자원지향 워크플로우 통합 프레임워크

김 현 아* 김 광 훈 백 수 기

◈ 목 차 ◈

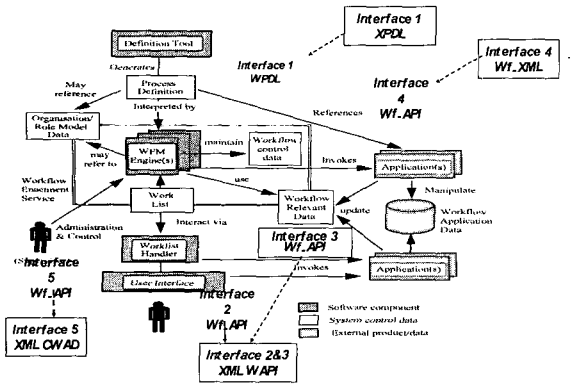
1. 서 론

최근의 기업 환경은 네트워크와 컴퓨팅 환경의 비약적인 발달로 많은 변화를 초래 하였다. 특히, 네트워크의 발달과 함께 폭발적인 대중화가 이루어진 인터넷은 초기에 기업 내의 분산 시스템을 지원하는데 큰 효과를 거두었고, 오늘날 서로 다른 기업간의 상호 작용을 가능하게 하고 있다. 그러나, 많은 기업들은 자신에게 적합한 정보 시스템을 채택함으로써 거시적인 관점에서의 기업 환경을 이기종 시스템들이 존재하는 비효율적인 방향으로 방치될 수 밖에 없는 상황적 결과를 낳았다. 그럼에도 불구 하고 기업간의 경쟁이 치열해 짐에 따라 보다 나은 효율적 가치의 생산 결과를 도출하기 위해 많은 대단위 정보 시스템이 도입 되었다. 워크플로우 관리 시스템, 비즈니스프로세스 관리 시스템, 기업자원계획 시스템, 고객관계관리 시스템, 기업응용시스템통합 시스템 등이 대표적이며 이들 간의 통합 문제가 대두되고 이다.

특히, 최근에 기업간 전자거래, 전자무역, 전자물류, 공급망 관리 등과 같은 기업간의 전자적인 거래 및 시장의 활성화와 더불어 워크플로우 및 비즈니스 프로세스 관리 기술이 기업정보화의 핵심 솔루션으로 인식되기 시작하면서 기업 내에 존재하는 기존의 정보화 시스템들을 워크플로우 및 비즈니스 프로세스 중심으로 통합시키는 것의 긴급히 해결해야 하는 문제로 나타나고 있다. 기존의 기업정보화 솔루션 통합 문제는 동일 컴퓨팅 환경의 이기종 솔루션들간의 상호연동에 초점을 두고 있는 반면에, 최근에 대두되는 기업정보화 솔루션 통합 문제는 동일 컴퓨팅 환경의 이기종 솔루션들간의 상호연동 뿐 만 아니라 다른 컴퓨팅 환경(네트워크, 운영체계, 프로그래밍언어 등)의 이기종 솔루션들과 기업 내에 운용되고 있는 기존의 다른 솔루션 및 응용 프로그램들간의 통합 문제로 통합의 초점이 점점 더 복잡해 지고 있다. 또한, 최근에 대두되는 기업정보화 솔루션의 통합문제의 또 다른 주요 특성중의 하나는 기존의 통합이 브로커 중심의 일대일 통합 개념을 기반으로 이루어지는 반면에 워크플로우 또는 프로세스 중심의 일대다 통합 개념으로 요구되고 있다는 점이다.

워크플로우 중심의 일대다 통합 개념의 구현을 가능하게 하는 핵심기술은 XML 기술을 기반으로 한다. 즉, XML 기술은 자원지향(Resource-Oriented) 기업정보화 통합 프레임워크를 구현 가능하게 한다. 이와 대조적으로 지금까지의 기업정보화 통합문제는 표준 명령어인터페이스(Application Program Interface)를 기반으로 하는 이기종 솔루션간의 명령어지향(Operation-Oriented) 기업정보화 통합 프레임워크의 구축을 통해 그의 해결방안을 모

* 경기대학교 대학원 전자계산학과 박사



(그림 1) 워크플로우 통합 프레임워크 Paradigm Shift

색해왔다. 하지만, XML 기술을 기반으로 하는 웹 서비스와 응용프로그램통합 솔루션의 등장은 기업 정보화 솔루션 통합문제에 대한 새로운 해결방안을 제시하고 있다.

위의 그림 1은 워크플로우 통합 프레임워크가 변화되는 패러다임의 모습이다. 즉, 기존의 명령어 지향 기업정보화 통합 프레임워크는 동일 컴퓨팅 환경, 즉 동일 네트워크, 동일 운영체계, 동일 프로그래밍언어를 기반으로 하는 이기종 솔루션들간의 상호연동을 지원하지만, 자원지향 기업정보화 통합 프레임워크는 이기종 컴퓨팅 환경(이기종 네트워크, 이기종 운영체계, 이기종 프로그래밍언어)을 기반으로 하는 이기종 솔루션들간의 상호연동을 지원함으로써 통합의 유연성, 확장성 그리고 가용성을 크게 개선시킬 수 있다. 따라서 본 논문은 기존의 명령어지향 기업정보화 통합 프레임워크를 기반으로 하는 워크플로우 관리 시스템 구성요소들간의 통합에 대한 문제점이 핵심과제로 대두됨에 따라 이에 대한 해결책으로 현재의 기술 변화 추세를 반영한 새로운 기업정보화 통합 프레임워크의 필요성을 주요 연구 목표로 하고 있다.

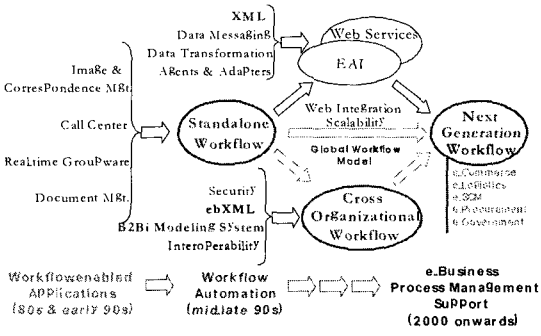
2. 관련연구

2.1 워크플로우 기술 및 표준

워크플로우 관리 시스템은 기업의 비즈니스 프로세스를 전자화하고 이를 자동으로 실행해주는 소프트웨어이다. 사무자동화의 실현이라는 개념으로부터 출발하여 IT로서 워크플로우 개념이 처음으로 등장한 것은 1980년대 초였으며, 1980년대 말과 1990년대 초에 이르러서는 관련 제품이 등장하기 시작하였다. 1990년대 초의 개념 정립기를 거쳐서 본격적인 시장 형성이 이루어진 것은 1990년대 중반이라고 할 수 있다.

오늘날 대부분의 정보기술은 데이터베이스의 기술을 기반으로 하고 있지만 데이터베이스 기술은 정보화의 해당 도메인의 데이터와 그 데이터를 중심으로 한 업무처리 프로그램 중심에 초점을 두고 있다. 그러나 조직 내, 업무처리의 생산성을 분석한 결과 업무처리의 시간보다 업무전이에 소요되는 시간의 비중이 막대하게 많이 차지 한다는 것을 인식하고 업무처리 프로세스에 대한 생산성 향상 문제를 정보기술의 핵심으로 그 초점이 바뀌게 되었다. 2000년대에는 프로세스 중심의 정보기술이 핵심으로 등장 하게 되면서 워크플로우 기술이 두각 되었고 프로세스 중심의 기술의 중심에 서게 되었음을 보여 준다. 워크플로우의 출발은 정보기술이 아니라 프로세스라고 하는 것이 올바른 해석으로 알려져 있으며, 정보공학 방법론이나 BPR(Business Process Reengineering)을 굳이 예로 들지 않더라도 모든 업무 분석의 첫 단계에 등장하는 것이 바로 업무 절차 분석으로 일컬어지는 프로세스 모델링이다.

그림 2는 워크플로우 기술의 발전 단계를 보여준



(그림 2) 워크플로우 기술 변화

다. 80년대 중반의 워크플로우 기술은 'Workflow-enabled Application'의 개념을 가지고 그룹웨어나 문서 관리 시스템 분야에 활용되었고 90년대에는 'Workflow Automation'의 개념으로 단일 워크플로우 시스템으로서 기업내의 비즈니스 프로세스 자동화를 위한 기술로 활용되었다. 2000년대 이후부터 워크플로우 기술은 'e-Business Process Support'의 개념을 기반으로 기업간의 전자적 협업을 지원하고 보다 원활하게 정보 교환을 할 수 있는 시스템으로써 EAI, BPR, ERP 등과 같은 최첨단 정보기술의 핵심 기반기술로서 인식되고 있다. 특히, 요즘 기업 환경은 업무의 흐름이 복잡하여 프로세스의 표준화 또는 정형화가 어렵고, 부문간 협업이 강조되어 워크플로우 기술은 비즈니스 프로세스를 통합 및 자동화한다는 점에서 기업의 핵심 기술이 될 것이다.

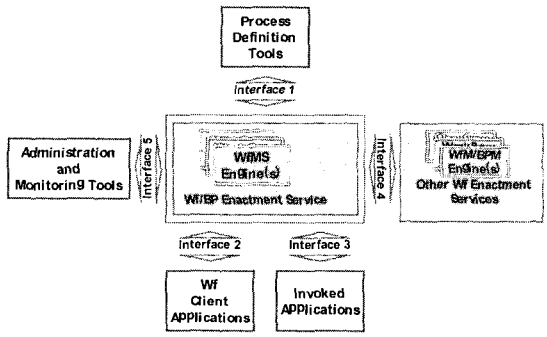
워크플로우 기술 표준화는 1993년 8월 비영리 기구 성격으로 설립된 WfMC(Workflow Management Coalition)를 중심으로 이루어지고 있고, 9개의 워킹 그룹으로 구성되어 있다. WfMC의 임무는 소프트웨어 용어 및 워크플로우 제품들간의 상호운용성과 연결성에 대한 표준을 제정함으로써 워크플로우 사용을 증진하고 워크플로우를 사용하고자 하는 사용자들에게 보다 많은 신뢰성을 제공하는 데 있다.

WfMC는 프로세스 정의 데이터와 이의 교환을 위한 표준 규격 개발, 이기종의 워크플로우 시스템간 상호운용성 지원, 다양한 IT 응용과의 상호 작용 지원, 사용자 인터페이스에 대한 표준화를 다루고 있으며, 현재 대부분의 워크플로우 개발 업체들과 사용자, 분석가들도 함께 참여하고 있다.

WfMC의 표준화 노력들은 추상적인 인터페이스 개념을 정리하고 메타 모델을 구체화하는 방향으로 진행되는 반면에 OMG(Object Management Group)에서 객체 지향 워크플로우 표준을 정의하여 워크플로우 모델 'Jflow'를 통해 실제 구현방안을 지원하고 있다. AIIM(The Association for Information and Image Management)은 비즈니스 프로세스를 지원하기 위한 기업 컨텐츠의 관리, 운송, 생성, 고객화, 저장 능력이 디지털 시대에서 효과적인 비즈니스 기반의 핵심이라는 측면에서 이를 확보하기 위해 필요한 기술들을 표준화하는데, 워크플로우 관리 시스템상의 문서 흐름에 대한 표준을 제정하고 있다. BPMI(The Business Process Management Initiative)는 모든 산업에 걸쳐 모든 규모의 기업들에게 인터넷과 방화벽을 통해 다양한 어플리케이션과 비즈니스 파트너들 간의 비즈니스 프로세스를 수행하고 개발하도록 하는 비영리 기구로서 비즈니스 프로세스 관리에 대한 표준을 제정하고 있다. 워크플로우 시스템의 구조난립을 막고, 서로 다른 워크플로우 제품들을 표준화하기 위해서 WfMC에서는 워크플로우 참조모델(Workflow Reference Model)을 제안하였다. 대부분의 상용 시스템들은 워크플로우 참조 모델을 기반으로 만들어지며 워크플로우 참조 모델에서 제안하는 표준 인터페이스를 준수하고 있다.

워크플로우 참조 모델은 다섯 개의 컴포넌트와 엔진 사이의 인터페이스를 명세하고 있다. 먼저 다섯 개의 컴포넌트는 그림 3과 같다.

- 워크플로우 엔진: 워크플로우 컴포넌트 가운데 핵심모듈로 중앙 서버에 위치하며 각 워크플로우 관리 시스템은 하나 이상의 엔진을 가진다. 엔진은 정의된 프로세스를 해석하여 시작시키며, 시작된 프로세스가 올바로 수행이 될 수 있도록 제어 관리한다.
- 프로세스 정의 도구: 프로세스 정의도구는 프로세스 모델을 설계할 수 있는 그래픽 사용자



(그림 3) WfMC의 워크플로우 참조 모델

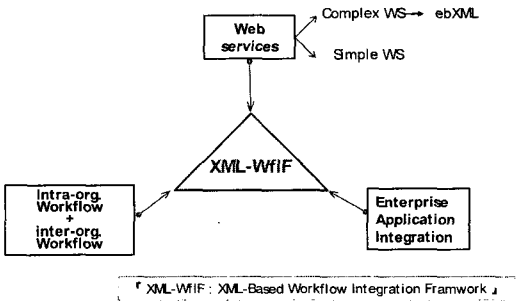
환경(GUI: Graphic User Interface)을 제공하고 이를 컴퓨터가 해석 가능한 형태로 만들어 저장한다. 제품에 따라서는 설계된 프로세스를 검증하는 기능과 다양한 프로세스 정의 언어로 저장해주는 기능도 포함한다.

- 워크플로우 클라이언트: 프로세스를 이루고 있는 단위업무를 사용자가 수행할 수 있는 사용자 인터페이스를 제공해 준다. 사용자는 클라이언트가 제공하는 사용자 인터페이스를 통해 자신에게 할당된 업무를 확인하고 이를 완료, 실행중지/재개, 반려 등을 수행하며, 할당된 자원에 대하여 어플리케이션을 호출하여 가공하기도 한다.
- 피호출 응용프로그램: 단위업무의 수행을 위해 엔진에 의하여 호출, 실행되는 프로그램 모듈이며, 대개 입력을 받아 출력을 반환해주는 형태이다.
- 관리 및 모니터링 도구: 프로세스를 관리, 감독할 수 있는 사용자 환경을 제공한다. 제품에 따라서는 프로세스와 관련된 정보를 통계적 수치로 사용자에게 제공하여 프로세스에 대한 전반적인 관리가 용이하도록 돕는다.

워크플로우 참조 모델은 이들 컴포넌트들에 대하여 워크플로우 시스템이 제공하는 서비스를 5개의 기능적인 인터페이스로 정의하여 표준적으로 정의하였다. 최근에는 WfMC에서 인터페이스 2와 3을 통합하여 실제로는 네 가지 표준 문서들을 제공한다.

2.2 워크플로우 통합 프레임워크를 위한 기술

기업 내 및 기업 간 워크플로우 통합이 비즈니스 협업을 증진시키기 위하여 어플리케이션과 자원 및 인적자원을 유기적으로 결합하여 협업 프로세스를 신속하고 유연하게 수행할 수 있도록 지원하기 위한 방법으로 XML 기술과 웹 서비스 기술을 기반으로 한다. 아래 그림 3은 XML 기반 워크플로우 통합 프레임워크를 지원하기 위한 기술들이다.



(그림 4) XML기반 워크플로우 통합 지원 기술

그림 4에서 어플리케이션 통합을 위한 기술로 EAI를 도입함으로써 기업의 복잡하고 다양한 어플리케이션과 데이터를 통합하는 것을 주요 기능으로 삼아 이기종이며, 독립적으로 개발된 어플리케이션 간의 통합이 이루어 질 수 있다. 이들 어플리케이션의 상호 운용이 불가능한 형태로 개발되어 있으며 EAI의 구현을 통해서 이러한 시스템들간의 정보 교환을 가능하게 할 수 있다. 어플리케이션 시스템들간의 정보교환은 메시지의 교류형태가 되며 메시지 기반의 제품들에서 메시지를 통한 정보 교환은 곧 워크플로우가 된다. 대부분의 개별 비즈니스 프로세스들은 이미 충분한 수준에서 자동화가 이루어져있다고 볼 수 있으나 이들은 서로 연결이 없는 상태이다. 따라서 이들간의 데이터 교환과 작업의 연결은 수동에 의존한다고 볼 수 있다. 이러한 데이터의 교환과 정보의 흐름을 자동화 해줄 수 있는 소프트웨어 모듈이 곧 워크플로우 이며 대부분의 EAI는 워크플로우 모듈을 핵심 모듈로 채택하고 있다.

정보의 흐름은 곧 자동화된 데이터의 이동을 의미하는데 비즈니스 프로세스는 항상 정보의 흐름을 포함한다. EAI의 목표도 곧 이러한 정보 흐름을 자동화하는 것이며 결국은 이를 통해 프로세스 자동화를 완벽히 하는 것이다. 사용자의 입장에서는 그들이 원하는 정보를 워크플로우를 통해서 자신의 작업에 필요한 형태로 자동으로 제공받을 수 있음을 의미한다. 결론적으로 EAI의 모듈로서 워크플로우는 어플리케이션 상호간에 데이터 전달 및 어플리케이션 연동을 미리 정의한 업무 프로세스에 따라 처리

할 수 있도록 해주는 없어서는 안될 핵심이다.

또한 비즈니스 협업을 위한 웹 서비스의 접목도 매우 중요한 기술 중에 하나이다. 웹 서비스는 다양한 의미에서 정의할 수 있다. 먼저 웹 서비스 관련 표준 기구들이 내리고 있는 웹 서비스 정의에 대해 알아보면, W3C의 Web Service Architecture Group에서는 다음과 같이 정의하고 있다.

웹 서비스는 URI에 의해 식별되는 소프트웨어 어플리케이션으로, 그 인터페이스와 바인딩은 XML에 의해 정의되고, 기술되며, 검색될 수 있다. 또한, XML 메시지를 사용하는 서로 다른 소프트웨어 어플리케이션들이 인터넷 기반의 프로토콜을 통해 직접 상호작용 할 수 있도록 지원한다.

WS-I(Web Services Interoperability Organization)에서는 좀더 일반적인 관점에서 웹 서비스를 정의하고 있다. WS-I는 웹 서비스를 일반적인 목적의 구조를 가진 것으로 간주한다. 즉, B2B 시나리오를 지원하지만, 범위나 기능 면에서는 더 확장된 상호운용성을 지닌 구조이다. 이 정의는 잠재적으로 문서 중심적이고, 느슨하게 결합된 비즈니스 협업의 의미를 내포하고 있다.

비즈니스 프로세스는 점점 더 협업에 의존하고 있다. 기업은 비즈니스의 생산성을 향상시키기 위하여 협업 프로세스를 자동화해야 한다. 정부나 교육 기관, 비영리 조직들도 웹 기반의 협업을 추진하고 있다. 웹 서비스의 최종적인 목표는 협업을 수월하게 하는 것이다. 웹 서비스가 협업에서 참여자 역할을 충분히 수행하기 위하여, 그리고 적절한 수준의 상호운용성을 보장하기 위하여 여러 계층을 살펴볼 필요가 있다. 그 중에서 프로세스나 협업 모델링 언어가 역할, 책임, 계약, 성과물, 상태 관리 및 상태 전이에 대한 충분한 정의를 지원하는데 반하여, WSCI의 목적은 주어진 웹 서비스가 여러 역할을 포함하는 복잡한 시나리오를 나타내는 인터페이스를 제공하는 데 있다. 즉, 협업 컨텍스트에 사용될 때, WSCI의 목적은 협업에서 하나 이상의 역할을 담당하는 특정 웹 서비스들에 관련된 메시지 교환 구성법을 명확하게 제시하는 것이다.

이러한 비즈니스 협업의 자동화를 이루기 위하여 웹 서비스와 워크플로우가 유기적으로 기술적 결합을 하면 협업 프로세스를 정의하는 것은 워크플로우를 정의하는 다른 표준들이 그 역할을 맡아준다는 것이다. 그리고 WSCI는 워크플로우에서 상호작용 하는 웹 서비스의 외부적인 행동들만 기술하겠다는 것이다. 즉, 워크플로우를 구현하는 시스템들의 가시적인 행동만을 정의한다. 또한, WSCI는 웹 서비스가 수행하는 액티비티 내에서 이루어지는 액티비티를 통제하기 위한 행동들을 기술하지는 않는다. 단지 웹 서비스 그 자체의 가시적인 행동을 정의한다. 이처럼 WSCI는 다양한 참여자가 상호작용하는 비즈니스 모델에서, 참여자들의 경계에 필요한 명세를 정의하는 데 유용하게 사용될 수 있다.

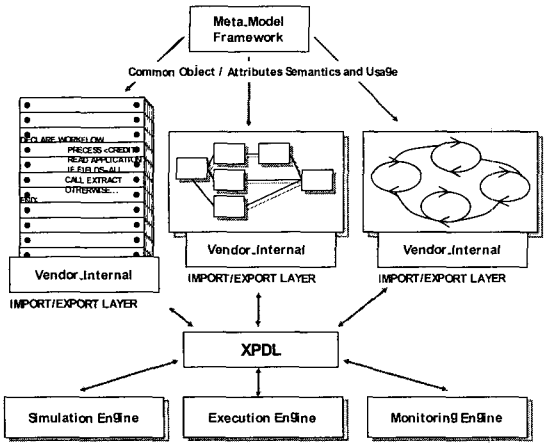
3. 자원 지향 워크플로우 통합 모델

XML기반의 자원지향 워크플로우 통합 프레임워크를 구현해서, 워크플로우 관리 시스템의 재 사용성과 효율적이고 강력한 통합 기능을 가진 표준화된 워크플로우 관리시스템을 제공하려 한다. 또한 XML기반으로 한 API가 워크플로우 엔진에 독립성이 보장되고, 웹 서비스 기술과 연계하여 시간, 장소에 제약 없이 각각의 워크플로우 클라이언트가 이기종의 엔진 상에서 유기적인 접근을 이룰 수 있도록 한다. 다음에서는 XML기반의 각각의 인터페이스 기술을 설명한다.

3.1 XML 기반 프로세스 정의 언어

XPDL(XML Process Definition Language)은 프로세스 정의 교환을 위한 워크플로우 프로세스 정의 언어이다. WfMC에서는 기존의 워크플로우 참조 모델에 근거하여 워크플로우에 관한 여러 가지 표준들을 제정하고 있는데, XPDL은 참조 모델의 인터페이스 1에 해당되는 프로세스 정의 모델을 XML로 표현한 것이다.

XPDL은 워크플로우에서 사용되는 다양한 프로세스 정의 요소를 포함한 프로세스 메타 모델을 제시하고 있다. 그리고 그림 5와 같이 워크플로우 제품들이 내부적으로 사용하는 프로세스 정의를 가져오기/내보내기(import/export) 할 수 있도록 공통의



(그림 5) 프로세스 정의 교환의 개념

XML 교환 포맷을 제공하고 있다. 또한, 다양한 비즈니스 시나리오에 따라서 프로세스 정의 데이터들이 교환될 때, 데이터 객체나 관계, 속성 집합을 일관성 있게 전달할 수 있도록 지원한다.

XPDL은 일종의 프로세스 정의 인터페이스이며, 워크플로우 시스템에게 다음과 같은 두 가지 의 이점을 제공한다. 워크플로우 시스템들에게 공통의 프로세스 정의 교환 포맷을 제공한다. 개발(development) 환경과 실행(run-time) 환경을 분리함으로써, 개발 도구와 실행 도구의 독립적인 개발과 호환을 지원한다.

XPDL을 구성하는 중심 요소인 패키지, 워크플로우 프로세스, 프로세스 액티비티, 전이 정보, 어플리케이션, 워크플로우 관련 데이터, 참여자 등이 있는데, 이들이 갖는 정보는 표 2와 같다.

3.2 워크플로우 런타임 클라이언트 인터페이스

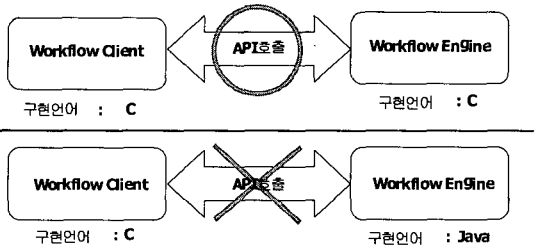
인터페이스 2를 사용하는 워크플로우 클라이언트 응용 프로그램은 사용자에게 워크플로우 수행 서비스에 대한 사용자 인터페이스를 제공한다. 그 중, 워크리스트 제어기를 통하여 사용자가 작업을 선택하고, 처리해야 하는 작업에 대한 상세 정보를 추출하고, 작업의 처리를 위해 필요한 관련 응용 도구를 호출할 수 있도록 한다. 또한 인터페이스 3은 응용 프로그램 호출은 워크플로우와 연동해야

(표 1) XPDL 구성요소 개요

Workflow Model Definition	Workflow Process Definition	Workflow Process Activity	Transition Information	Workflow Application Declaration	Workflow Relevant Data	Workflow Participant Definition
- Id - Name - Description	- Id - Name - Description	- Id - Name - Description	- Id - Name - Description	- Id - Name - Description	- Id - Name - Description	- Id - Name - Description
- WPDL-Version - Source Vendor ID - Creation Date - Version - Author - Characterset - Codepage - Country Key - Publication Status - Extended Library - Conformance Class	- Creation Date - Version - Author - Characterset - Codepage - Country Key - Publication Status - Valid From Date - Valid To Date - Classification - Extended Library	- Characteristic - Automation Mode - Split - Join - Loop - Inline Block		- Tool Name	- Type	- Type - type related information
	- Parameters		*- Condition*	*- Parameters*	*- Value*	
- Responsible *- External Model Ref* *- Access Restriction*	*- Responsible* *- Access Restriction*	*- Participant assignment* *- Implementation* *- Application assignment* *- Access Restriction*	*- From* *- To*			
- Documentation - Icon	- Documentation - Icon	- Documentation - Icon				
- Priority Unit - Cost Unit -	- Priority -Cost -Duration Unit - Duration - - Waiting Time - Working Time	- Priority - Instantiation -Cost - Duration - Waiting Time - Working Time				- Strategy - Capacity-Cost - Prepare Time
...	...	...	...	...	...	...

하는 서로 다른 환경에서 제공중인 다양한 기존 서비스로 구성되어 있다. 인터페이스 3은 자동 수행 액티비티와 워크플로우 수행 서비스에 직접 연동할 수 있는 워크플로우 사용 가능 응용 프로그램의 호출을 위한 인터페이스이다. 그리고 인터페이스 5는 모든 워크플로우 수행 서비스를 모니터링 하고 관리 할 수 있는 인터페이스를 제공하기 위함이다. 워크플로우 시스템은 현재 진행 중인 워크플로우 인스턴스의 원활한 진행을 위하여 시스템 및 현재의 인스턴스 상태를 모니터링하고 그 정보를 바탕으로 인스턴스를 관리 감독할 수 있는 인터페이스를 제공한다.

워크플로우 런타임 클라이언트가 워크플로우 엔진과 상호작용하는 방법의 정의는 이기종의 사용자 환경을 지원하는데 많은 어려움을 지니고 있다. 이는 이기종의 사용자 환경을 개발 시 엔진에서 제공하는 API에 의해 개발되어야만 한다는 제한이 있다. 이러한 형태는 기존의 다른 워크플로우 제품들을 통해서 확인 할 수 있다. 이렇게 워크플로우 엔진에서 제공하는 API를 통해서 사용자 환경을 개발 한다는 것은 워크플로우 엔진과 상호작용하게



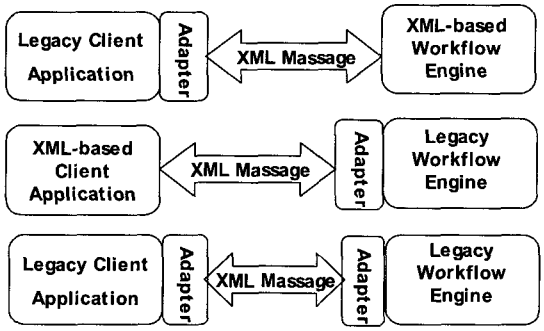
(그림 6) 기존의 API접근 방법

하는데 있어 많은 불편을 초래할 뿐만 아니라, 이기종의 사용자 환경 개발 시 제한적인 형태의 개발을 할 수 밖에 없다.

위의 그림 6에서 보면 각각의 사용자 환경을 위해서는 엔진에서 제공하는 API를 사용하여 사용자 환경을 개발하여야만 했다. 이런 경우 만약 워크플로우 엔진에서 제공하는 API가 C언어나 다른 개발언어로 된API를 제공한다면 사용자 환경 개발은 엔진에서 제공하는 API를 사용할 수 있는 언어로 개발하여야만 한다.

그러나 현재 IT 관련 산업은 정보 시스템이 기업 경쟁력에 미치는 영향력이 증대되고 B2B 전자상거래가 일반화되었고, IT 환경의 변화 다양한 이기종 접속환경이 등장함에 따라 기업의 변화된 환경에서는 이러한 이기종 접속환경에서 사용자가 워크플로우를 사용하게 하려는 요구가 증가하고 있는 실정이다. 이러한 이유로 기존의 API호출 방식에서 벗어나 XML메시지 형태의 API를 호출하는 방식으로 바뀌어야 한다. 그러기 위해선 워크플로우 통합 프레임워크에서 인터페이스 2와 3에 대한 통합 플랫폼은 사용자 환경 통합 기능을 제공하며 다양한 사용자 환경의 지원을 위한 어댑터 기능을 제공하여야 한다. 워크플로우 엔진의 API에 의존적이지 않으면서 이기종의 워크플로우 런타임 클라이언트 사용자 환경을 통합할 수 있는 매개체로써 XML을 고려할 수 있다.

워크플로우 런타임 클라이언트를 XML기반 하에 통합하고 엔진과 상호 작용 할 수 있도록 하기 위해선 워크플로우 런타임 클라이언트에 대한 적응, 변환, 그리고 마지막으로 통합 기능이 제공 되어야 하고 이를 각각 설명 한다.



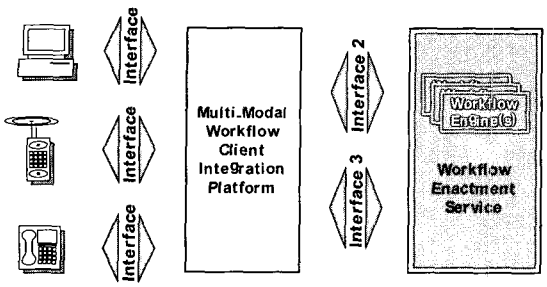
(그림 7) 어댑터를 이용한 변환

적응 기능은 런타임 클라이언트에 대한 적응 기능이다. 새로운 사용자 환경의 런타임 클라이언트에 대한 지원 시 어댑터를 이용하여 이를 지원할 수 있게 하는 기능이다.

변환 기능은 새로운 환경의 런타임 클라이언트에 대해 어댑터를 통해서 변환되는 기능이다. 어댑터를 이용함으로써 서로 상호작용이 가능하도록 변환해 줄 수 있다. 다음 그림 7은 기존 클라이언트 응용과 XML기반 워크플로우 엔진, XML기반 클라이언트 응용과 기존 워크플로우 엔진이 어댑터를 이용하여 XML Message로 Translation되어 상호작용하는 것을 보여주고 있는 그림이다.

통합 기능은 다양한 이기종의 워크플로우 런타임 클라이언트 사용자 환경을 통합하는 기능이다. 이러한 통합은 기존의 존재하는 시스템에도 적용할 수 있는 큰 장점이 있다. 그리고 이러한 통합 기능은 후에 워크플로우 시스템이 시스템 통합에 용이한 기초가 될 수 있다. 또한 이러한 통합은 새로이 개발하는 워크플로우 런타임 클라이언트 사용자 환경은 기본이고 기존에 존재하는 워크플로우 런타임 클라이언트에 적용하는데 용이하다. 다음 그림 8은 다양한 이기종의 워크플로우 런타임 클라이언트 사용자 환경을 통합한 그림을 보여주고 있다.

위에서 살펴본 바와 같이 워크플로우 런타임 클라이언트를 세가지 기능적 역할로 통합 하고 XML기반으로 적용 시키기 위해선 XML WAPI가 사용되게 된다. XML WAPI는 시스템 독립적이고 상호운용 가능한 특징을 가지고 있는XML을 기반으로 하는 XML 메시지 형태의 API이다.



(그림 8) 다양한 이기종 런타임 클라이언트 통합

3.3 워크플로우 상호 운용성

Wf-XML의 목적은 서로 다른 워크플로우 시스템간의 상호운용성을 구현하는데 사용되는 언어를 기술하는 것이다. 특히, WfMC의 상호운용성 추상 명세서에서 제시하고 있는 세 가지 모델, 단순 연결 워크플로우와 중첩 워크플로우, 병렬 동기화 워크플로우를 주요 지원 대상으로 삼고 있다. 이러한 세 가지 모델에서 동기적이거나 비 동기적인 상호 전달을 모두 지원하고, 메시지 교환을 개별적으로, 또는 배치로 수행하는 것을 모두 허락한다. 나아가 프로그램 언어, 데이터 전송 체계, 플랫폼, 하드웨어 등과 같은 특정 구현에 독립적인 언어를 제공함으로써 실제 구현 업체들의 기술 활용 범위를 최대한 보장하고 있다. Wf-XML은 표준의 이해를 돕기 위하여, HTTP 프로토콜을 이용하여 Wf-XML 메시지들이 상호 교환되는 방식을 서술함으로써 구체적인 적용을 고려하여 사용자를 지원하고 있다.

워크플로우의 상호운용을 위해서는 여러 종류의 메시지를 교환하게 된다. Wf-XML에서는 크게 두 가지 모델을 중심으로 메시지를 명세하고 있다. 하나는, 논리적 자원 모델로서 메시지를 주고받는 주체("그룹")에 따라서 메시지 오퍼레이션들을 구분하고 있다. 그리고 논리적 상호작동 모델은 메시지 전송에 관련된 동기화와 비 동기화, 그리고 배치 전송에 관해 설명하고 있다.

논리적 자원 모델에서 메시지는 여러 오퍼레이션으로 구성된다. 오퍼레이션이란 개별적으로 상호운용 가능한 기능 단위를 말하는데, 이 오퍼레이션들은 자신들의 컨텍스트에 따라서 여러 그룹들로 나누어진다. 이들은 세 개의 기본적인 그룹(프로세스 정의, 프로세스 인스턴스, 관찰자)과 하나의 추가적인 그룹(통제)으로 구성된다. 이 그룹들의 역할과 관련된 오퍼레이션은 표 2와 같다.

(표 2) 논리적 자원 모델의 역할 및 오퍼레이션

그룹	역할	포함된 오퍼레이션
통제 Control	서비스들의 상호운용에 요구되는 프로토콜 수준의 기능을 지원	GetBatchMessageState ChangeBatchMessageState
프로세스 정의 *ProcessDefinition*	포괄적인 서비스의 가장 기본적인 기능을 표현하며 프로세스인스턴스를생성하는 자원	CreateProcessInstance
프로세스 인스턴스 ProcessInstance	주어진 프로세스 정의의 실제 구동이며, 프로세스 정의에 대하여 여러 개가 호출, 실행될 수 있음.	GetProcessInstanceData *ChangeProcessInstanceState*
관찰자 Observer *Observer*	프로세스 인스턴스의 완료나 종료를 전달하고, 결과를 전달받음.	ProcessInstanceStateChanged Notify

논리적 상호작동 모델에서 상호작동이란 프로토콜에 따르는 두 서비스간의 정보 교환을 의미한다. 그리고 Wf-XML에서는 정보 교환의 수단으로 XML 메시지를 이용하고, 상호작동의 세 가지 형태로서 "요청", "확인", "응답"이 사용된다.

Wf-XML에서 사용되는 메시지 방식은 동기적 메시지 방식과 비동기적 메시지 방식이 있다. 동기적 메시지 방식은 하나의 요청 메시지를 보내면 하나의 대칭되는 응답 메시지를 받는 방식이다. 그 응답 메시지에는 상태 변화나 예외 정보 등이 포함된다. 반면에, 비동기적 메시지 방식은 요청 메시지에 대해 확인 메시지만 발송된다. 이 확인 메시지에는 상태 처리나 예외 정보 등이 포함되지 않는다. 즉, 응답 메시지에 대한 예외나 추가적인 응답은 별도의 응답 메시지를 통해 이루어진다. 또한, 한 메시지 안에 Wf-XML 상호작동에 관한 오퍼레이션이 포함되어 교환될 수 있도록 배치 메시지 교환을 지원하고 있으며, 이는 대용량 트랜잭션 환경에서 유용하게 사용될 수 있다.

Wf-XML 메시지는 워크플로우 연동에 참여하는 시스템간의 오퍼레이션을 중심으로 전개된다. Wf-XML에서 제공하는 오퍼레이션은 그룹의 성격에 따라 크게 네 가지로 분류할 수 있는데, 통제 오퍼레이션, 프로세스 정의 오퍼레이션, 프로세스 인스

(표 3) Wf-XML 오퍼레이션 정의

오퍼레이션 / 그룹	통제	프로세스정의	프로세스인스턴스	관찰자
GetBatchMessageState	X			
ChangeBatchMessageState	X			
CreateProcessInstance		X		
GetProcessInstanceData			X	
ChangeProcessInstanceState			X	
ProcessInstanceStateChanged				X
Notify				X

턴스 오퍼레이션, 관찰자 오퍼레이션이다. 표 3은 총 7가지 오퍼레이션이 어떤 그룹에 속하는지 보여주고 있다.

3.4 워크플로우 관리/모니터링 인터페이스

워크플로우 표준 기관인 WFMC의 인터페이스 5의 어드민 / 모니터링(Admin/Monitoring)과 엔진간의API이용방식을 XML 메시지 형태로 처리하여 엔진 종속적인 문제를 해결하고, 워크플로우 연구 이슈인 XML인터페이스를 정의하여 웹 서비스와 워크플로우 기술의 통합을 통해 워크플로우 엔진에 독립적이고, 이기종 워크플로우 엔진과 호환성, 시간과 장소에 자유로운 범용시스템으로 패키지화 한다.

4. 워크플로우 런타임 클라이언트 통합 플랫폼 구현

워크플로우 런타임 클라이언트 통합 플랫폼의 실제적인 구현에 관하여 기술하고자 한다.

본 시스템의 구현 환경은 Java로 개발하였기 때문에 플랫폼뿐만 아니라 응용프로그램에 독립적이다. 프로그램 안에서 메시지의 구조는 XML형태로 구성이 되기 때문에 XML의 처리를 하기 위해서 사용된 XML 파서(Parser)는 xerces 파서를 사용했다.

OS는 Windows 2000과 Windows XP Professional을 사용했고 구현언어는 JDK 1.3.x, JDK 1.4.1이며, XML 파서는 xerces 파서를 사용 했다.

(표 4) JSP 기반의 런타임 웹 클라이언트 사용 함수

함 ▯	WMConnect
형	public String WMConnect(String usetIdentification, String password), throws Exception
기	워 크 플 엔 G 로 G 한 .
매 개	String userIdentification: 사 G 아 G String password: 사 G 패 스
리 ▯	사 G 세 정
예	사 G 세 정 G 받 아 도 G 생 G 모 예

함 ▯	WMGetProcessDefinitionsList
형	public ProcessDefinition[] WMGetProcessDefinitionsList(String sessionId) throws Exception
기	프 로 정 리 스 가 져 .
매 개	String sessionId: 사 G 세 정
리 ▯	프 로 정 클 G 배 (ProcessDefinition[])
예	프 로 정 클 G 배 G 받 아 도 G 생 G 모 예

(표 5) RMI/IIOP 기반의 e-chautauqua 워크플로우 엔진 사용 함수

함수 명	log_in
형식	public Object log_in(String userId, String password) throws Exception
기능	워크플로우 엔진에 로그인 한다.
매개변수	*String userId* : 사용자 아이디 String password : 사용자 패스워드
리턴 값	사용자 인증키 객체 (Identification)
예외	사용자 인증키 객체를 받아오는 도중에 생기는 모든 예외

함수 명	WMOpenWorkflowList
형식	public ECDefWorkflow[] WMOpenWorkflowList(Identification id) throws Exception
기능	프로세스 정의 리스트를 가져온다.
매개변수	Identification id: 사용자 인증키 객체
리턴 값	프로세스 정의 클래스 배열 (ECDefWorkflow[])
예외	프로세스 정의 클래스 배열을 받아오는 도중에 생기는 모든 예외

기타 JBuilder 9을 이용했다.

현재 구현된 어댑터를 가지고 실제 환경에 적용시켜 보면 일단 런타임 클라이언트는 JSP기반의 웹 클라이언트라고 하고 엔진은 RMI/IIOP 방식의 e-chautauqua 엔진이라고 하겠다. 각각의 사용하게 되는 함수는 표 4, 5와 같다.

이렇게 서로 다른 방식의 함수가 다중 모달 워

(표 6) 변환되는 워크플로우 엔진 로그인 관련 XML 요청/응답 메시지

요청 메시지	POST /xxxx HTTP/1.1 User-Agent: Client Content-Length: 146 Content-Type: text/xml <?xml version="1.0"?> <WMGetProcessDefinitionsList.Request> <SessionID>dreamer</SessionID> </WMGetProcessDefinitionsList.Request>
응답 메시지	HTTP/1.1 200 OK Server: Server Content-Length: 971 Content-Type: text/xml <?xml version="1.0"?> <WMGetProcessDefinitionsList.Response> <ProcessDefinition> <ProcessName>N_Education Check</ProcessName> <ProcDefID>ECDefWorkflow_1020932762531</ProcDefID> <ProcDefState>1</ProcDefState> </ProcessDefinition> <ProcessDefinition> <ProcessName>Hiring</ProcessName> <ProcDefID>ECDefWorkflow_1021287261406</ProcDefID> <ProcDefState>1</ProcDefState> </ProcessDefinition> <ProcessDefinition> <ProcessName>WorkshopDemo</ProcessName> <ProcDefID>ECDefWorkflow_1029412812343</ProcDefID> <ProcDefState>1</ProcDefState> </ProcessDefinition> <ProcessDefinition> <ProcessName>TripReservation</ProcessName> <ProcDefID>ECDefWorkflow_1031566711395</ProcDefID> <ProcDefState>1</ProcDefState> </ProcessDefinition> </WMGetProcessDefinitionsList.Response>

크플로우 런타임 통합 플랫폼을 거치면서 다음과 같은 XML 메시지로 변환되는데 TCP Monitor에서 모니터링 된 XML 메시지는 표 6과 같다. 이는 변환된 프로세스 정의관련 요청과 응답 메시지를 XML로 변환시킨 모습을 보여준다. 표 7은 워크플로우 엔진에 로그인 관련 요청과 응답 메시지를 XML현태로 변환된 형태를 보여준다.

다음 그림 9는 Integrator가 실행되고 Adaptor들과 연결되어 이기종의 런타임 클라이언트 사용자 환경과 상호작용하고 있는 그림이다. 여기서는 워크플로우 런타임 클라이언트에 대해서 JAX-RPC 어댑터와 RMI 어댑터, 워크플로우 엔진에서 RMI 어댑터를 사용하고 있다.

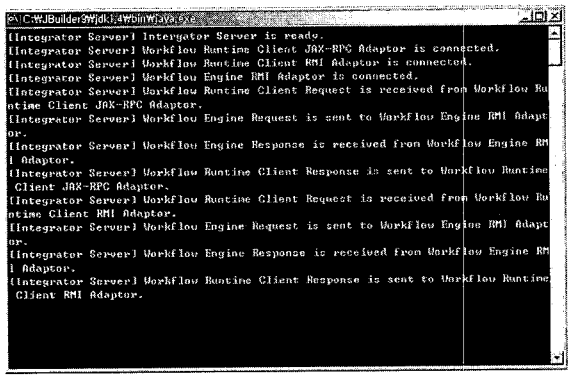
이렇게 Integrator를 실행한 후에는 이기종의 워크플로우 런타임 클라이언트를 엔진과 상호작용하게 할 수 있다. 다음 그림 10은 JAX-RPC를 사용한 순수 Java 워크플로우 런타임 클라이언트 응용을 보여준다.

또한 웹 기반의 워크플로우 런타임 클라이언트도 워크플로우 엔진과 상호작용하는 것이 가능한데 다음 그림 11은 JSP 기반의 웹 워크플로우 런타임 클라이언트를 보여준다.

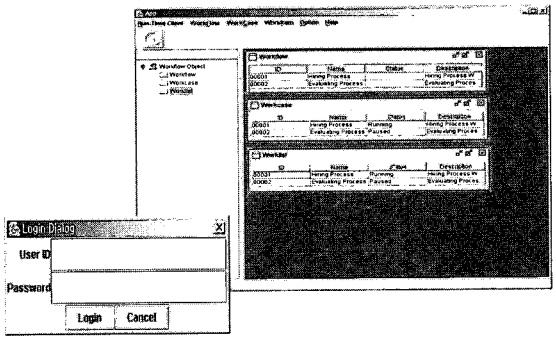
이러한 JAX-RPC를 사용한 순수 Java 워크플로우 런타임 클라이언트 응용과 JSP 기반의 웹 워크플로우 런타임 클라이언트가 엔진과 상호작용하는 모습을 그림 12에서 보여준다.

5. 결 론

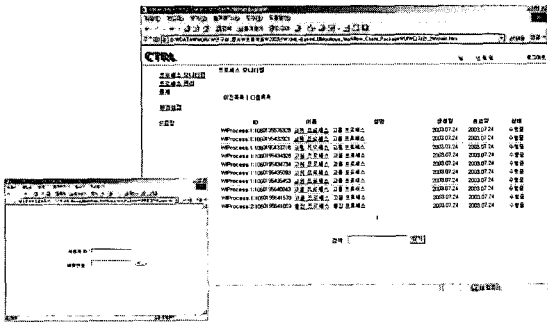
본 논문에서는 워크플로우 관리 시스템들간의 상호운용성을 보장하기 위해 표준화된 인터페이스와 데이터 교환 포맷의 문제점을 극복하기 위해서 정의된 프로세스의 상호교환을 위한 인터페이스에



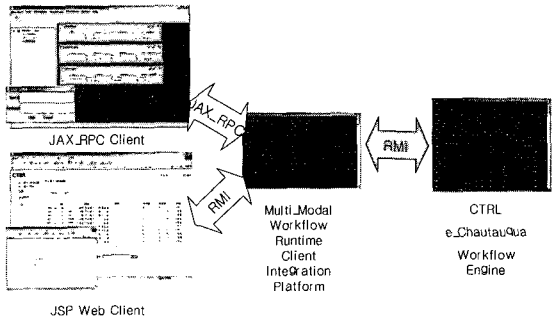
(그림 9) Integrator의 실행 화면



(그림 10) Pure Java Application 워크플로우 런타임 클라이언트



(그림 11)JSP 기반 웹 워크플로우 런타임 클라이언트



(그림 12) 클라이언트들의 엔진과의 상호작용

서는 워크플로우 프로세스 정의 메타모델이 정적 저장소를 통하여 프로세스 정의 데이터를 엔진에 전송하는데 있어 로컬지역간의 신뢰성 있는 데이터 전송이 가능하지만, 데이터의 유연성과 확장성에 문제가 대두되며 이를 극복하기 위해 프로세스 정의 모델을 XML로 표현하여 워크플로우 시스템들에게 공통의 프로세스 정의 교환 포맷을 제공하고, 개발환경과 실행환경을 분리함으로써 개발도구와 실행도구의 독립적인 개발과 호환을 지원한다.

워크플로우 런타임 클라이언트를 위한 인터페이스에서는 함수 호출 방식으로 명세되어 있어 이러한 함수 호출 방식의 인터페이스 상호방식은 응용 프로그래밍 인터페이스가 C, C++, COM과 같은 방식일 경우 워크플로우 엔진에 시스템 종속적인 문제점을 가져오고 또한 CORBA와 같은 방식일 경우에도 웹 서비스의 활성화가 급속하게 확장됨에 따라 개선된 상호작용 방법이 운용되어야 하므로 XML을 기반으로 하여 사용자 환경 통합기능을 제공하고 다양한 사용자 환경을 위한 어뎁터 기능을 제공하여 워크플로우 엔진의 API에 의존적이지 않으면서 이기종의 워크플로우 런타임 클라이언트 사용자 환경을 통합한다.

워크플로우 엔진들 간의 상호연동을 위한 인터페이스에서는 상호운용성이 확보되지 않을 경우 전체 비즈니스 프로세스의 단편화가 발생하고 워크플로우 관리 시스템간의 통신능력의 부재로 인해 프로세스의 자동수행의 흐름이 끊어지는 단점을 보안하기 위해 XML를 기반으로 하여 상호운영을 위한 단순 연결 및 중첩, 병열 동기화 모델을 지원하고, 동기적, 비동기적 상호작동을 모두 제공하며 개별 작동과 배치 작동을 모두 지원한다. 그리고 구현방식에 독립적으로 상호운용성을 제공하여 가볍고 구현 용이한 프로토콜을 정의한다.

본 논문에서 제안한 자원지향 워크플로우 통합 프레임워크는 워크플로우 엔진에 의존적이지 않으면서 워크플로우 사용자 환경을 통합 할 수 있는 매개체로 XML을 기반으로 한다. XML기반으로 한 API가 워크플로우 엔진에 독립성이 보장되고 웹 서비스 기술과 연계하여, 시간과 장소에 제약 없이 유기적으로 접근을 이룰 수 있을 것이라 기대한다.

앞으로 이러한 XML기반으로 하는 워크플로우 통합 프레임 워크에 대해 주변환경들이 급속하게 발전하므로 지속적인 연구가 필요하며 또한 웹 서비스 기반의 차세대 워크플로우 및 비즈니스 프로세스 관리 시스템도 연구를 해볼 수 있다.

참 고 문 헌

[1] K.Ogris, Applications Invocation Interface 0.9 Superceded 17 July 96

[2] WfMC-TC-1015, Audit Data Specification", The Workflow Management Coalition, 1.1, 22 Sep. 1998.

[3] ISO 8601:1988, Data elements and interchange formats - Information interchange - "Representation of dates and times", International Standards Organization, 1, 4 March 1999.

[4] eXtensible Markup Language(XML) 1.0(second edition), W3C Recommendation, Oct 6, 2000.

[5] W3C Note, Simple Object Access Protocol(SOAP) 1.1, May 8, 2000, http://www.w3.org/TR/SOAP/.

[6] Keith Swenson, "Simple Workflow Access Protocol (SWAP)", Internet-Draft, 7 Aug. 1998.

[7] Web Services Choreography Interface(WSCI) 1.0, BEA, Intalio, Sun, SAP, June 2002.

[8] Erik Christensen, Francisco Curbera, Greg Meredith, Sanjiva Weerawarana, Web Services Description Language(WSDL) 1.1, Editors. World Wide Web Consortium, 15 March 2001.

[9] Workflow Management Facility, Object Management Group, 1998.

[10] Hampshire, United Kingdom, Terminology & Glossary, Workflow Management Coalition, 1999.

[11] Brussel, Belgium, The Workflow Reference Model, Workflow Management Coalition, 1999.

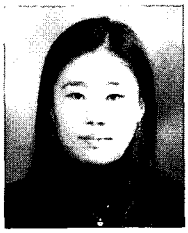
[12] Workflow Standard - Interoperability Wf-XML Binding version 1.1, Workflow Management Coalition, 2001.

[13] Workflow Process Definition Interface - XML Process Definition Language, Workflow Management Coalition, 2002.

[14] Winchster, Workflow Standard - Interoperability Abstract Specification, Workflow Management Coalition, United Kingdom, 1999.

[15] "Workflow Management Facility", Joint Submission, bom/98-06-07, revised submission, 4 July 1998.

◐ 저 자 소 개 ◑

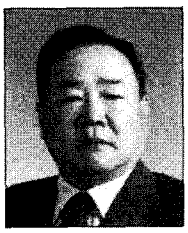


김 현 아

2001년 나사렛대학교 전산정보학과 학사
2003년 경기대학교 대학원 전자계산학과 석사
2004년 경기대학교 대학원 전자계산학과 박사 입학
2004~현재 경기대학교 대학원 전자계산학과 박사과정 재학 중

김 광 훈

1984년 경기대학교 전자계산학과 학사
1986년 중앙대학교 대학원 전자계산학과 석사
1994년 University of Colorado at Boulder, Computer Science, MS
1998년 University of Colorado at Boulder, Computer Science, Ph.D
1986년 2월~1991년 8월 한국전자통신연구원
1993년 5월~1994년 8월 American Educational Products, Inc., Professional DB Consultant
1994년 9월~1995년 8월 Colorado Advanced Software Institute, Research Assistant
1995년 9월~1997년 2월 Aztek Engineering, Inc., Software Engineer
1998년 3월~현재 경기대학교 정보과학부 조교수, 부교수
관심분야 : 워크플로우, 그룹웨어, 컴퓨터네트워크, 데이터베이스



백 수 기

1973년 연세대학교 토목공학과 학사
1979년 동국대학교 경영대학원 정보처리전공 석사
1992년 동국대학교 대학원 통계학과 박사
1973년 농업진흥공사 전자계산실
1976년 동국대학교 전자계산원 전임강사
1980년~현재 경기대학교 교수
관심분야 : 컴퓨터네트워크, 데이터구조및알고리즘, 데이터베이스