능동문서에 대한 새로운 접근법과 그 응용
(A New Approach to Active Documents and its Application)

남 철 기 † 배 재 학 †† 장 길 상 †††
(Chul-Ki Nam) (Jae-Hak Bae) (Gill-Sang Jang)

요 약 웹은 중요한 정보원천이며, 대부분의 웹 응용프로그램은 HTML로 작성된 서식문서를 기반으로 한다. 이러한 HTML 기반의 서식문서는 사용자 인터페이스를 제공하는 역할을 할 뿐, 문서서식 설계자가 지향하는 업무처리 절차나 로직을 내포하지는 않는다. 그러나 서식문서에는 그것에 대한 처리방법이 함축되어 있고, 이렇게 내재된 절차적 지식을 업무처리 과정의 자동화에 적극적으로 활용할 수 있다. 이에, 본 논문에서는 서식문서에 기반한 업무처리 절차를 자동화시키기 위해 인지과학적인 측면에서 문서의 능동성을 파악하였다. 이를 통해 능동문서(Active Documents)에 대한 새로운 개념과 그 적용 가능성을 제시하고자 한다. 이 능동문서는 문서에 함축되어 있는 업무규칙과 문서처리의 자동화를 지원하기 위한 선언적 지식을 문서 자체에 내포하고 있다. 또한, 본 논문에서는 제시된 능동문서를 처리하기 위한 프레임워크를 제안한다. 제안된 프레임워크는 크게 생성단계와 실행단계로 구성된다. 본 논문에서 제안한 프레임워크의 유용성을 보이기 위해, 인터넷 구매 시스템의 구매 요청서 처리에 능동문서를 적용한 ActiveForm이라는 원형시스템을 설계하고 구현하였다. 그 결과, 본 논문은 서식설계자의 지식이 Prolog로 명시적으로 표현되어 있는 능동문서를 추론엔진에서 처리함으로써 인터넷 응용프로그램의 지능화에 기여할 수 있음을 확인하였다.

키워드 : 능동문서, 업무규칙, WfMS, XML, 논리 프로그래밍

Abstract The web is an important source of information and most of Web applications are based on form documents in HTML. These HTML based form documents only play a role as user interfaces, and they do not involve the procedures or rules of business process which form document designers assume. However, form documents imply methods for treating documents, and these embedded procedural knowledge can be utilized actively in automation of business process. In this respect, we investigate the activeness of documents with cognitive science to automate business processes based on form documents. Through this, we have a new concept and applicability of active documents. Our active documents include business rules and declarative knowledge to support the automation of document processing. Also, we propose a processing framework for the active documents. The framework has two phases: build time and run-time. In order to demonstrate the usefulness of the proposed framework, a prototype called ActiveForm is designed and implemented for requisition processing of an Internet Procurement system. The result shows that our approach which explicitly represent knowledge of form designer in active documents with Prolog and processing them in an inference engine can enhance the intelligence of Internet applications.

Key words : Active document, Business rule, WfMS, XML, Logic programming

1. 서 론

· 이 논문은 2002년 울산대학교의 연구비에 의하여 연구되었음.
† 정 회 원 : 울산대학교 컴퓨터·정보통신공학부
cholki.nam@oracle.com
†† 종신회원 : 울산대학교 컴퓨터·정보통신공학부 교수
jhjbae@ulsan.ac.kr
††† 비 회 원 : 울산대학교 경영학부 교수
gsjang@ulsan.ac.kr
논문접수 : 2002년 10월 15일
심사완료 : 2003년 1월 28일

웹은 정보원천으로서 중요한 역할을 하며, 대부분의 웹 응용프로그램은 문서중심(Document-Centric)이다. 전통적으로 정보를 표현하고 교환하며 저장하기 위해 주로 문서를 사용한다. 문서 중에서 특히 서식문서(Form Document)는 사용자 인터페이스, 데이타 구조, 트랜잭션 레코드를 효과적으로 표현할 수 있기 때문에 널리 사용된다. 조직내의 대부분 업무 프로세스는 이러한 문서처리 흐름을 반영하고 있다[1]. 정보처리의 매체

로서 문서의 중요성은 컴퓨팅 패러다임에도 영향을 주고 있다. 기존의 처리지향(Process-oriented) 및 객체지향(Object-oriented) 컴퓨팅에서 문서지향(Document-oriented) 컴퓨팅으로 전환되고 있다[2].

일상적으로 사용하고 있는 업무용 종이 서식문서에는 문서 설계자가 문서에 반영하고자 하는 능동적인 행위가 함축되어 있다. 그러나 이러한 능동적인 행위는 현재 많이 사용되는 HTML 기반의 서식문서에는 간과되어 있다. HTML로는 문서의 외양(Presentation)과 내용(Content)만을 기술할 수 있다. 따라서 이러한 HTML 기반의 서식문서로는 사용자 인터페이스와 같은 수동적인 역할을 할 수 있을 뿐, 문서의 서식 설계자가 염두에 둔 업무처리 절차나 로직을 내포하기가 어렵다. 이를 개선하여 (1) 서식문서에 함축된 처리과정을 명시적으로 표현하고, (2) 문서에 내재된 절차적 지식과 업무규칙을 업무처리 과정의 자동화에 적극적으로 활용함으로써, 문서처리의 효율성을 기대할 수 있다.

이러한 취지로, 본 논문에서는 (1) 능동문서에 대한 개념을 정제하였으며, (2) 이 개념에 입각한 능동문서를 처리하기 위한 프레임워크를 제안하고, (3) 이 프레임워크의 유용성을 ActiveForm 이라는 구매요청서 처리시스템의 구현을 통하여 확인하였다. ActiveForm은 인터넷 구매조달 시스템의 하부구조를 형성한다. 이러한 능동문서는 동적인 특성을 가지고 있을 뿐만 아니라, 응용프로그램 인터페이스와 유사한 상호작용하는 요소를 프로그램 형태로 포함하고 있다[3]. 일반적으로 프로그램과 문서는 다르지만, 능동문서에는 이러한 이질적인 두 요소를 함께 내포하고 있다. 문서에 대한 능동적 접근법을 통해 웹 응용프로그램의 지능화에 기여할 수 있다고 본다.

2. 연구동기

일반적으로 문서는 행위자체에 초점을 둔 기능적인 관점과, 문서의 제목, 장, 단락 등과 같은 외형에 주안점을 둔 물리적인 관점으로 나누어 볼 수 있다[4]. 문서의 기능적인 관점에서 볼 때, 문서는 수동문서(Passive Document)와 능동문서로 대별된다[2]. 수동문서는 내용과 구조화된 표현만을 가지고 있는 문서이다. 한 예로 HTML은 웹 브라우저 내에서 수동문서를 표현하기 위해 사용된다. 능동문서는 내용과 구조화된 표현 뿐만 아니라, 문서처리의 동적인 특성 및 응용프로그램 인터페이스와 유사한 상호작용하는 요소를 포함하고 있다.

본 논문에서는 인지과학적인 측면에서 문서의 능동성을 파악하고자 한다. 즉, 일반적으로 자료처리의 대상이 되는 문서는 수동적인 것이 아니고, 그 문서의 설계자가 문서 작성자와 상호작용을 할 때 필요한 지식을 함축하고 있다는 점에서 능동적이라고 본다. 이러한 생각을 검증하기 위해 종이 서식문서를 일차적인 실험대상으로 하였다. 그림 1은 구매주문과 관련된 종이 서식문서이다. 문서 사용자인 구매 요청자는 단순히 문자를 해독하여 그 문서를 작성하는 것이 아니다. 서식문서 기입에는 주어진 서식에 함축되어 있는 바, 서식설계자가 지향하는 업무처리과정 및 업무규칙을 추론하여 그의 의도를 능동적으로 재구성해 가는 이해과정이 필요하다. 서식문

구매 요청서 ( )신규 ( )갱신

구매자	상 호			전 화	
	담 당 자			팩 스	
	주 소				
품 목	제품코드	제 품 명	수 량	금 액	총 액
사용환경	O/S종류	Unix( ) Windows( ) Linux( )			
	사용자수				
구매자정보	상 호			대표자명	
	사업자 등록번호				
	업 태			종 목	

- 조건
① 납품일자 : 계약 후 15일 이내
② 대금 지불 : 설치 인도 후 30일 이내 현금
③ 구매 유효기간 : 구매 요청일로부터 30일
④ 신규인 경우에는 구매자 정보 항목을 전부 기입할 것

그림 1 구매요청 종이서식문서

서에는 데이타 무결성의 제약조건, 필드간의 의존관계, 그리고 업무규칙 등과 같은 서식 설계자가 문서에 반영하고자 하는 지식이 내포되어 있다. 이러한 지식은 문서에 명시적으로 표현되어 있지 않을 수도 있다. 따라서 서식 기입자는 문서작성 시에 서식에 함축되어 있는 선언적 또는 절차적인 지식을 이해할 필요가 있다.

그림 1의 구매요청서 하단에 보이는 "조건"은 판매정책으로서 서식문서의 제약조건과 업무규칙을 나타낸다. 이는 본 서식에 대한 서식설계자의 의도가 명시적으로 표현되어 있다. 자연어로 서식문서의 제약조건과 업무규칙을 기술할 수 있지만, 문서처리를 자동화하기 위해서는 서식설계자와 기입자의 다양한 배경 및 도메인 지식이 문서에 명시적으로 반영되어 있어야 한다. 서식문서의 처리 자동화를 성취하기 위해서는 (1) 서식에 함축된 문서처리 과정을 기계가 읽고 이해할 수 있어야 하며 또한, (2) 서식과 관련된 업무처리 과정을 묘사하는데 사용할 지식표현 언어와 추론엔진이 필요하다. 이에 본 연구는 서식문서 처리를 위한 자동화 요건을 만족시키기 위하여 능동문서 개념을 새롭게 정의하고, 능동문서 처리를 위한 프레임워크를 제안한다.

3. 능동문서의 모형화

3.1 능동문서 모델

본 논문에서 제안하는 능동문서는 그 내부에 선언적 지식을 포함하며 문서제어와 처리에 대한 자동화를 지원하는 문서이다. 이러한 능동문서를 구성하는 요소는 문서의 구조, 표현, 데이타 그리고 행위이며 각각 XML로 일관되게 표현되어 문서자체에 함께 포함되는 서식문서이다. 본 논문에서 제안하는 능동문서 모델은 그림 2와 같다.

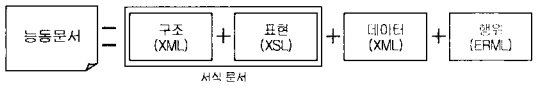

그림 2 능동문서 모델

- 구조(Structure)
  문서의 논리적인 구조이며 XML로 표현된다. 즉, 서식의 필드집합과 그 집합을 구성하는 필드명, 필드형태(예를 들면, TEXT, SELECT, TEXTAREA, RADIO, CHECKBOX)와 필드길이를 정의하는 부분이다. 각 필드는 유일한 ID를 가지고 있다.
- 표현(Presentation)
  XML로 표현된 문서의 논리적 구조를 시각적으로 표현하는 XSL문서이다. 구조를 표현하는 XML문서에 표현을 나타내는 XSL문서를 적용하여 HTML 서식문서를 생성한다. 이 서식문서는 사용자 인터페이스의 역할을 한다.
- 데이타(Data)
  데이타는 사용자가 서식문서에 입력한 데이타를 의미하며 XML문서로 저장된다. 이 XML문서는 Prolog의 추론엔진에서 문서처리를 위해 Prolog Goal을 생성할 때 사용되며 또한, 작업흐름(Workflow) 호출을 위해 사용된다.
- 행위(Behavior 또는 Activity)
  행위란 업무문서에 함축되어 있는 문서에 대한 처리 방법을 의미하며, (1) 일반적인 계산 로직, (2) 업무규칙과 같이 업무 프로세스를 정의하고 제어하는 규칙, (3) 데이타 검증에 필요한 데이타 무결성의 제약조건, 그리고 (4) 작업흐름 수행 등이 있다. 이렇게 내재된 절차적 지식을 업무처리 과정의 자동화에 적극적으로 활용하기 위해 지식표현에 효과적인 Prolog 규칙으로 기술하며, 이종의 규칙기반 시스템과 지식을 공유하고 교환하기 위해 ERML 문서로 변환한다. ERML을 통한 구체적인 행위의 표현방법은 다음 절에서 자세히 기술한다.

3.2 행위 표현언어 : ERML(Executable Rule Markup Language)

본 논문에서는 문서의 능동적인 행위를 나타내는 업무규칙과 데이타 무결성의 제약조건을 처리하기 위해 수행가능한 명세 언어(Executable Specification Language)로서 Prolog를 사용하였다. Prolog와 XML과의 관계[5]를 파악함으로써 ERML[6]을 고안하였다. ERML은 Prolog 규칙의 XML 표현이며 제안하는 프레임워크 기반하에서 수행이 가능한 규칙 표시언어이다. XML 문서는 요소(Element)로 구성되어 있고 각각의 요소는 <태그> ... </태그>의 형식이다. Prolog 규칙을 XML형식으로 변환하기 위해서는 Prolog에서 사용하는 상수, 변수, 구조체와 같은 Herbrand 항(Term)과 사실(Fact), 규칙(Rule)과 같은 Horn 절(Clause)을 XML 요소로 변환하기 위한 명명규칙이 필요하며 표 1과 같다.

표 1 Prolog-XML 요소 명명규칙

Herbrand 항	Element	Horn 절	Element
원자	<atom>	술어	<functor>
숫자	<number>	관계기호	<functor>
논리변수	<variable>	사실	<hn><relationship>
구조체	<structure>	규칙	<hn><relationship>

그림 3은 표 1의 명명규칙에 근거해서 ERML의 DTD (Document Type Definition)를 나타낸다.

```
<!ELEMENT   ruleset      (hn) >
<!ELEMENT   hn           (relationship*) >
<!ELEMENT   relationship (functor, relationship*,
                         (variable|atom|number)*) >
<!ELEMENT   functor      (#PCDATA) >
<!ELEMENT   variable     (#PCDATA) >
<!ELEMENT   atom         (#PCDATA) >
<!ELEMENT   number       (#PCDATA) >
```

그림 3 ERML의 DTD

한 예로 "고객번호가 100이상 200이하이면 우수고객이다." 라는 Prolog 규칙은 다음과 같다.

PremiumCustomer(CustomerNumber):-CustomerNumber > 100, CustomerNumber < 200.

위의 Prolog 규칙은 표 1의 명명규칙에 의해 다음과 같이 변환되며, Prolog2XML[6] 변환기를 통해 자동적으로 생성된다.

```
<?xml version="1.0" encoding="euc-kr"?>
<!DOCTYPE ruleset SYSTEM "erml.dtd">
<ruleset>
    <hn>
        <relationship>
            <functor> :- </functor>
            <relationship>
                <functor> premiumCustomer </functor>
                <variable> CustomerNumber </variable>
            </relationship>
            <relationship>
                <functor> , </functor>
                <relationship>
                    <functor> &gt; </functor>
                    <variable> CustomerNumber </variable>
                    <number> 100 </number>

                </relationship>
                <relationship>
                    <functor> &lt; </functor>
                    <variable> CustomerNumber </variable>
                    <number> 200 </number>
                </relationship>
            </relationship>
        </relationship>
    </hn>
</ruleset>
```

그림 4 ERML의 예

그림 4의 ERML 문서는 XML2Prolog[7, 8] 변환기를 통해 원래의 Prolog 규칙으로 변환되어 Prolog의 추론엔진에서 수행이 가능하다. 이 문서는 XML로 규칙을 표현하고 있으므로 이종의 규칙기반 시스템과 규칙을 공유하기가 용이하다. 또한, 업무 프로세스의 자동적인 처리를 위해 WfMS와 연동도 가능하다.

4. 능동문서 처리를 위한 프레임워크

본 장에서는 앞 장에서 논의한 능동문서를 처리하기 위한 전반적인 프레임워크에 대해 기술한다. 프레임워크의 전체 구조는 그림 5와 같다.

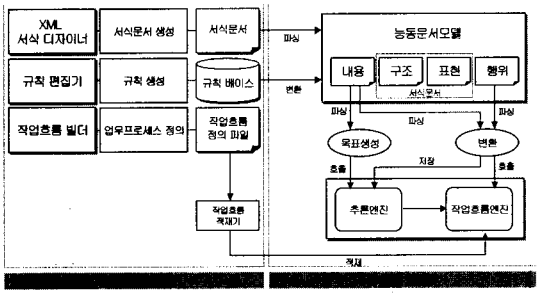

그림 5 능동문서 처리를 위한 프레임워크

제안된 프레임워크는 크게 생성단계와 실행단계로 구분된다. 생성단계에서는 본 논문의 실험을 위해 구현한 XML 서식 디자이너와 규칙 편집기, 그리고 오라클의 작업흐름 빌더[9]를 사용한다. 이 단계는 서식문서 생성 및 업무프로세스의 정의와 관련된 구성요소로 이루어져 있다. 구체적인 것은 5.3절에 기술되어 있다. 실행단계는 변환기 및 추론엔진, 그리고 작업흐름 엔진[9] 등과 같은 능동문서 처리와 관련된 구성요소로 이루어져 있다. 각 단계에 대하여 세부적으로 설명하면 다음과 같다:

• 생성단계

이 단계에서는 크게 세 가지 작업이 이루어진다. 먼저, 서식 디자이너를 이용하여 업무전문가가 WYSIWYG 환경에서 서식문서를 디자인 및 생성한다. 다음으로, 서식 디자이너에 포함되어 있는 규칙 편집기를 사용하여 업무규칙을 기술한다. 규칙 편집기로 업무전문가가 포인트와 클릭(Point & Click)과 목록상자(Listbox)와 같은 직관적인 인터페이스를 사용하여 쉽게 규칙을 만들 수 있다. 마지막으로, 작업흐름 빌더를 이용하여 업무 프로세스를 정의한다. 정의된 업무 프로세스는 작업흐름 적재기(Loader)에 의해 데이타베이스 또는 파일로 저장된다.

• 실행단계

이 단계에서는 크게 세 가지 변환 작업이 시스템에 의해서 자동적으로 수행된다. 먼저, 사용자가 서식문서에 입력한 데이타가 XML형식의 문서로 저장된다. 다음으로, 생성단계에서 규칙편집기로 생성한 업무규칙과 데이타 무결성의 제약규칙은 Prolog 규칙이므로, 이것을 지식의 공유 및 교환을 위하여 변환기를 통해 XML 문서로 저장된다. 따라서, 능동문서는 데이타와 업무규칙, 그리고 데이타에 대한 무결성 제약조건을 각각의 해당 파일로 가지고 있다. 마지막으로, 이러한 능동문서는 데이타 검증과 문서처리를 위해 Prolog의 추론엔진에서 수행된 후, 업무프로세스의 자동화와 효과적인 수행을 위해 WfMS와 연동되어 처리된다.

5. 원형 시스템의 설계 및 구현: ActiveForm

본 장에서는 제안된 능동문서 처리를 위한 프레임워크에 기반하여 기업간 거래(B2B)에서 가장 일반적인 구매관련 응용프로그램 중에서 구매요청서 처리와 관련된 ActiveForm이라는 원형시스템 구현절차에 대해 설명한다.

5.1 시스템 구조

서식 디자이너와 규칙편집기로 만들어진 웹 기반의 능동문서가 XML형식의 문서로 데이타와 규칙을 저장하고, 추론엔진에서 데이타 검증을 통해 WfMS와 연동되어 처리되는 ActiveForm 시스템의 구조는 그림 6과 같다.

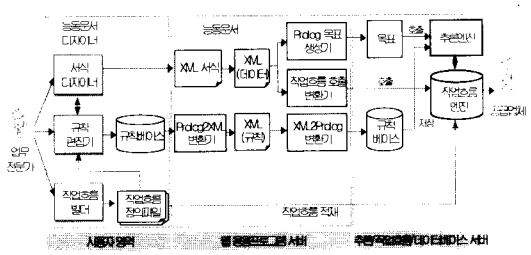

그림 6 ActiveForm 시스템 구조

5.2 시스템 구성요소

그림 6의 ActiveForm 시스템 구조를 변환기, 생성기, 그리고 엔진 부분으로 나누어서 설명하면 아래와 같다.

• 변환기

(1) Prolog2XML 변환기: 업무규칙과 데이타 무결성의 제약조건을 표현하는 Prolog 규칙을 ERML이라는 수행 가능한 규칙 마크업 언어로 변환한다. ERML은 이종의 규칙기반 시스템과 WfMS사이에서 규칙을 교환하기에 적당한 언어이다.

(2) XML2Prolog 변환기: Prolog2XML에 의해 변환된 ERML을 Prolog의 추론엔진에서 수행하기 위해 DCGs(Definite Clause Grammars)를 이용하여 원래의 Prolog 규칙으로 변환하다. DCG는 다양한 구문분석 응용프로그램에 대해 문법적인 관계를 표현하는데 효과적인 방법을 제공한다.

(3) 작업흐름 호출(Workflow Call) 변환기: 작업흐름 엔진에 적재되어 있는 업무 프로세스를 수행하기 위해서 작업흐름 엔진 API를 이용하여 작업흐름을 수행 하는 역할을 한다.

• 생성기

(1) Prolog Goal 생성기: XML2Prolog에 의해 변환되어 Prolog 지식베이스로 저장되어 있는 Prolog규칙에 대한 데이타 무결성 검증과 문서처리를 위한 Goal을 생성한다. 이러한 Goal은 데이타가 저장되어 있는 XML 문서에 XSLT를 적용하여 생성된다.

• 엔진

(1) 추론엔진: 역방향(Backward) 추론 엔진을 가지고 있는 SWI-Prolog[10]를 사용하였다.

(2) 작업흐름 엔진: 오라클 데이타베이스 서버에 포함되어 있는 작업흐름 엔진을 사용하였다. 이 엔진은 작업흐름 빌더에 의해 생성된 업무프로세스를 수행하며 또한, 업무 프로세스 사이의 각각 활동상태를 감시한다.

5.3 능동문서의 생성

사용자영역에서는 서식 디자이너를 사용하여 문서생성자인 업무전문가가 문서를 생성한다. 생성된 문서는 문서의 구조와 표현을 포함하는 XML 문서로 각각 저장된다. 그림 7에서 서식 디자이너로 작성된 문서는 웹

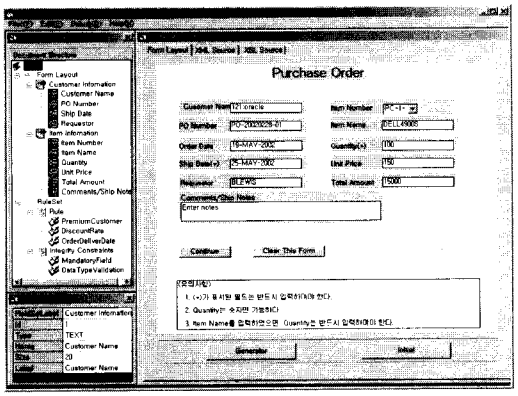

그림 7 서식 디자이너(구매 요청서식 작성)

표 2 판매정책 (Sales Policies)

규 칙	표 현	
(규칙 1) 고객등급	의 미	고객번호가 100이상 200이하이면 우수 고객이다.
	Prolog 규칙	***premiumCustomer***(CustomerNumber) :- CustomerNumber > 100, CustomerNumber < 200.
(규칙 2) 할인율	의 미	우수고객이면서 총 구매금액이 1000만원 이상이면 15%를 할인한다. 그 외의 경우에는 10%를 할인한다.
	Prolog 규칙	***discountRate***(CustomerNumber, OrderNumber, OrderDesc, ItemNumber, Quantity, DeliverDate, TotalAmount, Requestor, '15.0 percent') :- ***premiumCustomer***(CustomerNumber), TotalAmount > 1000, !. ***discountRate***(CustomerNumber, OrderNumber, OrderDesc,ItemNumber, Quantity, DeliverDate, TotalAmount, Requestor, '10.0 percent').
(규칙 3) 승인여부	의 미	모든 구매주문은 반드시 공급업체 승인자의 결재(승인, 거절)가 필요하다.
	Prolog 규칙	***needOrderNotice***(CustomerNumber, OrderNumber, OrderDesc,ItemNumber, Quantity, DeliverDate, TotalAmount, Requestor) :- ***purchaseOrder***(CustomerNumber, OrderNumber,ItemNumber, Quantity).
(규칙 4) 납기일	의 미	우수고객 또는 총 구매금액이 1000만원 이상이면 납기일이 구매일자로부터 10일이다. 그 외의 경우에는 납기일이 20일이다.
	Prolog 규칙	***orderDeliverDate***(CustomerNumber, OrderNumber, OrderDesc,ItemNumber, Quantity, DeliverDate, TotalAmount, Requestor,'day10') :- ***premiumCustomer***(CustomerNumber) ; TotalAmount > 1000. ***orderDeliverDate***(CustomerNumber, OrderNumber, OrderDesc,ItemNumber, Quantity, DeliverDate, TotalAmount, Requestor,'day20').

표 3 데이타 무결성의 제약조건 (Data Integrity Constraints)

규 칙	표 현	
(규칙 5) 필수필드	의 미	(*)로 표시된 필드는 반드시 입력하여야 한다.
	Prolog 규칙	***mandatoryField***(CustomerNumber, OrderNumber, OrderDesc,ItemNumber, Quantity, DeliverDate, TotalAmount, Requestor, '납기일자를 반드시 입력하세요') :- ***var***(DeliverDate).
(규칙 6) 데이타형	의 미	수량(Quantity)은 숫자(Number)값만 가능하다.
	Prolog 규칙	***datatypeValidate***(CustomerNumber, OrderNumber, OrderDesc, ItemNumber, Quantity, DeliverDate, TotalAmount, Requestor, '숫자만 입력하세요') :- ***not***(number(Quantity)).
(규칙 7) 필드간 상호 의존관계	의 미	품목명(Item Name)이 입력되었으면 수량(Quantity)은 반드시 입력되어야 한다.
	Prolog 규칙	***interrelationship***(CustomerNumber, OrderNumber, OrderDesc, ItemNumber, Quantity, DeliverDate, TotalAmount, Requestor, '수량은 반드시 입력하세요') :- ***nonvar***(ItemNumber), var(Quantity).

브라우저를 통해 동일한 인터페이스를 제공하게 된다.

위의 구매 요청서는 다음 표에서처럼 공급업체의 판매정책을 내포하고 있으며 데이타 무결성의 제약조건을 명시적으로 표현하고 있다.

상기와 같은 공급업체의 판매정책에 근거해서 그림 7의 고객은 우수고객이며 15%의 할인을 받고 납기일이 주문날짜로부터 10일이 된다는 것을 알 수 있다. 또한, 데이타 무결성의 제약조건에 위배되지 않게 데이타를 입력하였다. 표 2와 표 3의 Prolog 규칙표현은 그림 8과 같이 규칙 편집기에 의해 생성된다.

그림 8에서는 판매정책의 여러 가지 규칙가운데 (규칙 2)인 할인율 규칙을 작성하고 있다. 규칙편집기를 통

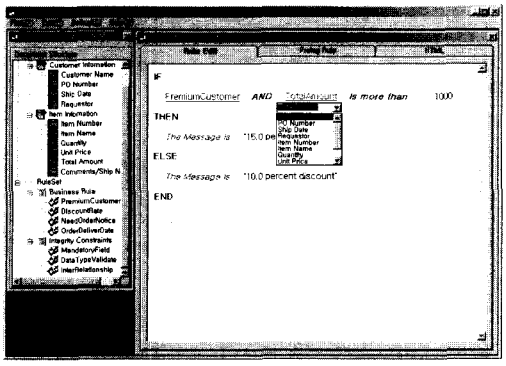

그림 8 규칙 편집기

해 업무규칙은 각각 추론엔진에서 추론 가능한 Prolog 규칙과 인간이 이해하기 쉬운 HTML 문서로 변환되어 저장된다. 업무규칙과 서식문서를 생성한 다음에는 WfMS와 연동이 필요한 규칙을 위해 업무프로세스를 정의한다. 표 2의 (규칙 3)은 공급업체 승인자의 통지가 필요하다. 본 논문에서는 (규칙 3)을 처리하기 위해 오라클 작업흐름 빌더를 사용하여 업무 프로세스를 정의한 후에 작업흐름 엔진에 적재하였다.

5.4 능동문서의 실행

능동문서의 실행은 다음과 같이 크게 네 단계의 과정을 거친다: 1단계에서는 업무규칙을 포함하는 서식문서를 제안하는 능동문서 모델로 변환하는 단계이며, 2단계에서는 1단계에서 생성된 ERML을 Prolog 규칙으로의 변환한다. 그리고 3단계에서는 추론엔진을 통한 데이타 검증 및 문서처리를 하는 단계이며, 마지막 4단계에서는 작업흐름 엔진에 저장되어 있는 작업흐름을 수행하는 단계이다. 각 단계에 대해 세부적으로 기술하면 다음과 같다.

- 1단계: 능동문서 모델로 변환

  이 단계에서 서식문서에 기입한 데이타는 data.xml로 저장된다. 그 다음으로는 Prolog 규칙으로 표현되어 있는 업무규칙과 데이타 무결성의 제약조건을 Prolog2XML 변환기를 통해 각각 business_rule.xml, integrity_constraint.xml로 저장되는 ERML 문서를 생성한다. 그림 9는 능동문서 모델의 요소 중에서 XML문서로 저장되는 데이타, 그리고 업무규칙과 데이타 무결성의 제약조건을 나타내는 행위를 표현한다. 나머지 요소인 구조와 표현은 사용자 인터페이스를 제공하는 서식문서이므로 그림 9에서는 표현되지 않았다.

그림 10은 그림 9를 웹 브라우저에서 확인한 모습이다. 사용자가 능동문서에 입력한 데이타는 <BusinessData>, 그리고 업무규칙은 <BusinessRule>, 데이타 무결성을 위한 제약조건은 <Constraints> 요소로 표현되어 있다.

```
<?xml version="1.0"?>
<!DOCTYPE active_doc [
<!ENTITY data SYSTEM "data.xml">
<!ENTITY business_rule SYSTEM
 "business_rule.xml">
<!ENTITY integrity_constraint SYSTEM
"integrity_constraint.xml">
]>
<Active_document>
&data;
&business_rule;
&integrity_constraint;
</Active_document>
```

그림 9 능동문서 모델의 XML 통합표현

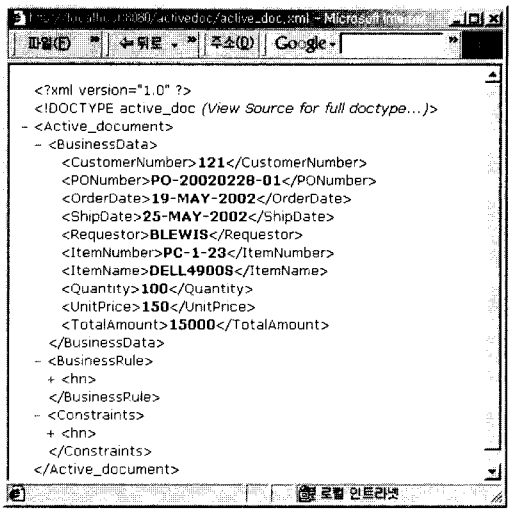

그림 10 능동문서의 XML 표현

- 2단계: ERML을 Prolog 규칙으로의 변환

  이 단계에서는 서식문서의 데이타 검증과 처리를 위해 XML2Prolog 변환기를 사용하여 ERML 문서(business_rule.xml, integrity_constraint.xml)로부터 원래의 Prolog규칙으로 변환하여 Prolog의 추론엔진에 저장한다.

- 3단계: 추론엔진을 통한 데이타 검증 및 문서처리

  데이타가 저장되어 있는 XML문서로부터 XSLT로 만들어진 PrologGoal 생성기를 사용하여 Goal을 생

```
:- tell('result.html').
:- write('<HTML>').
:- write('<BODY>').
...<중략>...
% 우수고객이면서 3000만원을 구매한 경우(할인율이 15%가
된다)
:- (discountRate(150, 'PO-100', 'PC구입', 'PC-114-32',
          100, '2002/10/05', 3000, 'james', Message),
    format('<FONT COLOR="RED">할인율:</FONT>
    ~q~n',[Message]),write('<BR>'),
    fail
   ; true
  ).
...<중략>...
:- write('</HTML>').
```

그림 11 생성된 Goal의 예

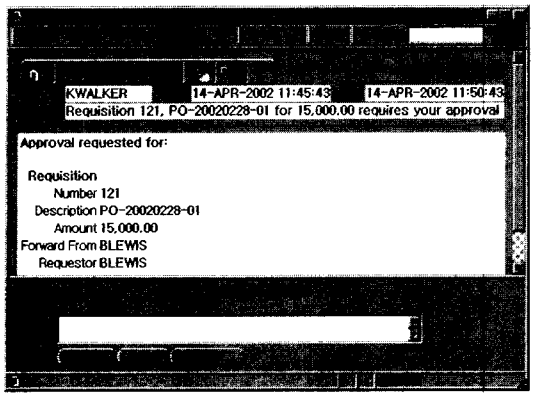

그림 12 작업흐름의 수행결과

성한다. 그림 11과 같이 생성된 Goal을 추론엔진에서 수행 시에, 만약 표 3의 데이타 무결성의 제약조건을 만족한다면 표 2의 판매정책에 근거하여 문서를 처리하게 된다. 처리결과는 result.html에 저장된다.

• 4단계: 작업흐름 수행

표 2의 판매정책에서 (규칙 3)의 승인여부 규칙을 처리하기 위해서 WfMS와 연동한다. 능동문서 생성단계에서 작업흐름 빌더로 정의된 업무 프로세스는 작업흐름 엔진에 적재되어 있다. 이러한 업무 프로세스를 수행하기 위한 방법으로 생성단계에서 오라클 작업흐름 엔진 API를 사용하여 프로세스를 기동하는 저장 프로시저(Stored Procedure)를 생성해 두었다. 이 저장 프로시저를 수행하면 엔진에 적재되어 있는 업무 프로세스를 수행하게 된다. 이를 위해 작업흐름 수행 변환기가 사용된다. 그림 12는 작업흐름의 수행결과이며 구매요청서 처리에 대한 공급업체 승인자의 작업목록에서 승인 결정을 받기 위해서 대기중인 세부 구매요청 정보를 보여준다.

6. 관련연구의 비교

여러 연구에서 능동문서의 개념을 사용하였다[11, 12]. 하지만, 능동문서의 응용 영역에 따라 서로 다른 개념을 사용한다. 접근방법에 의해 나누어 볼 때 클라이언트 영역에서 능동문서처리를 하는 연구와 서버 영역에서 처리하는 연구로 나누어 볼 수 있다. 또한, 전자출판, WfMS, 그리고 메일 시스템에 능동문서의 개념을 도입하기도 하였다. 문서의 능동적인 행위를 표현 및 처리하기 위해 문서자체에 처리 로직을 포함시킨 대표적인 연구는 표 4와 같으며 ActiveForm이 본 논문의 접근법에 해당한다.

표 4에 기술된 바와 같이 비록 기존의 능동문서에 대한 연구가 스크립트언어나 자바언어를 사용하여 문서의 능동적인 행위를 표현하지만 다음과 같은 약점을 내포하고 있다: (1) 문서의 행위를 표현하기 위해 절차적인 언어만 사용하였으며 특정 언어에 종속되어 있다. 따라서 다른 능동문서 처리 시스템과 호환성을 유지하기가 어렵다. (2) 능동문서를 구성하는 요소들을 표현하기 위

표 4 능동문서에 대한 접근법 비교

연구 / 비교항목	AHDM(Active Hypertext Document Model) [13] : Eckhart K ppen)	Displet[14] : Luca Bonfiani	Shadow & Jima[15] : Patrik Werle	ActiveForm (본 논문)
개 념	능동문서는 구조화되고 하이퍼링크된 정보를 하이퍼 텍스트 문서에 포함한다. 또한, 스크립트 또는 프로그램의 일부 형태로 연계된 절차적인 정보를 포함하고 있는 문서이다.	능동문서는 동일한 문서가 화면상에 출력, 검색, 애니메이션, 계산 등과 같은 자동적이며 능동적인 행위를 지원할 수 있는 문서이다.	능동문서에는 정보 저장과 표출에 관련된 코드가 내장된다. 이 코드는 문서자체의 내용과 그 용도를 반영하며, 상황에 맞는 정보저장과 표출에 관련된 행위를 할 수 있게 한다.	능동문서는 문서서식 설계자가 문서에 반영하고자 하는 업무처리 지식을 함축하고 있다. 지식표현은 규칙으로 하며 문서의 구성요소 (구조, 내용, 표현, 행위) 를 각각의 XML파일로 표현한다.
사용 기술	XML, Xlink, Xpointer, CSS, DOM	XML, XSLT, Java	Jini	XML, XSLT, JSP, Prolog
기본 사상	응용프로그램 로직을 포함한 하이퍼텍스트 문서를 통해 응용프로그램 구현이 가능하다.	XML 요소에 행위를 수행할 수 있는 자바 클래스를 연결시킬 수 있다.	문서는 정보 저장과 표출에 관련된 작업에 능동적으로 참여하므로 여러 서비스를 제공할 수 있다.	서식 문서에는 문서 서식 설계자자 의도한 문서의 취급법이 함축되어 있다.
행위 표현 언어	스크립트 언어(Tcl/OTcl)	Java 언어	Java 언어	ERML (Executable Rule Markup Language)
적용 가능 분야	컴퓨터기반의 작업흐름 시스템	다중-문서-에이전트 응용프로그램	이동컴퓨팅 환경	지능적인 웹 응용프로그램 규칙기반 어플리케이션

표 5 능동문서 행위 표현 언어별 특징 비교

특 징	HTML	Script	Java	HTML+Java	XML+XSL	(본 논문) XML+XSL+ERML
폼구조, 로직, 데이터의 문서내 포함	×	△	△	△	×	○
작은 파일크기(ASCII)	○	○	×	△	○	○
빠른 개발	○	×	×	△	○	○
데이터검증 및 처리자동화	×	○	○	○	×	○
업무규칙 관리	×	△	○	○	△	○
업무변경사항 수정의 유연성	×	△	△	△	△	○

(○ : 지원 △ : 다소 지원 × : 어려움)

표 6 능동문서처리 시스템의 비교

시스템 / 비교항목	Active Tioga: Edit Notifications[17]	Scripted Documents[18]	Interleaf[16]	Quill[19]	ActiveForm (본 논문)
행위 (Activity)	일반적인 계산	일반적인 행위(action)의 순서	일반적인 계산	SGML의 복잡한 의미 루틴	• 일반적인 계산 • 데이타 검증 • 업무규칙에 따른 문서처리 • WfMS의 작업흐름 수행
트리거링 (Triggering)	사용자 행위; 문서 열기, 스크롤, 편집	명시적으로 스크립트 동작을 시작할 때	출력, 편집, 선택, 메뉴를 클릭할 때	출력, 공유데이타 수정시	• 실행요구 발생시
코드 위치 (Code location)	별도의 등록된 프로시저에 저장됨	스크립트는 별도의 데이타베이스에 저장	문서	디자인 문서 (문서 스타일)	• 별도의 외부 XML 파일에 저장되며 능동문서에서 참조되어 포함됨
문서모델 (Document model)	• 노드 구조 WYSIWYG • 풍부한 확장성	• 메타 문서 편집기	• 계층적, 확장가능한 WYSIWYG	• 계층적 SGML WYSIWYG	• 계층적 XML WYSIWYG • 문서는 구조, 내용, 표현, 행위를 나타내는 각각의 XML 파일로 구성됨

해 서로 다른 언어를 사용하였다. 즉, 문서의 내용은 HTML, 표현은 HTML 또는 스타일시트, 그리고 행위는 스크립트 또는 자바 언어를 사용함으로써 일관되게 하나의 언어로 표현하지 않았다. 마지막으로, (3) 자바와 스크립트 언어로 문서의 능동적인 행위를 표현하고 수정하는 것은 시간적인 비용이 많이 발생한다. 따라서, 현재와 같은 급변하는 비즈니스 환경에서 사용하기에는 유연성이 부족하다. 이 약점을 해결하기 위해 본 논문에서는 Prolog와 XML 기반으로 능동문서를 처리하였다. 표 5에서는 기존의 절차적 언어와 마크업 언어를 다섯 가지 특징에 대해 지원여부를 비교하였다. 본 논문의 접근법(XML+XSL+ERML)이 업무문서가 갖추어야 할 중요한 특징을 지원한다는 것을 알 수 있다.

표 6에서는 Terry 와 Bakers가 제시한 능동문서의 주요문제[16]에 근거해서, 본 논문의 원형 시스템인 ActiveForm과 대표적인 능동문서 처리 시스템과 비교하였다.

한편, 문서의 능동성이 표출될 비즈니스 내용의 패턴 및 다양성에 대한 한계는, 업무 프로세스를 선언적 업무규칙으로 명세한 관련연구를 통해 가늠할 수 있다. 이들 연구는 (1) 거래 당사자간의 계약에 대한 규칙 중에서 환불정책, 납기일, 할인정책에 관한 것을 예시하였으며[20], (2) 인터넷 서점의 가격정책과 회원등급 관리를 위한 업무로직을 60개의 규칙으로 표현하기도 하였다[21]. 또한, (3) 업무규칙을 무결성 제약규칙, 유도규칙, 반응규칙 등으로 분류하였다[22]. 이들 대부분은 IF-THEN 형식으로 표현할 수 있으며, 이는 Prolog 규칙으로 효과적으로 표현 및 처리가 가능하다. 이러한 선언적 업무규칙은 본 논문의 표 2, 3에서 예시한 것과 크게 다르지 않다. 따라서 능동문서의 응용범위는 업무프로세스 명세화 수준에 따라 크게 확장될 수 있음을 알 수 있다.

7. 결론 및 향후연구

본 논문에서는 웹 기반의 서식문서를 효과적으로 처

리하기 위하여 능동문서의 개념을 새로이 정립하였고, 제시된 능동문서를 처리하기 위한 프레임워크를 제안하였다. 제안된 능동문서의 처리를 위한 프레임워크에 근거하여, 인터넷 구매 시스템의 구매요청서 처리에 능동문서를 활용한 *ActiveForm*이라는 원형 시스템을 설계하고 구현하였다. 그 결과, 본 논문에서 제시한 접근법은 서식문서 기반의 웹 응용프로그램 구축시 다음과 같은 일곱 가지의 장점이 있다: (1) XML을 단순히 자료교환 목적으로 사용하는 것이 아니라 문서에 함축되어 있는 지식교환 용도로 사용함으로써, 이종의 규칙기반 시스템간의 지식공유를 더욱 용이하게 하였다. (2) 문서 모델을 내용, 구조, 표현, 행위의 네 가지 구성요소로 분리함으로써 문서 요소의 재사용성을 향상시킬 수 있다. (3) 업무에 대한 요구사항의 수정이 필요할 때, 서식문서 설계자가 개발자에게 업무 로직의 수정을 요청하지 않고 바로 업무로직을 수정/변경함으로써 업무요구사항을 유연하게 만족시킬 수 있을 것으로 기대할 수 있다. 또한, Prolog를 사용하여 선언적으로 정의된 업무 규칙은 프로그래밍 언어로 복잡한 구현 없이 즉시 수행이 가능하므로 급변하는 비즈니스 환경에 유연하게 대처할 수 있다. (4) 업무규칙을 포함하는 XML문서를 WfMS와 연동시킴으로써, 문서처리의 자동화를 가능하게 한다. (5) 바람직한 전자문서 요건은 문서에 함축된 의미를 기계가 읽고 이해할 수 있으면서 또한, 자연어에 못지않은 의미 표현력이 필요하다. 이를 위해 Prolog를 사용하여 문서에 함축되어 있는 지식을 선언적으로 표현함으로써 논리 프로그램언어의 장점인 선언적 의미와 절차적 의미의 해석을 가능하게 하였다. 그러므로 절차적 언어(Java, C, Script Language)에 비해 처리의 유연성과 풍부한 지식표현, 그리고 지식표현의 표준화와 인간의 이해력 향상을 도모할 수 있다. (6) 본 논문에서는 구매요청서 처리를 위해 표 2와 표 3과 같이 단지 7개의 업무규칙만을 사용하였다. 이는 요청서 처리를 위해 절차적인 언어를 사용하는 일반적인 방법보다 개발 복잡성이 덜 하며 개발 및 유지보수에 드는 시간적인 비용을 절감할 수 있다. 마지막으로, (7) 문서처리에 논리프로그래밍 언어를 사용함으로써 문서처리의 지능화에 기여하였고, 이를 통해 기업의 지능적인 응용프로그램 개발의 토대를 제공하였다.

향후 연구방향으로는, 현재 구매요청서 처리에 국한한 원형 시스템을 규모가 큰 실제 웹 기반 시스템에 적용해 볼 수 있을 것이다. 또한, ERML을 현재 W3C에서 제안한 의미웹(Semantic Web)의 규칙마크업 언어로 확장할 수도 있다. 이러한 연구들은 현재 진행 중에 있으며 조만간 의미 있는 결과를 제시할 수 있을 것으로 기대한다.

참 고 문 헌

[1] Sprague and Ralph, "The Electronic Document: A New Frontier for Corporate IS", I.S. Business Partners, 1997.

[2] Paolo, C. and Robert, T. and Franco, Z., "Coordination Middleware for XML-centric Applications", 16th ACM symposium on Applied Computing, 2002.

[3] Quint, V., "Editorial, active document issue", Electronic Publishing, Vol.7, pp. 53-54, 1994.

[4] Buckland, M. K., "What is a document?", Journal of the American Society for Information Science, Vol.48, pp. 804-809. 1997.

[5] Harold Boley, "Relationships between Logic Programming and XML", Proc. 14th Workshop Logische Programmierung, W"urzburg, 2000.

[6] 남철기, 양재군, 배재학. "검증규칙을 포함한 XML문서", 한국 정보과학회 추계학술발표대회 논문집, 제28권, 제2호, pp. 709-711, 2001.

[7] 남철기, 배재학, "업무규칙을 포함한 능동문서", 한국정보과학회 춘계학술발표대회 논문집, 제29권, 제1호, pp. 352-354, 2002.

[8] Nam, C.-K. and Bae, J.-H. J., "A Framework for Processing Active Documents", The 6th Russian-Korean International Symposium On Science and Technology, Proc. KORUS-2002, Vol.1, pp. 122-125, 2002.

[9] Oracle Workfow, http://otn.oracle.com/products/integration/workflow/workflow_fov.html.

[10] SWI-Prolog's Home, http://www.swi-prolog.org/.

[11] Helena Ahonen et al, "Intelligent Assembly of Structured Documents", Technical Report C-1996-40, University of Helsinki, Department of Computer Science, 1996.

[12] Paul Dourish et al, "Extending Document Management Systems with User-Specific Active Properties", Transactions on Information Systems, ACM, Vol.18, No.2, 2000.

[13] Eckhart K. and Gustaf N., "Active Hypertext for Distributed Web Applications" Proc. The 8th IEEE International workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 1999.

[14] L. Bompani, P. Ciancarini, F. Vitali, "Active Documents in XML", SigWeb News letter, ACM, Vol.8, No.1, pp. 27-32, 1999.

[15] Patrik W. and Carl Gustaf Jansson, "Active Documents Supporting Teamwork in a Ubiquitous Computing Environment", PCC Workshop 2001 &

NRS 01, Nynashamn, Sweden, April 2001.
[16] English, P. M. and Tenneti, R., "Interleaf active documents", Electronic Publishing, Vol.7, No.2, pp. 75-87, 1994.
[17] Terry, D. B. and Baker, D. G., "Active Tioga documents: an exploration of two paradigms", Electronic Publishing, Vol.3, No.2, pp. 105-122, 1990.
[18] Zellweger, P. T. "Active paths through multimedia documents", Proc. The international Conference on Electronic Publishing, pp. 20-22, 1988.
[19] Chamberlin, D. D., Hasselmeier, H. F. and Paris, D. P. "Defining document styles for WYSIWYG processing", Proc. The international Conference on Electronic Publishing, Document Manipulation, and Typography, pp. 1-18, 1988.
[20] Benjamin N. Grosof, Y. Labrou, and H.Y. Chan. "A Declarative Approach to Business Rules in Contracts: Courteous Logic Programs in XML", Proc. The 1st ACM Conference on Electronic Commerce (EC-99), ACM Press, 1999.
[21] Kuldar Taveter, "Agent-Oriented Business Rules: Deontic Assignments", Proc. The international workshop on Open Enterprise Solutions: Systems, Experiences, and Organizations (OES-SEO2001), Rome, 2001.
[22] Benjamin N. Grosof, "DIPLOMAT: Compiling Prioritized Default Rules into Ordinary Logic Programs, for E-Commerce Applications", IBM Research Report, 1999.

남 철 기
1996년 울산대학교 전자계산학과 졸업(학사). 1999년 울산대학교 정보통신대학원 졸업(석사). 2002년 8월 울산대학교 대학원 컴퓨터·정보통신공학부 박사과정수료. 1996.년 1월~현재 한국오라클(주) 기술서비스본부 근무. 관심분야는 데이타베이스, 지식관리, 시맨틱웹, 웹서비스 등

배 재 학
1981년 중앙대학교 전자계산학과(이학사). 1983년 한국과학기술원 전산학과(공학석사). 2003년 포항공과대학교 컴퓨터공학과(공학박사). 1985년~현재 울산대학교 컴퓨터·정보통신공학부, 1996년 정교수. 관심분야는 자동문서요약, (자동, 논리)프로그래밍, 지식경영 및 기술, 전략경영정보시스템, 교육인적자원정보시스템 등

장 길 상
1986년 울산대학교 산업공학과 학사 1988년 한국과학기술원 산업공학 석사 1997년 한국과학기술원 경영정보공학 박사. 현재 울산대학교 경영학부 경영정보학전공 교수. 관심분야는 DB 응용, ERP, ebusiness, DW, 생산정보시스템, 객체지향 분석 및 설계 등