 확정.

크플로 관리 시스템을 BPR 성공을 위한 핵심 정보 기술이라고 평가하고 있다.

그러나 최근까지 기업에 적용되고 있는 워크플로 관리 시스템은 몇 가지 중요한 문제점을 노출하고 있다. 첫째, 시스템 구조 측면에서 객체지향 시스템 이전의 호스트 중심이거나 서버 중심의 클라이언트/서버 구조에서는 하나의 단일 엔진이 조직 내에서 발생하는 모든 프로세스의 실행을 관리하는 중앙집중식 구조로 이루어져 있어서 대상 조직에 맞게 시스템 적용 규모를 조정하기 어렵고 기능 수정이나 확장이 어려운 취약점을 갖고 있다(Wheater *et al.*, 1998). 그리고 이러한 중앙집중식 구조는 단일 엔진으로 구성되어 있기 때문에 시스템 운영 관리는 용이하나 모든 실행 프로세스를 하나의 엔진에서 관리함으로써 서버에 큰 부하를 주며 워크플로 엔진에 장애가 발생하게 되면 실행되고 있는 모든 프로세스가 진행을 정지하게 되는 문제점이 있다. 이를 해결하기 위해 분산형 워크플로 엔진에 관한 연구가 다수 수행되었으나 분산형 워크플로 엔진을 운영하기 위해서는 분산 엔진 간 통신을 위한 미들웨어 비용이 추가적으로 소요되고 통신 오버헤드 때문에 수행 시간이 증가하는 문제점을 갖고 있어서 기업 간 워크플로나 B2B 통합과 같은 경우가 아닌 단일 기업 내에서의 사용에는 적합하지 않을 수 있다.

둘째, BPR 추진 시에 사용되는 기업 프로세스 모델링 도구와 워크플로 관리 시스템 간의 통합에 있어서 현재까지 대부분의 워크플로 관리 시스템은 고유의 프로세스 정의 포맷을 사용하고 있어서 제3의 프로세스 정의 도구에서 산출한 프로세스 정의 데이터를 수정없이 해석하여 실행하기 어려운 구조를 갖고 있다. 그래서 대부분의 BPR 프로젝트에서 생성된 프로세스 재설계 데이터를 상용 워크플로 관리 시스템에서 실행하기 위해서는 해당 시스템이 가지고 있는 별도의 워크플로 정의 도구를 가지고 다시 모델링해야 하는 낭비가 발생하고 있는 실정이다.

본 연구에서는 위에서 언급한 현행 워크플로 관리 시스템의 두 가지 중요한 문제점을 해결하기 위해, 1) 경량 컴포넌트 구조의 워크플로 엔진을 제안하며, 2) WfMC(Workflow Management Coalition)에서 제정한 표준 프로세스 교환 형식인 XPDL (XML Process Definition Language) 데이터를 입력받아 프로세스를 실행할 수 있는 워크플로 관리 시스템의 구조 및 기능을 제시하고, 3) 이를 바탕으로 정보 시스템으로 개발하고 사례 연구를 통해 그 유용성을 보이고자 한다.

2. 관련 논문 연구

2.1 워크플로 엔진 구조

중앙집중식 엔진 구조는 전체 시스템을 복수 개의 하위 시스템으로 분리하기 어려운 단일 구조로 구성되어 있고 적용 대상에 적합하게 시스템의 규모를 조정하기 어려우며 타 시스템과의 상호 운영이 어려운 문제점을 갖고 있다(Wheater *et al.*, 1998). 최근에는 기업의 글로벌화와 정보 기술의 발달에 따라 중앙집중식 엔진 구조의 단점을 극복하고 지리적으로 분산되어 있는 조직원들 간의 워크플로를 원활하게 하기 위해 분산형 워크플로에 관한 연구가 다수 수행되었다(Miller *et al.*, 1996; Das *et al.*, 1997; Muth *et al.*, 1998a; Silva Filho *et al.*, 1999). 완전 분산 구조에서는 프로세스를 구성하는 단위 업무마다 이를 관리하는 단위 업무 관리자가 존재하고 복수 개의 단위 업무 관리자가 지역적으로 분산되어 상호 통신에 의해 하나의 실행 프로세스 생애 주기를 관리하는 구조를 취하고 있다(Miller *et al.*, 1996). 이러한 분산 구조는 각 서버에서 실행 프로세스의 일부를 관리하므로 서버의 부하를 경감시킬 수 있고 하나의 서버에 장애가 발생하더라도 일부의 프로세스에만 영향을 미치게 된다. 그러나 분산형 구조의 워크플로 관리 시스템을 운영하기 위해서는 분산 객체들 간의 메시지 통신을 위해 CORBA(Common Object Request Broker Architecture), RMI(Remote Method Invocation), DCOM(Distributed Component Object Model)과 같은 별도의 추가 기능이 필요하고 분산 객체 간 통신 오버헤드 때문에 수행 시간이 길어질 확률이 높다.

워크플로 관리 시스템의 경량 구조에 관한 연구로는 Muth *et al.*(1998b), Alonso *et al.*(1998)과 Manolescu(2002) 등이 있는데 이들 연구는 시스템을 수행 기능에 의해 소수의 핵심 커널과 부가 기능으로 분리하였다는 공통점이 있고, 특히 Manolescu (2002)는 여기에 부가하여 핵심 컴포넌트와 부가 컴포넌트들을 객체지향 모델로 설계하였다는 특징이 있다. 본 연구에서는 시스템 구성 요소들을 객체지향 모델로 설계했다는 점에서 Manolescu(2002)와 유사하나 시스템을 기능면으로 나누지 않고 프로세스 관리자를 실행 프로세스 인스턴스 단위로 생성 및 삭제함으로써 경량화를 추구했다는 점에서 차이가 있다.

2.2 프로세스 정의 형식의 해석 및 실행

기업 프로세스의 정의와 정의된 프로세스의 시스템 간 교환을 위한 언어로는 WPDL(Workflow Process Definition Language) (WfMC, 1999), PSL(Process Specification Language)(Schlenoff *et al.*, 2000), PIF(Process Interchange Format)(Lee *et al.*, 1998) 등이 있으며 zur Muehien and Becker(1999)에서는 이들 언어에 대한 상호 비교 분석이 이루어졌다. 최근에는 XPDL(WfMC, 2002)과 BPML (Business Process Modeling Language)(Arkin, 2002)이 발표되었는데 XPDL은 XML(eXtensible Markup Language) 스키마를 기반으로 WPDL을 재정의한 것으로 WPDL에 비해 프로세스 정의 데이터의 유효성을 사전 검증할 수 있다는 점에서 주목을 받고 있으며 BPML 역시 XML 스키마를 기반으로 하고 있다.

표준적인 기업 프로세스 정의를 산출해내는 도구에 관한 연구로는, 우선 WPDL에 기반한 연구로 대상 프로세스를 고유의 모델링 방법으로 모델링하고 그 결과를 WPDL 파일로 변환하

여 저장하는 방법에 대한 연구가 이루어졌고(이창수 외, 2001; 신동일과 신동규, 2000), 김상배 외(2000)에서는 모델링 도구로 ICN(Information Control Net)(Ellis, 1979)을 이용하여 모델링하고 그 결과를 WPDL 파일로 저장하였다. XPDL에 기반한 연구로는 UML(Unified Modeling Language) 활동 다이어그램으로 대상 프로세스를 모델링하고 이를 XPDL 파일로 저장하는 프로세스 모델링 도구의 개발에 관한 연구가 수행되었다(한관희와 황태일, 2003).

표준 프로세스 정의 포맷을 해석하여 프로세스를 실행하는 워크플로 엔진에 관한 연구로는 고유의 모델링 도구인 프로세스 디자이너에 의해 모델링된 결과를 WPDL 파일로 저장하고 이 정보가 워크플로 엔진에 보내져서 프로세스 정의 해석기에 의해 해석되어 정의 데이터를 데이터베이스에 저장하고 이를 바탕으로 프로세스를 실행하는 워크플로 관리 시스템 구조에 대한 연구가 이루어졌으나(신동일과 신동규, 2000), XPDL 기반의 워크플로 엔진 구조는 아직까지 발표되고 있지 않다.

3. 경량 컴포넌트 구조

본 연구에서 제안하는 워크플로 엔진 구조는 단일 조직 내에서 워크플로 관리 시스템을 효율적으로 사용하기 위해 기존 중앙집중식의 단점을 개선한 경량 컴포넌트 구조로서 중앙집중식에서는 <그림 1>과 같이 하나의 프로세스 관리자가 모든 실행 프로세스 인스턴스의 생성에서 종료까지를 모두 관리하는 것에 비해 경량 컴포넌트 구조에서는 각 실행 프로세스 인스턴스 단위로 이를 관리하는 프로세스 관리자가 하나씩 생성되어 각 실행 프로세스들의 생애 주기를 관리하고 실행 프로세스가 완료되면 해당 프로세스 관리자도 시스템에서 소멸된다.

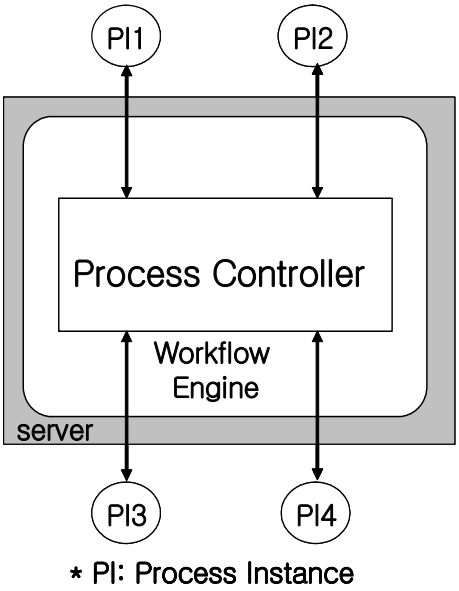

그림 1. 중앙집중식 구조.

즉, 중앙집중식에서는 실행중인 모든 프로세스 인스턴스들이 단일 엔진의 통제하에 실행됨에 비해 경량 컴포넌트 구조에서는 실행 프로세스의 생성을 요청받으면 해당 프로세스 인스턴스를 관리하기 위한 전담 프로세스 관리자 인스턴스를 먼저 생성하며, 생성된 프로세스 관리자 인스턴스가 요청받은 실행 프로세스 인스턴스를 생성하는 구조를 갖고 있다. 전담 프로세스 관리자 인스턴스는 해당 프로세스 인스턴스의 생애 주기를 관리하며 프로세스 인스턴스가 종료되면 시스템에서 삭제된다. <그림 2>는 경량 컴포넌트 구조를 도식화한 것으로 워크플로 엔진에서 실행 프로세스 기동이 필요한 시기에만 해당 프로세스 관리자 인스턴스를 생성함으로써 실행 프로세스가 적은 경우 중앙집중식 구조보다 부하가 적게 걸린다. 또한, 이 구조에서 분산 객체 미들웨어를 추가하면 분산 엔진 구조로 이전하기에 용이한 환경을 제공할 수 있다. 2장과 3장에서 분석한 중앙집중식 구조, 경량 구조, 분산 구조의 특성들을 정리하면 <표 1>과 같다.

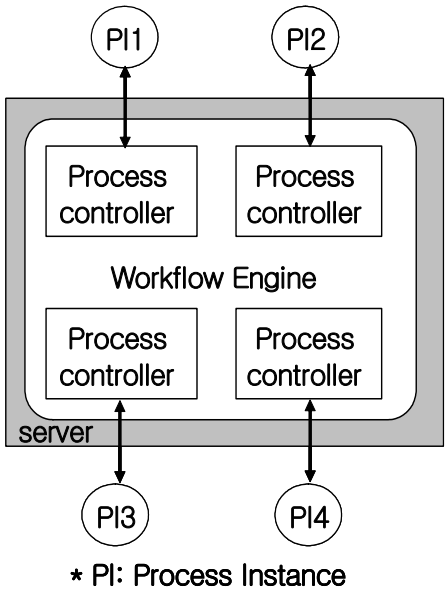

그림 2. 경량 컴포넌트 구조.

표 1. 워크플로 엔진 구조 비교

	중앙집중식 구조	경량 구조	분산 구조
시스템 복잡도	간단	중간	복잡
확장성	어려움	중간	쉬움
시스템 신뢰성	중간	중간	높음
관리 용이성	용이	용이	어려움
커스터마이제이션	어려움	용이	용이
유연성	낮음	중간	높음
통신부하	낮음	낮음	높음

4. 워크플로 관리 시스템 설계

본 연구에서 목표로 하는 시스템은 XPDL 형식의 프로세스 정의 데이터를 입력으로 받아 프로세스 실행을 관리하는 워크플로 관리 시스템(이하에서는 C³-P:EXE라 칭함)으로서 4장에서는 시스템 개발을 위한 설계 과정을 설명한다.

4.1 유즈케이스 다이어그램

문제 영역의 필요 기능을 판별하기 위해서는 우선 관련 액터와 시스템 사이의 상호 작용을 표현한 UML 유즈케이스 다이어그램을 작성하는데, 이는 <그림 3>에 나타나 있다. 시스템의 기능은 크게 XPDL 파일 관리와 실행 프로세스 관리로 나눌 수 있는데, 우선 파일 관리 부분에서는 프로세스 정의 담당자가 웹 상에서 시스템에 접속하여 작성된 프로세스 정의 XPDL 파일을 등록한다(XPDL file register). 관리자는 등록 파일의 유효성을 체크하여 이상이 없을 경우에는 시스템에서 XPDL 파일을 해석하여(XPDL file parse/interpret), 파일의 정보를 프로세스 정의 DB에 갱신한다(Build-time process database update). 실행 프로세스 관리 부분에서는 담당자가 프로세스를 실행시키기 위하여 프로세스 정의 DB에 저장되어 있는 프로세스 목록 중에서 필요한 프로세스를 선택하여 이를 실행 환경에 적합하게 재정의함으로써 프로세스를 기동시킨다(Process redefine & Process execute). 기동된 실행 프로세스는 업무 순서에 따라 담당자가 지정되어 담당자의 업무 목록(Worklist)에 저장된다. 담당자는 업무 목록을 참조하여 할당된 업무를 수행하고 수행 결과를 시스템에 피드백해 주는 작업을 반복 수행하게 된다(Worklist handle). 이 과정에서 관리자와 업무 담당자는 실행 프로세스들의 상태를 모니터링하고 필요 시에는 조정 행위를 한다(Monitor & control).

4.2 기능적 요구 사항 도출

<그림 3>의 유즈케이스 다이어그램을 분석해 보면 요구되는 기능은 XPDL 파일의 등록, 저장 및 해석과 관련된 기능과 일반적인 실행 프로세스를 관리하는 워크플로 엔진 기능 및 단위 업무의 수행을 위해 담당자들이 사용하는 워크플로 사용자 기능으로 분류될 수 있음을 알 수 있는데, 이를 정리하면 아래와 같다.

첫째, XPDL 파일 관리 부분에서는 <그림 4>에서와 같이 XPDL 파일을 원격지에서도 등록할 수 있게 웹 기반으로 구성되어야 하고 XPDL 파일 등록 기능, 저장소 관리 기능, XPDL 파일 해석 및 프로세스 정의 DB 갱신 기능 등이 있어야 한다.

둘째, 워크플로 엔진에서 요구되는 기능들은 아래와 같다.

① 프로세스 정의 DB에 저장된 프로세스 정의 정보의 조회 기능.

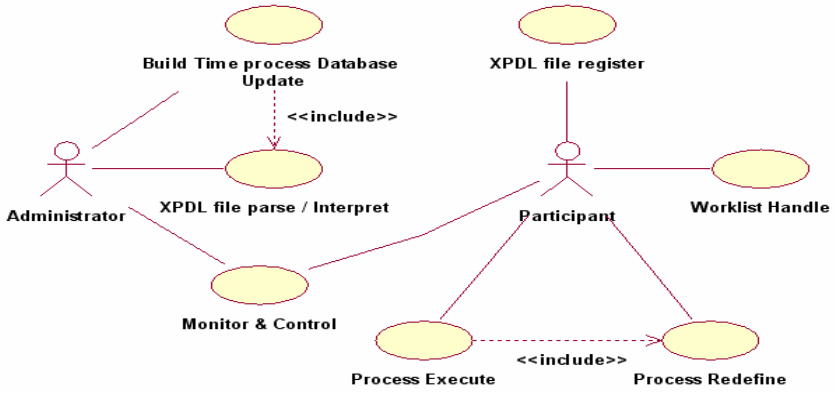

그림 3. C³-P:EXE 유즈케이스 다이어그램.

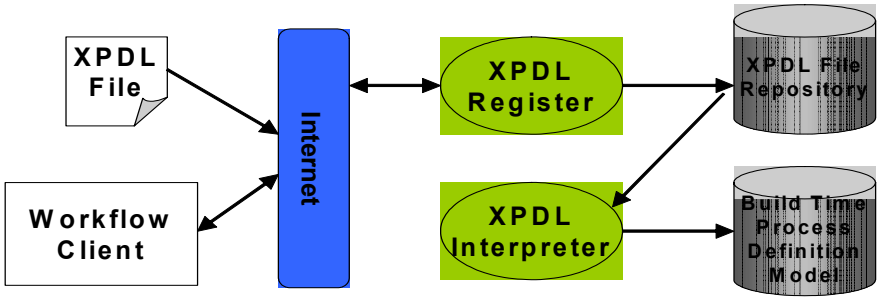

그림 4. XPDL 파일 관리 기능.

② 저장된 프로세스 정의 데이터 중에서 필요한 프로세스를 선택하여 기동시키는 기능. 이 때 단위 업무 담당자, 관련 데이터, 사용 s/w 등은 재정의가 가능해야 한다.
③ 실행 프로세스의 생애 주기 관리를 담당하는 프로세스 관리자의 생성 및 삭제 기능. 즉, 실행 프로세스가 하나 생성되면 그 프로세스의 생성에서부터 종료까지의 활동을 관리하기 위한 전담 프로세스 관리자가 생성되며 이 프로세스 관리자는 해당 프로세스가 종료되면 삭제된다. 이와 같은 구조가 기존의 단일 워크플로 엔진이 모든 실행 프로세스의 생성에서 종료까지를 모두 관리하는 중앙집중식 구조와 대별되는 경량 컴포넌트 구조의 핵심이다.
④ 수행해야 할 단위 업무를 해당 담당자의 업무 목록에 저장하는 기능. 이 때, 담당자가 역할이나 조직으로 정의되어 있으면 담당자 선정 규칙에 의해 구체적인 담당자를 정할 수 있는 기능이 있어야 한다.
⑤ 프로세스 진행에 맞추어 프로세스와 단위 업무 상태 변경을 관리하는 상태 관리 기능.
⑥ 실행 프로세스별 진행 상태 정보 제공 및 업무 조정 기능.

셋째, 워크플로 사용자 기능은 아래와 같다.
① 보안 및 사용자 역할에 따른 접근 관리 기능.
② 업무 목록 조회 및 새로운 업무 항목 선택 기능.
③ 담당 업무에 필요한 s/w 구동 기능.
④ 업무 수행 결과를 시스템에 피드백할 수 있는 기능. 수행 결과는 완료, 반려, 일시 중지 및 삭제 중에서 선택할 수 있어야 한다.

4.3 클래스 다이어그램

기능적 요구 사항을 만족시키기 위해 필요한 문제 영역 내의 객체와 객체가 갖고 있는 속성 및 객체 간의 관계를 표현하기위해 클래스 다이어그램을 작성하는데, 이는 <그림 5>에 나타나 있다. <그림 5>에서 점선으로 구분된 부분이 프로세스를 정의하는 모델(build time model)이고 프로세스 정의 모델을 포함한 전체가 워크플로 관리 시스템의 핵심인 실행 시간 모델(run time model)에 해당한다. 프로세스 정의 모델은 프로세스 정의 데이터를 저장하고 있는 모델로서 프로세스, 단위 업무, 단위 업무의 전이, 업무 담당자, 사용하는 응용 s/w 및 관련 데이터 등의 정적인 정보를 관리한다. 이에 반해 실행 시간 모델은 실행 프로세스가 생성되어 실행되는 과정에서 필요한 동적인 정보를 관리하는 모델로서 각 프로세스 인스턴스의 생성에서 종료까지의 전 과정을 관리한다.

담당자나 외부 이벤트에 의해 실행 프로세스의 생성을 요청받으면 워크플로 관리자(WorkflowMgt) 클래스에서 해당 프로세스를 관리하는 전담 프로세스 관리자(Process Controller)를 생성하며 워크플로 관리자 클래스는 복수 개의 프로세스 관리자를 관리한다. 해당 프로세스 관리자는 요청받은 실행 프로세스(Process)를 생성하며 하나의 프로세스는 복수 개의 단위 업

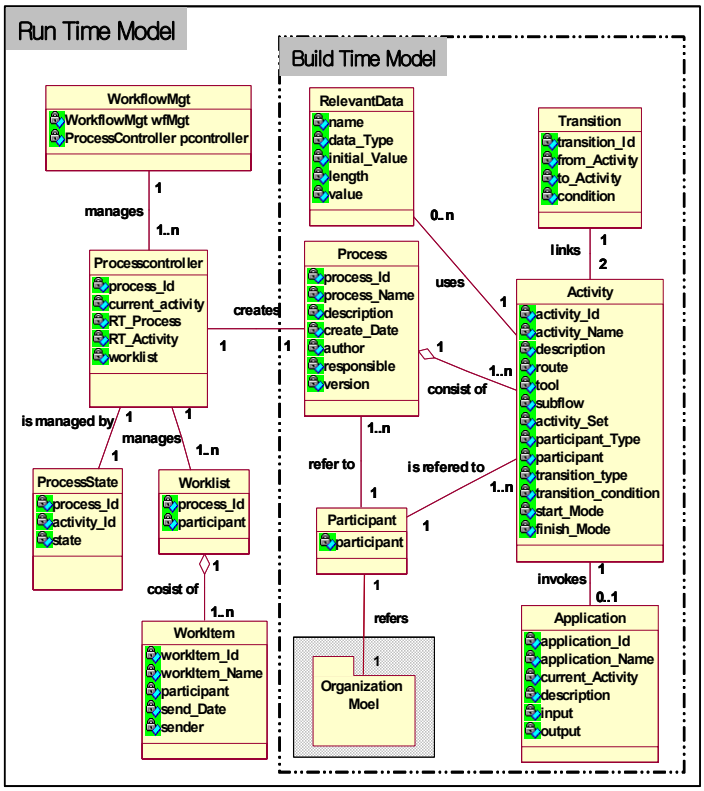

그림 5. C³-P:EXE 클래스 다이어그램.

무(Activity)로 구성된다. 즉, 하나의 실행 프로세스는 하나의 프로세스 관리자에 의해 관리되며 실행 프로세스가 종료되면 이를 관리하는 프로세스 관리자도 삭제되는 경량 컴포넌트 구조를 갖고 있다.

프로세스 관리자는 생성된 실행 프로세스의 상태(Process State) 클래스와 업무 목록(Worklist) 클래스를 관리한다. 즉, 기존의 중앙집중식 엔진 구조에서는 하나의 워크플로 관리자가 실행되고 있는 모든 프로세스의 상태와 업무 목록을 관리하는 것에 비해 경량 컴포넌트 구조에서는 하나의 프로세스 관리자는 하나의 실행 프로세스에 관한 상태 관리 및 업무 목록 관리만을 담당함으로써 실행 프로세스 관리에 필요한 부하를 분산시킨다.

단위 업무는 해당 업무 수행에 필요한 담당자(Participant), 관련 데이터(Relevant Data) 및 사용 s/w(Application) 클래스와 연결되고, 각 단위 업무는 이를 수행하는 담당자가 결정되어 담당자의 업무 목록에 저장된다. 업무 목록에는 복수 개의 업무 항목(Work item)이 저장될 수 있다. 담당자는 업무 목록을 참조하여 할당된 업무를 수행하고 수행 결과를 시스템에 피드백해 주는 작업을 반복 수행하게 된다. 담당자 클래스는 조직 모델과 연결되는데 조직 모델에서는 담당자와 역할, 담당자와 조직 단위 간의 관계를 다대다 관계로 모델링함으로써 CFT (Cross Functional Team)와 같은 한시적이고 유동적인 조직을 표현할 수 있게 한다(한관희와 박찬우, 2002).

5. 워크플로 관리 시스템 구현 및 사례 연구

5.1 시스템 구조

본 연구에서 개발한 시스템인 C^3-P:EXE 시스템은 분산 환경을 지원하기 위하여 웹 기반 시스템으로 구성하였고 범용성과 확장성을 고려하여 3계층(Tier) 구조로 설계하였으며 객체지향 개발을 위해 UML 모델링 도구로 래셔널 로즈를 사용하였다. 개발 언어로는 플랫폼 독립적인 자바를 사용하였으며 JDK (Java Development Kit)는 J2SE v1.4를 이용하였다. J2SE v1.4에는 자체적으로 XML을 파싱(Parsing)하는 클래스가 내장되어 있어 XPDL을 파싱할 때 이를 사용하였다. 사용자 인터페이스 부분은 JSP(Java Server Page)를 사용하였고, 응용 로직 부분은 자바 클래스와 자바 빈즈(Java Beans)를 이용하였다. 웹 서버로는 Apache 1.3.19과 서블릿(Servlet) 엔진으로 Tomcat 3.2를 사용하였으며 정보 객체들의 영속적인 저장 관리를 위해서 MS SQL 2000 서버를 사용하였다..

<그림 6>에서 개발된 시스템의 구조를 보여주고 있는데, 1) 사용자가 프로세스 정의 XPDL 파일을 웹상에서 중앙 저장소에 등록하는 XPDL 등록기와, 2) 관리자 권한을 가진 사용자에 의해 유효성 검증을 거친 XPDL 파일을 프로세스 정의 모델에 저장하는 XPDL 해석기, 3) 프로세스 정의 정보나 실행 프로세스 정보를 텍스트 형식으로 볼 수 있는 텍스트 기반 프로세스

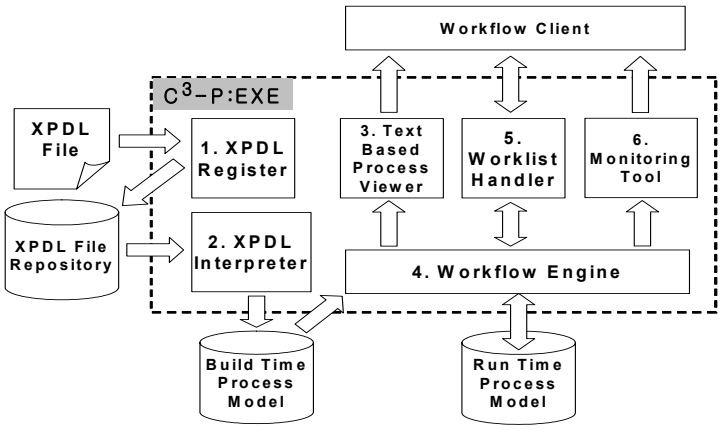

그림 6. C^3-P:EXE 시스템 구조도.

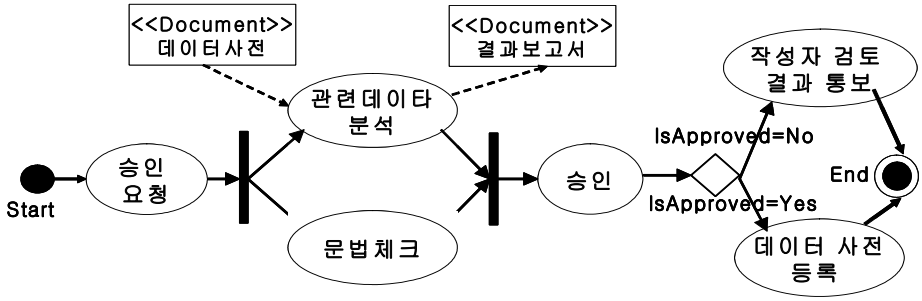

그림 7. 단위 데이터 등록 프로세스 시나리오.

뷰어, 4) 실행 프로세스의 생애 주기를 동적으로 관리하는 워크플로 엔진, 5) 프로세스의 진행 상태에 따라 담당자의 업무 목록에 단위 업무를 할당하고 담당자와 상호 작용하면서 단위 업무의 진행 상태를 관리하는 업무 목록 처리기 및 6) 실행 프로세스 및 단위 업무의 상태를 조회하는 모니터링 도구로 구성되어 있다.

5.2 사례 연구

대상이 되는 사례는 기업 정보 시스템을 개발하는 데 있어서 개발 생산성 제고와 유지보수를 용이하게 하기 위해 소스 프로그램에 사용되는 각종 데이터들을 표준화하여 여러 개발 프로젝트에서 표준화된 데이터만을 사용하도록 하는 '데이터 사전 관리' 중의 일부 프로세스이다(한관희, 2002). 여기서 데이터라 함은 프로그램 작성 시에 사용되는 각종 클래스명이나 속성명/메서드명 혹은 DB 테이블명/컬럼명 등을 지칭한다. 대상 프로세스는 데이터 사전 관리 중 단위 데이터를 데이터 사전에 등록하는 데 요구되는 '단위 데이터 등록' 프로세스이며 이를 UML 활동 다이어그램으로 나타내면 <그림 7>과 같다. 데이터 사전에 없는 단위 데이터를 신규로 사용할 필요가 있는 개발자는 승인 과정을 거쳐 신규 단위 데이터를 중앙 데이터 사전에 등록한 후에 사용해야 한다. 프로세스가 시작되면 첫 번째 업무가 '승인 요청'이고 승인 요청 이후에는 두 가지 단위 업무가 병렬적으로 수행되는데, 하나는 이미 존재하는 유사 데이터를 대신 사용할 수는 없는지를 판단하는 '관련 데이터 분석'이고 다른 하나는 승인 요청된 단위 데이터가 데이터 생성 규칙에 부합하는지를 판단하는 '문법 체크' 업무이다. '관련 데이터 분석' 업무를 위해서는 '데이터 사전' 문서를 입력으로 필요로 하며 업무 수행 결과로 '결과 보고서' 문서가 산출된다. 이 두 업무가 모두 완료되어야만 다음 '승인' 업무가 가능한데, 이 업무의 결과에 따라 승인이면 자동으로 데이터 사전 DB에 등록되며 부결되면 승인 요청자에게 반려 통지가 이메일로 발송된다.

시스템 입력 파일인 '단위 데이터 등록' 프로세스 정의 XPDL 파일은 UML 활동 다이어그램과 XPDL을 기반으로 하는 프로세스 모델링 도구인 C^3-P:DEF(한관희와 황태일, 2003)를 이용하여 산출하였는데 <표 2>에서는 단위 데이터 등록 프로세스 정의의 일부인 '관련 데이터 분석' 업무의 XPDL 파일 예를 나타낸다. 이 파일에는 업무 실행 유형, 업무 시작/종료 방법, 필요한 입 · 출력 문서 등의 정보가 표현되어 있다. 이하에서는 C^3-P:EXE 시스템의 기능을 사례 중심으로 설명한다.

5.2.1 프로세스 등록 및 해석기

제3의 프로세스 정의 도구에서 산출된 '단위 데이터 등록' 프로세스 정의 XPDL 파일을 <그림 8>의 상단 화면을 이용하여 시스템에 등록한다. <그림 8>의 하단 화면에서는 등록된 XPDL 파일의 상세 정보를 볼 수 있고 파일명을 클릭하면 브라우저 상에서 XPDL 파일 내용을 볼 수 있다. 그 후 관리자가 'XPDL Parsing'이라는 버튼을 클릭하면 시스템에서 XPDL 파일을 해석하여 필요한 정보를 프로세스 정의 DB에 저장한다.

표 2. 단위 업무 정의 XPDL 파일 예

```
<Activity Id="PDF00000005" Name="관련데이터분석">
        <Description/>
        <Implementation> <No/> </Implementation>
        <StartMode>      <Manual/> </StartMode>
        <FinishMode>     <Manual/></FinishMode>
        <ExtendedAttributes>
          <ExtendedAttribute Name="InputFile">
            <smi:InputFile Id="PDF00000006" Name="데이터사전"
            Type="Word" xpos="103.0" ypos="59.0">
              <smi:Description>데이터사전
                              </smi:Description>
            </smi:InputFile>
            </ExtendedAttribute>
          <ExtendedAttribute Name="OutputFile">
            <smi:OutputFile Id="PDF00000008" Name="결과보고서"
            Type="Word" xpos="509.0" ypos="73.0">
              <smi:Description>결과보고서
                              </smi:Description>
          </smi:OutputFile>
        </ExtendedAttribute>
    </ExtendedAttributes>
</Activity>
```

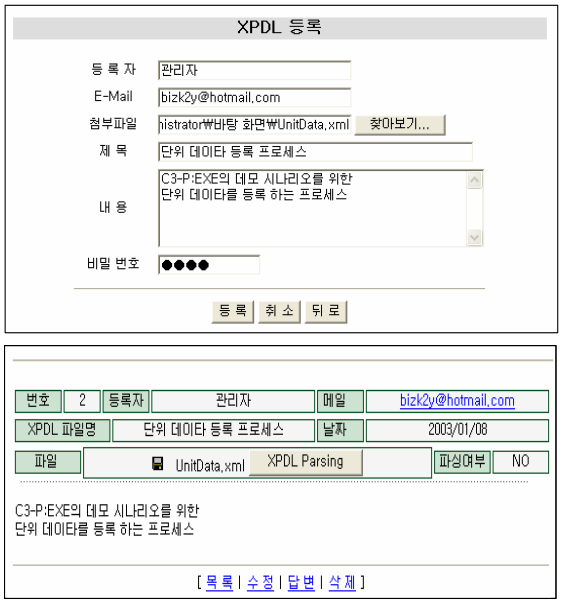

그림 8. XPDL 파일 등록 및 해석.

5.2.2 텍스트 기반 프로세스 뷰어

프로세스 뷰어에서는 프로세스 정의 DB에 저장되어 있는 프로세스들의 메타 정보와 프로세스를 구성하는 상세 단위 업무들 간의 선행 관계와 업무 실행 유형 및 담당자 정보 등을 <그림 9>에서와 같이 텍스트 형식으로 조회할 수 있다.

그림 9. 텍스트 기반 프로세스 뷰어.

5.2.3 프로세스 실행

프로세스 실행을 위해서는 프로세스 정의 DB에 저장된 프로세스 정의 템플릿을 선택하여 실행 대상 프로세스에 적합하게 몇 가지 정보를 재정의하여 사용한다. 재정의가 가능한 정보로는 실행 프로세스에서 사용하는 관련 데이터, 사용 s/w 및 단위 업무 담당자 정보이다. 프로세스 정의 시 업무 담당자가 조직이나 역할 이름으로 정의되어 있거나 정의되어 있지 않으면 실행 시에 실제 담당자를 결정해야 하는데, 현재 구현되어 있는 규칙은 모든 담당자에게 해당 업무를 할당하여 그 중 업무 항목을 먼저 선택한 사람에게 작업을 할당하는 선입선출(FCFS) 규칙과 담당자를 임의로 결정하는 임의지정(RANDOM) 규칙의 두 가지가 있다.

<그림 10>에서와 같이 프로세스를 재정의한 후 하단에 있는 '프로세스 실행' 버튼을 클릭하여 프로세스를 기동시킨다.

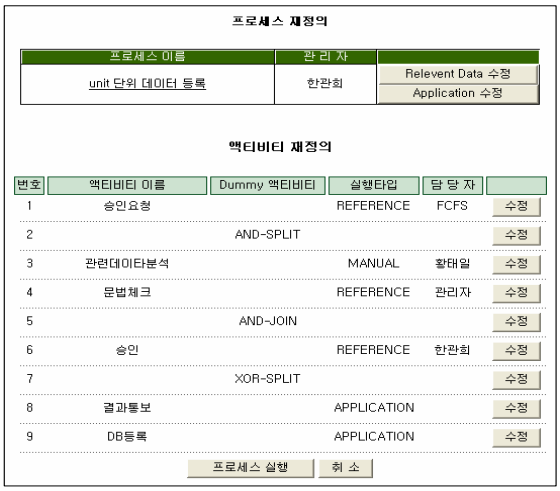

그림 10. 실행 프로세스 재정의.

5.2.4 작업 목록 처리기

각 업무 담당자들은 할당된 업무를 수행하기 위해 주기적으로 각자의 업무 목록을 조회하게 되는데 특정 담당자의 업무 목록이 <그림 11>에 나타나 있다. 업무 목록에는 이전 업무 담당자, 수행해야 할 업무, 수신 일자, 업무 상태 등이 표현되어 있는데 <그림 11>의 첫 번째 라인의 업무 항목은 '문법 체크'로서 이를 클릭하면 해당 업무를 수행하는 데 필요한 작업 정보들이 <그림 12>와 같이 나타나는데 왼쪽 화면에는 작업 수행에 필요한 관련 데이터들이 나타나며, 이 업무에 필요한 응용 s/w가 있으면 오른쪽 화면에 'Application' 버튼이 활성화되어 이 버튼을 누르면 해당 업무 수행에 필요한 프로그램이 구동된다. 이 경우에는 독립적인 응용 시스템인 데이터 사전 관리 시스템 중 '단위 데이터 등록' 화면이 <그림 13>에서와 같이 뜨게 된다. 담당자가 '단위 데이터 등록' 프로그램을 참조하여 업무를 수행한 후 수행 결과를 시스템에 피드백하기 위해서는 <그림 12>에서의 '프로세스 상태'라는 버튼을 클릭하면 <그림 14> 화면이 뜨는데, 여기에서 완료된 업무 항목('문법 체크')을 클릭하면 해당 업무 상태가 'running'에서 'completed'로 바뀌며 반려 버튼을 클릭할 경우에는 'running'에서 'aborted'로

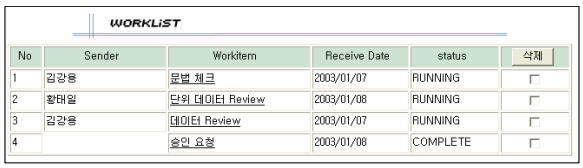

그림 11. 업무 목록.

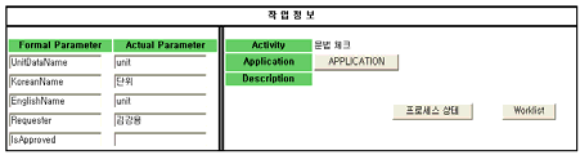

그림 12. 작업 정보.

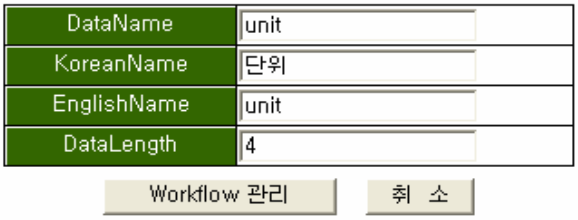

그림 13. 단위 데이터 등록 화면.

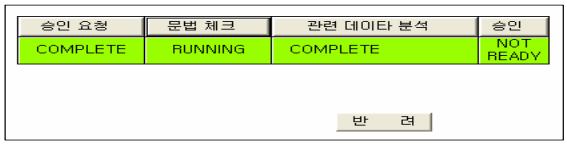

그림 14. 실행 프로세스 상태 관리.

바뀐다. 오직 'completed' 상태에서만이 다음 단위 업무로 이동할 수 있다. 이와 같이 각 프로세스나 단위 업무들은 여러 가지 상태를 갖고 있어서 각 상태에 따라 처리해야 할 절차가 다르게 되는데 이러한 상태 관리는 워크플로 엔진의 각 실행 프로세스 관리자가 담당한다. <그림 15>에 단위 업무의 상태 전이도를 나타낸다.

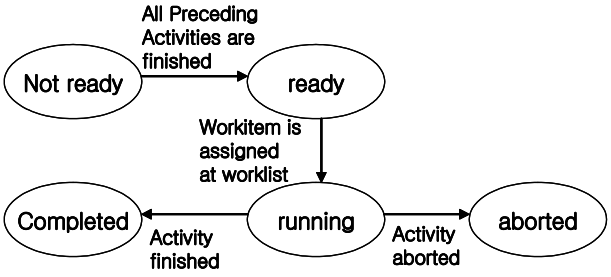

그림 15. Activity State Diagram.

5.2.5 프로세스 모니터링

실행 프로세스나 단위 업무의 진행 상태를 파악하기 위해서는 현재 상태를 모니터링하는 것이 필요한데 <그림 16>의 상단 화면에서는 현재 진행중이거나 완료된 프로세스 목록을 보여 주고 하단 화면은 단위 업무들의 구체적인 진행 상태를 나타낸다.

실행 프로세스 목록

번호	프로세스명	관리자	상 태	시작 시간	완료 시간
1	unit 단위 데이터 등록	김강용	RUNNING	2003/01/07 10:45:38	
2	Workflow 데이터 등록 프로세스	황태일	RUNNING	2003/01/08 05:03:05	
3	Data 단위 데이터 등록 프로세스	황태일	COMPLETE	2003/01/06 10:26:59	2003/01/07 15:22:12

실행 액티비티 목록

번호	액티비티명	Dummy액티비티	담당자	상 태	시작 시간	완료 시간
1	승인요청		김강용	COMPLETE	2003/01/07 10:45:38	2003/01/07 10:46:12
2		AND-SPLIT			2003/01/07 10:46:12	2003/01/07 10:46:12
3	관련데이타분석		관리자	RUNNING	2003/01/07 10:46:12	
4	문법체크		황태일	COMPLETE	2003/01/07 10:46:12	2003/01/07 10:50:12
5		AND-JOIN				
6	승인		한관희	NOT READY		
7		XOR-SPLIT				
8	결과통보			NOT READY		
9	DB등록			NOT READY		

그림 16. 실행 프로세스 진행 상태 정보.

6. 결론 및 추후 연구 과제

본 연구에서는 하나의 조직 내에서 워크플로 관리 시스템을 운영할 때 기존의 워크플로 관리 시스템의 중앙집중식 엔진 구조와 최근의 분산형 엔진 구조가 가지는 한계점을 분석하고 이를 개선하기 위해 효율적인 경량 컴포넌트 구조의 엔진 구조를 제안하고 이를 시스템으로 구현하였다. 또한 각 워크플로 관리 시스템마다 각자 고유한 프로세스 정의 형식을 갖고 있는 현행 방식의 문제점을 지적하고 이를 해결하기 위해 WfMC에서 제시한 표준 XPDL 교환 형식을 해석하여 프로세스를 실행시킬 수 있는 워크플로 관리 시스템의 기능적 요구 사항을 도출하고 바람직한 시스템의 구조를 제시하였으며 이를 바탕으로 시스템으로 개발하고 사례 연구를 통해 성공적으로 운영됨을 보였다. 이렇게 함으로써 제3의 프로세스 모델링 도구가 모델링 결과를 XPDL 파일로 산출할 수 있다면 본 연구에서 개발한 시스템은 프로세스 재입력 노력 없이 곧바로 프로세스를 실행할 수 있다. 개발된 시스템은 크게 XPDL 등록기, XPDL 해석기, 텍스트 기반 프로세스 뷰어, 워크플로 엔진, 업무 목록 처리기 및 상태 모니터링 도구로 이루어져 있다.

그러나 본 연구에서 개발된 시스템은 아직 프로토타입 수준이어서 실제적으로 기업에서 사용하기 위해서는 보완해야 할 점이 많이 있다. 첫째, 엔진 구조 측면에서 현재는 각 프로세스 관리자가 단독으로 돌아가고 있지만 프로세스 관리자간의 협동 작업을 통한 프로세스 실행에 관한 연구가 보완되어야 한다. 둘째, XPDL 관리 부분에서 현재는 사전 정의된 XPDL 파일을 해석하여 사용하는 기능만 구현되어 있는데 실행 단계에서 수정된 프로세스 정의 데이터를 다시 XPDL 파일로 갱신하는 기능에 대한 추가 연구가 필요하다. 마지막으로 현재 개발된 시스템에서 프로세스 모니터링 기능은 단순히 현 상태를 보여주는 기능에 머물고 있는데, 특정 업무에 대한 조정이 필요할 때 신속히 조정 행위를 할 수 있는 프로세스 제어 기능 보완이 시급하게 요구된다.

참고문헌

김상배, 배송용, 김광훈, 백수기(2000), 실시간 협업 지원 그룹 워크플로우 모델링 도구, *2000년 한국정보처리학회 추계 학술발표 논문집*, 7(2), 125-128.

신동일, 신동규(2000), 워크플로우 관리 시스템의 설계 및 구현, 한국*정보처리학회 논문지*, 7(5), 1609-1619.

이창수, 최혁승, 김한중, 김정수, 김선호, 조학래(2001), WfMC 표준 기반의 Web-Based Process Designer 개발, *제9회 첨단 생산시스템 Workshop 논문집*, 한국 생산 기술원.

한관희(2002), 원격 분산 환경에서의 소프트웨어 개발을 위한 통합 정보 객체 관리, *한국정보처리학회 논문지*, **9-D**(3), 427-434.

한관희, 박찬우(2002), 제품 정보 관리 시스템 개발을 위한 기능 분석에 관한 연구, *한국CAD/CAM 학회 논문집*, 7(1), 42-56.

한관희, 황태일(2003), UML/XML 기반의 비즈니스 프로세스 정의 도구, *산업공학지*, 16(2), 156-166.

Arkin, A.(2002), Business Process Modeling Language Version 1.0, http://www.bpmi.org.

Alonso G., Hagen C., Schek, H.-J. and Tresch, M.(1988), Towards a Platform for Distributed Application Development, Workflow Management Systems and Interoperability 164, pp.195-221.

Das, S., Kochut, K., Miller, J., Sheth, A. and Worah, D.(1997), ORBWork: A Reliable Distributed CORBA-based Workflow Enactment System for METTOR$_2$, Technical Report UGACS-TR-97-001, Department of Computer Science, University of Georgia.

Ellis, C. A.(1979), Information Control Nets: A Mathematical Model of Office Information Flow, *Proceedings of the Conference on Simulation, Measurement and Modeling of Computer Systems*, 225-239, ACM Press, New York.

Hammer, M. and Champy, J.(1993), *Reengineering the Corporation: a Manifesto for Business Revolution*, Harper Business, New York.

Hollingsworth, D.(1995), The Workflow Reference Model Version 1.1, Document Number TC00-1003, Hamsphire, UK.

Lee, J., Grunninger, M, Jin, Y., Malone, T., Tate, A., Yost, G.(1998), The PIF Process Interchange Format and Framework Version 1.2, *The Knowledge Engineering Review*, 13(1), 91-120, Cambridge University Press.

Manolescu, D. A.(2002), An Extensible Workflow Architecture with Object and Patterns, *TOOLS Eastern Europe 2001*, Sofia, Bulgaria.

Miller, J. A., Sheth, A. P., Kochut, K. J. and Wang, X.(1996), CORBA-Based Run-Time Architectures for Workflow Management System, *Journal of Database Management*, 7(1), 16-27.

Muth P., Wodtke, D., Weissenfels, J., Dittrich, A. K. and Weikum, G.(1998a), From Centralized Workflow Specification to Distributed Workflow Execution, *Journal of Intelligent Information Systems*, 10(2), 159-184.

Muth, P., Weissenfels, J., Gillmann M. and Weikum, G.(1998b), Mentor-lite : Integrating Light-Weight Workflow Management Systems within Business Environments, *1st European Workshop on Workflow and Process Management(WPM)*, Zurich, Switzerland.

OMG(2000), Workflow Management Facility Specification V1.2, Object Management Group, http://www.omg.org.

Schlenoff, C., Gruninger M., Tissot, F., Valois, J., Lubell, J., Lee, J.(2000), The Process Specification Language(PSL): Overview and Version 1.0 Specification, NISTIR 6459, National Institute of Standards and Technology, Gaithersburg, MD.

Silva Filho, R. S., Wainer, J., Madeira, E. R. M. and Ellis, C.(1999), CORBA-based Architecture for Large Scale Workflow, *4th International Symposium on Autonomous Decentralized Systems*, Tokyo, Japan.

WfMC(1999), Workflow Management Coalition Interface 1: Process Definition Interchange Process Model Version 1.1, Document Number WfMC-TC-1016-P, Hamsphire, UK.

WfMC(2002), Workflow Process Definition Interface - XML Process Definition Language Version 1.0, Document Number WfMC-TC-1025, Hamsphire, UK..

Wheater S. M., Shrivastava S. K. and Ranno S.(1998), A CORBA Compliant Transactional Workflow System for Internet Applications, *IFIP International Conference on Open Distributed Processing(Middleware '98)*, The Lake District, England.

zur Muehien, M. and Becker, J.(1999), Workflow Process Definition Language Development and Directions of a Meta-Language for Workflow Processes, *Proceedings of the 1st KnowTech Forum*, Potsdam, Germany.

한 관 희
아주대학교 산업공학과 학사
한국과학기술원 산업공학과 석사
한국과학기술원 자동화 및 설계공학과 박사
대우전자, 대우정보시스템 근무
현재: 경상대학교 산업시스템공학부 교수
관심분야: 워크플로, EAI, PLM/CPC, EA

김 강 용
경상대학교 산업시스템공학과 학사
경상대학교 산업시스템공학과 석사
현재: 이놉스(주) 연구원
관심분야: PDM/PLM, 워크플로