IE Interfaces
Vol. 20, No. 3, pp. 257-266, September 2007.

서비스 지향 아키텍쳐 기반의 자재관리시스템 설계

안민정 · 이홍철†

고려대학교 정보경영공학부

Design of the Material Control System based on Service Oriented Architecture

Min Jeong An · Hong Chul Lee

Division of Information Management Engineering, Korea University, Seoul, 136-713

To survive in rapidly changing business environment, the enterprises of all over the world are trying to integrate their IT infra structure and are trying to automate their business process. Service Oriented Architecture (SOA) is being accepted as an IT standard to support efficient system integration and flexible business process automation between enterprises or departments. The material control system is required this software architecture between manufacturing company and materials supply vendors. This paper introduces concept of SOA, relevant technology about SOA such as Web Services and Business Process Execution Language (BPEL) and Enterprise Service Bus (ESB) and describes how to automate materials control process by designing the material control system based on SOA. The analysis of requirements is proceeded by Unified Modeling Language (UML) and SOA delivery strategy is selected the top-down strategy. And this paper describes how to derive services and operations and how to arrange services in three service layers and how to design business process.

Keyword: Material Control System, e-business, Service Oriented Architecture, SOA, Web Services, BPM, BPEL

1. 서 론

최근 IT업계 최고의 화두는 '서비스 지향 아키텍쳐(SOA)'이다. 1996년 가트너(Gartner) 그룹에 의해 처음 소개된 지 10여년이 지난 현재 BEA, IBM, Oracle, Microsoft, SAP, TmaxSoft 등과 같은 벤더를 중심으로 SOA를 지원하는 솔루션이 활발하게 출시되었으며 POC(Proof Of Concept) 형태의 프로젝트가 많이 진행된 상태이다. Gartner 그룹은 2008년까지 신규 개발 프로젝트의 80%이상이 SOA를 기반으로 개발될 것으로 전망하였다(Gartner, 2005). 이렇게 SOA가 소프트웨어 시장뿐 아니라, 기업환경 전반에 걸쳐 최고의 화두가 되고 있는 이유는 그것이 최근의 급변하는 비즈니스 환경 변화에 빠르게 대응할 수 있는 최적의 대안이라고 보고 있기 때문이다.

가장 두드러진 비즈니스 환경변화는 비즈니스 생태계의 개념과 실시간 기업의 등장이다. 비즈니스 생태계(Business ecosystem)란 비즈니스 환경이 과거 독립적인 조직 및 프로세스에 의해 주도되는 수직적 통합에서 고객, 공급자, 파트너 등 다수 기업과의 관계적 협업관계가 중시되는 환경에서 유기체로서의 기업을 의미한다. <Figure 1>은 이러한 비즈니스의 진화를 보여준다.

그리고 실시간 기업(Real-time Enterprise)이란 기업의 기회, 위협요인을 경쟁사보다 먼저 파악해 효율적인 의사결정을 하고

본 논문은 2007년도 두뇌한국 21사업에 의하여 지원되었음.
†연락저자 : 이홍철 교수, 136-713 서울시 성북구 안암동 5가 1번지 고려대학교 정보경영공학부, Fax : 02-929-5888,
E-mail : hclee@korea.ac.kr
2007년 07월 접수, 1회 수정 후 2007년 08월 게재확정.

민첩하게 대응할 수 있도록 하는 개념이다.

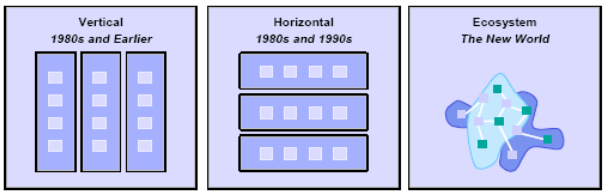

Figure 1. The evolution of business(Mark *et al*., 2004)

이러한 관계적 협업이 중시되는 비즈니스 환경에서 경쟁력 있는 기업은 급변하는 비즈니스 환경 변화와 시장요구에 민첩하게 대응할 수 있어야 한다. 이를 위해서는 기업 간, 조직 간에 실시간 정보공유가 가능하도록 적재적소에 실시간 데이터를 수집하고 접근하여, 업무프로세스의 중단이 없도록 하는 것이 필수적이다. 즉 기업 내 모든 애플리케이션과 데이터가 필요할 때 곧바로 접근할 수 있는 통합 환경이 되어야 한다. 현재 수준에서 이러한 실시간 기업에 대한 요구에 가장 적합한 것이 바로 SOA인 것이다.

본 논문에서는 제조 기업과 자재 공급 업체 간에 위와 같은 통합 환경이 필수로 요구되는 자재관리 시스템의 프로토타입을 SOA 기반으로 설계하였다. 제 2장에서는 관련 기술을 요약 설명하였고, 제 3장에서는 자재관리시스템의 분석 내용을 기술하였다. 제 4장에서는 ActiveBPEL Designer를 활용하여 자재관리시스템의 프로토타입을 설계하였고, 제 5장에서 결론 및 향후 연구 과제를 기술하였다.

2. 관련기술

2.1 SOA의 개념 및 특징

OASIS(the Organization for the Advancement of Structured Information Standards) 그룹에서는 2006년 8월에 다음과 같이 SOA에 대한 공식적인 정의를 하였다.

> Service Oriented Architecture (SOA) is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains (Matthew *et al*., 2006).

IT 관점에서 SOA는 분산 객체의 형태로 존재하는 컴포넌트들 간에 Message 통신을 통해 정보를 교환하는 느슨하게 연결된(Loosely Coupling) 형태의 Software Architecture이다(Thomas, 2006). 여기서의 서비스는 독립적인 비즈니스 기능을 구현한 소프트웨어 컴포넌트로서, 어플리케이션이나 다른 서비스가 외부에서 그 서비스를 찾을 수 있고, 공개된 인터페이스를 통해 접근하며, 주로 메시지 기반의 비동기 커뮤니케이션 방식으로 사용하도록 구현되는 소프트웨어 개체이다.

<Figure 2>는 SOA의 기본 개념을 보여준다. 서비스 제공자가 서비스 저장소에 서비스를 공개(publish)하면 서비스 요청자는 서비스 저장소에서 원하는 서비스를 찾아(find) 해당 서비스를 호출(bind)하여 사용하게 된다.

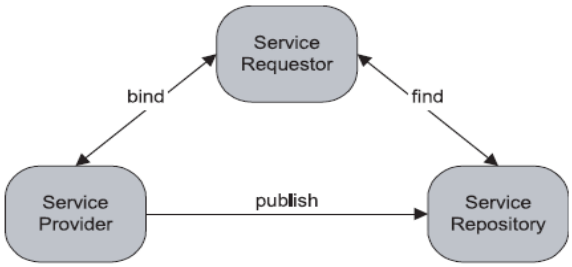

Figure 2. The Service Oriented Architecture(Jens *et al*., 2003)

SOA는 다음과 같은 네 가지 특징을 가지고 있다. 첫 번째, 서비스는 네트워크상에 접근할 수 있는 인터페이스(Interface)를 지녀야 한다. 이러한 인터페이스는 Public 인터페이스와 Published 인터페이스로 구분된다. 전자는 특정 시스템의 내부 네트워크에서 접근하는 인터페이스이다. 후자는 기업 시스템의 외부에서 네트워크를 통하여 접근할 수 있는 인터페이스이다. 웹 서비스의 WSDL은 이러한 Published 인터페이스의 대표적인 사례이다. 후자는 불특정한 원격지의 프로그램이나 비즈니스 기능들이 네트워크를 통하여 접근할 수 있으므로 버전 관리, 성능, 서비스 품질(QoS: Quality of Service), 보안 등 여러 가지 복잡한 문제들을 고려해야 한다.

두 번째, 클라이언트가 네트워크를 통하여 원격지의 서비스를 호출할 수 있어야 한다. 또한 서비스는 네트워크 환경설정(Network Configuration) 기능을 지원해야 한다.

세 번째, SOA는 상호운용성이 강조된다. 서비스를 사용하려는 잠재적인 모든 클라이언트들이 이해할 수 있는 프로토콜과 데이터 규약을 사용해야 한다는 것을 의미한다.

네 번째, 서비스는 동적으로 발견될 수 있어야 한다. 이것은 제 3자가 서비스 발견을 위한 메커니즘을 제공해야 한다는 것을 의미한다.

이러한 SOA의 특징을 구현하기 위하여 표준화 단체와 소프트웨어 벤더가 참여하여 표준을 만들고 있으며, SOA 관련 소프트웨어 제품은 이러한 표준을 채택하여 제작되고 있다. 현재 표준으로 활용되는 기술은 BPEL, JSR 168, JSR 181, SCA, SDO, SOAP, UDDI, WS-Addressing, WSDL, WS-I, WSIF, WS-Reliability, WSRP, WS-Security, XML, XPath, XQuery, XSLT 등이 있다.

2.2 웹 서비스(Web Services)

웹 서비스는 SOA의 주요한 특징들을 구현하는 기술 중의 하나로써 상호운용성을 위하여 주요한 네트워크 프로토콜로 HTTP/SOAP를 사용하고, 데이터의 규약으로 XML을 사용하며, 서비스 등록 및 동적 발견을 위한 UDDI, 그리고 서비스 인터페이스를 기술하는 WSDL로 SOA를 구현한다.

기술적인 관점에서 웹 서비스는 “XML과 인터넷 프로토콜을 이용하여 네트워크상에 분산된 서비스 간의 상호 연동이 가능하도록 하는 표준기술”이다. 비즈니스 관점에서 웹 서비스는 좀 더 포괄적으로 “진정한 분산 환경에서 개인 혹은 기업, 정부 간의 협업을 통하여 지속적인 수익을 창출하는 서비스 지향적 패러다임”으로 그 개념이 재정립될 수 있다.

SOA를 구현하기 위한 웹 서비스 기본 표준은 <Table 1>과 같다. 서비스 제공자가 WSDL로 표현된 서비스의 인터페이스, 데이터 타입, 위치 정보 등 서비스 호출 시 필요한 상세 정보를 UDDI 저장소에 등록한다(Publish). 서비스 사용자는 UDDI 저장소에서 원하는 서비스 상세 정보를 파악한(Find) 후, 확인된 서비스 위치에 접속하여 서비스를 호출한다(Bind). 서비스를 등록하고 호출할 때 SOAP을 이용한다.

초기의 웹서비스 기술은 주로 비즈니스 분야에서 다양한 애플리케이션들의 통합 도구로서 이용되어 왔으나, 현재 웹서비스는 단순히 비즈니스 영역의 애플리케이션뿐만 아니라, 유무선 통합, 통신-방송 융합, 정보가전, 임베디드 환경 등 IT839정책의 다양한 서비스 분야에서 핵심 소프트웨어 인프라 기술로 활용될 것으로 기대되고 있다.

Table 1. Web Services Standards

표준	내 용
XML	(eXtensible Markup Language) XML은 SGML의 복잡성을 단순화시키고 확장성이 없는 HTML의 문제점을 해결하고자 각 마크업 언어의 장점만을 수용한 것으로 데이터의 표현과 구조를 분리한다. XML은 문서의 내용을 기술하며 그 표현은 XSL(eXtensible Stylesheet Language)로 구조는 XML Schema로 나타낸다.
HTTP	(Hyper Text Transfer Protocol) 인터넷의 월드 와이드 웹(WWW) 서버와 WWW 브라우저가 파일 등의 정보를 송수신 하는데 사용되는 클라이언트/서버 규약이다.
SOAP	(Simple Object Access Protocol) XML과 HTTP 등을 기반으로 하여 다른 컴퓨터에 있는 데이터나 서비스를 호출하기 위한 통신 규약이다.
WSDL	(Web Services Description Language) 비즈니스 서비스를 기술하여 비즈니스들끼리 전자적으로 서로 접근하는 방법을 제공하기 위해 사용되는 XML 기반의 언어이다.
UDDI	(Universal Description Discovery and Integration) 인터넷에서 전 세계 비즈니스 목록에 서비스를 등재하기 위한 XML 기반의 레지스트리이다.

2.3 BPEL(Business Process Execution Language)

BPEL은 웹 서비스 환경에서 비즈니스 프로세스를 정의하고 실행하기 위한 표준 언어이다. BPEL은 웹 서비스의 composition, orchestration, coordination을 통해 Top-Down 방식으로 SOA를 구현한다. BPEL을 이용하면 웹 서비스에 “비즈니스 프로세스(business process)”라 불리는 컴포지트 서비스(composite service)를 쉽고 직관적으로 구현할 수 있다.

웹 서비스 작업은 게시(publish)와 조정(orchestration)의 2단계 작업 과정으로 이루어진다. 게시란 기존 시스템의 일부를 이용하여 그 시스템을 서비스로 노출하는 것을 의미하고, 조정이란 개별적인 여러 서비스를 엔드 - 투 - 엔드 프로세스 흐름으로 구성하는 것을 의미한다. BPEL은 조정을 위한 업계 표준이다. 일련의 서비스를 엔드-투-엔드 프로세스 흐름으로 조정하는 일에는 여러 새로운 기술 요구 사항들(이기종 시스템으로의 바인딩, 비동기식 및 동기식 메시지 교환 패턴, 데이터 조작, 흐름 조정, 예외 관리, 비결정적 이벤트, 트랜잭션 보상, 병렬 버전 확인, 진행 중인 인스턴스 관리 및 감사 등)이 수반된다. 현재 WS-BPEL 2.0이 최신 버전이다.

2.4 ESB(Enterprise Service Bus)

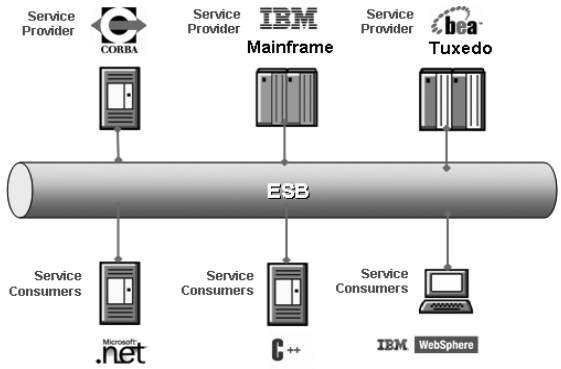

Figure 3. ESB provides a lightweight distributed architecture(IONA Technologies PLC, 2007)

Table 2. ESB 기능(Cholhong *et al*., 2006)

ESB 기능	설 명
Transformation	XSLT에 따라서 message 변환
Content-Based Routing	정해진 Rule에 따라 Process 상의 Path를 분기하여 처리
Web Services	외부 컴포넌트와 통신
Legacy	외부 Legacy와 통신
B2B	B2B 프로토콜 기반으로 통합
JMS	비동기 통신 message 처리
Work-list	Human Interaction(전자결제형태)

ESB는 SOA의 기본 원리를 따르면서 통합 인프라 구조를 제공하는 논리적 구조로, 이벤트 중심의 분산 시스템 환경을 가능하도록 SOA를 지원하는 논리적인 버스의 개념이다(Eunyoung *et al*., 2004). ESB는 각 endpoints를 WSDL과 같은 표준화된 서비스로 표현함으로써 인터페이스의 표준화가 가능하고, <Figure 3>에서 보여주는 lightweight distributed architecture를 구성한다. 단위 서비스들은 ESB안에 설치되어 프로세스 기반 하에 Message

Transformation, Routing, 외부 서비스 호출, Legacy 연동의 기능을 수행하게 된다. ESB의 주요 기능은 <Table 2>와 같다(Cholhong *et al*., 2006).

2.5 SOA 설계 방법론

SOA 설계 방법으로는 하향식(Top-Down) 접근 방법, 상향식(Bottom-Up) 접근 방법, 절충식(Meet-in-the-middle) 접근 방법이 있다.

하향식 접근 방법은 "분석 우선"인 접근 방법으로, 조직의 전반적인 업무 모델을 개발하게 한다. 이 방식을 통해 SOA의 품질을 가장 높은 수준으로 올릴 수 있으나, 시간과 비용이 많이 소요된다.

상향식 접근 방법은 서비스가 애플리케이션 중심의 요구사항이 충족되도록 기존 시스템에 서비스 래퍼(wrapper)를 붙이는 방식이다. 이 방식은 수행하기는 쉽지만 SOA가 추구하는 아키텍쳐를 진전시키는 효과는 없다.

절충식 접근 방법은 하향식과 상향식 접근방법의 조합이다. 지속적으로 분석을 하면서 즉각 필요한 서비스를 구축하는 방식이다(Thomas, 2006).

본 논문에서는 <Figure 4>의 하향식 접근 프로세스를 적용하여 자재관리시스템의 프로토타입을 설계하였다.

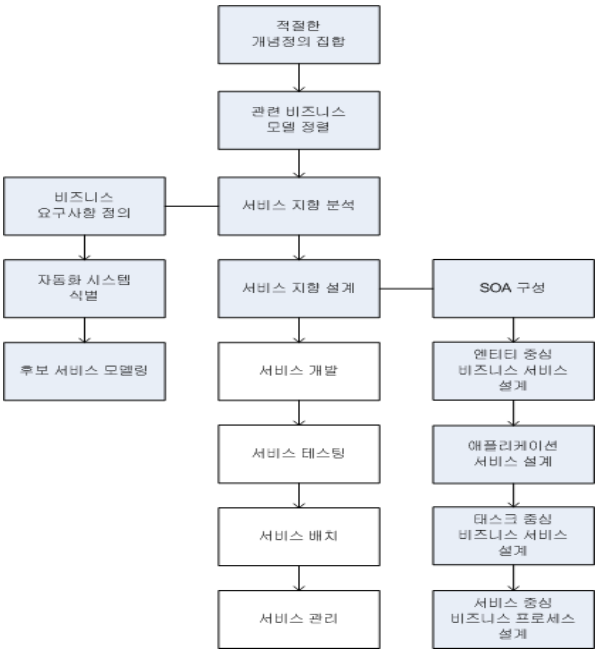

Figure 4. 하향식 SOA 구축 전략 프로세스(Thomas, 2006)

3. 자재관리시스템의 분석

자재관리시스템은 기업 내 시스템 통합과 기업 간 전자상거래가 요구되는 시스템으로써 SOA 도입의 효과를 가장 크게 볼 수 있는 분야 중의 하나이다.

3.1 자재관리 업무 흐름 분석

제조기업과 관련된 자재관리 업무흐름은 고객업체의 발주 데이터를 FAX나 VAN망을 통해 받아서 제조기업 생산부서의 부하분석을 통하여 생산계획을 수립하고, 제품에 대한 Sub자재를 BOM을 통해 확인한 후, 최종적으로 재고를 감안한 자재 소요량을 산출한다. 그리고 자재 소요량을 각 거래업체별로 FAX나 Web EDI를 통하여 다시 전달한다. <Figure 5>는 제조기업과 자재공급사간의 자재관리의 업무흐름을 보여준다(Junghyuk *et al*., 2003).

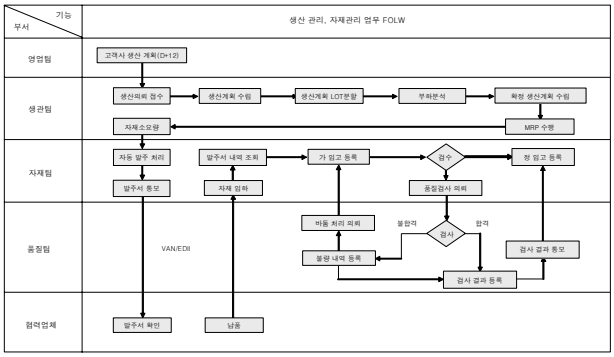

Figure 5. 자동차부품산업 업무현황(Junghyuk *et al*., 2003)

본 논문에서는 <Figure 5>와 같이 복잡한 업무프로세스를 자동화하고 비즈니스 환경 변화에 민첩하게 대응할 수 있도록 실시간 자재재고관리와 자재 스케쥴링, 그리고 자재발주 프로세스의 자동화에 초점을 두어 자재관리시스템의 프로토타입을 SOA기반으로 설계하였다.

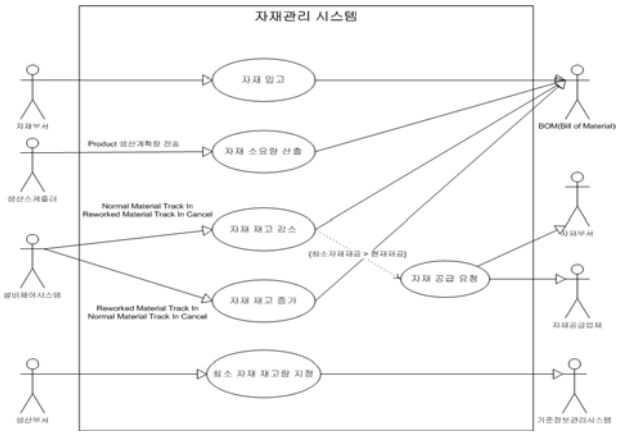

Figure 6. Use Case Diagram

3.2 자재관리시스템 요구사항 분석

시스템의 범위와 요구사항 정의를 명확히 하기 위해 <Figure

6>과 같이 Use Case Diagram을 작성하였다.

3.3 비즈니스 프로세스 정의

각 Use Case의 비즈니스 프로세스를 다음과 같이 정의하였다. <Figure 7>은 자재입고 프로세스이다. 자재 입고 시 자재부서에서 User Interface Portal을 통해 자재 입고 내역을 입력하면 자재재공정보가 자재 창고와 BOM에 자동으로 업데이트된다.

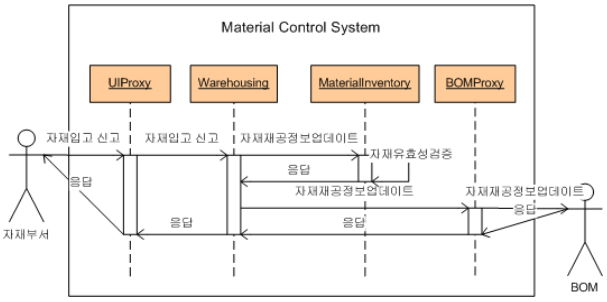

Figure 7. Warehousing Process

<Figure 8>은 자재의 자동 공급을 위한 기준정보인 최소자재재고량을 등록하는 프로세스이다. 생산부서에서 최소자재재고량을 User Interface Portal을 통해 입력하면 기준정보관리시스템에 저장된다. 최소자재재고량은 자재의 자동공급요청 프로세스 개발 시 참조 데이터로 활용된다.

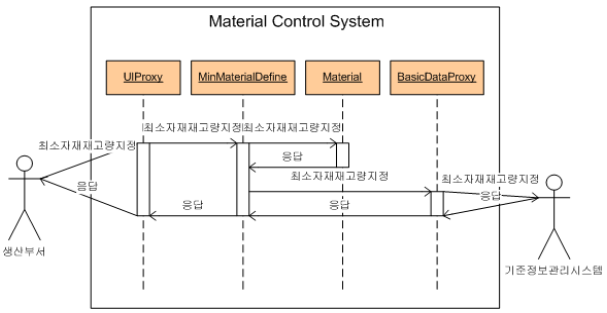

Figure 8. Minimum Material Define Process

<Figure 9>는 제품 생산 계획에 따른 자재 소요량을 산출하는 프로세스이다.

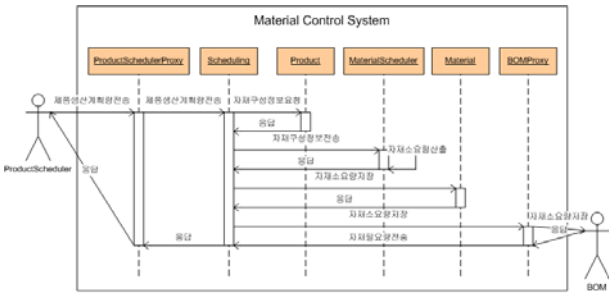

Figure 9. Scheduling Process

Product Service를 통해 제품별로 필요한 자재 목록과 수량을 파악할 수 있으므로 자재 스케쥴러를 구현할 수 있다.

<Figure 10>은 실시간으로 자재재공을 파악하기 위한 프로세스이다. 설비에 자재가 투입되면 그 수량만큼 자재재공을 감소시킨다. 수리된(Reworked) 자재를 투입했을 경우에 자재재공을 증가시킴으로써 과잉 자재주문을 방지한다. 이 프로세스가 전제되어야 <Figure 11>의 자재자동공급 프로세스가 실행될 수 있다.

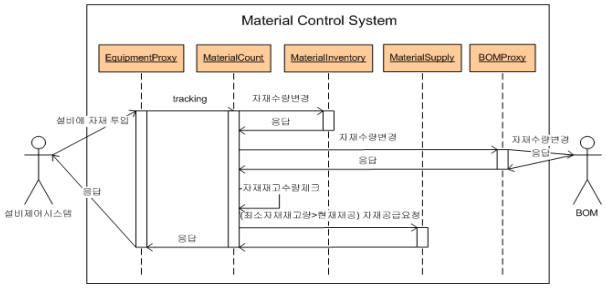

Figure 10. Material Count Process

<Figure 11>은 자재 자동공급 요청 프로세스로써 자재관리시스템의 핵심 프로세스이다. <Figure 10>의 프로세스에서 현재 자재재공이 최소자재재고량보다 작으면 <Figure 11>의 자재 자동공급 요청 프로세스를 호출한다. 이 프로세스에서는 자재별로 자재공급업체의 서비스를 검색하여 해당 업체의 서비스를 호출하는 방식으로 자재를 발주한다. 자재의 발주 결과를 자재부서에 통보한다. 이 때 자재 공급 업체의 시스템이 SOA 기반이라고 가정한다. 만약 자재 공급 업체가 다른 구조의 자동주문접수 시스템을 가지고 있다면 그것과의 인터페이스를 위한 Proxy 서비스를 개발함으로써 자재 자동 발주 프로세스를 구현할 수 있다. 이 업무는 지속적으로 반복되는 프로세스이므로 SOA기반으로 자동화프로세스를 구축하면 자재부서의 업무를 줄이고, 자재 발주 시간을 절약할 수 있는 장점이 있다.

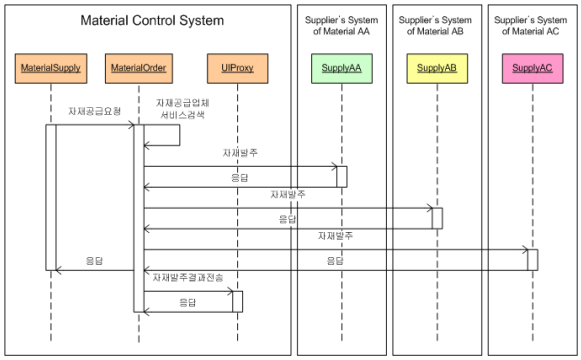

Figure 11. Material Supply Process

3.4 후보 서비스 모델링

3.3에서 정의한 비즈니스 프로세스를 분석하여 후보 서비스를 도출하였고, 이를 효과적으로 구현하기 위하여 세 계층으로

추상화하였다. 각 Layer별 특징과 서비스 목록은 다음과 같다.

Application Service Layer는 기술에 특화된 기능을 표현하기 위한 서비스로써 신규 혹은 기존 애플리케이션 환경에서 데이터를 처리할 수 있게 한다. 기존 애플리케이션의 컴포넌트 인터페이스를 반영하여 WSDL 정의를 제공한다. 이를 통해서 해당 컴포넌트에 대한 endpoint를 구축할 수 있으며, 그 컴포넌트로 하여금 SOAP 통신을 할 수 있게 한다(Thomas Erl, 2006). <Table 3>은 도출된 Application Service의 목록이다.

Table 3. Application Service Layer

Service Name	Operation
BasicDataProxy	기준정보관리 시스템과의 인터페이스
BOMProxy	BOM 데이터베이스와의 인터페이스
EquipmentProxy	설비제어시스템과의 인터페이스
ProductSchedulerProxy	생산스케줄러와의 인터페이스
UIProxy	User Interface Portal과의 인터페이스

Business Service Layer는 SOA의 가장 핵심적인 요소로써 비즈니스 서비스 모델을 직접적으로 구현한 비즈니스 서비스로 구성되며 애플리케이션 서비스를 조합하는 컨트롤러로서 동작하는 것이 일반적이다. 비즈니스 로직을 캡슐화하는 태스크 중심 비즈니스 서비스와, 특화된 비즈니스 엔티티를 캡슐화하는 엔티티 중심 비즈니스 서비스로 구분될 수 있다(Thomas Erl, 2006). <Table 4>는 도출된 Business Service의 목록이다.

Table 4. Business Service Layer

	Service Name	Operation
Entity	MaterialInventory	자재재고관리 서비스
	Product	생산제품관련 서비스
	Material	생산자재관련 서비스
Task	MaterialScheduler	자재소요량산출 서비스
	MaterialSupply	자재공급 서비스

Table 5. Orchestration Service Layer

Service Name	Operation
MaterialCount	자재 투입에 따른 자재 수량 관리 서비스
MaterialOrder	자재 자동공급 요청 서비스
MinMaterialDefine	자재 발주 시점에 참조되는 최소자재재공량 정의 서비스
Scheduling	Product 생산계획에 따른 자재 소요량 산출 서비스
Warehousing	자재 입고 처리 서비스

Orchestration Service Layer는 SOA 구성요소 중에서 프로세스 역할을 수행한다. 하나 이상의 프로세스 서비스들로 구성되고, 이 프로세스 서비스는 프로세스 정의 안에 내장된 비즈니스 규칙과 비즈니스 로직에 따라 비즈니스와 애플리케이션 서비스를 조합한다. 오케스트레이션은 다른 서비스로부터 비즈니스 규칙과 서비스 실행 순서 로직을 추상화하고, 기민성과 재사용성을 높여준다(Thomas Erl, 2006). <Table 5>는 도출된 Orchestration Service의 목록이다.

3.5 자재관리시스템 구성

<Figure 12>는 3.4에서 정의한 자재관리시스템 내의 서비스와 외부 시스템과의 인터페이스를 표현한 자재관리시스템 구성도이다. Business Process Layer에서는 이러한 자재관리시스템 내의 서비스와 동일 기업 내 타 시스템과의 인터페이스를 위한 서비스, 타 기업인 자재공급업체의 서비스를 조합하여 비즈니스 로직을 수행한다.

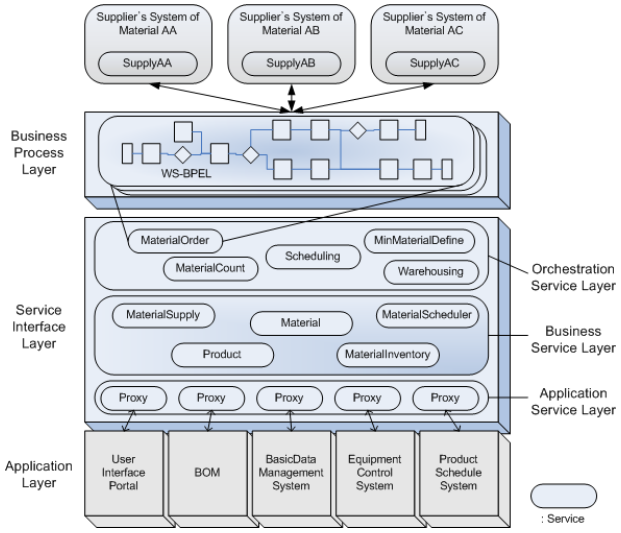

Figure 12. Material Control System Configuration

4. 자재관리시스템 설계

제 3장에서 분석한 시스템을 Active Endpoints, Inc.의 Freeware SOA 개발 도구인 ActiveBPEL Designer를 이용하여 설계하였다.

Table 6. MCS에서의 SOA 특징 구현

SOA 특징	MCS에서의 구현
네트워크상에 접근할 수 있는 인터페이스	WSDL에서 PortType(operation)과 binding(통신 방식)을 정의하였다.
컴포넌트(서비스) 간에 느슨한 결합	분산된 WSDL의 PortType들을 BPEL에서 조합하여 다양한 비즈니스 컴포넌트(서비스)를 생성하였다.
클라이언트가 네트워크를 통하여 원격지의 서비스를 호출	기업 외부 시스템(자재공급업체)의 WSDL을 호출하였다.
상호운용성	표준 프로토콜로 HTTP/SOAP을 사용하였고 데이터 규약은 XML을 이용하였다.

<Table 6>은 MCS에서 구현한 SOA의 특징에 대한 설명이다.

4.1 Message 설계

서비스 명세서인 WSDL 설계에 앞서 서비스의 입출력에 사용될 Message를 정의하는 XML Schema를 설계하였다. <Figure 13>에서 materialInfo element는 string type의 materialId와 int type의 qty로 구성됨을 알 수 있다.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace="http://163.
    <element name="orderResultSendInfo">
        <complexType>
            <sequence>
                <element ref="tns:orderInfo"></element>
                <element name="sendAddress" type="tns:sendAddress"></element>
            </sequence>
        </complexType>
    </element>
    <element name="trackingInfo">
        <complexType>
            <sequence>
                <element name="materialId" type="string"></element>
                <element name="status" type="string"></element>
                <element name="trackingType" type="string"></element>
                <element name="qty" type="string"></element>
            </sequence>
        </complexType>
    </element>
    <complexType name="materialInfo">
        <sequence>
            <element name="materialId" type="string" />
            <element name="qty" type="int" />
        </sequence>
    </complexType>
```

Figure 13. MCSData.xsd

4.2 서비스 설계

3.4에서 정의한 15개의 서비스와 자재공급업체의 서비스 3개를 설계하였다. <Figure 15>와 같이 XML 형태의 서비스 명세서인 WSDL은 가독성이 떨어지므로 SOA 개발 툴은 그래픽 인터페이스를 제공한다. <Figure 14>은 Orchestration Service Layer의 MaterialOrder.wsdl이다. Services는 MaterialOrder 서비스에서 이용할 타 서비스 목록이고, PortTypes는 이용할 operation들의 목록이다. Bindings 항목은 각 서비스의 operation에 접근할 방식을 정의하며, Messages는 operation의 입출력 형태를 정의한다.

```
<wsdl:message name="QtyResponse">
  <wsdl:part name="QtyResponse" type="MCSData:qtyResponse"/>
</wsdl:message>
<wsdl:message name="OrderInfo">
  <wsdl:part name="OrderInfo" type="MCSData:orderInfo"/>
</wsdl:message>
<wsdl:message name="Response">
  <wsdl:part name="Response" type="MCSData:response"/>
</wsdl:message>
<wsdl:portType name="SupplyAAPT">
  <wsdl:operation name="orderSupply">
    <wsdl:input message="tns:MaterialInfo"/>
    <wsdl:output message="tns:QtyResponse"/>
  </wsdl:operation>
</wsdl:portType>
```

Figure 15. MaterialOrder.wsdl

4.3 비즈니스 프로세스 설계

3.3에서 정의한 5개의 비즈니스 프로세스를 BPEL로 설계하였다. XML 형태의 비즈니스 실행 언어인 BPEL 또한 가독성이 떨어지므로 SOA 개발 툴들은 그래픽 인터페이스를 제공한다.

<Figure 16>은 Business Process Layer의 Scheduling.이다. ProductScheduler에서 메시지를 받으면 제품 구조를 조회하여 관련 자재의 소요량을 계산한 후 각 자재별로 계산 결과를 셋팅하는 프로세스이다. Product, Material, MaterialScheduler, ProductSche

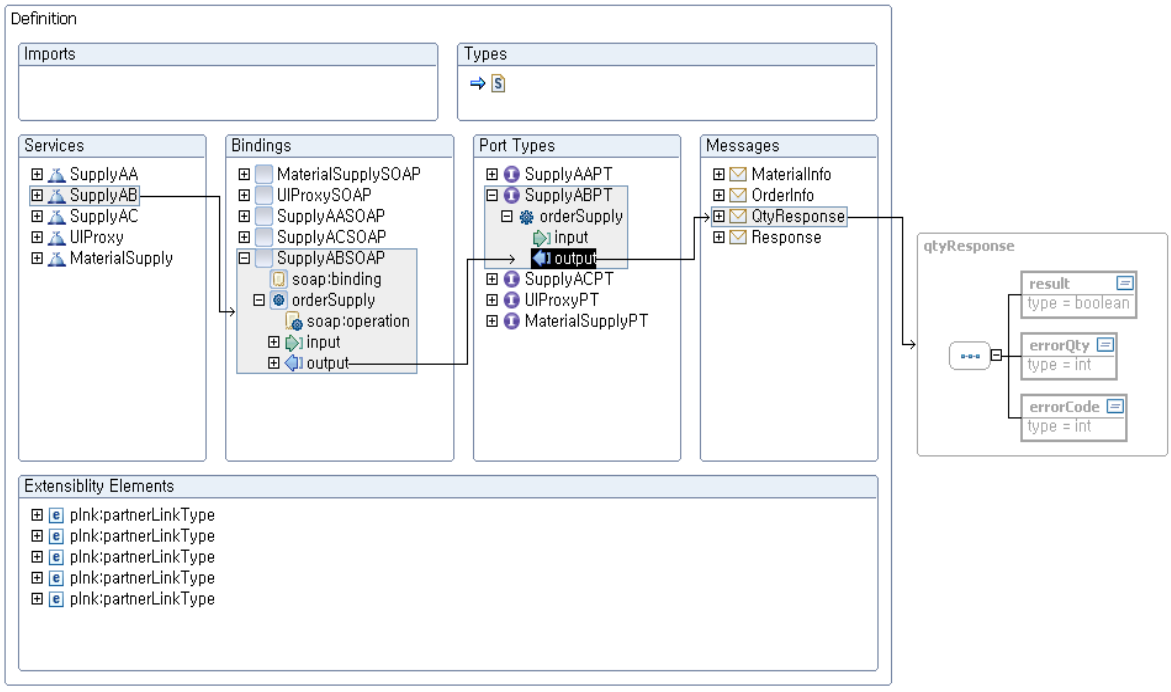

Figure 14. MaterialOrder.wsdl

dulerProxy, BOMProxy의 5개 서비스를 조합하여 <Figure 16>과 같은 비즈니스 프로세스를 설계하였다. 이러한 프로세스는 생산 정책에 따라 자주 바뀔 수 있고, 이에 신속히 대응할 수 있어야 한다. 또한 전산 부서가 아닌 생산 관리자도 쉽게 프로세스를 정의할 수 있어야 한다.

즉, 변화에 빠르게 대응하고, 서비스 개발자와 비즈니스 프로세스 개발자가 독립적으로 업무를 수행할 수 있어야 한다. 이러한 요구 사항을 해결할 수 있는 방안으로 BPEL이 큰 역할을 수행할 수 있고, 이를 가능하게 하는 IT 기반으로 SOA가 가장 적합한 것이다.

<Figure 17>은 본 논문에서 가장 중점적으로 다루는 자재 자동 공급 프로세스를 구현한 BPEL이다. 설비에 자재가 투입되거나 투입이 취소될 때 MaterialCount.bpel이 실행된다. TrackingType이 trackin이고 status가 normal이면 즉, 정상자재가 투입되면 MaterialInventory의 outMaterial과 BOMProxy의 setWipQty를 병렬로 실행한다. 이러한 조건을 bpel은 다음과 같이 표현한다.

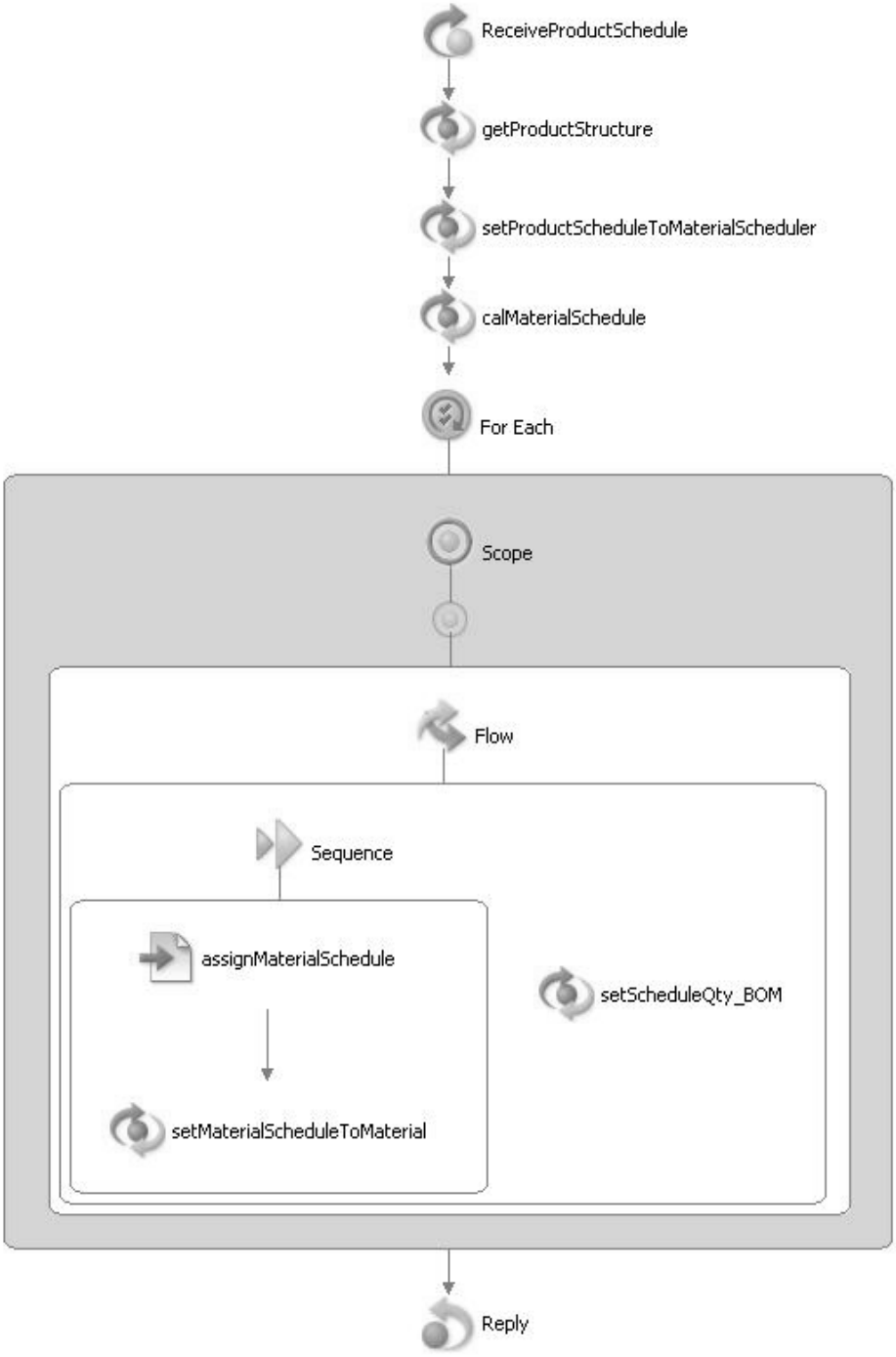

Figure 16. Scheduling.bpel

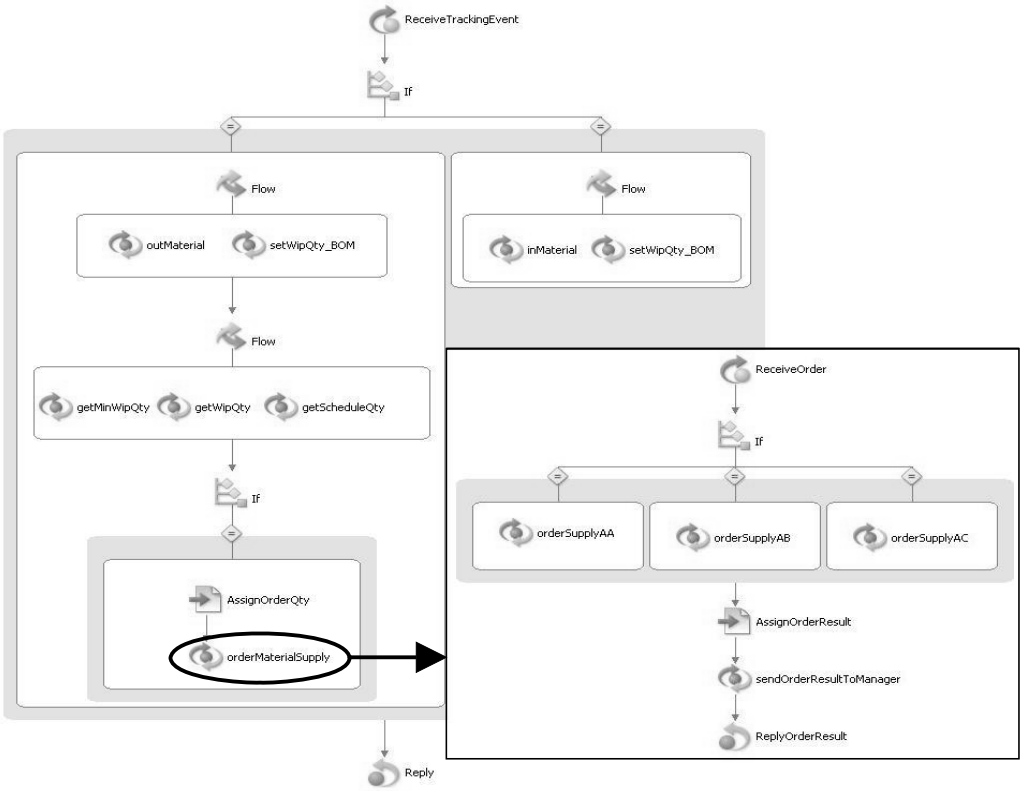

Figure 17. MaterialCount.bpel & MaterialOrder.bpel

```
<bpel: condition
expressionLanguage =“urn:oasis:names:tc:wsbpel: 2.0: sublang:
xpath1.0” > ($TrackingInfo.TrackingInfo/trackingType =“trackin” and
$Trackin gInfo.TrackingInfo/status =“normal”)
</bpel: condition>
```

WipQty.qty가 MinWipQty.qty보다 작으면 즉, 현재재공이 최소자재재공량보다 작으면 자재공급요청 수량을 지정하여 MaterialSupply 서비스의 orderMaterialSupply operation을 호출함으로써 MaterialOrder.bpel을 실행한다.

자재공급요청 수량을 지정하는 Assign은 다음과 같다.

```
<bpel: assign name =“AssignOrderQty”>
 <bpel: sources>
          <bpel: source linkName =“L4”/>
 </bpel: sources>
 <bpel: copy>
          <bpel: from part =“ScheduleQty”
variable =“ScheduleQty”>
           <bpel: query>$ScheduleQty.ScheduleQty/qty -
$WipQty.WipQty/qty</bpel: query>
          </bpel: from>
          <bpel: to part =“MaterialInfo”
variable =“MaterialInfo”>
           <bpel: query> qty </bpel: query>
          </bpel: to>
 </bpel: copy>
</bpel: assign>
```

MaterialSupply 서비스의 orderMaterialSupply operation을 호출하는 invoke는 다음과 같다.

```
<bpel: invoke
 inputVariable =“MaterialInfo”
name =“orderMaterialSupply”
operation =“orderMaterial” outputVariable =“QtyResponse”
partnerLink =“MaterialSupply”
portType =“ns1: MaterialSupplyPT”>
```

MaterialOrder.bpel에서는 자 재공급업체를 판별하여 자재공급업체의 서비스를 호출함으로써 자재공급요청을 하고 그 결과를 UIProxy 서비스를 통해 자재부서에 전달한다. <Figure 17>의 프로세스는 기업 내, 기업 간 업무프로세스를 자동화하기 위한 시스템 통합의 중요한 예로써 SOA 도입의 장점를 가장 잘 나타내는 부분이다.

실제 제조 기업에서는 많은 종류의 자재와 자재공급업체가 있으므로 동적으로 서비스를 찾아 연결하는 로직이 추가로 개발되어야 할 부분이다.

5. 결론 및 향후 연구과제

본 논문에서는 하향식 SOA 접근 방법을 이용하여 분석 및

설계한 자재관리시스템의 프로토타입을 제안하였다. 자재관리시스템의 분석 단계에서는 Use Case를 정의하였고, 각 Use Case별 비즈니스 프로세스를 분석하여 후보서비스를 모델링하였으며, 이러한 서비스를 효과적으로 구현하기 위하여 Application Service Layer, Business Service Layer, Orchestration Service Layer의 세 계층으로 추상화하였다. 설계 단계에서는 Active Endpoints, Inc.의 Freeware SOA 개발 도구인 ActiveBPEL Designer를 이용하여 Message, WSDL, BPEL을 설계하였다. BPEL 설계 단계에서는 Message와 WSDL 설계시 고려하지 못했던 부분을 발견하여 이 부분의 수정 작업을 여러 번 함으로써 BPEL의 완성도를 높였다. 본 논문에서 제안한 자재 입고 프로세스와 자재스케쥴링 프로세스는 기업 내 조직 간 업무프로세스 혁신에 도움이 될 것이며, 자재자동공급요청 프로세스는 기업 간 업무프로세스 혁신에 도움이 될 것으로 본다.

SOA는 전사적 관점에서의 접근이 이루어져야 하며, 이에 따른 조직의 업무 역할도 재분배되어야할 필요가 있다. IT 조직은 기존의 정보시스템이 가지고 있는 비즈니스 프로세스를 분리하여 해당 부서에 이관하는 작업을 해야 할 것이고, 각 부서에서는 BPEL을 워드프로세서를 쓰듯이 작성할 수 있어야할 것이다. IT 조직은 서비스 인프라 구축과 서비스 품질관리 및 보안에 중점을 두어 업무를 해야 할 것이다.

향후 SCA, SDO, WS-Addressing, WS-Security 등의 SOA 관련 표준 기술의 연구와 오픈 소스 SOA 플랫폼을 연구하여 본 논문에서 설계한 자재관리시스템의 프로토타입을 개발할 예정이다. SOA는 기업 내 시스템 통합과 기업 간 시스템 연계를 넘어 개인과 개인의 서비스를 연계할 수 있는 기반을 제공할 것이므로 양질의 공개 서비스 기반 확충을 위해 오픈 소스 SOA 플랫폼 연구는 활발히 진행되어야 할 것으로 본다.

참고문헌

Cholhong, I. and Doseok, H. and Jeongjoon, C. (2006), A Study of a Scheme to Assess and Improve ESB-based SOA Applications from the S/W Architecture Perspective, *Korea Society of IT Services Journal*, **5**(2), 169-178.

David A. Chappell (2004), Enterprise Service Bus, O'Reilly, U.S.

Eunyoung, M and Byoungju, C., and JungWon, L. (2004), Developing Service for Managing Data Quality on ESB, *Korea Information Science Society*, **31**(2), 517-519.

IONA Technologies PLC (2007), The Evolution from EAI to ESB, 5, IONA Technologies PLC, Dublin, Ireland.

Jens, H. and Mathias, W. (2003), Web Services: Foundation and Composition, *Electronic Markets*, **13**(2), 108-119.

Junghyuk, P. and kichul, S., and Taesoo, M. (2003), Design and Implementation of UML-Based Material Management System for Automotive Part Company, *Journal of information systems*, **12**(2), 129-149.

Mark, E. and Jenny, A., and Ali, A. and Sook, C. and Philippe, C. and Pal, K. and Min, L., and Tony, N. (2004), Patterns: Service-Oriented Architecture and Web Services, 19, IBM Corp., U.S.

Matthew, C. and Ken, L. and Francis, M., and Peter, F. and Rebekah, M. (2006), Reference Model for Service Oriented Architecture 1.0, Committee Specification 1, OASIS.

OASIS (2006), Reference Model for Service Oriented Architecture 1.0.

Thomas, E. (2006), Service-Oriented Architecture Concept Technology and Design, acorn, Seoul, Korea.

안 민 정
숭실대학교 전자공학 학사
현재: 고려대학교 정보경영공학부 석사과정
관심분야: 시스템 분석, 설계 및 통합

이 홍 철
고려대학교 산업공학 학사
Univ. of Texas 산업공학 석사
Texas A&M Univ. 산업공학 박사
현재: 고려대학교 정보경영공학부 교수
관심분야: 생산 및 물류 정보시스템, SCM