B2B 전자 상거래 환경에서 기업 사이의 적기 협력 지원을 위한 능동 기능 컴포넌트

(An Active Functionality Component to Support Timely Collaboration among Businesses in B2B EC Environment)

이 동 우 † 이 성 훈 †† 황 종 선 †††
(Dong Woo Lee) (Seong Hoon Lee) (Chong Sun Hwang)

요 약 B2B 전자 상거래 환경에서 기업들은 긴밀한 협력을 필요로 한다. 특히 기업 사이의 긴급 협조 요청이나 긴급 정보는 즉시 처리 방식으로 협력하여야 한다. 그러나 대부분의 시스템들은 시스템의 독립성과 보안을 위한 방화벽 때문에 이를 적절히 다루지 못하고 있고 *ad hoc* 방식으로 처리하고 있다.

본 논문에서는 B2B EC 환경에서 기업들의 적기 협력 방안과 이를 지원하는 능동 기능 컴포넌트를 제안한다. 능동 기능 컴포넌트는 고수준의 ECA 규칙 패턴과 사건 기반의 즉시 처리 방식을 지원하므로 시스템 관리자나 프로그래머가 응용 로직과는 별도로 즉시처리 협력을 쉽게 구축하고 유지 보수할 수 있다. 제안하는 능동 기능 컴포넌트는 기업의 방화벽을 통해서도 적용될 수 있도록 HTTP 프로토콜을 사용하였고, 실용성을 위하여 상업용 DBMS를 이용하여 설계하였다.

키워드 : B2B 전자 상거래, 적기 협력, ECA rule, 방화벽

Abstract Close collaboration among businesses is required in B2B EC environment. Furthermore, emergency requests or critical information among businesses should be processed in an immediate mode. Most current systems, however, due to firewalls for the systems' security and autonomy, can not handle these requirements appropriately, but handle them in an *ad hoc* manner.

In this paper a method of timely collaboration among businesses and an active functionality component to support it in B2B EC environment are proposed. Since the active functionality component supports high level ECA rule patterns and event-based immediate processing, system administrators and programmers can easily program and maintain the timely collaboration independently to the application logic. The proposed active functionality component uses HTTP protocol to be applied through firewalls and is designed using a commercial DBMS for practical purpose.

Key words : B2B EC, Timely Collaboration, ECA Rule, Firewall

1. 서 론

EDI(전자문서교환)로 시작된 전자상거래 시스템은 인터넷의 확산, WWW의 등장, 그리고 분산 시스템 기술의 발달로 변화를 거듭하고 있다. 특히 최근에는 기업들의 경쟁력 제고를 위한 노력과 인터넷의 확산, 지속적인 최신 기술의 등장 및 표준화 노력, 등으로 기업 사이의 B2B 전자 상거래 시스템의 사용은 더욱 확대될 것으로 예상되고 있다[1].

이러한 B2B 전자 상거래 환경에서 기업들은 공동의 사업 목적을 달성하기 위해서 서로 긴밀히 협력하여야 한다. 특히 기업사이의 긴급 협조 요청이나 긴급 메시지에 대해서는 즉시처리 협력을 하여야 한다. 예를 들면, 그림 1의 쇼핑 몰에서 어느 제품이 갑자기 재고가 없을 경우 협력 업체(공급자)에 긴급 공급 요청을 하여야 하고, 협력 업체(공급자)는 요청된 제품을 약속된 기일 이내에 즉시 공급하여야 한다. 만약 협력 업체가 이를 제때에 공급할 수 없다면, 즉시 이 사실을 알려주어 쇼핑 몰이 다른 공급업체를 찾도록 해야 한다. 그러나 현재 대부분의 시스템들은 이를 적절히 다루지 못하고 있고 주로 ad hoc 방식으로 해결하고 있다.

현재 각 기업들은 자신들의 시스템을 독립성과 안정

† 정 회 원 : 우송대학교 컴퓨터학과 교수
dwlee@woosong.ac.kr
†† 정 회 원 : 천안대학교 정보통신공학부 교수
shlee@cheonan.ac.kr
††† 종신회원 : 고려대학교 컴퓨터학과 교수
hwang@disys.korea.ac.kr
논문접수 : 2004년 7월 14일
심사완료 : 2004년 11월 10일

성 및 보안을 위하여 쉽게 외부에 개방하고 있지 않으며, 대부분 방화벽을 설치하고 있다. 따라서 기업간 협력 기능의 구현은 주로 허용된 사용자에 의한 로그인 방식과 Email 또는 EDI 시스템을 이용한 일괄 처리 방식(batch-processing)으로 이루어지고 있고, 업무 협조가 신속히 이루어져야 할 경우에도 이를 즉시 처리 방식으로 구현하고 있지 않거나, ad hoc 형태의 저 수준 프로그램으로 이를 구현하고 있다[2]. 따라서 기업사이의 즉시 처리 협력 내용이 응용 로직에 코드화 되어 시스템의 모듈성이 확보되지 않는다. 각 기업들의 정책이나 전략 그리고 사업적 제약조건이나 규약은 자주 변할 수 있는데, 기업들의 이러한 새로운 변화에 따라 시스템을 유지 보수할 경우, 응용 로직을 중단하고 프로그램을 재 컴파일 하는 등, 과다한 비용이 들게 된다. 따라서 기업간 업무 협조 기능을 업무 처리 기능과 분리하여 고 수준의 프로그램으로 구현하는 것이 시스템의 모듈성, 재사용성 등 효율성을 기할 수 있다[3].

본 논문에서는 B2B 전자상거래 환경에서 기업들 사이의 적기 협력 방안과 이를 지원하는 능동 기능 컴포넌트를 제안한다. 제안된 능동 기능 컴포넌트는 고수준의 ECA(Event-Condition-Action) 규칙 프로그래밍 방식과 사건 기반의 즉시 처리 방식을 제공하므로, 응용 로직에 독립적으로 기업사이의 협력 기능을 즉시 처리 방식으로 구현할 수 있다. 즉, 협력 업체 사이에서 즉시 처리 방식을 필요로 하는 상황을 ECA 규칙에 따라, 상황의 발생을 사건(Event)으로, 그리고 이 사건이 즉시 처리를 필요로 하는지 여부를 조건(Condition)으로, 그리고 이에 대한 조치를 행위(Action)로 표현하면, 능동 기능 컴포넌트가 사건의 발생을 자동으로 탐지하여, 이에 대한 조치를 취하는 협력 시스템에 사건 발생을 즉시 통보하고, 통지 받은 협력 시스템의 능동 기능 컴포넌트는 조건을 검사하여 해당 상황 여부를 판단하여 조치(협력 행위)를 취함으로써 업무 협조를 사람이나 응용 프로그램의 간섭 없이 즉시 처리 방식으로 실행할 수 있다. 따라서 시스템 관리자와 프로그래머는 적기 협력 기능 구현과 유지 보수를 쉽게 할 수 있다.

그리고 시스템들이 방화벽을 통해서도 상호 운용되도록, 사건 기반 메시지를 HTTP 프로토콜 위에서 전달되도록 구현하여 각 기업의 시스템의 독립성을 보장하고자 한다. 특히 제안된 능동 기능 컴포넌트는 실용성을 위하여 상업용 DBMS가 제공하고 있는 기본적인 트리거(trigger) 기능을 이용하여 구현할 수 있도록 설계하였다.

이 논문의 나머지는 다음과 같이 구성되어 있다 : 2장에서 B2B 전자상거래 시스템을 통한 기업 사이의 업무 협력 방식을 분석하여 ECA 규칙 시스템에 의한 적기 협력 방안을 제시하고, 3장에서는 적기 협력을 표현하는 ECA 규칙 구성요소와 프로그램 예를 다룬다. 4장에서는 ECA 규칙으로 표현된 적기 협력을 지원하는 능동 기능 컴포넌트의 아키텍처를 설명하고, 5장에서 이의 적용 예를 다룬다. 6장에서는 본 논문의 연구내용과 관련된 연구들을 검토하고, 끝으로 7장에서 요약과 앞으로의 연구 방향을 논한다.

2. B2B EC 환경에서 기업 사이의 적기 협력

2.1 B2B 전자상거래 시스템 환경

전자상거래 환경에서, 전자상거래 행위의 생명 주기(life cycle)는 4 단계로 구분할 수 있는데, 중개(matching), 흥정(negotiation), 계약 체결(contract formation), 계약 이행(contract fulfillment) 등이다. 본 연구에서는 이러한 전자상거래 행위의 생명 주기에서 계약을 체결하고 계약이행 단계의 실제 거래 상황만을 고려하기로 한다. 그림 1은 본 논문에서 적용할 전형적인 전자상거래의 사례로 [4]의 예를 사용한다. 그림 1에 있는 각 개체(기업 및 고객)의 행위를 간단히 살펴보면, 우선 실선은 모든 시스템이 연결되어 정보교환을 하기 위한 인터넷을 의미하고 점선 화살표는 상품의 이동을 의미한다. 각 기업은 독자적으로 B2B 및 B2C 시스템을 운영한다. 즉, 쇼핑몰(shopping mall)의 경우, B2C를 통하여 일반 고객의 주문을 받고 이에 관한 각종 서비스를 하며, B2B 시스템을 통하여 공급자(supplier)에게 제품 주문, 택배 업체(shipper)에게는 발송 의뢰, 은행에는 자금 이체 확인, 신용카드 회사에는 신용 및 거래 승인 확인 등, 계약 체결 단계에서 이미 정해진 업무 협약에 따라 거래에 관한 업무 요청과 협조를 한다. 그리고 공급자는 쇼핑몰의 주문 요청을 자체 B2B 시스템을 통하여 확인하고 서비스한다. 기타 개체들도 상호 정해진 거래 규칙에 따라 유사한 행위를 한다.

그림 1의 각 개체의 시스템은 각 조직이 시스템의 보안을 위하여 모두 방화벽을 사용하므로 이를 통하여 외

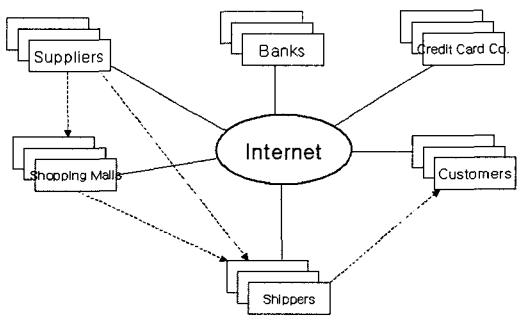

그림 1 전형적인 B2B 전자상거래 환경

부와 연결된다. 일반적으로 방화벽은 HTTP 80 포트와 같은 표준적인 것을 제외하고는 모든 포트의 사용을 차단한다. 따라서 동적으로 포트를 할당해야 하는 DCOM과 같은 분산 객체 프로토콜은 위와 같은 전자상거래 환경에서 사용하기가 어렵다[5]. 따라서 방화벽을 사용하는 위와 같은 환경에서도 각 기업의 시스템들이 상호작용이 가능하려면, HTTP 프로토콜 같은 방화벽에서 허용하는 프로토콜과 고정된 포트를 이용하여 상호 연동하여야 한다.

2.2 기업 사이의 업무 협력 방식과 적기 협력의 필요성

기업에서의 업무 처리 방식은 같은 종류의 일을 모아서 한꺼번에 일괄적으로 처리하는 방식(batch processing)과 일이 발생할 때마다 바로 처리하는 즉시 처리 방식(immediate processing)으로 구분할 수 있다. 일괄 처리 방식의 경우는 일하는 사람이나 시스템(또는 프로그램)의 효율성을 기할 수 있으나 각 일들의 처리 시간이 늦어지게 된다. 즉시 처리 방식의 경우 일하는 사람이나 시스템(프로그램)이 일에 대하여 항시 대기해야 하는 문제점이 있으나 각 일거리는 빠르게 처리된다. 이러한 업무 처리 방식의 선택은 업무의 특성이나 비용 등을 고려한 기업의 정책과 상호간의 계약(업무 협약)에 따라 달라진다. 기업들이 서로 업무 협력을 하기 위해서는 협조 요청과 협조 행위를 통한 상호 작용이 필요하다. 각 기업의 업무처리 방식에 따라 기업간 업무 협력 방식은 다음과 같이 4 가지 방식이 있을 수 있다: 일괄요청-일괄처리협조, 일괄요청-즉시처리협조, 즉시요청-일괄처리협조, 즉시요청-즉시처리협조. 이러한 여러 가지 방식에서 기업들이 업무 협조를 어떤 형태로 하느냐의 결정은 기업 내부의 정책을 고려한 상호간의 계약에 따라 달라진다.

기업사이의 적기 업무 협력(즉시요청-즉시처리협조)이 필요한 경우는 긴급 협조 요청과 이에 대한 협조, 긴급 정보의 전달 및 이에 대한 조치를 취해야 하는 경우로, 통상적인 업무 처리 과정에서 예외적인 상황으로 볼 수 있다. 이러한 예외적인 상황의 발생은 아주 빈번하지는 않지만, 기업과 고객의 이익에 크게 영향을 미치고 있다. 계약을 통한 계약이행으로 기업들이 상호 협력을 하므로 적기 방식의 협력을 필요로 하는 경우는 다음과 같이 분류할 수 있다:

- 약속된 협력 업무를 제대로 처리할 수 없는 경우(Unable-Service) : 협력 업체의 내부 사정으로 업무 협조를 제대로 수행할 수 없는 경우, 이 사실을 협조 서비스를 요청한 업체에 즉시 통보해 주어야 상대방 업체가 대책을 세울 수 있다. 예를 들면, 쇼핑몰 업체가 제품 공급 협력 업체에 어느 제품의 공급을 요청했을 경우, 제품 공급 업체가 이 제품의 생산이 중단되었거나 내부 사정으로 공급할 수 없다면, 이를 쇼핑몰에 즉시 통보해 주어야 쇼핑 몰은 다른 공급 업체에 해당 품목을 주문하여 확보할 수 있다.
- 약속된 협력 업무의 처리를 보완 또는 수정해야 하는 경우(Modify-Service) : 약속된 협력 업무 내용 중의 일부를 보완 또는 수정해야 하는 경우로, 예를 들면, 신용 카드 회사의 거래 은행이 거래 고객들의 신용카드 사용 금액 결제를 일괄 자동 이체하는데, 그중 어느 고객이 신용 카드 회사의 즉시 입금 방식으로 결제를 하게 되면, 2 중 결제를 방지하기 위해서는 이 고객의 자동 이체를 은행의 일괄 자동 이체 업무에서 제외 시켜야 한다.
- 약속된 협력 업무 처리를 취소해야 하는 경우(Cancel-Service) : 기업 내부의 사정에 의하여 요청했던 업무 협조를 취소해야 하는 경우로, 예를 들면, 쇼핑 몰의 고객이 주문을 취소할 경우, 이 제품에 대한 공급과 운송업무 협조를 취소하여야 한다.
- 약속된 일반 협력 업무 처리와 별도로 특수 처리를 해야 하는 경우(Special-Service) : 통상적인 업무 처리와 별도로 특별 서비스를 필요로 하는 경우로, 예를 들면, 쇼핑 몰에서 어느 고객이 추가 비용을 지불하더라도 제품의 긴급 구매를 원한다면, 이에 대한 제품의 우선 확보와 특송 서비스가 필요하다.

위와 같은 적기 협력이 필요한 경우에 대하여 현재 각 기업들은 시스템의 독립성과 안정성 및 보안을 위하여 쉽게 외부에 시스템을 개방하고 있지 않기 때문에, 주로 허용된 사용자에 의한 로그인 방식이나 Email로 이루어지고 있고, 또는 ad hoc 형태의 저 수준 프로그램으로 이를 구현하고 있다[2]. 따라서 기업사이의 즉시 처리 협력 내용이 응용 로직에 코드화 되어 시스템의 모듈성이 확보되지 않는 등 적절히 다루어지지 못하고 있다.

2.3 ECA 규칙 시스템에 의한 적기 협력

B2B 전자상거래 환경에서 기업사이의 적기 협력을 위한 절차를 정리하면 그림 2와 같다. 이 절차는 다음과 같이 4 단계로 구성 된다:

1. 탐지 : 한 기업이 다른 기업에 적기 협력을 필요로 하는 긴급 요청을 해야 하거나, 다른 기업에게 즉시 통보해야할 긴급 정보가 발생한 경우를 인식하는 단계이다.
2. 전달 : 탐지된 상황을 협력 업체에 전달하는 단계.
3. 평가 : 요청된 협조 업무의 수행가능 여부와 통지된 긴급 정보에 대한 조치 가능성을 평가하는 단계로, 협력 가능 제약 조건을 점검하여야 한다. 협력 업체가 협력을 할 수 없는 경우가 있기 때문이다. 협력 가능 제약 조건에는 제

한된 시간 이내에 또는 절차상 시간이 경과하여 협조할 수 없는 시간 제약 조건과 요청한 제품이나 서비스를 실행할 자원이 없어 협조할 수 없는 자원(resource) 제약 조건이 있다.

4. 협력 : 요청된 협조 업무를 실행하거나 통지된 긴급 정보에 대한 조치를 취하는 단계이다.

B2B 전자상거래 환경에서 기업들이 적기 협력을 하기 위해서는 위와 같은 절차를 따라야 하는데, 그러기 위해서는 기업의 시스템은 기업 사이에 즉시 처리를 필요로 하는 상황을 탐지하고 이를 서로 통보할 수 있는 기능과 통보된 상황을 인식하고 이에 대한 조치(협력 행위)를 취할 수 있는 기능이 있어야 한다. 이 것은 능동데이타베이스의 ECA 규칙 메커니즘에 적합한 응용임을 알 수 있다[6]. 즉, 협력 업체 사이에서 즉시 처리 방식을 필요로 하는 상황을 ECA 규칙에 따라, 상황의 발생을 사건(Event)으로, 그리고 이 사건이 즉시 처리를 필요로 하는지 여부를 조건(Condition)으로, 그리고 이에 대한 조치(협력 행위)를 행위(Action)로 표현할 수 있다. 그러면, ECA 규칙을 실행하는 능동 기능 컴포넌트가 사건의 발생을 자동으로 탐지하여, 이에 대한 조치를 취하는 협력 업체의 시스템에 사건 발생을 즉시 통보하고, 통지 받은 협력 시스템의 능동 기능 컴포넌트는 조건을 검사하여 해당 상황 여부를 판단하여 조치(협력 행위)를 취함으로써 업무 협조를 즉시 처리 방식으로 구현할 수 있다.

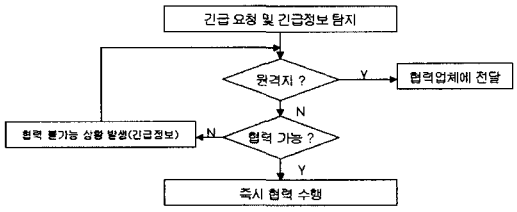

그림 2 기업사이의 적기 협력 절차

위와 같은 ECA 규칙 기법을 이용하면 다양한 형태의 다중 시스템 사이의 즉시 처리 방식의 협력을 형성할 수 있다. ECA 규칙은 시스템 관리자 및 프로그래머가 시스템이 다른 여러 시스템들과 협력하도록 프로그램하는데 사용할 수 있는 인터페이스(프로그램 도구) 역할을 하게 된다. 적기 협력을 위한 사건과 조건 및 행위는 규칙으로 표현되고, 이를 능동 기능 컴포넌트가 실행하게 된다. 따라서 능동 기능 컴포넌트는 ECA 규칙을 관리하고 실행할 수 있도록 사건 관리 및 탐지 기능, 조건 평가 기능, 행위 실행 기능을 포함해야 한다. 그리고 B2B 전자상거래 환경의 방화벽을 통해서 시스템 사이의 상호운용을 할 수 있도록 HTTP 프로토콜로 구현된 서버-서버 프로토콜로 접근 가능한 통신 기능이 있어야 한다. 다음 장에서 B2B EC 환경에서 기업사이의 적기 협력을 표현하기 위한 ECA 규칙 요소를 설명하고, 5장에서 이를 실행하는 능동 기능 컴포넌트에 대하여 설명한다.

3. 적기 협력을 위한 ECA 규칙

B2B 전자상거래 환경에서 기업 사이의 적기 협력을 ECA 규칙으로 표현하게 되는데, ECA 규칙 구성 요소가 이를 반영할 수 있어야 한다. 본 연구에서는 새로운 ECA 언어를 개발하지 않고 일반적인 ECA 규칙언어 개념을 도입하여 적기 협력을 표현하기 위한 최소한의 확장을 한다. 그리고 실용성을 위하여 상업용 DBMS로 구현하고자 한다.

3.1 구성요소

본 연구에서는 이러한 ECA 규칙 모델의 하나인 ECAA (Event-Condition-Action-Alternative Action) [6] 규칙 모델을 채용한다. 그림 3은 ECA 협력 규칙의 구조이다. 규칙의 이름은 규칙이 실행되는 동안은 중요한 역할을 하지 않고 주로 관리 목적으로 쓰인다. 규칙의 정의는 그림 4의 능동 가능 컴포넌트의 사건-규칙-인터페이스(Event-Rule Interface)를 통하여 시스템에 정의된다.

```
Rule 규칙-이름(rule-name);
        Event 사건-표현식(event-expression);
        Condition 조건-표현식 (condition-expression);
        Action Begin 행위 블록(action-block)End
        [Alternative Action Begin 다른 선택 행위 블록 End]
```

그림 3 ECA 규칙 구조(ECA Rule Structure)

3.1.1 사건(Event)

규칙의 실행은 사건-표현식과 같은 사건의 발생을 탐지하면서 시작된다. 이러한 사건은 일반적으로 데이타베이스 연산(접근, 삽입, 삭제, 갱신) 사건, 시간 사건, 그리고 응용-정의(Application-defined) 사건으로 분류된다[6]. 본 연구에서는 B2B 전자상거래 환경에서 기업사이의 적기 협력을 필요로 하는 경우를 각각 사건화(abstract event)하여 ECA 규칙으로 표현한다. 그리고 이 적기 협력 사건은 기존의 사건 분류 방식 이외의 발생 방식(occurrence mode)과 구독 방식(subscription mode)에 따라 지역 사건과 원격지 사건으로 분류할 수 있다. 즉 사건의 발생이 지역인지 원격지인지의 구분과 발생된 사건을 필요로 하는 것이 지역 규칙인지 원격지 규칙인지의 분류이다. 이러한 정보는 사건을 정의할 때

그림 4의 사건-규칙-인터페이스를 통하여 사건 관리기(Event-Manager)의 사건 관련 스키마 테이블에 저장되고, 사건이 발생하면 사건 관리기가 이 테이블들을 참조하여 구독자에 따라 지역 규칙 관리기(Rule-Manager)에게 전달할 것인지 통신 관리기(Communication-Manager)를 통하여 원격지 협력 시스템에게 전달할 것인지를 결정한다. 따라서 2.2에서 분석된 적기 협력을 필요로 하는 각 경우에 대한 사건을 표 1과 같이 요청하는 사건(Request-Service)과 통보하는 사건(Notify)으로 구분하여 표현할 수 있다. 'Notify-Unable-Service'는 약속된 협조 업무를 수행할 수 없을 때 이 사실을 업무 협조를 요청한 시스템에게 긴급히 통보하는 원격지 사건이다. 'Notify-Able-Service'는 협조 가능을 통보하는 원격지 사건으로 통신상의 'ack' 표현을 위한 컨스트럭트(construct)로 추가된 것이다. 그리고 나머지 사건들(Request-Modify-Service, Request-Cancel-Service, Request-Special- Service)은 협력 시스템에게 업무 협조를 긴급히 요청하는 사건들이다.

즉시 협력에 필요한 사건들은 상업용 DBMS에서 지원하지 않으므로, 협력을 필요로 하는 업무 응용에서 일으켜야 한다. 따라서 이 사건 생성을 위한 포장 코드(wrapper code)가 필요하다. B2B 전자상거래 시스템 사이의 적기 협력을 위한 사건은 시스템들이 협력하는 데 필요한 정보를 포함해야 하므로 이를 파라미터 형태로 포함하여 메시지로 전달된다. 따라서 사건 관리기가 관리하는 사건 관련 스키마 테이블에는 이에 대한 정보들이 함께 저장 관리된다. 사건 표현식의 구문은 다음과 같다 :

event-expression ::= event-name(type$_1$ par$_1$, type$_2$ par$_2$, .. type$_n$ par$_n$);

예를 들면, Supplier가 item n개에 대한 공급 요청을 협조할 수 없다는 사건은 다음과 같다.

Event unable-supply(string supplier, string item, integer n);

3.1.2 조건(Condition)

규칙에 있는 사건의 발생이 탐지되면 조건(condition)이 평가된다. 이 조건은 해당되는 사건이 발생하였을 때 규칙의 행위(협조 행위)를 실행해야 하는 상황인지의 여부와 다른 행위의 선택 여부를 좀더 상세하게 제어할 수 있게 한다. 따라서 조건에 따라 다양한 조치를 규칙으로 표현할 수 있다. 그리고 B2B EC에서 기업사이의 적기 협력은 협력업체가 협조를 할 수 있어야 하므로, 협력 가능 조건을 평가해야 한다. 협력 가능 제약 조건에는 제한된 시간 이내에 또는 절차상 시간이 경과하여 협조할 수 없는 시간 제약 조건과 요청한 제품이나 서비스를 실행할 자원이 없어 협조할 수 없는 자원(resource) 제약 조건이 있다. 본 연구에서는 상업용 DBMS의 트리거(trigger)에서 SQL 조건문을 지원하므로 이를 이용하여 ECA 규칙의 조건을 표현한다.

3.1.3 행위(Action)

ECA 규칙의 행위 부분은 규칙이 활성화되고 조건이 만족되었을 때 실행될 연산들을 명시한다. 즉, 필요한 조치를 취하는 것으로 데이타베이스 조작 명령이나 기타 데이타베이스 관련 명령이 될 수 있고, 또, 일련의 응용 프로그램일 수도 있다. 그리고 때에 따라 새로운 사건을 발생시키는 행위가 될 수 있다. 이러한 것들을 이용하여 B2B 전자 상거래 환경에서 기업 사이의 즉시처리 협력을 표현할 수 있다. 표 1에 2.2절에서 언급한 적기 협력을 위한 행위들을 해당되는 사건과 함께 보이고 있다. 'Find-Alternate-Service' 행위의 경우는 협력 시스템에서 요청한 협조 업무가 실행될 수 없음을 통보받고 다른 대안을 찾기 위한 행위이다. 나머지 행위들(Modify-Service, Cancel-Service, Special-Service)은 협조를 요청하는 시스템의 요구를 수행하는 원격지 행위 종류들이다. 따라서 이러한 원격지 행위는 협력 시스템의 관리자의 엄격한 통제 하에 정의되어야 한다. 행위에 대한 구문은 상업용 DBMS로 쉽게 변환하기 위하여 상업용 DBMS의 트리거 프로시저 구문을 따른다. 적기 협력을 위한 ECA 규칙의 행위 부분은 협조 행위이므로 이를 트리거의 프로시저나 내장 프로시저에 대한 함수콜(function call)로 구현하도록 한다.

표 1 적기 협력을 위한 사건과 행위

즉시 협조 요청 및 통보 사건	즉시 협조 행위
Notify Able Service	No Action
Notify Unable Service	Find Alternate Service
Request-Modify-Service	Modify-Service
Request Cancel Service	Cancel Service
Request Special Service	Special Service

3.1.4 선택적 행위(Alternative Action)

사건(협력 요청)이 발생하면 규칙이 활성화되고, 조건(협력 제약 조건)이 만족하면 행위가 실행이 된다. 그러나 조건이 만족하지 않을 경우, 즉 협력 제약 조건을 만족하지 못하여 협력 업체가 협조를 할 수 없는 경우 이를 요청한 업체에 이 사실을 즉시 통보하는 등의 행위가 필요하다. 선택적 행위 부분은 이러한 협력 제약 조건이 만족되지 않았을 경우를 표현하는 부분이다. 이 것은 'Notify-Unable-Service'와 같은 사건을 발생하기도 한다. 예를 들면,

Alternative Action Begin Call raise-event ('Notify-Unable-Service') End;

이러한 선택적 행위 부분은 모든 규칙에서 다 필요한 것은 아니다. 따라서 필요한 경우에만 명시한다.

ECA 규칙 기법을 이용하면 지역과 원격지의 사건 및 행위를 조합하여 B2B 전자상거래 환경에서 웹 고유의 분산성질을 유지하면서 융통성 있고 다양한 형태의 다중 시스템 사이의 즉시 처리 방식의 협력을 형성할 수 있다.

3.2 규칙 예

이 절에서는 ECA 규칙 패턴을 사용하여 2.2절에서 언급한 적기 협력이 필요한 경우에 대한 규칙 프로그램의 예를 보인다.

예1) 쇼핑 몰에서 어느 제품이 갑자기 재고가 없어 협력 업체(공급자)에 긴급히 공급 요청을 하는 경우를 보자. 협력 업체(공급자)는 요청된 제품을 약속된 기일 이내에 즉시 공급하여야 한다. 그리고 만약 협력 업체가 이를 제 때에 공급할 수 없다면, 즉시 이 사실을 알려주어 쇼핑몰이 다른 공급업체를 찾도록 해야 한다. 이 예는 쇼핑 몰과 공급 업체가 적기 협력을 해야 함을 보이고 있다. 다음 두 규칙 'Request-Special-Service'와 'Find-Alternate-Service'로 위와 같은 경우의 적기 협력을 표현할 수 있다.

```
Rule Request-Special-Service /* rule on a Supplier */
  Event request-special-supply(string requester, string item, integer n);
  Condition no. of item > n;
  Action Begin special-order-processing(string requester, string item) End
  Alternative Action Begin raise-event('notify-unable-special-supply') End
Rule Find-Alternate-Service /* rule on a Shopping Mall */
  Event notify-unable-special-supply(string supplier, string item, integer n);
  Condition true;
  Action Begin find-alternate-service(string item, integer n) End
```

위 두 규칙이 제대로 동작하기 위해서는 사건과 규칙이 적절히 정의 및 구독되어야 한다. 즉, 사건 'request-special-supply'는 쇼핑몰이 지역 사건으로 정의하고 구독자로 공급업체를 등록한다. 그리고 공급업체 시스템에는 이 사건을 원격지 사건으로 정의하고 구독자를 공급업체 자신으로 등록하도록 요청하고, 이 사건에 대한 조치로 행위 'special-order-processing'을 실행할 수 있도록 공급 업체 시스템 관리자에게 규칙의 정의 및 구현을 요청하여야 한다. 여기서 협력 제약 조건은 자원 제약 조건이다. 그리고 이에 협조할 수 없을 경우의 대체 행위로 사건 'notify-unable-special-supply'를 발생하고 있다. 그리고 사건 'notify-unable-special-supply'는 공급업체에 지역 사건으로 정의되어야 하고 쇼핑몰이 이를 구독하여야 한다. 쇼핑몰은 이 사건을 자신의 사건 관리기를 통하여 원격지 사건으로 등록한 다음, 이 사건에 대한 규칙 'Find-Alternate-Service'를 정의하여야 한다.

예2) 다음 규칙 'Modify-Batch-Transfer'는 신용카드 회사가 즉시 입금 방식으로 결제한 고객을 자동 이체자 명단에서 제외 시켜 달라는 요청을 은행에서 수행하는 예로 기존 협조 업무를 수정하는 경우이다. 여기서 사건 'request-modify-batch-transfer'는 신용카드 회사가 발생시키지만 은행 시스템에는 이 사건을 구독하게 원격지 사건으로 등록하여야 한다. 그리고 이 사건에 대한 조치로 행위 'delete-batch-transfer'를 실행할 수 있도록 은행 시스템 관리자에게 규칙의 정의 및 구현을 요청하여야 한다. 이는 상대 시스템에 영향을 미치므로 엄격한 통제 하에 정의 및 실행되어야 한다.

```
Rule Modify-Batch-Transfer /* rule on a Bank */
  Event request-modify-batch-transfer(string requester, string delete, string name, string account-no);
  Condition before batch-processing-done;
  Action Begin delete-batch-transfer(string transfer-list, string name, string account-no) End
```

예3) 규칙 'Cancel-Service'는 쇼핑 몰이 item-list의 주문을 철회 요청하였을 때 공급업체가 이의 처리를 취소하는 경우이다. 여기서 공급업체는 무조건 철회하는 것이 아니고 철회가 가능한 상태(즉, 주문처리 전)에만 응하고 있다. 그리고 사건 'request-cancel-supply'는 쇼핑몰이 발생시키지만 공급업체 시스템에다 이 사건을 구독하게 원격지 사건으로 등록하여야 한다. 그리고 이 사건에 대한 조치로 행위 'cancel-order-processing'를 실행할 수 있도록 공급업체 시스템 관리자에게 규칙의 정의 및 구현을 요청하여야 한다.

```
Rule Cancel-Service /* rule on a Supplier */
  Event request-cancel-supply(string requester, string item-list);
  Condition before Order-Processing Done;
  Action Begin cancel-order-processing(string requester, string item-list) End
```

4. 능동 기능 컴포넌트

기업사이의 적기 협력을 표현한 ECA 규칙 프로그램은 능동 기능 컴포넌트에 의하여 실행되게 된다. 즉시 처리 협조를 요청하는 사건이 발생하면 이 것이 탐지되어

이를 필요로 하는 규칙이 지역이면 지역 규칙 관리기에, 원격지 규칙이면 이를 필요로 하는 원격지 능동 기능 컴포넌트에게 전달한다. 이 장에서는 제안된 능동 기능 컴포넌트의 구성 요소와 이들의 상호 작용을 설명한다.

4.1 아키텍처

B2B 전자상거래 시스템 사이의 적기 협력을 표현한 ECA 규칙은 그림 4와 같은 능동 기능 컴포넌트에 의하여 처리된다. 능동 기능 컴포넌트는 통신 관리기(Communication_Manager), 사건 관리기(Event_Manager), 규칙 관리기(Rule_Manager), 사건 규칙 인터페이스(Event_Rule_ Interface), 행위/응용(Actions/Applications) 모듈로 이루어진다.

각 모듈의 중요 기능과 구조는 다음과 같다:

- Communication_Manager : HTTP 프로토콜을 이용하여 적기 협력을 위한 사건 메시지를 다른 서버의 통신 관리기와 주고받는 역할을 담당한다. 웹 서버의 Java servlet으로 작성된 모듈로 2가지 역할을 한다. 우선, 웹 서버를 통하여 외부에서 오는 XML 메시지에서 사건을 추출하여 사건 관리기에게 전달하는 역할(그림 5의 Receive-Event)과 두 번째로, 내부 사건 관리기가 전달하는 사건을 XML로 변환하여 HTTP post 명령으로 상대 능동 기능 컴포넌트의 통신 관리기에게 전달하는 역할(그림 5의 Send-Event)이다. XML 메시지를 처리하기 위하여 Xerces2 Java parser가 이용된다. 통신 관리기의 상세 구조도는 그림 5에 있다.
- Event_Manager : 사건에 대한 정의 스키마를 관리하며 지역 사건과 원격지 사건을 인식하여 해당(구독하는) 규칙 관리기에게 전달하는 역할을 한다. 웹 서버의 Java servlet으로 작성된 모듈로 필요한 사건 스키마 등록과 실행 시간 사건을 관리한다. 사건의 등록과 구독은 사건 규칙 인터페이스를 통하여 시스템 관리

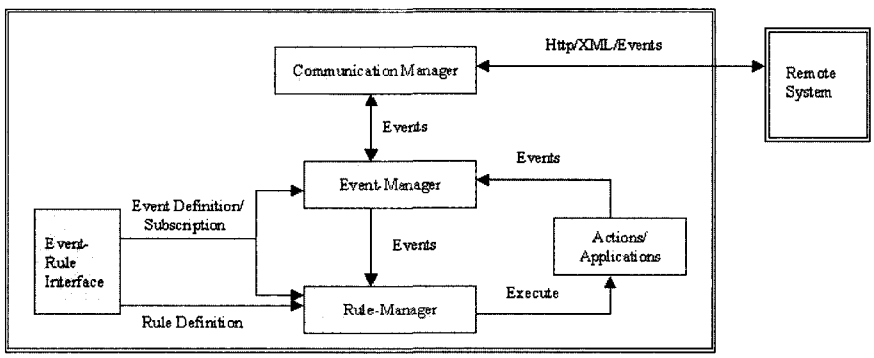

그림 4 능동 기능 컴포넌트 아키텍처

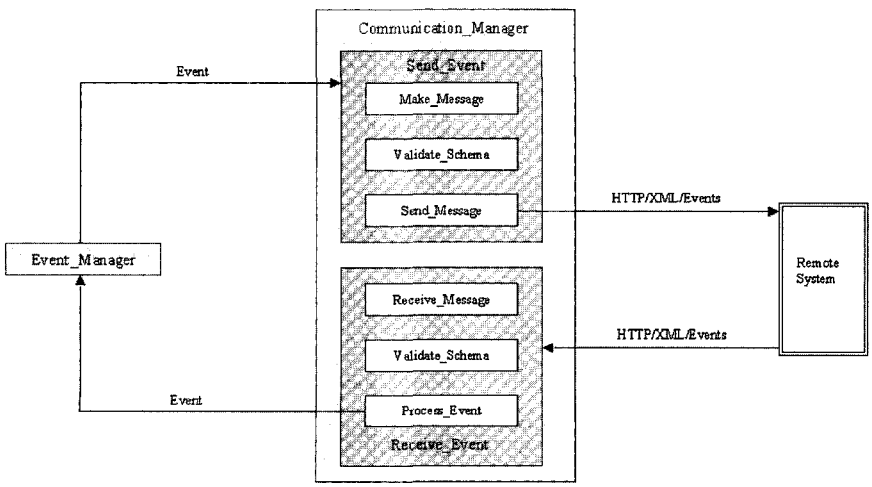

그림 5 Communication Manager

자나 프로그래머의 입력에 의하여 사건 관련 테이블에 사건에 관한 정의와 구독에 관한 사항을 등록한다. 그리고 실행 중에는 전달되어 오는 사건 사례를 Event-Comparator(그림 6)가 사건 관련 테이블의 subscriber-schema에서 확인하여 구독자가 지역인지 원격지인지를 결정하여 지역 규칙 관리기나 통신 관리기의 Send_Event(그림 5)에게 파라미터와 함께 각각 전달한다. 사건 관련 테이블은 다음과 같이 4 개의 테이블로 구성된다.

event-schema=(event-name, no-of-parameters)
publisher-schema=(event-name, publisher)
subscriber-schema=(event-name, subscribers)
parameter-schema=(event-name, para-name, type, position, length)

이들 테이블은 데이타베이스에 저장된다. 따라서 사건 관리기는 JDBC를 이용하여 데이타베이스에 연결하여 이 테이블들을 관리한다. 사건 관리기의 구조도는 그림 6에 있다.

• Rule_Manager : 사건 사례 테이블과 규칙 테이블을 관리하며 규칙을 평가 실행하는 역할을 한다. 하부 DBMS의 트리거를 이용하여 구현된 것으로 사건 사례 테이블(Event-Instance table)과 규칙 테이블(Rule Table)을 갖고 있다. 그리고 해당 규칙의 평가 및 행위의 실행이 하부 DBMS에 의하여 이루어진다. 사건 규칙 인터페이스를 통하여 사건이 정의될 때 사건 정의에 의하여 해당 사건 사례 테이블을 형성한다. 규칙 역시 사건 규칙 인터페이스를 통하여 정의되고, 이 규칙들을 규칙 테이블에 저장하고 사건 사례와 내부 트리거로 연결된다. 사건 관리기로부터 전달되어 오는 사건 사례를 사건 사례 테이블에 저장 및 탐지를 하여 해당 규칙을 구동시킨다. 그리고 이 규칙의 조건을 평가한 다음 조건이 만족되면 규칙의 행위를 실행한다. 규칙 관리기의 상세 구조도는 그림 7에 있다.

능동 기능의 주요 구성 요소는 사건의 탐지, 조건의 평가, 그리고 행위의 실행이다. 본 연구에서는 조건의 평가와 행위의 실행 부분을 하부 DBMS의 트리거 기능

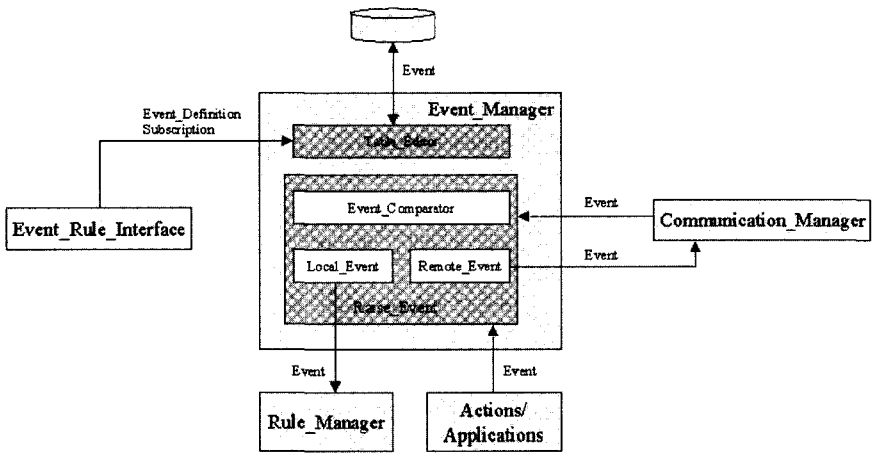

그림 6 Event Manager

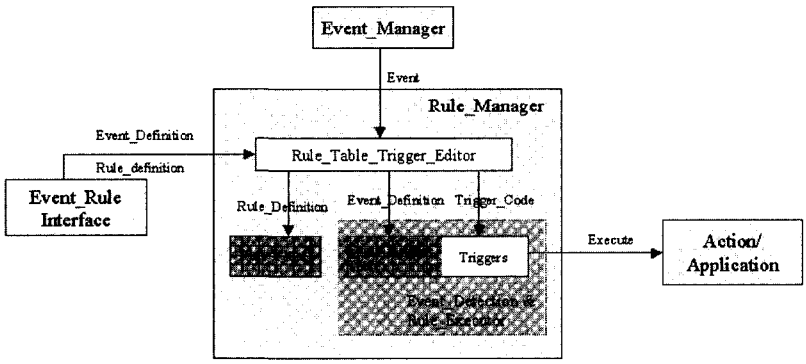

그림 7 Rule Manager

을 이용하여 설계 구현하였다. 즉, 각 사건에 대한 사건 사례 테이블을 데이타베이스에 만들고 이 테이블에 대한 트리거 코드를 작성한다. 그리고 트리거의 몸체(body)에 조건의 평가와 행위의 실행 코드를 삽입한다. 그러면 사건 사례가 사건 사례 테이블에 삽입될 때 관련된 트리거가 하부 DBMS에 의하여 구동되어 조건을 평가하고 조건이 만족되면 행위 부분을 실행하게 된다. 다음은 Oracle 9i 트리거문을 이용하여 작성된 예이다.

```
create trigger find-alternate-service
        after insert on unable-service-instance-table
begin
    if (true)
    then call find-alternate-service();
end;
```

여기서 'find-alternate-service()'는 stored procedure로 이루어진 프로시저로 실제 행위(Action)를 수행하는 코드이다.

- Event-Rule Interface : 시스템 관리자와 프로그래머를 위한 JSP로 작성된 인터페이스로 사건의 정의와 규칙의 정의 뿐 아니라 관리 즉, 검색 및 삭제와 수정을 할 수 있는 도구이다. 사건 관리기와 규칙 관리기의 기능을 이용하여 작업을 수행한다. 그림 8에 사건 규칙 인터페이스 구조도가 있다.
- Actions/Applications은 DBMS의 API로 이루어진 내부 행위와 그 이외의 응용인 외부 행위를 의미한다. 즉시 협력을 요청하는 사건은 응용 관련 사건이므로 이를 발생시키는 포장 코드(wrapper code)가 생성되어야 하고 이를 사건 관리기에게 통보하는 raise-event() 함수 콜(function call)이 필요하다. ECA 규칙의 행위 부분은 즉시 협력을 위한 조치(협조 행위)를 취하는 부분이므로 앞의 규칙 관리기에서 설명한 것처럼 trigger body에 이에 대한 프로시저나 저장 프로시저 콜로 이루어진다. 그리고 이러한 행위(action)는 또 다른 사건을 발생시킬 수 있다. 그림 9에 Action/Application의 구조도가 있다.

4.2 구성 요소 사이의 상호 작용

위 구성 요소 사이의 상호작용 과정을 build-time과 run-time으로 구분하여 정리하면 다음과 같다:

- Build-time

1. 관리자나 프로그래머에 의한 사건 규칙 인터페이스를 통한 적기 협력 사건의 정의 입력
2. 사건의 정의는 사건 관리기에 의하여 사건 관련 스키마 테이블에 기록된다.
   사건을 발생하는 wrapper code 생성
3. 사건의 정의는 또한 규칙 관리기에게 전달되어 사건 사례 테이블이 생성된다.
4. 사건 규칙 인터페이스를 통한 사건 구독 입력
5. 사건 구독 정보는 규칙 관리기에 의하여 사건 관련 스키마 테이블에 기록
6. 사건 규칙 인터페이스를 통한 규칙의 정의
7. 규칙의 정의는 규칙 관리기에 의하여 규칙 테이블에 기록되고 해당 사건 사례 테이블에 대한 트리

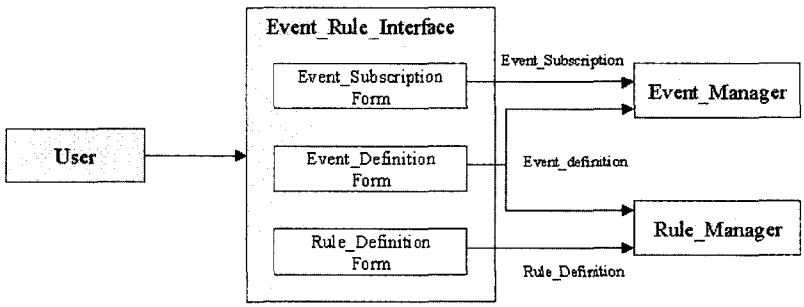

그림 8 Event Rule Interface

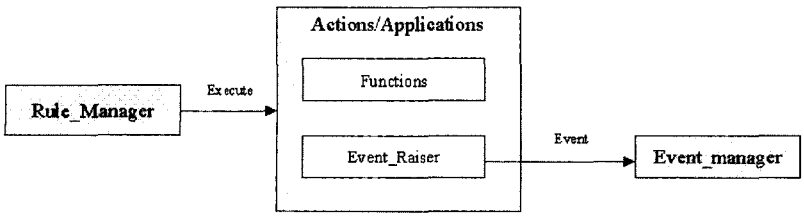

그림 9 Actions/Applications

거를 생성하고 규칙의 행위 부분에 해당하는 코드 생성

• Run-time

1. 지역 응용이나 행위에 의한 사건 발생이 사건 관리기에게 전달되거나 또는 원격지로부터의 사건이 통신 관리기를 통하여 사건 관리기에게 전달됨
2. 사건 관리기는 사건 관련 스키마 테이블에서 해당 사건의 구독자를 확인
3. 구독자가 원격지 서버이면 통신 관리기에게 전달하고 지역이면 규칙 관리기에게 전달
4. 통신 관리기에게 전달된 사건은 파라미터와 프로토콜 포장기가 추가되어 원격지 통신 관리기에게 전달됨
5. 규칙 관리기는 사건을 사건 사례 테이블에 삽입하여 트리거에 의한 해당 규칙을 구동시킨다.

4.3 프로토타입 구현

제안된 능동 기능 컴포넌트의 프로토타입을 구현하였다. 제안된 능동 기능 컴포넌트의 설계시 고려 사항은 실용성, 데이타베이스와의 연동, 그리고 플랫폼 독립성이다. 따라서 구현 언어로는 Java, 상업용 DBMS로는 Oracle을 택하였고, Apache Tomcat Web Server, Xerces2 Java Parser, Windows2000 서버, Linux 서버를 사용하였다.

앞에서 언급한 능동 기능을 처음부터 모든 것을 설계하고 구현하거나, 기존 DBMS의 커널에 직접 융합시킨다면, 시간과 비용이 너무 많이 소모된다. 그리고 일반적으로 시스템의 안정성과 유지보수의 확실성을 고려하여, 현재 시스템 시장에서 널리 사용되고 있는 제품들을 사용하고 있다. 또 상업용 DBMS들은 최소한 기본적인 트리거 기능을 제공하므로 이를 이용하여 능동 기능 컴포넌트를 구축하면, 시스템 구축비용이 크게 절감되면서 기존 시스템을 그대로 적용할 수 있다. 본 연구에서는 상업용 DBMS 중에서 Oracle을 예로 하여 능동 기능을 구축하였다. 그러나 다른 제품들도 동일하게 적용될 수 있다.

오라클의 데이타베이스 트리거는 테이블, 스키마, 데이타베이스에 관련된 저장 PL/SQL 블록이나 임의의 PL/SQL 블록 또는 PL/SQL 이나 Java로 구현된 프로시저를 call 하는데 사용된다. 그리고 지원되는 사건은 데이타베이스 사건들(insert, delete, update 등)이다[7]. 따라서 4.1절에서 설명한 것처럼 각 사건에 대한 사건 사례 테이블을 데이타베이스에 만들고 이 테이블에 대한 트리거 코드를 작성한다. 그리고 트리거의 몸체(body)에 조건의 평가와 행위의 실행 코드를 삽입한다. 그러면 사건 사례가 사건 사례 테이블에 삽입될 때 관련된 트리거가 하부 DBMS에 의하여 구동되어 조건을 평가하고 조건이 만족되면 행위 부분을 실행하게 된다. 이처럼 사건 사례 테이블을 데이타베이스에 생성하고 이에 대한 트리거를 작성하여 적기 협력 규칙을 구현하였다.

5. 적 용

제안된 적기 협력 방안과 능동 기능 컴포넌트의 타당성을 보이기 위하여 그림 1의 전형적인 B2B 전자 상거래 시나리오에 적용해 보았다.

5.1 적용 시나리오

적용된 예는 그림 1과 같은 환경에서 전형적인 B2B 전자상거래의 한 경우인 쇼핑몰과 공급자, 택배회사 사이의 적기 협력 구현에 적용하였다. 이들은 각자 자신의 사이트를 소유하고 있고 이를 통하여 B2B 전자상거래를 수행하고 있다. 인터넷상에는 다수의 쇼핑몰, 다수의 공급자, 다수의 택배회사가 있고 이 들은 각각 독립적으로 영업 파트너(business partner)와 사업을 하고 있다. 적용의 편이성을 위하여 은행 부문은 제외하였다.

적용된 시나리오는 쇼핑몰에서 고객에게 다양한 서비스를 제공하기로 하고 **일정 기한 동안은 고객이 주문 내역을 변경(상품 추가 또는 일부 취소) 하거나 주문을 취소할 수 있고 또 배송지를 변경할 수 있도록 허용**하고자 한다. 이러한 경우 고객의 서비스를 만족시키기 위해서는 쇼핑몰과 협력 업체인 공급자, 택배회사가 함께 협조를 하여야 한다. 즉 고객의 상품 주문 행위는 쇼핑몰이 공급자에게 이의 공급을 요청하고, 공급자는 이를 공급하는 협조 행위를, 그리고 택배회사에게는 상품의 배송 요청을, 택배회사는 이를 배송 협조 행위를 수행하도록 한다. 그리고 이 것은 이미 수행한, 현재 수행 중인, 또는 수행할 업무에 영향(변경)을 미치므로 즉시 협조 요청과 협조 행위가 이루어져야 한다. 따라서 새로운 서비스를 제공하기 위해서는 쇼핑몰은 공급자 및 택배회사와 이에 대한 새로운 업무 협약을 체결하고 이를 시스템에 반영하여야 한다.

5.2 적기 협력 규칙 유도

적기 협력 규칙을 유도하기 위해서는 각 업체들의 협력 가능성(협력 제약 조건), 요청 사건, 그리고 이에 상응하는 행위를 파악하여야 한다. 우선 각 협력 업체의 업무 흐름을 분석하여 새로운 고객 서비스에 대한 협조 가능성을 분석하면 표 2와 같다. 여기에서 협조 가능한 시간(시간 제약 조건)을 유도할 수 있다. 즉, before send item 이나 before ship 동안 새로운 서비스에 대한 협조 행위를 할 수 있다.

각 업체 별로 적기 협력 규칙을 유도하기 위해서는 각 업체 별로 2.2절에서 언급한 적기 협력을 필요로 하는 경우 각각에 대한 사건과 이에 대한 조치를 분류하

표 2 새로운 서비스에 대한 협조 가능성

새로운 서비스 / 협력업체	상품 추가/ 일부 취소	주문 취소	배송 변경
Supplier	가능 (before send items)	가능 (before send items)	불가능
Shipper	불가능	불가능	가능 (before ship)

여 규칙으로 표현한다.

그림 10에 전체 적기 협력 관계가 나타나 있다. 화살표는 사건 메시지의 전달을 의미한다. 협력 업체 사이에 이동하는 사건과 이 때 수행되는 규칙이 함께 표시되어 있다. 이렇게 정의된 사건과 규칙을 각 시스템의 능동 기능 컴포넌트가 실행하면 각 협력 업체가 지원하고자 하는 적기 협력을 구현할 수 있다.

5.3 System Test(Walk-through)

앞에서 구현된 시스템을 실제 상황처럼 동작 시켜 정확성을 확인하였다. 쇼핑몰에서 고객이 제품을 주문한 후, 이 주문에 대하여 다른 제품을 추가하거나 일부 품목을 취소하여 주문을 변경하는 경우, 주문 자체를 취소하는 경우, 그리고 배송지를 변경하는 경우에 대하여 정확히 동작하였다. 고객이 주문에 제품을 추가하는 경우를 살펴보면 그림 11과 같은 순서도로 표현할 수 있다.

그림 11에서 우선 1과 같이 고객이 쇼핑 몰의 웹에서 이미 한 주문에다 추가로 품목을 추가하는 요청을 하게 된다. 쇼핑 몰은 이를 즉시 2와 같이 'request-modify-service' 사건을 발생시켜 Supplier에게 전달한다. Supplier는 이를 자체 능동 기능 컴포넌트에 의하여 처리하는데, 협조 가능할 경우 즉시 3과 같이 ack용 'able-service' 사건을 쇼핑몰에 전달한다. 그리고 협조가 불가능할 경우에는 3′과 같이 'unable-service' 사건을 발생시켜 전달한다. 이러한 사건을 전달받은 쇼핑 몰은 사건 종류에 따라 4나 4′ 같이 즉시 client에게 요청한 결과를 통보한다. 이러한 요청은 Supplier나 쇼핑몰의 업무 처리 절차에 따라 처리되어 점선 화살표로 되어 있는 5와 6 같이 주문 품목이 각각 전달된다.

그림 10 쇼핑몰, 공급자, 택배회사 사이의 즉시 협력을 위한 사건과 규칙

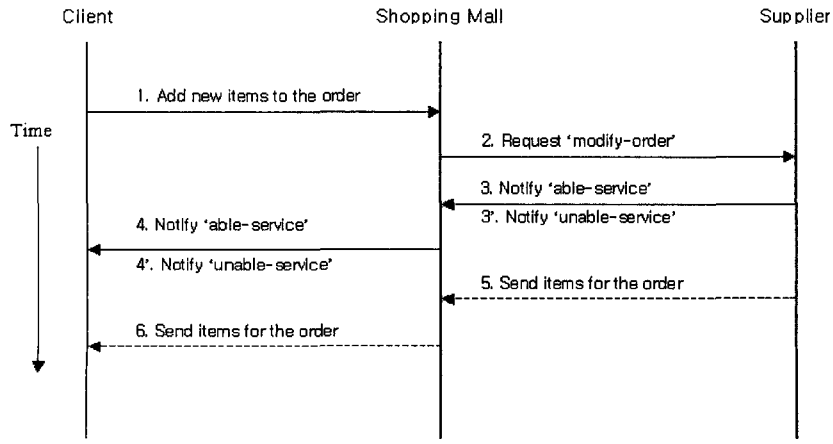

그림 11 고객, 쇼핑몰, 공급자 사이의 상호작용 순서도

5.4 분석

앞 절의 적용 예는 본 논문의 적기 협력 메커니즘을 이용하여 B2B 전자 상거래 환경에서 기업사이의 적기 협력을 체계적으로 구현할 수 있음을 보여 주고 있다. 이 것은 기업 사이의 business level 협약 체결에 따라 시스템을 변경할 때, 기존 업무 시스템에 전혀 영향을 주지 않고 적용할 수 있는 이점이 있어 시스템 관리자나 프로그래머의 부담을 크게 경감시키고 있다. 여기서 주목할 점은 적기 협력 메커니즘의 목표는 긴급하고 비동기적인 적기 업무 협조를 지원하는데 있다. 따라서 기존의 일반적이고 동기적인 일상 업무 협조를 관리하는 WFMS나 BPMS(business process management system)와 보완 관계에 있다.

본 논문에서 제안하고 있는 능동 기능 컴포넌트는 시스템의 다른 부시스템(subsystem)과 느슨하게(loosely) 결합된 형태이므로 적기 협력 메커니즘을 필요로 하는 다른 어떤 시스템에도 쉽게 적용될 수 있다. 특히 Java Servlet과 상업용 데이타베이스 시스템의 기본적인 트리거를 이용한 능동 기능 컴포넌트 구현은 실용적이다. 즉, 대부분의 전자상거래 시스템에 채용되고 있는 웹 플랫폼에서 Java Servlet을 지원하고 있고 또 데이타베이스 시스템을 사용하고 있기 때문이다. 따라서 능동 기능이 데이타베이스와 융합되는 이점이 있어 기존의 상업용 데이타베이스 시스템의 능동 기능을 확장할 수 있다.

HTTP 프로토콜과 사건 메시지 방식을 이용한 B2B 전자상거래 시스템 사이의 적기 협력 구현 방식은 방화벽을 통해 느슨하게 결합되어 있는 다른 이질 분산형 시스템에도 적용할 수 있다. 즉, Grid 컴퓨팅 같은 경우도 ECA 규칙을 사용하여 시스템 사이의 상호 협력을 지원할 수 있다.

그러나, 적기 협력에 관련된 사건은 응용 관련 사건이므로 관리자나 응용 프로그래머가 사건 발생 및 통보를 위한 코딩을 하여야 한다. 이를 간단히 할 수 있는 built-in 사건과 행위 또는 template을 제공하여 프로그램 부담을 경감할 필요가 있다. 그리고 B2B 전자 상거래 시스템 사이의 업무 요청과 협조에서는 사건이 단순 사건이지만 복합 사건에 대한 탐지와 통보 그리고 기타 다른 사건(데이타베이스 사건, 시간 사건 등)과의 결합 문제가 고려되어야 한다.

6. 관련 연구

본 논문과 관련된 연구로는 B2B 상호운용성(interoperability), 비즈니스 프로세스에서의 예외 상황(exception), 분산 환경에서의 능동 데이타베이스 응용, 상업용 데이타베이스의 능동 기능 확장에 관한 연구가 있다.

B2B 전자상거래 기술은 Web이 존재하기 이전부터 EDI(Electronic Data Interchange)형태로 존재하였으나 그 제한성 때문에 널리 사용되지 못하였다. 그러나 Web의 등장과 성장으로 B2B 전자상거래는 크게 활성화되었고 많은 연구가 이루어졌다[8]. 이들 연구는 주로 B2B의 상호운영성에 대한 연구가 대부분인데, 그 것은 B2B 특성이 시스템 구성이 독립적이고 분산 이질형의 시스템들이기 때문이다. 물론 분산 이질형 환경에 대한 연구가 있어왔지만, B2B 전자 상거래만의 특성이 있다. B2B 상호운용성 문제는 통신 계층(communication layer), 내용 계층(content layer), 경영 과정 계층(business process layer)로 분류할 수 있다. 즉, B2B 전자 상거래를 하기 위해서는 서로의 업무처리에 동의해야 하고, 서로 주고받는 업무 내용을 이해할 수 있어야 하고 또 메시지를 주고받으려면 통신 프로토콜에 동의하여야 한다.

[9,10,11]에서 워크플로우 시스템에서의 예외상황(exception)에 대하여 다루고 있다. 여기서 정의하는 예외 상황은 정상적인 워크플로우(업무 절차)에서 발생하는 비정상적인 것으로 시스템에서 발생하는 실패나 에러 등에 의하여 워크플로우의 정상적인 처리를 방해하는 경우를 말한다. 이러한 예외 상황은 시스템 레벨(OS, network, DBMS 등)의 기본 실패(basic failure), 응용 실패(application failure), 워크플로우 레벨의 예상된 예외(expected exception), 그리고 비예상 예외(unexpected exception)로 분류하고 있다. 특히 [9,11]에서는 이를 ECA 규칙으로 해결하는 방안을 제안하고 있다. 그러나 이 들 연구는 위와 같은 예외 상황이 발생할 때 워크플로우의 정상적인 처리에 중점을 두고 있다. 즉 고장 허용(fault-tolerant) 워크플로우 처리(workflow processing)에 관한 연구들이다. 이것은 본 논문의 연구와 보완 관계가 있다고 볼 수 있다.

최근의 능동 데이타베이스 연구 분야에서는 능동 기능을 비 데이타베이스 응용 분야와 분산 환경에서도 제공하는 방법들을 연구하고 있다. C^{2}offein[12] 프로젝트에서는 CORBA 분산 환경에서 사용 가능한 컴포넌트를 제안하고 있다. 이 시스템은 파라미터화된 포장기(wrapper)들로 구성되어 있는데, 이를 이용하여 사건 처리기를 구성할 수 있고 어떤 특정 응용에 맞추어진 규칙 엔진을 구성할 수 있다. 그러나 이 연구에서는 방화벽을 고려하고 있지 않다. FRAMBOISE[13] 프로젝트에서는 특정 DBMS로부터 분리된 ECA 서비스를 구축하는 시스템을 제안하고 있다. 이 시스템에서 사건 탐지는 특정 DBMS에 특화된 사건 탐지기가 할 수 있고, 규칙 서비스는 규칙 베이스의 유지보수와 규칙 실행을 담당하고 있다. 그러나 분산 환경을 고려하고 있지 않다. CoopWARE[14]와 TriggerMan[15] 프로젝트들은

분산 규칙 처리 메커니즘들을 제안하고 있다. 전자는 워크플로우 환경에서 개발되었고, 후자는 객체-관계형 DBMS를 위한 실행 모듈로 비동기 트리거 프로세서를 제안하고 있다. [16]은 CORBA 분산이질 시스템을 위한 ECA 규칙 지원 프레임워크 구조를 제안하고 있다. 그리고 분산 환경에서의 복합사건 탐지를 위한 시간 개념을 다루고 있으나, 이들 모두 방화벽을 다루고 있지 않다. [17]에서 분산 환경에서의 개방형 능동 서비스(open active service)를 제안하고 있으나 CORBA의 사건 서비스를 이용하여 구현하고 있다.

[18]에서 웹 환경에서의 ECA 규칙을 다루고 있다. 일반 사용자가 Web에서 사건 등록과 구독을 할 수 있고 ECA 규칙도 구독할 수 있다. 이에 따라 서버가 사건을 구독자 시스템에 전달을 하게 된다. 여기서 지원하고 있는 사건은 웹 접근(web access) 사건, CORBA 사건, 그리고 데이타베이스 사건이다. 그리고 [11]에서 웹 환경에서 기업간의 워크플로우의 동적인 조정에 ECA 규칙을 적용하고 있는데, 이들 모두 중앙 집중 Broker 서버를 필요로 하고 있다. 그러나 본 연구에서의 능동 기능 컴포넌트는 분산 형태에서 중앙 서버가 필요하지 않다. [19]에서는 웹에서의 동적인 문서정보 서비스를 하기 위하여 서버 사이의 협력을 ECA 규칙으로 구현하고 있다. 그러나 본 논문에서 제안하는 능동 기능 컴포넌트의 구조와 구현 방식과는 다르다.

[20]은 [11]과 유사한 연구로 e-Business 응용(Auction, Personalized Car and Driver Portal 등.)을 위한 능동 기능 서비스를 제공하는 미들웨어를 제안하고 있다. 이것은 온톨로지 기반 인프라(ontology-based infrastructure), 사건 통지(event notification), 서비스 기반 구조 3 가지로 구성되어 있다. 이 연구의 주목적은 급변하는 기업 환경에서 응용 로직과 별도로 변화하는 기업 규칙에 따른 업무 프로세스의 적응성을 높이고 이질 소스(heterogeneous source) 들의 사건과 데이타에 대한 공통 온톨로지를 사용하는 것이 특징이다. 이 연구 역시 기업 사이의 적기 협력 문제를 다루고 있지 않다.

현재 시장에서 널리 쓰이고 있는 상업용 DBMS들은 트리거에 의한 기본적인 능동 기능만을 제공하는 한계점이 있다[7]. 따라서 상업용 RDBMS의 트리거 기능을 확장하는 연구들이 이루어졌다[21,22]. 이들의 방법은 계층화 에이전트(layered agent) 방법으로 RDBMS의 트리거를 이용하여 복합 사건의 탐지를 위한 에이전트를 설계 구현하고 있다. 그러나 이들의 연구는 데이타베이스 사건만을 다루고 있고, 응용 관련 사건은 언급이 없다. 본 논문의 능동 기능 컴포넌트는 응용 관련 사건을 탐지할 수 있고 규칙의 실행 부분을 데이타베이스의 테이블과 트리거를 이용하여 DBMS 내에서 구현하고 있다.

7. 결 론

본 연구에서는 B2B 전자상거래 환경에서 기업 사이의 적기 협력 방안과 이를 지원하는 능동 기능 컴포넌트를 제안하였다.

B2B 전자상거래 환경에서 기업 사이의 업무 협력 방식은 일괄요청-일괄처리협조, 일괄요청-즉시처리협조, 즉시요청-일괄처리협조, 즉시요청-즉시처리협조로 구분할 수 있다. 이러한 기업사이의 협력 방식에서 본 논문은 비즈니스 레벨의 업무 협약(계약체결) 관점에서 적기 협력(즉시요청-즉시처리협조)의 필요성을 분석하였다. 즉, 적기 협력이 필요한 경우는 약속된 업무 협조를 할 수 없는 경우, 약속된 협조 업무를 변경해야하는 경우, 약속된 협조 업무를 취소해야 하는 경우, 그리고 약속된 일반 협조 업무와 별도로 특수한 서비스를 필요로 하는 경우이다.

그리고 이를 지원하는 능동 기능 컴포넌트는 사건(즉시 업무 협조 요청)의 발생을 자동으로 탐지하여, 이에 대한 조치를 취하는 협력 업체의 시스템에 사건 발생을 즉시 통보하고, 통지 받은 협력 시스템의 능동 기능 컴포넌트는 조건을 검사하여 해당 상황 여부(즉시 업무 협조 여부)를 판단하여 조치(협조 행위)를 취함으로써 업무 협조를 즉시 처리 방식으로 구현할 수 있다. 능동 기능 컴포넌트는 고수준의 ECA 규칙 프로그램을 제공하므로, 응용 프로그램과는 독립적으로 기업 사이의 업무 협조와 사건 기반의 즉시 처리 협력을 구현할 수 있다. 따라서 시스템 관리자 및 프로그래머는 비즈니스 규칙(업무 협약)의 변화에 따른 기업 사이의 조정 기능 구현과 유지보수를 쉽게 할 수 있다.

제안된 능동 기능 컴포넌트는 실용적이다. 일반적으로 모든 B2B 시스템에서 상업용 데이타베이스 시스템을 사용하고 있는데, 제안된 능동 기능 컴포넌트는 상업용 DBMS의 기본적인 트리거를 이용하여 응용 관련(즉시 협력) 사건을 탐지하고 행위를 구현하고 있다. 그리고 제안된 능동 기능 컴포넌트는 기업의 방화벽을 통해서도 서로 사건 메시지를 전달할 수 있도록 HTTP 프로토콜을 이용하므로, 기업들의 시스템 보안 유지와 독립성을 보장하고 있다. 그리고 능동 기능 컴포넌트에 의한 B2B 전자상거래 시스템의 적기 협력을 위해서 중앙 제어 기능이 전혀 필요 없다. 즉 peer-to-peer 방식의 상호 작용을 하고 있다.

앞으로 본 연구를 확장 보완하기 위한 연구는 다음과 같다. 전자상거래 생명주기 상의 다른 단계에 대한 지원을 위한 다른 여러 가지 형태의 협력 지원을 위한 협력 개념의 확장과, 서버 지향적인 규칙 뿐 아니라 클라이언트에 의한 규칙 지원이 필요하다. 그리고 사건 전달과

규칙의 실행에 관한 보안 문제의 연구가 필요하다. 이것은 사건의 통보가 곧바로 규칙의 실행을 야기하기 때문이다. 이 것은 곧 응용과 데이타베이스에 영향을 미치기 때문이다. 따라서 적절한 보안 대책이 필요하다.

그리고 적기 협력에 관련된 사건은 응용 관련 사건이므로 관리자나 응용 프로그래머가 사건 발생 및 행위를 위한 코딩을 하여야 한다. 이를 간단히 할 수 있는 built-in 사건과 행위 또는 template을 제공하여 프로그램 부담을 경감할 필요가 있다. 또한, 최근 등장하고 있는 여러 가지 표준(ebXML, Web Services 등)과의 결합 문제가 있다. 즉, 이들이 전자상거래 시스템의 표준으로 제안되고 있기 때문이다.

참 고 문 헌

[1] A. Albuquerque, "E-Commerce Websites: a Qualitative Evaluation," in proceedings of WWW'2002, 2002. http://www2002.org/

[2] Nobuyuki Kanaya, et. al., "Distributed Workflow Management Systems for Electronic Commerce," proceedings of 4th International Enterprise Distributed Object Computing Conference(EDOC'00), IEEE 2000.

[3] Paolo Ciancarini, "Coordination Models and Languages as Software Integrators," ACM Computing Surveys, vol.28, No.2, June 1996.

[4] H. Kim, "Modeling Inter- and Intra-Organizational *Coordination in Electronic Commerce Deployments*," Information Technology and Management no,2, Kluwer Academic pub. 2001. pp.335-354.

[5] Aaron Skonnard "SOAP:The Simple Object Access Protocol," http://www.microsoft.com/mind/0100/soap/soap.asp, Microsoft Internet Developer, January 2000.

[6] Norman W. Paton and Oscar Diaz, "Active Database Systems," Computing Surveys, ACM, 1999.

[7] Oracle9i SQL Reference Release 2 (9.2) Part Number A96540-02, October 2002.

[8] Brahim Medjahed, et al., "Business-to-business interactions: issues and enabling technologies," VLDB Journal, Springer-Verlag, April, 2003. pp.59-85.

[9] Fabio Casati, S. Ceri, S. Paraboschi, and G. Pozzi, "Specification and Implementation of Exceptions in Workflow Management Systems," ACM Tr. on Database Systems, Vol. 24, No. 3, September 1999, pp.405-451.

[10] Zongwei Luo, Amit Sheth, Krys Kochut, and Budak Arpinar, "Exception Handling for Conflict Resolution in Cross-Organizational Workflows," Technical Report, LSDIS Lab, Computer Science, University of Georgia, April 10, 2002.

[11] J. Meng, Stanley Y.W. Su, herman Lam and A. Helal, "Achieving Dynamic Inter-Organizational Workflow management by Integrating Business processes, Events and Rules," IEEE HICSS-35'02, 2002.

[12] A. Koschel, et al., "Configuration of Active Functionality for CORBA," proceedings of the ECOOP'97 workshop, Finnland, June 1997.

[13] H. Frithschi, S. Gatziu, and K.R.Dittrich, "Framboise-an approach to construct active database mechanisms," Technical Report 97-04, Department of CS, University of Zurich, 1997.

[14] J. Mylopoulos, et al., "A generic integration architecture for cooperative information systems," proceedings of the 1st IFCIS (CoopIS'96) IEEE 1996, pp.208-217.

[15] E. N. Hanson and S. Khosla, "An introduction to the triggerman asynchronous trigger processor," Technical Report TR-97-007, CISE Department, University of Florida, 1997.

[16] S. Chakravarthy, R. Le, and R. Dasari, "ECA Rule Processing in Distributed and Heterogeneous Environments," proceedings of 14th ICDE, 1998.

[17] C. Collet, et al., "Open Active Services for Data-Intensive Distributed Applications," IDEAS'2000, IEEE, September, 2000.

[18] Minsoo Lee, Stanley Y. W. Su, and Herman Lam, "Event and Rule Services for Achieving a Web-Based Knowledge Network," Technical report UF CISE TR00-002, University of Florida, 2000.

[19] I. Ben-Shaul and S. Ifergan, "WebRule: An Event-based Framework for Active Collaboration among Web Servers," proceedings of WWW'97, 1997, pp.521-531.

[20] M. cilia and A. P. Buchmann, "An Active Functionality Service for E-Business Applications," ACM SIGMOD Record, vol.31, No.1, pp.24-30, March 2002.

[21] Zecong Song, "A Generalized Approach for Extending the Active Capability of RDBMS," MS Thesis, CISE, University of Florida, 2000.

[22] Ganesh A. Gopalakrishnan, "Making Sybase Fully Active : Supporting Composite Events and Prioritized Rules," MS Thesis CSE 2002-5, Department of Computer Science and Engineering, University of Texas, Arlington, 2002.

이 동 우

고려대학교 전자공학과 졸업(공학사). 고려대학교 전자공학과(공학석사). 고려대학교 컴퓨터학과(이학박사). 1995년~현재 우송대학교 컴퓨터학과 부교수. 관심분야는 분산처리 및 분산시스템, 데이타베이스

이 성 훈

한남대학교 컴퓨터공학과 졸업(학사). 고려대학교 컴퓨터학과 졸업(석사). 고려대학교 컴퓨터학과 졸업(박사). 1998년~현재 천안대학교, 정보통신공학부 부교수 관심분야는 인공지능, Bioinformatics, 분산시스템

황 종 선

고려대학교 수학과(학사, 석사). University of Georgia, 전산학 박사. 1982년~현재 고려대학교 컴퓨터학과 교수. 관심분야는 Mobile Computing, 분산처리 및 분산시스템