EAI(Enterprise Application Integration)와 Web Service 환경에서 트랜잭션의 효율적인 처리 방안

정 지 호† · 윤 청††

요 약

기업이 시장 환경 변화에 빠르게 대응하기 위해서는 기업 업무 프로세스의 자동화가 필요하고 이러한 자동화를 위해서는 기업 응용 체계를 통합해야 한다. 기업 응용 체계를 통합하는 방법으로는 동기적 통합(Synchronous Integration) 방식과 비동기적 통합(Asynchronous Integration) 방식이 있으며 비동기적 통합 방식으로써 EAI(Enterprise Application Integration)와 Web Service는 기업 업무 프로세스를 통합할 수 있는 방법으로써 최근 대두되고 있는 방식 중에 하나이다. 비동기적 통합(Asynchronous Integration)방식을 이용하여 기업의 응용 프로그램들을 통합한 후에는 업무 프로세스의 처리 과정인 비즈니스 트랜잭션을 자동으로 처리하기 위한 트랜잭션 관리가 필요하게 된다. 이를 위해서 2PC 프로토콜(2-Phase Commit Protocol)을 근간으로 한 비즈니스 트랜잭션 처리 모델들이 제안되고 있으나 기업 자원을 효율적으로 활용할 수 있는 데는 한계가 있다. 따라서 본 논문에서는 기업의 응용 프로그램들을 통합할 수 있는 환경으로써 EAI와 Web Service와 같은 유연한 결합(Loosely Coupled)의 분산 환경에서 기업 자원을 효율적으로 관리하면서 트랜잭션을 처리하기 위한 모델을 제안한다. 이 방식은 2PC 프로토콜 방식을 보완하기 위하여 Classify Phase를 추가한 3PC 프로토콜(3-Phase Commit Protocol)방식으로서 유연한 결합 환경에서 트랜잭션을 효율적으로 관리하고 트랜잭션 처리 자원을 절약할 수 있도록 한 방식이다. 본 논문에서는 기존의 분산 트랜잭션 처리방식인 2PC 프로토콜 기반의 트랜잭션 처리 모델의 문제점을 제시하고 이를 해결하기 위한 방법을 제시함으로써 제안한 모델의 성능을 확인하였다.

An Efficient Method of Transaction Process for EAI(Enterprise Application Integration) and Web Service

Jeeho Jeong† · Cheong Youn††

ABSTRACT

It is important to integrate an enterprise application for automating of the business process, which is responded by a flow of market environment. There are two categories of method that integrate enterprise applications. One is Synchronous Integration, and the other is Asynchronous Integration. EAI(Enterprise Application Integration) and Web service which of the asynchronous integration is focused in the automating method of the business process. After we construct the application integration for automating of the business process, we have to concern about managing of the business transaction. Many Organizations have proposed the process method of business transaction based on 2-phase commit protocol. But this method can't supply the phase that classify the transaction by transaction weight. In this paper, we propose an efficient method of transaction process for business transactions, which is composed by 'Classify Phase' that classify transactions. We called this model "3-Phase Commit Method Applied by Classify Phase," we design this model to manage an resource of enterprise efficiently. The proposed method is compared by the method based on 2-Phase commit that could be a problem of management the resource of enterprise, and the advantage of this method is certified to propose the solution of that problem.

키워드 : 비지니스 트랜잭션(BTP), EAI, 웹서비스(Web service)

1. 서 론

기업의 시장 환경은 시간이 갈수록 더욱 경쟁적으로 바뀌어 가고 있고 고객의 요구가 다양화됨에 따라 얼마나 빨리, 그리고 얼마나 효율적으로 고객의 요구를 만족시켜 줄 수 있느냐가 과거나 현재나 변함없는 기업의 경쟁력을 가름하는 중요한 요소이다. 특히 산업시대에서 정보화시대로 사회의 패러다임이 변화하는 시기에 있어서 기업의 가장 큰 변혁은 기능(function) 중심에서 프로세스(process) 중심으로 경영의 패턴이 움직이고 있다는 것이다.

프로세스 중심의 기업경영은 기업 내부의 업무 프로세스와 기업과 기업간의 업무 프로세스가 시장 변화에 빠르게 대처할 수 있도록 모든 기업 환경을 변화시키고 업무의 프로세스를 자동화시키는 것이다. 업무 프로세스의 자동화를 위해서는 프로세스 내의 기능별 응용체계 간에 필요한 자료의 흐름과 업무처리순서를 자동화해야 하며 기업의 응용

† 준 회 원 : 충남대학교 대학원 컴퓨터공학과
†† 정 회 원 : 충남대학교 전기정보통신공학부 교수
논문접수 : 2003년 10월 10일, 심사완료 : 2004년 2월 5일

체계들 간의 통합이 이루어져야 한다. 현재, 업무 프로세스 자동화 관리와 관련하여 각종 표준을 주도적으로 제정하고 있는 기구인 WfMC(Workflow Management Coalition)에서 제시한 참조모델(Reference Model)을 만족하는 다양한 WFMS(Workflow Management System) 제품들이 기업 프로세스 자동화에 기여하고 있다[1, 3-5].

WFMS 기반으로 기업의 응용체계들을 통합할 때 새로 개발되는 응용 체계들에 대해서는 WfMC RM 인터페이스 2와 3에서 제공하는 API를 이용하면 된다. 그러나 기존 개발되어 운용되고 있는 응용 체계들을 표준 API를 사용하여 통합하려고 하면 응용 체계 내부를 수정하거나 체계 대부분을 재개발해야 하는 등 통합소요가 많이 발생하게 된다. 기존의 응용체계들을 거의 수정하지 않고 통합하려면 API을 이용하여 상대 응용체계를 직접적으로 통제하는 동기식의 강력한 결합(Synchronous & Tightly-Coupled)형태에서 벗어나 비동기식의 유연한 결합(Asynchronous & Loosely-Coupled)형태의 방법으로 변화해야 하는데 EAI(Enterprise Application Integration)는 비동기식의 유연한 결합 방법을 기반으로 하여 개발된 통합 기술이다[4-6, 9].

EAI는 MOM(Message-Oriented Middleware)을 기반으로 한 통합 브로커를 중심으로 메시지 교환 및 데이터 변환작업을 통하여 상대방 응용체계 내부에 적합한 형태로 데이터를 제공하고 그 처리 결과를 메시지 형태로 받는 방식으로 이기종 시스템들을 통합한다. 즉, 프로세스 통합을 위해서는 WFMS가, 기업 정보시스템의 통합을 위해서는 EAI가 핵심적인 역할을 수행하고 있다.

EAI가 통합에 소용되는 시간과 비용을 상당히 감소시켰다면 이를 온라인 실시간 통합이 가능하도록 하는 시도가 웹 서비스(Web Services) 형태의 통합이다. 웹 서비스란 개별 응용체계가 하나의 서비스 단위로서 인터넷에 연결되어 누구나 마치 자기 회사의 응용체계처럼 사용될 수 있도록 지원함으로써 필요한 정보를 제공하는 통합체계이다. EAI기술을 이용한 통합 방법은 어댑터나 데이터 변환 도구들을 사용해야 하는 반면에 웹 서비스 형태의 통합은 이런 것에 대한 요구가 불필요한 대신 다양한 형태의 표준 인터페이스 사용을 통해 지원하게 된다. 웹 서비스를 정의하고(WSDL : Web Service Description Language), 그것을 일정한 인터넷 장소에 등록하여 필요한 사용자가 찾아볼 수 있도록 해야 하고(UDDI : Universal Description, Discovery, and Integration), 해당 웹 서비스를 수행시키기 위한 연결이 필요하며(SOAP : Simple Object Access Protocol), 데이터 변환이 필요 없도록 표준 데이터 형식을 사용토록 해야 하는 것(XML) 들이 웹 서비스를 가능케 하는 표준들이다.

이처럼 EAI와 웹 서비스 기술이 고객의 요구에 대한 반응시간을 단축시키는데 크게 기여하고 있으며 이런 경향은 B2B를 통해서 더욱 더 확대되는 추세이다. 이제는 기업의 업무 트랜잭션 처리가 기업 내부에만 국한되는 것이 아니라 해당 기업의 통제 밖에 있는 타 기업과 연계되어 그 결과가 관련되는 모든 회사에게 영향을 끼치고 있다는 것이다. 기업의 비즈니스 수행간 발생되는 트랜잭션은 장시간 내지는 장기간의 처리가요구되는 것이 대부분이기 때문에 기존의 DBMS에서 수행되는 트랜잭션 특성인 ACID(Atomicity, Consistency, Integrity, Durability)원칙을 그대로 적용하기 곤란한 문제가 많다. 특히 EAI와 웹 서비스 환경하에서는 비즈니스 프로세스에 연계되어 기업간에 발생되는 트랜잭션은 그 처리의 정확성이 관련 기업간에 보장되어야 하므로 비즈니스 트랜잭션 처리가 더욱 더 중요한 문제로 대두되고 있다[11, 12, 15].

본 논문에서는 기업정보체계(EIS)를 통합하는 플랫폼으로서의 업무프로세스 자동화 체계(WFMS : Workflow Management System)와 주변 응용체계들과의 통합을 담당하는 통합 브로커(Integration Broker)를 기반으로 하는 통합 환경으로서 EAI와 Web Service 기반의 업무 프로세스 수행간 발생하는 장기 트랜잭션(Long-Term Business Transaction)을 처리할 수 있는 방안을 제시한다. 또한, EAI와 Web Service와 같은 유연한 결합(Loosely Coupled)의 분산 환경에서 발생하는 비즈니스 트랜잭션의 특징을 살펴보고, 기존에 연구되어 온 비즈니스 트랜잭션 처리 모델들이 갖고 있는 문제점과 해결방안을 제시한다.

준비 단계(Prepare Phase)와 수용 단계(Commit Phase)로 구성된 2PC 프로토콜(2-Phase Commit Protocol)을 기반으로 비즈니스 트랜잭션을 처리할 경우 기업의 자원이 많이 소요된 트랜잭션들을 모두 취소하는 경우가 빈번하게 발생할 수 있고 전체적인 트랜잭션 소요시간을 줄일 수 있는 방안을 제시할 수 없기 때문에 기업 자원의 효율성이 떨어지게 된다. 그 이유는 자원이 많이 소요된 트랜잭션과 적게 소요된 트랜잭션을 구분하는 단계가 없고 트랜잭션의 소요시간을 통계적으로 확인할 수 있는 단계가 포함되어 있지 않기 때문이다. 따라서 이러한 문제를 해결하기 위해서는 트랜잭션에 가중치를 부여하여 트랜잭션을 분류할 수 있는 단계를 추가하여 준비 단계와 수용 단계 사이에 분류 단계(Classify Phase)를 적용함으로서 수용 단계에서 트랜잭션을 끝내기 전에 효율적으로 트랜잭션을 처리할 수 있도록 하는 것이다.

이와 같이 제안한 방식의 이름을 "분류 단계(Classify Phase)를 포함한 3PC 프로토콜(3-Phase Commit Protocol)"이라 명명하였으며 본 논문에서는 이 모델의 성능을 확인하기 위하여 기존의 분산 트랜잭션 처리방식인 2PC 프로토콜 기반의 트랜잭션 처리 모델의 문제점을 보다 자세히 살펴본 후 이를 해결하기 위한 방법을 구체적으로 제시함으로써 제안한 모델 기존 모델의 문제점들을 해결할 수 있다는 가능성을 통해서 본 모델의 성능을 확인하였다.

2. 트랜잭션과 비즈니스 트랜잭션

전통적으로, 트랜잭션이란 개념은 "전부이거나 전무(All

or Nothing)"인 결과를 생성하는 하나의 작업 단위를 말한다. 트랜잭션 모델을 정의하는 중요한 요소로서 ACID가 있으며, 이들은 트랜잭션 수행의 완전성을 보장하는 "원자성(Atomicity)"과 데이터의 무결성을 보장하는 "일관성(Consistency)", 트랜잭션의 동시성을 보장하는 "고립성(Isolation)", 그리고 수행결과의 지속성을 보장해주는 "영속성(Durability)"등으로 구성된다[11, 12].

ACID속성을 만족하기 위해서는, 하나의 트랜잭션 관리 시스템이 그 작업에 참가하고 있는 모든 트랜잭션과 자원들을 통합적으로 조정(Coordination)할 수 있는 강력한(tightly) 결합방식의 시스템을 가져야 한다. 그러나 웹 서비스와 같은 유연한 결합 방식에서는 트랜잭션을 처리하는데 짧게는 수분에서 길게는 수일, 수주까지의 시간이 필요한 상황이 발생할 수 있고 이럴 경우 트랜잭션에 참여한 자원들을 긴 시간 동안 잠글(Locking)수 없기 때문에 통합적인 조정이 불가능하게 된다. 또한, 웹 서비스는 B2B(Business to Business)를 활성화할 수 있도록 제안된 방식이기 때문에 자율적으로 트랜잭션에 참여한 참가자(Participant)들은 임의로 상대의 자원을 잠글(Locking) 수 있는 권한이 없어서 전통적인 의미의 ACID속성을 그대로 적용할 수 없게 한다[17-19].

트랜잭션의 처리 과정에서 문제가 발생할 경우, "전부이거나 전무"를 만족하기 위해서는 발생 시점 이전 상태로 되돌아가야(Rollback) 한다. 그러나 유연한 결합 환경에서는 비즈니스에 참가한 참가자들이 철회(Undo)할 수 없는 작업을 수행해 버린 상태일 수 있기 때문에 통합적인 중앙 통제를 통해서 되돌아가기를 실행할 수 없다. 따라서 이럴 경우에는 ACID의 원자성을 보장하기 위해서 취소(Cancel)를 대신할 수 있는 있는 보상 트랜잭션(Compensating Transaction)이 필요하게 된다.

기업에서 비즈니스 프로세스를 처리하는데 사용되는 자원(Resource)들은 데이터베이스와 같은 시스템 단위의 자원뿐만 아니라 인적 자원 등의 여러 가지 기업 자원들을 포함하고 있기 때문에, 시스템 단계에서 적용했던 ACID 속성을 기준으로 트랜잭션을 정의할 수 없다. 비즈니스 기능(Function)을 처리하는 응용(Application) 단계에서 트랜잭션을 정의하기 위해서는 처리시간(Duration)과 기업자원의 잠금(Locking), 그리고 되돌림(Rollback)을 고려해서 ACID를 만족할 수 있는 비즈니스 트랜잭션을 정의해야 하는 것이다[18-20].

특별히 장기(Long-term) 트랜잭션에 대해서, 확장 트랜잭션(Extended Transaction)이나 ATM(Advanced Transaction Model)은 느슨한 고립성(Relaxed Isolation)이라는 새로운 속성을 정의하여, 모든 참가자들이 동의(Agree)하기 전까지는 트랜잭션의 결과를 확인할 수 없었던 전통적인 고립성의 개념에, 서비스의 중간 결과를 확인할 수 있는 유연함을 추가하였다. 또한, 부분적인 복귀(Rollback)와 철회(undo)가 가능한 느슨한 원자성(Relaxed Atomicity)을 정의하였다. 그 밖에도 범위성(Scability), 일시적인 데이터 관리(Temporal Data Management), 접근 제어(Access Control), 시간 제어(Time Control)등을 정의하여 유연한 결합 환경에서 트랜잭션의 신뢰성을 보장할 수 있는 속성들을 제시하고 있다. 그러나 이 속성을 시스템 단계에서 적용할 경우, 시스템이 복잡하게 구성될 수 있기 때문에 성능을 떨어뜨릴 수 있다. WFMS를 이용해서 ATM을 표현하는 것이 가능하다는 것을 [21]이 보여주고 있으나 트랜잭션이 완결되지 못할 경우를 보장하기 위해서 보상 트랜잭션을 수동으로 정의해야 하기 때문에 관리가 복잡해지는 단점을 갖고 있다[21, 22].

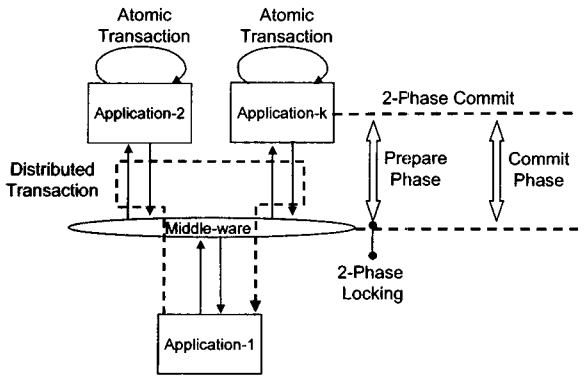
비즈니스 트랜잭션에 참여하는 응용들이 이기종(Heterogeneous)이면서 다양한 표준과 기술의 비동기적(Asynchronous)인 분산 환경이며 각기 다른 기업이나 조직에 포함되어 있기 때문에 시스템 기반의 잠금 방식을 통해서 ACID를 만족시키는 것이 아니라, 응용 기반에서 확장 트랜잭션이나 ATM의 트랜잭션 속성들을 자동으로 적용할 수 있는 통합된 메시지 관리 규약이 필요하게 된다. 따라서 웹 서비스와 같은 유연한 통합 환경에서 신뢰성을 부여하면서 비즈니스 트랜잭션을 관리할 수 있는 모델은 장기 트랜잭션(Long Term Transaction)과 분산 트랜잭션(Distributed Transaction), 그리고 보상 트랜잭션(Compensating Transaction)의 특성들을 고려한 메시지 프로토콜이 되어야 한다.

3. 비즈니스 트랜잭션 처리 모델과 문제점

비즈니스 트랜잭션(Business Transaction)은 "비즈니스 프로세스의 진행 상태를 일관성 있게 변화시키는 것"을 의미한다. 예를 들어, '주문 처리'와 같은 경우, 하나의 주문 처리가 완료되면 비즈니스는 하나의 일관성 있는 변화가 이루어졌다고 할 수 있다. 이는 주문을 받아 관련된 데이터베이스가 갱신되고 하나의 구매주문서가 추가되는 것을 의미한다. 유연한 결합 환경에서 발생하는 비즈니스 트랜잭션을 처리하는 모델에 대하여 여러 기관에서 2PC 프로토콜(2-Phase Commit Protocol)을 이용한 처리 모델을 제안하고 있다. 2001년 HP, Oracle, BEA는 OASIS(Organization for Advanced Structured Information) 위원회를 구성하여 웹 서비스와 같은 유연한 결합 환경에서 B2B(Business to Business) 트랜잭션을 처리할 수 있는 작업을 시작하여 2002년 4월 BTP(Business Transaction Protocol)라는 트랜잭션 표준안을 제안하였고 IBM, Microsoft, BEA에서는 WS-C(Web Service Coordination)와 WS-T(Web Service Transaction)라는 그들 자신만의 스팩을 만들어 배포하였다.

비즈니스 트랜잭션이 분산 트랜잭션 속성을 갖기 때문에 BTP는 분산 데이터베이스 환경에서 제안되어 왔던 2PC 프로토콜을 적용하였다. 로컬 시스템(Local System)에서 트랜잭션 처리를 위한 준비상태를 점검한 후 전체 트랜잭션의 처리 완료를 수행하는 방식을 2PC 프로토콜(2-Phase Com-

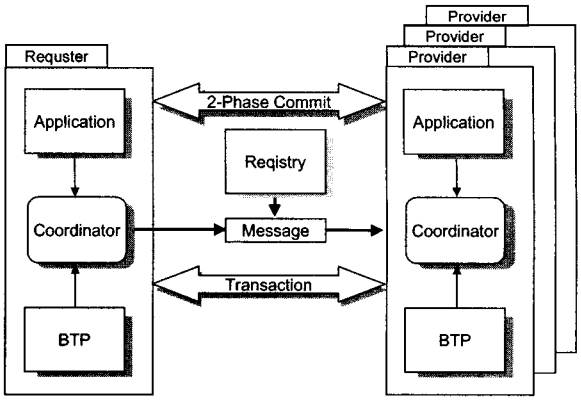
mit Protocol)이라 하며 준비 단계(Prepare Phase)와 완료 단계(Commit Phase)로 구성된다.



(그림 1) Two-Phase Commit Protocol의 구성

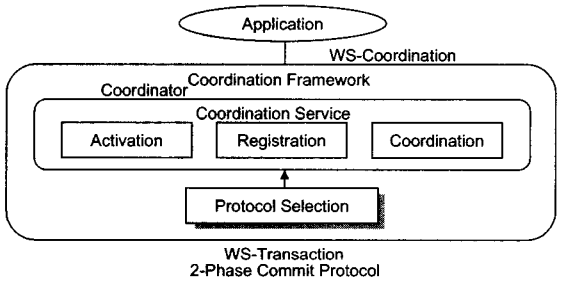
(그림 1)에서 확인할 수 있듯이 분산 트랜잭션을 발생시킨 Application-1은 Atomic Transaction을 2단계(2-phase) 상태로 확인하면서 트랜잭션을 처리한다. 준비 단계(Prepare Phase)에서 Application-1에 있는 조정자(Coordinator)는 트랜잭션에 참가한 응용들이 수용(Commit)이나 복귀(Rollback)를 수행해도 되는가를 확인하는 단계이고, 수용 단계(Commit Phase)는 준비 단계(Prepare Phase)의 상태를 확인하여 완료(Commit)나 복귀(Rollback)를 수행하는 단계이다. 준비 단계(Prepare Phase)는 모든 부분 트랜잭션(Local Transaction)이 완료(Commit) 직전단계에 이르렀다는 준비된(Prepared) 응답과 부분 트랜잭션(Local Transaction)에 의하여 아무런 변화가 없다는 읽기 전용(Read-Only) 응답, 그리고 준비(Prepared)되지 않았다는 중단(Abort) 응답 등으로 구성되어 있다. 완료 단계(Commit phase)는 모든 부분 트랜잭션이 준비된(Prepared) 응답을 받으면 완료(Commit)를 수행하고 하나라도 중단(Abort) 응답을 받게 되면 중단(Abort)되어 모든 부분 트랜잭션(Local Transaction)이 복귀(Rollback) 된다. 이 2PC 프로토콜을 기반으로 BTP 모델이 제안되었으며 (그림 2)와 같다.

(그림 2)는 OASIS Committee에서 제안하고 있는 비즈니스 트랜잭션 처리 방식인 비즈니스 트랜잭션 프로토콜(BTP : Business Transaction Protocol)의 처리 구조를 나타내고 있다. BTP는 유연한 결합 환경에서 발생하는 메시지를 인식할 수 있는 조정자(Coordinator)를 포함하고 있으며 분산 트랜잭션을 처리할 수 있는 2PC 프로토콜을 기반으로 메시지를 처리할 수 있게 하였다. 조정자는 비즈니스 트랜잭션에 참여한 참가자(Participant)에게 트랜잭션 프로토콜을 통해서 비즈니스 기능들을 수행할 수 있도록 메시지를 전달하고 준비 단계(Prepare Phase)를 수행시킨 후 그 결과 메시지를 전달받아 비즈니스 트랜잭션을 완결하는 하는 완료 단계(Commit Phase)를 수행시킨다. 그러나 2PC 기반의 프로토콜은 각 참가자가 비즈니스 기능들을 수행할 때 소요된 시간과 자원을 확인하는 방법이 없기 때문에 비



(그림 2) BTP 처리 구조

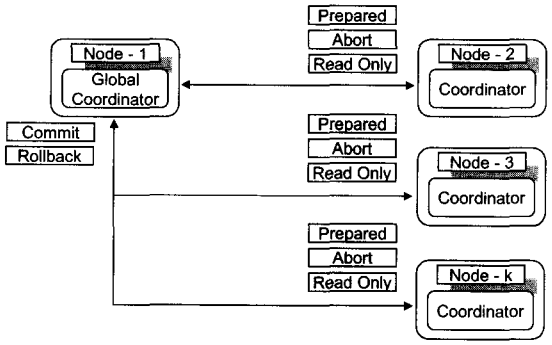
즈니스 트랜잭션을 처리하는데 소요되는 전체적인 비용을 인식할 수 없고 이것은 기업 경영자들에게 비즈니스를 수행하는데 효율성을 자동으로 측정할 수 있는 기준을 제시하지 못하는 문제점을 발생시키게 된다. 이와 같은 방식을 기반으로 특별히 웹 서비스에서의 트랜잭션 처리 방안을 제안하고 있는 모델이 웹 서비스 트랜잭션 모델(WS-Tx)이며 (그림 3)에서 표현되고 있다.



(그림 3) WS Transaction과 WS Coordination

WS-T(WS Transaction)와 WS-C(WS Coordination)는 웹 서비스 환경에서 발생하는 비즈니스 트랜잭션을 안정적으로 처리할 수 있는 모델과 시스템 구성이다. (그림 3)은 비즈니스 기능을 수행하는 응용 프로그램과 연동하는 시스템들 중에서 하나만 떼어낸 모습을 나타내고 있다. WS-C에 존재하는 조정자는 응용 프로그램이 서비스를 수행하는 액티비티 부분과 서비스를 등록하는 등록 부분, 그리고 메시지를 총괄하는 조정 부분으로 구성된다. 모든 구성 요소들은 자신의 작업을 인식할 수 있도록 프로토콜을 탑재해야 하기 때문에 프로토콜 선택기(Protocol Selection)가 WS-C에 포함된다. 이 시스템 구성으로 비즈니스 트랜잭션을 처리할 수 있는 기반이 구성되어 프로토콜에 따라서 비즈니스 트랜잭션의 속성들을 처리할 수 있는 모든 기준들이 준비될 수 있다. 그러나 이 시스템에서도 트랜잭션이 처리되는데 소요되는 자원을 측정할 수 있는 구성요소가 존재하기 않기 때문에 기업이 비즈니스 트랜잭션 처리에 소요되는 비용을 측정할 수 없는 문제점을 갖고 있다.

비즈니스 트랜잭션 처리 모델들은 트랜잭션 처리에 소요되는 자원을 측정할 수 없는 문제점을 갖고 있다. 그 이유는 2PC 프로토콜을 기반으로 트랜잭션을 처리하기 때문이다. (그림 4)는 2PC 프로토콜 방식을 기반으로 한 비즈니스 트랜잭션 처리 모델의 특징을 나타내고 있다.



(그림 4) 유연한 결합에서 기존 모델들의 처리 특성

(그림 4)와 같이 2PC 프로토콜 기반으로 트랜잭션을 처리한다면, 트랜잭션에 참여한 참여자들 중에서 자원이 많이 소요된 참여자는 준비(Prepared) 되어 있으나 자원이 적게 소요된 참여자의 자원이 중단(Abort)되어 있는 경우 전무(Nothing)를 위해서 자원이 많이 소요된 참여자가 무조건 원래 상태로 돌아가(Rollback)야 한다면 얼마 지나지 않아 다시 이 작업이 수행될 때 이중으로 자원이 낭비될 수 있다. 따라서 이 2PC 프로토콜 자체적으로 비즈니스 트랜잭션 처리에 대한 비효율적인 특성이 있기 때문에 문제점들을 구체적으로 정리하면 다음과 같다.

① 적은 자원이 소요되는 부분 트랜잭션이 Abort되면 상대적으로 많은 자원을 소모했던 다른 부분 트랜잭션까지 Rollback되는 경우가 있다. 이럴 경우 전체적인 자원의 낭비가 이루어 질 수 있다.

② 하나의 노드에서 전역 트랜잭션의 Commit Phase 일 때만 문제가 생겼을 경우 Initiator에 의하여 다시 전역 트랜잭션을 실행시키지 않으면 큰 문제가 없었던 전역 트랜잭션이 완전히 소멸될 수 있다.

③ 희소가치가 있는 트랜잭션은 Prepared 되었으나 상대적으로 적은 희소가치를 갖는 트랜잭션이 Abort 되었을 경우 Initiator 관리자에 의하여 다시 전역 트랜잭션을 수행한다 하더라도 기회를 상실할 수 있다.

④ 시간이 많이 소요되었던 부분 트랜잭션이나 여러 개의 부분 트랜잭션을 갖는 부분 트랜잭션은 Prepared 되었으나 시간이 적게 소요되는 부분 트랜잭션이 Aborted 되었으면 전역 트랜잭션의 Rollback 이 발생하기 때문에 다시 전역 트랜잭션을 시작한다면 다시 많은 시간과 많은 부분 트랜잭션들을 발생시키기 된다.

⑤ 관리자는 다양한 부분 트랜잭션들의 자원 소요 비중을 Monitoring 할 수 없기 때문에 트랜잭션의 효율적인 관리가 이루어지기 어렵다.

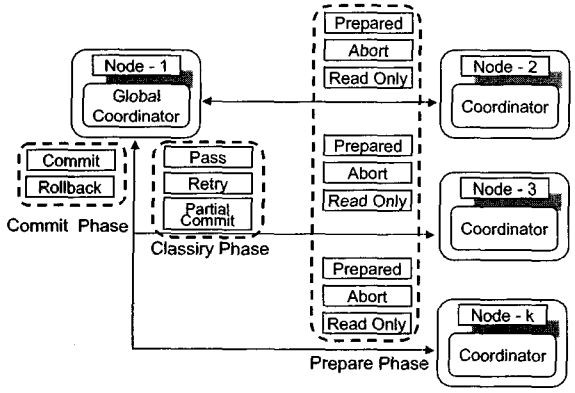
이와 같은 문제들 때문에 2-Phase Commit을 기반으로 하는 기존 모델들을 이용하여 유연한 결합 환경에서 비즈니스 트랜잭션을 효율적으로 처리하는데 한계가 있다. 따라서 2-Phase Commit Protocol을 이용하여 유연한 결합 환경에서 비즈니스 트랜잭션을 처리하는 모델들의 한계를 극복하기 위해서는 새로운 형태의 Commit Protocol을 적용한 모델을 연구하여 기업 자원 활용에 대한 효율성을 극대화하고 신뢰성을 제공하면서 비즈니스 트랜잭션을 처리할 수 있는 새로운 모델이 제안되어야 할 것이다.

4. 분류 단계(Classify Phase)를 적용한 3-Phase Commit Protocol

지금까지 제안되고 있는 비즈니스 트랜잭션 처리 모델은 2PC를 기반으로 앞서 기술한 BT(Business Transaction) 특성을 고려하여 변형된 ACID를 만족함으로서 신뢰성을 부여하도록 하였다. 그러나 2PC를 적용할 경우 "전부이거나 전무"를 만족하기 위해서 트랜잭션 전체를 무조건 처리하는 경우, 기업 자원의 활용 측면에서 한계가 발생할 수 있다. 트랜잭션에 참여한 참여자들 중에서 자원이 많이 소요된 참여자는 준비(Prepared) 되어 있으나 자원이 적게 소요된 참여자의 자원이 중단(Abort)되어 있는 경우 전무를 위해서 자원이 많이 소요된 참여자가 무조건 원래 상태로 돌아가야 한다면 얼마 지나지 않아 다시 이 작업이 수행될 때 이중으로 자원이 낭비될 수 있기 때문이다. 물론, 자원 소요를 고려하여 트랜잭션 처리에 대하여 정의할 수 있지만, 이것은 수동으로 트랜잭션을 처리하게 하는 방식이기 때문에 비즈니스 트랜잭션 자동처리 모델에 대한 근본적인 해결방안과 부합될 수 없는 이유가 된다.

본 논문에서는 이러한 한계를 극복하기 위해서, 비즈니스 트랜잭션의 비중을 분류할 수 있는 Classify Phase 개념을 BTP의 2PC에 추가하여, 자원의 효율성까지 고려할 수 있는 새로운 3PC(3-Phase Commit)프로토콜의 비즈니스 트랜잭션 처리모델을 제안한다. 이 모델은 트랜잭션을 가중치별로 분류하는 단계(Classify Phase)를 2PC에 적용하여 3PC 프로토콜 방식을 구성하였으며 간과될 수 있는 자원의 낭비를 찾아 이를 제거함으로서 B2B 환경에서 비즈니스 자원 활용을 극대화할 수 있게 하기 위한 방법이다.

유연한 결합 환경은 메시지를 기반으로 비즈니스 프로세스를 처리하는 방식을 사용하는데, 3PC는 이러한 메시지 내에 비즈니스 트랜잭션의 가중치를 포함하도록 프로토콜을 구성하고 비즈니스 트랜잭션을 처리하는데 소요되는 자원의 가중치를 계산하여 트랜잭션의 유형을 분류할 수 있도록 하였다. 비즈니스 트랜잭션에 참가한 참가자들 중에서 트랜잭션을 다시 처리할 필요가 있는 참가자에게만 다시 메시지를 보내는 단계를 Classify Phase라고 정의 하였으며 이 Classify Phase를 포함한 3PC 프로토콜은 (그림 5)와 같다.



(그림 5) 3PC의 개념도

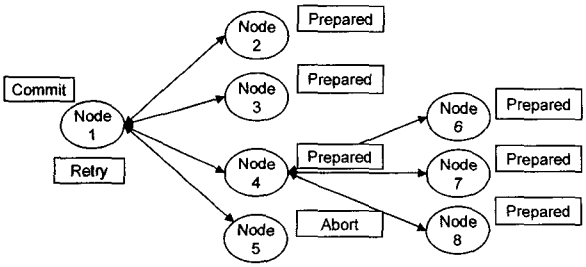
3PC 프로토콜의 처리 과정은 준비 단계(Prepare Phase), 분류 단계(Classify Phase), 수용 단계(Commit Phase)로 구성된다. 준비 단계(Prepare Phase)는 2PC단계와 일치하며 전역 조정자(Global Coordinator)가 나머지 모든 노드에게 준비(Prepare)하도록 요청하는 단계이다. 준비 단계(Prepare Phase)에서 각 참가자들은 준비(Prepared), 중단(Aborted), 단순 읽기(Read-Only)들 중에서 하나의 결과를 가질 수 있다. Classify Phase는 전역 트랜잭션을 분류하는 역할을 하며 Passed, Retry, Partial commit의 결과를 갖는다. Passed는 참가자들 모두 Prepared되었을 때 발생하는 결과이며, Retry는 가중치가 높은 부분 트랜잭션은 Prepared 되었으나 가중치가 낮은 부분 트랜잭션 Aborted 되었을 경우 이 부분 트랜잭션만 다시 Prepare 단계를 수행할 때 나타나는 결과이다. Aborted 된 경우 Partial Commit은 가중치는 높지만 시간이 지날 경우 자동으로 Rollback이 될 수 있는 경우에 발생하는 결과이다.

가중치를 계산하는 방식은 자동방식과 수동방식이 있는데, 수동방식은 WFMS에서 설정하는 것과는 다른 형태가 된다. WFMS에서는 트랜잭션의 모든 경우에 대하여 잘 못될 경우에 대한 정의가 필요하지만, 이 모델의 가중치 수동설정방식은 각 참가자(Participant)의 결과에 대한 중요도만 설정하면 되기 때문에 시나리오를 입력해야하는 WFMS 방식과는 보다 간편할 수 있다. 그러나 이 방식도 시스템을 복잡하게 할 수 있을 때는 Classify Phase가 자체적으로 가중치를 계산할 수 있는 방식을 내포하고 있는 자동방식을 사용할 수 있다. 가중치 자동설정 방식은 전체 소요 시간과 소요자원을 기준으로 각 참여자의 가중치를 계산한다. 그리고 처음 시스템을 설정하고 비즈니스 트랜잭션을 시작한 후 현재까지의 준비 단계 요청 후 각 참가자의 준비율(Prepared Ratio)을 통계적으로 저장할 수 있도록 해서 준비율이 낮은 경우, 즉 희소가치가 있는 참가자에 대한 가중치를 높일 수 있도록 하였다. 이 방식에 대한 정리는 식 (1)과 같다.

$$W_p = \left(K \times \frac{P_p}{R_p} + \frac{T_p}{T_t} + \frac{S_p}{S_t} \right) \times 100$$

식 (1) 각 참가자의 트랜잭션 가중치 계산식

이 계산은 조정자(Initiator)에서 이루어지며 메시지 프로토콜의 Classify Phase에서 이를 계산하여 Passed, Retry, Partial commit 중 하나를 선택할 수 있는 기준을 만들게 된다. K는 통계계수를 나타내고, R_p는 한 번 조정자에 등록된 참가자에 대하여 준비단계에서 요청한 총 요청수를 나타내고 P_p는 참가자가 그 요청에 준비 응답을 나타낸 총 수를 나타낸다. T_t는 비즈니스 트랜잭션 발생 이후 완료될 때까지 소요된 총시간이고 T_p는 참가자가 요청을 받고 응답할 때까지 소요된 시간을 나타낸다. S_t와 S_p는 각각 비즈니스 트랜잭션 전체자원과 참가자가 트랜잭션에 참여한 자원을 나타내지만, 수동입력이 없을 경우 S_p를 0으로 하여 식의 항목에서 제외될 수 도 있다. 이러한 가중치 분류를 적용하여 Classify Phase를 수행한 후, Commit Phase는 ATM이 갖고 있는 속성을 가지면서, BTP에서처럼 Relaxed Atomicity, Relaxed Isolation, Time Control 기능을 수행할 수 있도록 한다. 이 3PC 프로토콜의 트랜잭션 처리 효율성을 확인하기 위한 예제로서, 비즈니스 프로세스가 (그림 6)과 같이 발생한다고 가정한다.



(그림 6) 3-phase commit 예

Node 1에서 전역 트랜잭션을 시작하여 모든 나머지 노드들에 대해 Prepare Phase를 수행한다. Node 2~Node 4 까지는 Prepared 되는데 20일이 경과 되었다고 가정하고 1일 소요되는 Node 5에서 Abort가 이루어졌다면 Classify Phase는 Retry를 시도한다. 그러나 다시 Node 5가 Abort된 경우 Node 3은 시간이 지나면 자동으로 취소되는 부분 트랜잭션이라면 Classify Phase에서는 Partial Commit을 수행한 후 Commit Phase로 전달되고 Commit Phase는 Rollback을 수행하는데 Node 3에 대해서는 Rollback Message를 발생시키지 않는다.

2PC 프로토콜을 기반으로 이 비즈니스 트랜잭션을 처리한다면 Node5가 Abort된 경우 모든 나머지 Node들에게 Cancel 메시지를 발송하여 각 Node들의 작업 결과를 다시 원점으로 돌려서 비즈니스 트랜잭션이 전체 비즈니스 프로세스에 아무런 문제가 발생하지 않도록 한다. 만약, Abort된 Node5가 잠시 문제가 발생했을 경우나 다시 작업을 요청했을 때 Prepared될 수 있는 상황이라고 한다면 처음부터 모든 Node들에게 다시 작업 요청을 하는 메시지를 발송하여 처음 발송했을 때와 같은 자원들이 소요될 수 있을 것이다. 그러나 3PC(3 Phase Commit) 프로토콜과 같은 경

우 전체 트랜잭션 처리 소요 자원들 중에서 약 5% 정도의 소요자원을 차지하는 Node5만 Abort된 경우 Classify Phase에서 이를 계산하여 Node5에만 다시 작업을 요청하여 Prepared를 기다리기 때문에 앞서 기술한 문제가 발생할 경우 Node5만 자원이 소요되어 전체 비즈니스 트랜잭션을 처리하는 자원을 95% 감소시킬 수 있는 효율성을 보장할 수 있다. Node5가 계속해서 Abort되는 경우에는 전체 비즈니스 트랜잭션을 원상태로 복귀시키는 메시지를 모든 Node들에게 발송하여 2PC와 큰 차이가 없겠지만, 비즈니스 프로세스에서 반복적으로 발생하는 트랜잭션들을 계속해서 재활용할 수 있다는 측면에서는 3PC가 효율성을 증대시키는 역할을 할 것이다.

이 예제에서 가중치를 계산하는 과정을 구체적으로 확인하기 위하여, 각 Node들에 대한 비즈니스 트랜잭션 처리 소요 자원을 트랜잭션 처리에 참가하는 사람을 기준으로 상세하게 기술한다.

① Node1은 비즈니스 트랜잭션을 발생시킨 서비스 요청자가 되고 여기에 소요되는 자원은 1명이 된다.
② Node2는 2명이 작업에 참가하며 처리 기간은 약 3일 이라고 가정한다.
③ Node3도 5명이 작업에 참가하며 처리 기간은 약 5일 이라고 가정한다.
④ Node4는 하부 처리 Node들이 존재하며 하부 Node들(Node6, Node7, Node8)을 포함하여 총 12명이 작업에 참가하며 처리 기간은 약 20일 이라고 가정한다.
⑤ Node5는 1명이 작업에 참가하며 처리 기간은 약 1일 이라고 가정한다.

이 예제는 매우 극단적인 경우를 보여주는 예이며, Node1에서는 각 Node들이 어느 정도의 자원이 소요되는지 확인하지 못할 수도 있다. 이럴 경우에는 Node1이 인식할 수 있는 것은 처리 시간을 통해서 소요되는 자원일 것이다. 그러나 이번 예에서는 참가자의 자원 소요내용이 공개된다고 가정할 것이다.

이 예제를 통해서 2-Phase Commit Protocol을 기반으로 하는 비즈니스 트랜잭션 처리 모델과 본 논문에서 제안하는 3-Phase Commit Protocol의 비즈니스 트랜잭션 처리 모델의 자원 효율성을 비교하면 <표 1>, <표 2>와 같다.

<표 1>은 모든 Node가 비즈니스 트랜잭션이 발생한 후 한 번에 Prepared된 경우를 나타내며 이 경우는 모든 Node가 Commit되기 때문에 2PC나 3PC 모두 소요되는 자원이 일치한다. 그러나 <표 2>와 같이 가장 적은 자원이 소요되는 Node5만 Abort되고 나머지 Node들이 Prepared된 경우는 3PC가 2PC보다 19명 28일이 절약됨을 알 수 있다.

2PC를 기반으로 하는 비즈니스 트랜잭션 처리 모델은 트랜잭션 처리시 발생하는 처리 자원들의 중복되는 부분과 자원의 가중치를 계산하는 과정이 없기 때문에 기업 입장에서는 자원의 새어나가는 부분을 확인할 수 없고 이것은 기업 경쟁력을 향상시키는데 어려움을 만들 것이다. 그러나 3PC는 표에서 확인할 수 있듯이 중복되는 처리 자원들을 확인하여 가중치를 계산하고 가중치별로 트랜잭션을 관리할 수 있는 Classify Phase를 포함하고 있기 때문에 기업 경영에서 자원 관리에 보다 효과적인 대안들을 제시할 수 있게 된다.

<표 1> 모든 Node가 한 번에 Prepared된 경우

	2PC 소요 자원	3PC 소요 자원
Node2	2명 3일	2명 3일
Node3	5명 5일	5명 5일
Node4	12명 20일	12명 20일
Node5	1명 1일	1명 1일
합 계	20명 29일	20명 29일

<표 2> Node5만 Abort되고 나머지는 Prepared된 경우

	2PC 소요 자원	3PC 소요 자원
Node2	(2명 3일)×2 = 4명 6일	2명 3일
Node3	(5명 5일)×2 = 10명 10일	5명 5일
Node4	(12명 20일)× = 24 명 40일	12명 20일
Node5	1명 1일	1명 1일
합 계	39명 57일	20명 29일

5. 결 론

기업의 응용체계는 다양한 기술과 표준으로 구성되어 있어 동기식 방법으로 통합하기에는 통합비용이나 시간측면에서 많은 어려움이 있어 메세징 기술을 이용한 비동기식 방법위주의 EAI나 웹 서비스 기술을 적용하고 있는 추세이다. 본 논문에서는 유연한 통합 환경에서의 효율적인 비즈니스 트랜잭션을 관리하는 모델을 제안하였다. 이 모델은 트랜잭션이 잘못되었을 경우 2PC 프로토콜이 "전부이거나 전무"인 트랜잭션의 속성을 만족시키기 위하여 자원 소요 비중을 판단하지 않고 무조건 복귀(Rollback)를 실행하기 때문에 발생했던 효율성 저하라는 단점을 보완한 모델이다. 결국, 트랜잭션 소요 자원의 비중을 판단하여 복귀를 명령하기 전에 다시 한번 트랜잭션 처리 시도를 할 수 있는 Classify Phase를 추가하여 3PC 프로토콜로 개선하여 제시하였으며 이 모델의 향상된 성능을 확인하였다.

확인 결과, 3PC 프로토콜 방식은 각 서브 트랜잭션의 가중치별 정의와 분류를 가능하게 하였다. 이는 각 서브트랜잭션을 정밀하게 제어함으로써 모든 자원의 효율성을 극대화 시키는 관리방법을 제시하였다. 뿐만 아니라 각 트랜잭션들의 재활용성을 높여주고 트랜잭션의 처리과정에서 발생할 수 있는 비효율적인 오류의 원인을 찾을 수 있게 해주었다.

또한 트랜잭션 관리의 효율성을 증대시키는 트랜잭션 모니터링 기법을 제시하였으며 이는 복잡한 Multi-level Transaction에 능동적인 대처 방안으로써의 가능성도 갖고 있음을 보여 주고 있다.

향후 연구 과제는 개선된 3PC 프로토콜에 추가된 Classify Phase 단계에서 수행되는 가중치 계산을 보다 정확히 할 수 있는 계산 모델과 시스템 레벨에서의 계산으로 인한 추가 자원을 최소화 하는 연구, 그리고 이것의 결과가 신뢰성을 가질 수 있도록 비즈니스 레벨의 소요자원 측정의 정확도를 높이는 연구가 병행되어야 할 것이다.

참 고 문 헌

[1] Eder, J., Groiss, G, Liebhart, W. "The Workflow Management System Panta Rhei," *In Advances in Workflow Management Systems and Interoperability*, Springer-Verlag, 1998.

[2] P. W. P. J. Grefen, R. N. Remmerts de Vries ; *A Reference Architecture for Workflow Management Systems* ; *Journal of Data & Knowledge Engineering*, North Holland Elsevier, Vol.27, No.1, pp.31-57, 1998

[3] D. Georgakopoulos, M. Hornick, A. Sheth. "An Overview of Workflow Management : From Process Modeling to Workflow Automation Infrastructure," *Distributed and Parallel Databases,* Vol.3, No.2, pp.119-153, April, 1995.

[4] Gisolfi, Dan. Web services architect, Part 1 : *An introduction to dynamic e-business,* IBM developerWorks.

[5] Gisolfi, Dan. Web services architect, Part 2 : *Models for dynamic e-business,* IBM developerWorks.

[6] Snell, James. Web services insider, Part 1 : *Reflections on SOAP,* IBM developer Works.

[7] *Simple Object Access Protocol (SOAP).* W3C. (www.w3.org).

[8] *Web Services Description Language (WSDL).* (www.w3.org/TR/wsdl)

[9] Arkin, A., Askary, S., Fordin, S., Jekeli, W., Kawaguchi, K., Orchard, D., Pogliani, S., Riemer, K., Struble, S., Takacsi-Nagy, P., Trickovic, I., Zimek, S. *Web Service Choreography Interface 1.0 Specification*, BEA, Intalio, SAP and Sun, http://ftpna2.bea.com/pub/downloads/wsci-spec-10.pdf, June, 2002.

[10] Christensen, E., Curbera, F., Meredith, G., Weerawarana, S. *(Eds.) Web Services Description Language (WSDL) 1.1*, W3c Note, March, 2001, http://www.w3.org/TR/2001/NOTE-wsdl-20010315.

[11] J. Gray and A. Reuter, Transaction Processing : *Concepts and Techniques, Morgan Kaufmann*, San Francisco, 1993.

[12] The Open Group. *X/Open Distributed Transaction Processing Reference Model*, Version 3, February 1996 ; http://www.opengroup.org.

[13] M. H. Nodine and S. B. Zdonik, "Cooperative Transaction Hierarchies : A Transaction Model to Support Design applications," *Proceedings of the 16th International Conference of Very Large Databases*, Brisbane, Australia, August, 1990.

[14] D. Barbara, S. Mehrota, and M. Rusinkiewicz. "INCAS : Managing Dynamic Workflows in Distributed Environment," *Journal of Database Management*, 7(1), pp. 5-15, IDEA Group Publishing, 1996.

[15] A. K. Elmagarmid(ed.) "*Transaction Models for Advanced Database Applications*," Morgan-Kaufmann, 1992.

[16] M. Hsu. *Special Issues on Workflow and Extended Transaction Systems, Bulletin of the IEEE Technical Committee on Data Engineering*, 16(2), June, 1993 and 18(1), March, 1995.

[17] Sanjay Dalal, Pal Takacsi-Nagy, "*Business Transaction Protocol Version 1.0 Primer*," OASIS, 2001.

[18] OASIS Business Transaction Technical Committee, "Use Cases for the Business Transaction Protocol," OASIS, 2001.

[19] OASIS, "Business Transaction Protocol Primer," An OASIS Committee Supporting Document, 2002.

[20] Muhammad F. Kaleem, "Transaction over Web Services An Introduction to the Business Transaction Protocol," WebService.Org.

[21] G. Alonso 외 5명, "Advanced Transaction Models in Workflow Contexts," IBM Research Report.

[22] Mark Little, Thomas J. Freund, "A comparative analysis of WS-C/WS-Tx and OASIS BTP".

정 지 호

e-mail : apolosh@hanmail.net

1977년 육군사관학교 토목공학과(학사)

1986년 미해군 대학원 Computer Science (공학석사)

1988년~1992년 Delaware Computer Science (박사수료)

2004년 충남대학교 대학원 컴퓨터공학과 박사과정

관심분야 : 객체지향 모델링, 분산 컴퓨팅, 네트워크 관리, 워크플로우

윤 청

e-mail : cyoun@cs.cnu.ac.kr

1973년~1979년 서울대학교 물리학과

1982년~1983년 Illinois University Computer science 석사

1985년~1988년 Northwestern Universtiy Computer science 박사

1988년~1993년 Bell Communication Research 선임연구원

1993년~현재 충남대학교 전기정보통신공학부 교수

관심분야 : 소프트웨어공학, 객체지향 분석 및 설계