분산 환경에서 장기 트랜잭션의 효율적인 처리 방안

정지호, 엄기환*
충남대학교, 동국대학교*

Efficient Method of Processing Long-term Transactions for Distributed Environment

Jee-ho Jung, Ki-hwan Eom*
Chungnam National University, Dongguk University*
E-mail : reachj@empas.com

요 약

기업은 내부의 정보체계를 프로세스나 워크플로우 중심으로 통합함으로써 고객의 요구가 달성되기까지의 과정과 시간을 최소화하고 고객의 가치를 극대화하여 기업의 시장 경쟁력을 확보하려고 한다. 그러한 통합노력은 ERP, CORBA, DCOM등 패키지 또는 동기식의 강력한 결합(Synchronous & Tightly-Coupled)방식에서 시작하여, 인터넷 기술의 발전으로SCM(Supply Chain Management), CRM(Customer Relationship Management), e-Business, B2B(Business-to-Business)등이 확대됨으로써 이 기종의 다양한 플랫폼을 가진 기업간에 효율적인 통합이 가능토록 MOM(Message-Oriented Middleware)을 기반으로 한 비동기식의 유연한 결합(Asynchronous & Loosely-Coupled)방식인 EAI(Enterprise Application Integration)나 웹 서비스(Web Services)로 발전하고 있다. 하나의 프로세스는 기업의 목적달성을 위한 하나의 장기 트랜잭션으로 간주될 수 있다. 동기식 결합 보다는 비동기식 결합 방식에서 트랜잭션의 효율적인 관리가 더욱 중요한 데, 본 고에서는 기존의 2-Phase Commit(2PC) 방식보다 트랜잭션의 효과적인 모니터링과 처리자원 낭비를 감소시킬 수 있는 방법으로써 "Classify Phase"를 추가한 3PC Protocol를 제시하고 기존의 처리 방식과 비교하여 개선된 부분을 확인해본다.

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
분산 트랜잭션, 비즈니스 트랜잭션, 데이터베이스, Workflow, Web Service

1. 서 론

기업의 시장 환경은 시간이 갈수록 더욱 경쟁적으로 바뀌어 가고 있고 고객의 요구가 다양화 됨에 따라 얼마나 빨리, 그리고 얼마나 효율적으로 고객의 요구를 만족시켜 줄 수 있느냐가 과거나 현재나 변함없는 기업의 경쟁력을가름하는 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 대처할수 있도록 모든 기업 환경을 변화시키고 업무의 프로세스를 자동화시키는 것이다. 업무 프로세스의 자동화를 위해서는 프로세스 내의 기능별 응용 체계간에 필요한 자료의 흐름과 업무처리순서를 자동화해야 하며 기업의 응용 체계들간의 통합이 이루어져야 한다. 현재, 업무 프로세스 자동화 관리와 관련하여각종 표준을 주도적으로 제정하고 있는 기구인 WfMC(Workflow Management Coalition)에서 제시한 참조모델(Reference Model)을 만족하는다양한 WFMS(Workflow Management System) 제품들이 기업 프로세스 자동화에 기여하고 있다. [1], [3], [4], [5]

WFMS 기반으로 기업의 응용체계들을 통합할 때 새로 개발되는 응용 체계들에 대해서는 WfMC RM 인터페이스 2 와 3 에서 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 많이발생하게 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], [5], [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), 그 것을 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 EAI 와 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 환경 하에서는 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 구성된2-Phase Commit Protocol을 기반으로 비즈니스 트랜잭션을 처리할 경우 기업의 자원이 많이 소요된 트랜잭션들을 모두 취소하는 경우가 빈번하게 발생할 수 있고 전체적인 트랜잭션 소요시간을 줄일수 있는 방안을 제시할 수 없기 때문에 기업 자원의 효율성이 떨어지게 된다. 그 이유는 자원이 많이 소요된 트랜잭션과 적게 소요된 트랜잭션을 구분하는 단계가없고 트랜잭션의 소요시간을 통계적으로 확인할 수 있는 단계가 포함되어 있지 않기 때문이다. 따라서, 이러한 문제를 해

결하기 위해서는 트랜잭션에 가중치를 부여하여 트랜잭션을 분류할 수 있는 단계를 추가하여 준비 단계와 수용 단계 사이에 분류 단계(Classify Phase)를 추가하여 수용 단계에서 트랜잭션을 끝내기 전에 효율적인 트랜잭션을 처리할 수 있도록 하는 것이다.

이와 같이 제안한 방식의 이름을 "분류 단계를 포함한 3-Phase Commit Protocol"이라 명명하였으며 본 논문에서는 이 모델의 성능을 확인하기 위하여기존의 분산 트랜잭션 처리방식인 2-Phase Commit 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. 분산 환경과 트랜잭션

업무 프로세스의 자동화가 이루어지기 위해서 응용 체계들을 통합해야 하는 이유에 대해서는 앞서 기술하였다. 이러한 이유들 때문에 응용 체계들을 통합하고 나면 그 통합된 환경은 여러 처리 단위들이 지역적으로 분산되어 있는 분산 환경이 된다. 이와 같은 분산 환경에서 업무 프로세스 처리 과정으로써 트랜잭션을 처리하기 위해서는 분산 환경에서 발생하는 트랜잭션의 속성을 먼저 정리해야 한다. 따라서, 본 장에서는 응용 체계를 통합하는 방법으로써 EAI(Enterprise Application Integration)와 Web Service를 살펴보고, 업무 프로세스의 처리로 이루어진 트랜잭션으로써 ATM(Advanced Transaction Model)의 속성을 갖는 비즈니스 트랜잭션의 특징과 분산 환경에서의 트랜잭션 처리 모델인 2-Phase Commit에 대하여 기술한다.

2.1 EAI(Enterprise Application Integration)

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, 상대 응용체계를 직접 호출할 필요가 없고, 따라서 상대 응용체계 내부를 알 필요도 없으며 결과적으로 기존체계들을 통합하는 시간과 비용을 크게 단축할 수 있는 통합기술이다. 가트너 그룹에 의하면 EAI를 이용한 통합이 기존의 통합 방법과 비교해볼 때 전체 통합 비용을 약 1/3 정도 감소시키고 추후의 보수유지 단계까지 고려하면 총비용의2/3 까지도 절감시키는 것으로 보고되고 있다

프로세스 통합을 위해서는 WFMS가, 기업 정보시스템의 통합을 위해서는 EAI가 핵심적인 역할을 수행하고 있다. 기업의 경쟁적인 우위는 인터넷 상에서 관련 기업들이 서로 연결되어 하나의 가상적인 기업으로 운영될 수 있을 때 진정으로 확보될 수 있다. 그러기 위해서는 기업 내부 시스템 모두가 하나의 시스템으로 통합되어 있지 않으면 다른 기업과 연결될 수가 없다. 이처럼 기업정보시스템 통합을 위해서 WFMS와 EAI는 필수적인 소프트웨어 플랫폼이며 따라서 지금은 그 두개의 미들웨어가 하나의 시스템으로 통합되고 있는 추세다.

기업 경영의 핵심은 고객의 요구에 따라 수시로 변하는 시장 상황을 비즈니스 프로세스에 제때에 반영하여 고객이 원하는 상품(서비스)를 시간적으로 더 빨리 고객에게 제공함으로써 타기업보다 보다 나은 경쟁 우위를 확보하는 것으로 요약될 수 있다. 이 것은 비즈니스 프로세스를 얼마나 효율적으로 관리하느냐는 프로세스 자동화 측면 과 기업 내에서 운용되고 있는 각종 응용들을 비즈니스 프로세스 중심으로 얼마나 효과적으로 통합하느냐는 기업응용 통합 측면의 두 가지 관점으로 귀결될 수 있다.

프로세스를 자동화하는 관점에서는 프로세스 모델링 도구를 사용하여 기업의 비즈니스 프로세스를 가장 효율적으로 정의함(BPR)으로써 불필요한 업무 처리 절차를 줄이고, 실질적인 기업 활동과 관련하여서는 업무활동(activity)들의 처리 순서와 프로세스내의 응용들 사이에 필요한 자료의 흐름을 자동적으로 처리되게 함으로써 업무 처리 시간을 크게 단축시키는데 기여한다. 프로세스 중심으로 기업응용을 통합하는 측면에서는 워크플로우 표준기구인 WfMC가 구현모델 및 참조모델을 통하여 표준화된 인터페이스를 제시함으로써 워크플로우 제품간 프로세스 명세서(정의)를 상호 공유하면서 앞 절에서 정의된 4가지 방식의 연동 시나리오 모습을 사용하여 기업간 프로세스들을 연계시키는 워크플로우 기반의 기업 응용체계 통합 방법을 제공하고 있다. 그러나, 분산환경에서 워크플로우 기반의 응용 통합을 위한 통신 미들웨어로써는 COM, CORBA등의 표준들이 플랫폼으로 활용되고 있는데, 이런 표준들 사이의 차이점으로 인하여 그들간 호환성 있는 제품 구현이 힘들고, CORBA표준을 적용하여 구현된 ORB 제품 사이에도 호환성이 완벽하게 이루어지지 못하고 있다. 이런 기술이나 표준들은 대부분 자체 표준의 API를 이용한 동기식의강한 결합(Synchronous & Tightly-Coupled) 방식을 사용함으로써 같은 표준을 사용하지 않는 플랫폼간에는 호환성을 제공하기가 어렵다.

COM, CORBA, EJB 등에서 제공하는CBD(Component-Based Development) 방식은 새로운 응용들을 개발하는 데는 좋은 방식이지만기존 응용들을 CBD방법 중 어느 하나의 기술이나 표준으로 모두 통합하는 데는 기존 응용들에 대한 대체 개발 또는 변경 소요가 너무 크고 그 결과에 대한 성공을 보장하기에는 위험도가 높은 편이다. 또한 장차 새롭게 출현되는 미래의보다 나은 기술을 수용하기도 곤란할 것이다.

실제로 기업들이 운용하고 있는 응용들의 종류를 살펴보면 패키지 형태로 도입되는 시스템들인 ERP, CRM, SCM등이 있고, 옛 기술에 의해 오래

전에 개발되어 지금까지 핵심적인 비즈니스 프로세스 내에서 활용되고 있는 구 시스템(legacy systems)들도 아직 존재하고 있으며, 최근에는 CBD(Component-Based Development) 방식으로 개발된 시스템까지 다양한 기술과 표준들로 뒤섞여 있음을 알 수 있다. 하물며 기업간의 운용환경을 고려하면 기업내의 통합보다 더욱 더 상이한 기반체계와 기술, 표준들을 고려하지 않으면 안될 것이다. 즉, 기업들마다 응용 개발 방법이나 운용하는 기반체계에 적용되는 정보기술이 대부분 다르고 기업 단위로 독립적으로 기업정보체계를 발전시켜 나가고 있기 때문에 현실적으로 어느 하나의 기술이나 표준으로 기업 또는 기업간의 모든 응용체계 통합을 달성하기가 불가능한 것이 사실이다. 그리고, B2B, SCM, e-Business등으로 발전되고 있는 최근의 경향으로 볼 때 기업의 응용들을 통합하려는 요구는 날로 증가하고 있고, 기업마다 가지고 있는 다양한 표준이나 기술로 개발된 응용들을 통합하는데 소요되는 비용이나 시간은 그 복잡성으로 인하여 더욱 더 증가하고 있다.

이처럼 다양한 플랫폼의 분산 컴퓨팅 환경하에서 이 기종의 기술과 표준으로 구성된 응용들을 신속하게, 효율적으로 통합하려는 기업의 요구를 만족시켜 주기 위하여 제시된 기업정보체계 통합 방법이 EAI이다. 기업마다 독자적인 정보기술체계 및 표준들로 정보체계를 개발 운용하고 있는 현실을 감안해 볼 때, EAI는 기업에서 기존에 개발하여 운용 중인 정보자원들에 대한 투자를 최대한 보호하면서 또한 미래의 변화도 충분히 수용할 수 있는 유연성을 제공하는 통합기술이다.

EAI는 호환성이 없는 상이한 기술을 사용하여 독자적으로 개발되고, 독립적으로 관리되고 있는 응용들을 통합하는 절차와 기술을 의미한다. 위에서 본 것처럼 이기종의 다양한 플랫폼의 분산 컴퓨팅 환경하의 기업정보체계를 통합하기 위해서는 어느 하나의 기술이나 표준에의존하기 보다는 기존의 각종 기술이나 표준으로 개발된 시스템들을 다 아우르는 방법이 필요하다. 따라서, COM이나 CORBA등에서 제공하는 API를 이용하여 동기식으로 상대 응용을 직접적으로 호출하는 방법보다는 비동기식의 통신을 지원하는 MOM(Message -Oriented Middleware)을 기반으로 하여 응용간 메시지를 교환하는 간접적인 통합 방식이 EAI의 핵심이다. MOM 제품들은 분산 컴퓨팅 환경에서 동기식 방식과 달리 비동기식 방식으로 메시지의 전달이 보장될 수 있도록 특별히 설계되어 있다. 상대 응용을 직접 호출하지 않기 때문에 그 내부를 알 필요 없고 따라서 기존 응용의 수정을 별로 필요로 하지 않는 것이 EAI 기술의 주요 특징이다.

EAI 구조는 기본적으로 비즈니스 프로세스 수준에서 응용간 통합을 담당하는 통합 브로커(Integration Broker)와 응용간 비동기식의 실시간 메시지 교환을 보장하는메시지 브로커로서의 MOM, 응용과 통합 브로커간 연결 역할을 하면서 메시지에 포함된 데이터를 응용에게 필요한 형태로 변환하는 등 응용의 I/O를 담당하는 어댑터(Adapter)로 구성된다.

통합 브로커는 응용 내부에서 사용되는 데이터 형태를 그래픽 도구를 사용하여 공통 데이터 형식으로변환시킨다. 공통 데이터를 사용함으로써 어떤 하나의 응용 내부에 만들어진 변화가 통합체계 내의 타 응용에 영향을 미치지 않으며, 메시지가 해당 응용에게 전달될 수 있도록 사전에 정해진 절차에 따라 정의함으로써 지능적인 메시지 중개 기능도 제공한다. 통합 브로커는 CORBA, COM, EJB 등의 CBD방식을 지원하고, IBM의 MQSeries, MS의MSMQ, J2EE의 JMS등 MOM표준을 지원할 수 있어야 한다.

MOM은 분산 환경에서 통합 브로커와 응용 사이에 통신 수단을 제공하여 비동기식이지만 신뢰성 있는 메시지 전달을 보장하기 때문에 EAI에 아주 중요한 역할을 담당한다. 네트워크 구현에 따른 복잡성을 덜어주기 위하여대개의 MOM은 개발자를 위한 API를 제공한다.

어댑터는 응용과 통합 브로커 사이에 형성된 연결을 다루며 데이터의 I/O교환을 통하여 응용에 사용되는 다양한 기술과 표준 사이에서 다리 역할을 한다. 응용은 어댑터를 통하여 통합 브로커와 통신을 하며 통합 브로커도 응용과 통신을 할 때는 반드시 해당 응용의 어댑터를 통한다. 어느 하나의 응용에서 사용되는 기술과 표준이 다른 응용에 꼭 사용될 필요가 없으며 어느 하나의 응용이 수정될 때마다 다른 모든 응용들이 따라서 수정될 필요가 없게 되었다. 응용들은 어댑터와 통합 브로커를 통하여 느슨한 형태로 결합이 이루어 진다.

대부분의 EAI제품은 프로세스 자동화 관리 도구를 함께 제공하고 있다. 따라서, 응용간 데이터의 통합뿐만 아니라 기업간의 비즈니스 프로세스 통합 능력도 함께 제공할 수 있는 기능을 가지고 있다.

2.2 Web Service

EAI가 통합에 소용되는 시간과 비용을 상당히 감소시켰다면 이를 온라인 실시간 통합이 가능하도록 하는시도가 웹 서비스(Web Services) 형태의 통합이다. 웹 서비스란 개별 응용체계가하나의 서비스 단위로서 인터넷에 연결되어 누구나 마치 자기 회사의 응용체계처럼 사용될 수 있도록 지원함으로써 필요한 정보를 제공하는 통합체계이다. EAI기술을 이용한 통합 방법은 어댑터나 데이터변환 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), 그 것을 일정한 인터넷 장소에 등록하여 필요한 사용자가찾아볼 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 없도

록 표준 데이터 형식을 사용토록 해야 하는 것(XML) 들이 웹 서비스를 가능케하는 표준들이다.

앞에서도 언급한 것처럼 CRM, SCM 등B2B형태로 e-Business가 발전함에 따라 이런 웹 서비스 기술은 점차적으로 확대되어 갈 것이다. 기업마다 다양하고 서로 다른 하드웨어 및 소프트웨어 플랫폼에서 기업정보체계(Enterprise Information System)를 운용하고 있는 상황에서 이를 극복할 수 있는 방법은 웹 서비스가 현재로서는 가장 적절한 대안으로 보인다. 웹 서비스 기술의 이점은 소프트웨어 재사용성을 더 한층 향상시킬 수 있고 필요한 서비스를 실시간으로 활용할 수 있는 측면에서도 바람직하지만, 가장 큰 매력은 개발된 웹 서비스가 어떤 프로그래밍언어를 사용했건, 어떤 플랫폼에서 개발 되었건, 어떤 회사에서 개발된 제품이건 상관없이 누구나 그 서비스를 활용할 수 있다는 것이다. [6], [7]

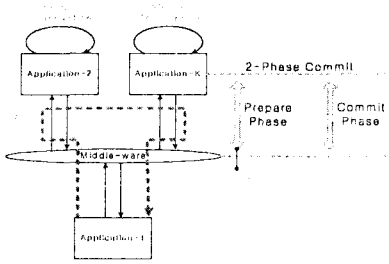
위에서 기술한 것처럼EAI는 다양한 기술과 표준을 사용하여 개발된 응용들뿐만 아니라 미래의 새로운 기술도 수용할 수 있도록 지원하고 있다. 그러한 융통성은 분산 이기종의 다양한 플랫폼에서 운용되고 있는 기존의 핵심적인 기업 응용을 크게 수정하지 않고 그대로 활용하게 함으로써 기업의 응용들을 통합하는데 소요되는 시간과 비용을 획기적으로 줄이고 있다. 그러나, EAI는 다양한 표준과 기술을 다 수용할 수 있도록 융통성이 많은 것만큼 통합을 이루기 위해서는 개별 응용에 대한 어댑터의 구현, 자료형식 변환 등 EAI도구에 맞도록 다소의 수정 소요도 발생하는 것이 사실이다.

웹 서비스는 통합에 필요한 요소를 사전에 표준화하여 정의함으로써 표준화를 만족하는 응용에 대해서는 통합을 위한 별도의 수정이 필요 없도록 하여 통합에 투자되는 시간과 노력을 더욱 더 최소화하고 필요한 응용을 곧 바로 체계 내에 포함시켜 활용할 수 있도록 함으로써 실시간으로 통합을 이루고자 하는 노력의 일환으로 만들어진 기술이다. 웹 서비스 기술은 기업 안팎의 응용 통합을 가속화시키는 수단으로서, 프로그램 언어 중립적이고 운용 환경 중립적인 통합 모델이다. 웹 서비스를 통한 응용 통합은 융통성 있으면서 유연한 결합 방식(loosely-coupled)의 비즈니스 통합 체계를 만들어 내고 있다. [8], [10]

웹 서비스는 네트워크 상에서실시간으로 정의되고, 공시되고, 저장되고, 호출될 수 있는 독립적으로 작동하는 규격화된 콤포넌트이다. 웹 서비스는 단순한 요구 기능에서부터 복잡한 비즈니스 프로세스까지를 망라한 어떤 기능도 수행한다. 웹 서비스가 개발되어 설치되면 기존의 다른 응용들은 설치된 웹 서비스를 바로 찾아서 호출할 수 있다. XML메시지가 웹 서비스 사이의 연결을 위해 사용된다. 웹 서비스는 현재 활성화되어 잘 사용되고 있는 인터넷 표준을 기반으로 하여 웹 서비스를 위해 새롭게 정의된 통신 연결 표준을 적용하는 일종의 응용이다. 웹 서비스는 CBD방식을 이용하여 미리 만들어진 콤포넌트를 기업의 필요에 따라 조합하여 신속하게 비즈니스 수행 방안을 제시함으로써 인터넷 상에서 실시간으로 소프트웨어 재사용성을 제고시키는 방법을 제공한다.

2.3 2-Phase Commit Protocol

분산 트랜잭션을 처리하기 위하여 Local System에서 트랜잭션 처리를 위한 준비상태를 점검한 후 전체 트랜잭션의 처리 완료를 수행하는 방식을 Two-Phase Commit Protocol이라 한다.



[그림1] Two-Phase Commit Protocol의 구성

Two-Phase Commit은 Prepare Phase와 Commit Phase로 구성된다. [그림 1]에서 확인할 수 있듯이 분산 트랜잭션을 발생시킨 Application-1은 Atomic Transaction을 2-phase상태로 확인하면서 트랜잭션을 처리한다.

Prepare Phase에서 Coordinator는 트랜잭션에 참가한 Application들이 Commit이나 Rollback을 수행해도 되는가를 확인하는 단계이고, Commit Phase는 Prepare Phase의 상태를 확인하고 Commit이나 Rollback을 수행하는 단계이다.

Prepare Phase는 모든 Local Transaction이 Commit 직전단계에 이르렀다는 Prepared 응답과 Local Transaction에 의하여 아무러변화가 없다는 Read-Only 응답, 그리고 Prepared되지 않았다는 Abort 응답 등으로 구성되어 있다.

Prepared 응답을 받으면 Commit phase에서는 Commit을 수행하고 Abort 응답을 받으면 Abort되어 모든 Local Transaction이 Rollback 된다.

2-Phase Commit 방식이 처리가 되기 위한 전제는 2-Phase Locking이 가능하다는 것이다. 모든 Local Transaction을 관리하는 Global Coordinator는 Prepare Phase에서2-Phase Locking을 동작시킨다. 따라서 전통적인 2-Phase Commit은 항상 2-Phase Locking Protocol을 포함하고 있다.

2.4 비즈니스 트랜잭션과 ATM(Advanced Transaction Model)

Business Transaction이란 "Business Function들로 구성된 Business의 진행 상태를 일관성 있게 변화시키는 것"을 의미한다. 이러한 Business Transaction들은 전체적으로 자동화될 수 있고 일부분만 자동화될 수 있다. Business Process는 여러 개의Business Transaction들로 구성된다. Business Transaction의 예로서 '주문 처리'를 들수 있다. 어

떤 회사로부터 주문을 받는 과정은 잘 정의될 수 있다. Business가 진행되고 있는상태에서 주문 하나가 완료되면 Business는 일관성 있는 변화가 이루어진다. 즉, 주문과 관련된 Database가 Update되고 하나의 구매주문서가 추가된다. 결국, Business Transaction은Business가 그 목적을 달성하기 위해서 일관성 있는 방향으로 변화하는 과정이라고 할 수 있다

ATM이란 과거의 트랜잭션 모델의 속성인 ACID(atomicity, consistency, isolation, Durability)에 수정과 트랜잭션 간에 상호 작용을 통하여 트랜잭션의 효율성을 극대화시키는 트랜잭션 모델이다. 하나의 트랜잭션이 수분 내지는 몇 일까지 걸려 결과를 도출하는 형태의 기능을 하고 있는 장기 트랜잭션(Long Term Transaction)이 필연적으로 발생하고 있는 실정에서 과거의 트랜잭션 모델의 경우 전체 성능의 저하를 발생시키는 주원인이 되고 있다. 또한 하나의 트랜잭션이 Un-Commit된 상태가 발생하여 실패한 경우 복귀(Rollback)에 대한 많은부하를 발생시키고 시간적 낭비를 초래할 수 있다. 그리고 트랜잭션의 속성에 의하여 트랜잭션들 사이에는 Message나 Control을 제공하지 않기 때문에 Deadlock에 빠질 위험이 커진다. 따라서 트랜잭션의 전통적인 모델을 개선한 진보한 트랜잭션 모델이 필요하게 되었고 이를 위해서 ATM을 정의하게 되었다.

ATM은 다음과 같은 속성을 만족한다.

(1) 느슨한 원자성(Relaxed atomicity)

전통적인 Transaction에서는 All or Nothing의 개념이었으나 부분적인 Rollback의 인정과 중요한 부분만을 Undo하는 융통성이 필요하게 된다. Transaction1이 만든 산출물을 Transaction2에서 사용을 하였으나 그 결과가 Transaction2의 일관성에 영향을 미치지 않는다면 Transaction1이 Uncommit되고 fail된 경우 Transaction2의 결과까지 Rollback 할 필요는 없는 것이다.

(2) 느슨한 고립성(Relaxed isolation)

자원의 공유와 트랜잭션 간의 메시지 통신 및 서로에 대한 컨트롤이 가능해져야 한다.

(3) 범위성(Scalability)

단일 트랜잭션(Single Transaction)이나 다계층 트랜잭션(Multi-level Transaction) 모두에게 적합한 환경이 되어야 한다.

(4) 일시적 데이터의 관리(Temporal data management)

데이터공유나 환경을 공유함으로써 생기는 지속적인 변화부분을 관리하여야 한다. 즉 Interrupted 되어도 전에 수행된 환경을 계속 수행할 수 있어야 한다.

(5) 접근제어(Access control)

데이터나 기능에 대한 접근을 제어할 필요가 생긴다.

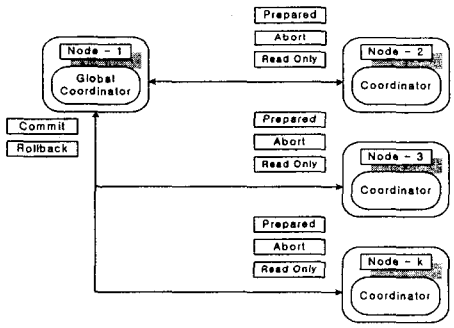
(6) 시간제어(Time requirement)

하나의 트랜잭션이 장시간 동안 시스템을 점유하지 못하게 할 필요가 생긴다. 특히 실시간 시스템(Real-Time System)에소 중요하다.

ATM은 위의 모델을 통합함으로써 상호 응용 프로그램(Cooperative Application)의 작업수행 능력을 극대화하는 논리적 모델이고 이 모델은 주로 CovaTM이라는 모델을 통해 물리적모델로 구현되어지고 있다. [11], [12], [13]

3. 2-Phase Commit 기반의 트랜잭션 처리 모델의 문제점

EAI와WS를 기반으로 한 통합방식에서 2-Phase Commit Protocol을 적용하여 트랜잭션을 처리하는 방법에 대한 연구가 다양하게 이루어지고 있다. HP, IBM, Oracle, Sun은 공동작업을 통해서 XAML(Transaction Authority Markup Language)라는 표준안을 제안했으며, OASIS는 BTP(Business Transaction Protocol)라는 트랜잭션 표준안을 제안하였고 IBM에서는 독자적으로WS Coordination과 WS Transaction을제안하였다. 이러한 표준 모델들의 공통적인 특징은 [그림 2]와 같다.



[그림 2] 유연한 결합에서 기존 모델들의 특성

기존 모델들은 다음과 같은 문제점들을 갖고 있다.

(1) 적은 자원이 소요되는 부분트랜잭션이 Abort되면 상대적으로 많은 자원을 소모했던 다른 부분 트랜잭션까지 Rollback되는 경우가 있다. 이럴 경우 전체적인 자원의 낭비가이루어 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) 회소가치가 있는 트랜잭션은 Prepared 되었으나 상대적으로 적은 회소가치를 갖는 트랜

잭션이 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.

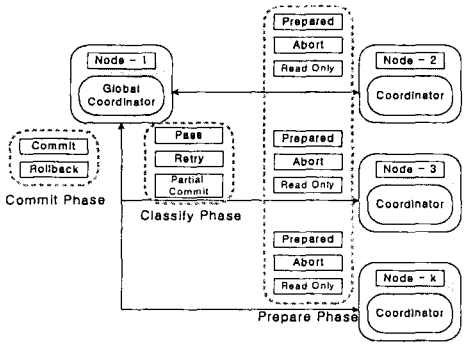
(5) 관리자는 다양한 부분 트랜잭션들의 자원 소요 비중을 Monitoring할 수 없기 때문에 트랜잭션의 효율적인 관리가 이루어지기 어렵다.

이와 같은 문제들 때문에 2-Phase Commit을 기반으로 하는 기존 모델들을 이용하여 유연한 결합 환경에서 트랜잭션을 효율적으로 처리하는데 한계가 있다.

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

유연한 결합과 같은 분산 환경에서 트랜잭션을 효율적으로 처리하기 위한 방식으로서 Classify Phase를 적용한 3-Phase Commit Protocol을 제안한다. 이 모델은 분산 환경에 적합한 트랜잭션 처리 방식인 2-Phase Commit Protocol을 보완한 방식이며, EAI와 Web Service환경에서 발생하는 Business Transaction 을 효율적으로 처리하기 위한 방식들인 WS-Coordination과 WS-Transaction 그리고 BTP 방식을 보안한 방식이다. 그리고 트랜잭션을 처리하는 자원의 비효율성을 최소화하는 것이 목적을 하고 있다.

유연한 결합 환경으로서 대표적인 Business System인 EAI와 Web Service방식은 메시지 통신 방식을 사용하는데, 이러한 메시지(혹은 문서) 내에 트랜잭션의 가중치를 포함하도록 프로토콜을 구성했으며 트랜잭션의 가중치를 계산하고 트랜잭션의 유형을 분류하며 트랜잭션의 가중치를 정의할 수 있도록 하였다. 트랜잭션은 자동과 수동, 두 방식 모두를 사용할 수 있으며 관리가 용이하도록 트랜잭션 모니터링 기능이 가능하도록 하였다. 또한 Multi-level Transaction에 대하여 능동적인 대처 방안을 제시할 수 있으며 고비용의 Local Transaction(가중치가 높은 Local Transaction)이 무조건 Abort되는 것을 막기 위해서 Classify Phase를 적용하였다.

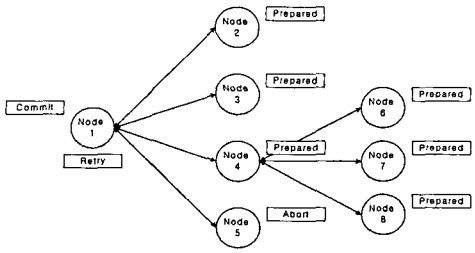


[그림 3] 3-phase commit 개념도

3-phase commit의 처리 과정은 Prepare Phase, Classify Phase, Commit Phase 로서 [그림 3]과 같다. Prepare Phase는 2-Phase Commit 방식과 일치하고 Global Coordinator가 나머지 모든 노드에게 Prepare 하도록 요청한다. Prepare Phase는 Prepared, Aborted, Read-Only와 같은 세 가지의 결과를 갖으며 2-Phase Commit 방식과 일치한다.

Classify Phase는 전역 트랜잭션을 분류하는 역할을 하며 Passed, Retry, Partial commit의 결과를 갖는다. Passed는 부분 트랜잭션들 모두 Prepared 되었을때 발생하는 결과이며, Retry는 가중치가 높은 부분 트랜잭션은 Prepared 되었으나 가중치가 낮은 부분 트랜잭션Aborted 되었을 경우 이 부분 트랜잭션만 다시 Prepare 단계를 수행할 때 나타나는 결과이다. Aborted 된 경우 Partial Commit은 가중치는 높지만 시간이 지날 경우 자동으로 Rollback이될 수 있는 경우에 발생하는 결과이다.

Commit Phase는 ATM이 갖고 있는 속성을 갖는다. Relaxed Atomicity, Relaxed Isolation, Time Control 기능을 갖을 수 있도록 한다.



[그림 4] 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.

5. 결 론

비동기적 통합(Asynchronous Integration)방식을 이용하여 기업의 응용 프로그램들을 통합한 후에는 비즈니스 프로세스를 자동으로 처리하기 위한 트랜잭션 관리가 필요하다. 본 논문에서는 기업의 응용 프로그램들을 통합할 수 있는 EAI와 Web Service 환경과 같은 유연한 결합(Loosely Coupled)의 분산 환경에서 효율적으로 트랜잭션을 처리하기 위한 모델을 제안하였다. 이 모델은 분산 환경에 적합한 2-Phase Commit 방식에 Classify Phase를 적용한 3-Phase Commit 방식으로써 유연한 결합 환경에서 트랜잭션을 효과적으로 관리하고 트랜잭션 처리 자원을 절약할 수 있게 하였다. 기존의 분산 트랜잭션 처리 방식인 2-Phase Commit 기반의 트랜잭션 처리 모델의 문제점으로써 트랜잭션을 분류하지 않기 때문에 트랜잭션을 처리하는 자원들을 낭비할 수 있는 부분과 트랜잭션의 정상적인 처리율이 낮아질 수 점들을 제시하고 이를 해결하기 위한 방법으로써 트랜잭션에 가중치를 부여하여 분류할 수 있는 Phase로써 Classify Phase를 포함한 3-Phase Commit Protocol을 제시하고 개선된 부분들을 확인하기 위하여 기존의 처리 방식과 비교하였다.

참고 문헌

[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,* Vol. 27, No. 1; North Holland Elsevier, 1998; pp. 31-57.

[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, April, 1995, pp. 119-153.

[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 developerWorks

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

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

[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, June 2002. http://ftpna2.bea.com/pub/downloads/wsci-spec-10.pdf

[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-wsº|-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):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.