[UCI]I804:24011-200000233047

2006年　8月
博士學位論文

온토로지 기반의 웹서비스를 이용한 포털 시스템 구현

朝鮮大學校 大學院

電算統計學科

崔　奉　漢

온토로지 기반의 웹서비스를 이용한 포털 시스템 구현

An Implementation Portal System of Base on Ontology to Using Web Service

2006年 8月 25日

朝鮮大學校 大學院

電算統計學科

崔 奉 漢

온토로지 기반의 웹서비스를 이용한 포털 시스템 구현

指導敎授　李　雄　基

이 論文을 理學博士學位申請 論文으로 提出함.

2006年　4月

朝鮮大學校 大學院

電算統計學科

崔　奉　漢

崔奉漢의 博士學位論文을 認准함

委員長　朝鮮大學校 敎授 _____ 印

委　員　湖原大學校 敎授 _____ 印

委　員　朝鮮大學校 敎授 _____ 印

委　員　朝鮮大學校 敎授 _____ 印

委　員　朝鮮大學校 敎授 _____ 印

2006年　6月　　日

朝鮮大學校 大學院

목 차

표 목차 ····· iv
그림 목차 ····· v
ABSTRACT ····· viii

제 1 장 서론 ····· 1
제 1 절 연구 배경 ····· 3
제 2 절 연구목적 및 필요성 ····· 4
1. 연구목적 ····· 5
2. 연구의 필요성 ····· 6
제 3 절 연구내용 및 구성 ····· 9
1. EIP의 핵심 기능 연구 ····· 9
2. 웹 서비스와 포틀릿의 상호 운용성 연구 ····· 10
3. 웹 서비스 기반 포털 프레임워크 설계 및 구축 ····· 10

제 2 장 온토로지의 개념 및 시멘틱 웹과 웹 서비스 ····· 12
제 1 절 온토로지의 개념 ····· 14
제 2 절 온토로지 모델링 ····· 15
제 3 절 온토로지 기술 언어 ····· 17
제 4 절 시맨틱 웹과 웹 서비스 ····· 20
제 5 절 시맨틱 웹의 개발배경 ····· 22
제 6 절 시맨틱 웹의 구성 ····· 24
1. 자원서술 ····· 24
2. 지식서술 ····· 25
3. 통합운용 ····· 26
제 7 절 시맨틱 웹의 활용 ····· 27

제 8 절 웹 서비스의 개요 ····· 29
제 9 절 웹 서비스의 표준기술 ····· 30
1. 웹 서비스의 데이터 표준, XML ····· 31
2. 웹 서비스의 데이터 전송 표준, SOAP ····· 32
3. 웹 서비스에 대한 위치 정보 서비스, UDDI ····· 34
4. 웹 서비스 기술 언어, WSDL ····· 36

제 3 장 포털 시스템의 개념 및 핵심기술 ····· 38
제 1 절 포털 시스템의 구조 ····· 38
1. 포털 시스템의 개념 및 정의 ····· 38
2. 포털 시스템의 종류 ····· 41
3. 상용 EIP 시스템 동향 분석 ····· 44
제 2 절 기존 포털 시스템의 기능 분석 ····· 46
1. 수직 포털의 기능 분석 ····· 46
2. 수평 포털의 기능 분석 ····· 48
제 3 절 EIP 시스템의 기능 분석 ····· 50
제 4 절 포틀릿의 개념과 기능 ····· 53
제 5 절 WSRP의 구조와 동작 ····· 57
1. WSRP의 정의 ····· 57
2. WSRP의 구현 ····· 61

제 4 장 포털 프레임워크 아키텍처 설계 및 구현 ····· 63
제 1 절 웹서비스 기반 EIP 프레임워크 아키텍처 ····· 63
제 2 절 웹서비스 기반 EIP 프레임워크 구현 방안 ····· 65
1. 핵심 서브시스템 ····· 66
2. 전단부 서브시스템 ····· 67
3. 후단부 서브시스템 ····· 68

제 3 절 포틀릿 합성 방법 분석 ····· 71
제 4 절 포틀릿 합성 방법의 구현 ····· 74
제 5 절 포털 프레임워크 구현 및 기본 화면설계 ····· 80

제 5 장 포틀릿 설계 및 구현 ····· 83
1. Personalization Management ····· 84
2. Collaboration Management ····· 84
3. Content Management ····· 85
4. Portal System Management ····· 86
제 1 절 포팅 시스템 관리 기능의 구현 ····· 86
1. SSO(Single Sign On) ····· 86
2. Cache Service ····· 89
제 2 절 개인화 관리 기능의 구현 ····· 91
1. 개인 정보 설정, 화면 레이아웃 ····· 91
2. 개인 일정관리 ····· 92
3. 메일관리 ····· 94
제 3 절 협업 관리 기능의 구현 ····· 95
제 4 절 콘텐츠 관리 기능의 구현 ····· 97
1. 파일관리 ····· 97
2. 이미지갤러리 ····· 101
제 5 절 포털 프레임워크의 서비스 ····· 102

제 6 장 결론 ····· 104
참고문헌 ····· 107

표 목차

표 1. 포틀릿과 서블릿의 차이점 ····· 56
표 2. Portlet.xml 파일 ····· 75
표 3. strutsPortlet.xml 파일 ····· 76
표 4. Display.xml 파일 ····· 77
표 5. 개인화 관리 기능 ····· 84
표 6. 협업 관리 기능 ····· 85
표 7. 컨텐트 관리 기능 ····· 85
표 8. 포털 시스템 관리 기능 ····· 86

그림 목차

그림 1. 시맨틱 웹 계층구조 ····· 13
그림 2. RDF 데이터 모델 ····· 19
그림 3. 웹 서비스 이용절차 ····· 30
그림 4. SOAP 메시지 구조 ····· 34
그림 5. UDDI 데이터 구조 ····· 36
그림 6. 엔터프라이즈 정보 포털(EIP) 시스템의 개념 ····· 41
그림 7. EIP 기능의 발전 ····· 42
그림 8. 포털 시스템의 발전 과정 ····· 43
그림 9. 업종에 따른 EIP의 구분 ····· 45
그림 10. 수직 서비스 계층 구조 ····· 47
그림 11. 포털 시스템의 구조 ····· 53
그림 12. 웹서비스 이용절차 ····· 58
그림 13. JSR168과 WSRP의 상호 운용 방식 ····· 59
그림 14. WSDL의 활용 ····· 60
그림 15. Axis 엔진의 상호 운용 방식 ····· 62
그림 16. Axis 엔진에 대한 클라이언트 호출 Sequence Diagram ····· 62
그림 17. EIP 프레임워크 아키텍처 ····· 64
그림 18. 웹 서비스 기반 EIP 프레임워크 구현 시스템 ····· 65
그림 19. JBOSS3.0 JMX Implementation ····· 67
그림 20. HTTP 상에서 JMS SOAP 메시지 ····· 69
그림 21. Hibernate Mapping DTD ····· 71
그림 22. 포털 페이지의 구성 ····· 72
그림 23. 포털 페이지의 생성 ····· 72
그림 24. Portlet의 Request Handling Sequence ····· 73
그림 25. 포틀릿과 포털 시스템 상호작용절차 Activity Diagram ····· 78

그림 26. 포털페이지 상에 포틀릿 디스플레이 절차 Activity Diagram ·····78
그림 27. 포털페이지 상에 포틀릿 추가절차 Activity Diagram ·····79
그림 28. 포털페이지 상에 포틀릿 삭제절차 Activity Diagram ·····79
그림 29. 화면 설계 및 파일 연관도 ·····80
그림 30. 파일 네비게이션 ·····81
그림 31. 전체 레이아웃 처리 Activity Diagram ·····82
그림 32. 포틀릿 박스 Activity Diagram ·····82
그림 33. EIP 프레임의 유즈 케이스 ·····83
그림 34. SSO가 적용된 상태도 ·····87
그림 35. SSO가 적용된 로그인 절차 및 EIP 리소스 사용절차
Activity Diagram ·····88
그림 36. Account 생성 Activity Diagram ·····88
그림 37. EJBhome의 Factory 패턴화 클래스 Diagram ·····89
그림 38. EJBhome의 Factory 패턴화 예제 ·····90
그림 39. 개인별 프로파일 설정 화면 ·····91
그림 40. 개인별 페이지 환경설정 화면 ·····92
그림 41. 개인 일정관리 메인화면 ·····92
그림 42. 일정관리 초기화면 Activity Diagram ·····93
그림 43. 일정관리 작업별 Activity Diagram ·····93
그림 44. 메일 UseCase Diagram ·····94
그림 45. 메일 포틀릿 메인화면 ·····95
그림 46. 그룹의 편집 및 추가 ·····96
그림 47. 그룹별 권한의 배정 ·····96
그림 48. 포틀릿의 그룹 배정 ·····97
그림 49. 파일관리 포틀릿 메인화면 ·····98
그림 50. 저장소 안의 내용보기 ·····98
그림 51. 저장소(Repository)의 생성 ·····99

그림 52. 문서의 관리 ·····100
그림 53. 이미지 갤러리 포틀릿 메인화면 ·····101
그림 54. 이미지 갤러리 내용보기 ·····101
그림 55. 로그인 후 첫 Home 탭의 구성 ·····102
그림 56. 사용자 개인이 설정한 Mypage ·····103

ABSTRACT

An Implementation Portal System of Base on Ontology to Using Web Service

Choi Bong-Han
Advisor : Prof. Lee Woong-Ki
Department of Computer Science and Statistics,
Graduate School of Chosun University

As the Internet-based E-Business is generalized, an information system at the enterprise is seriously expected the unification of heterogeneous applications and platforms for the B2B E-commercial transactions. There are many difficulties to unify the total business on account of the insufficient policy to effectively manage business data and each business to be treated as several transactions. Therefore, a new concept "EIP(Enterprise Information Portal)", as a new business information system to maximize a business production and pliability, was raised.

The staffs and customers can perform all tasks of enterprise through the enterprise portal. By using the portal, enterprises can perform tasks through the cooperation and can increase their productivities. Furthermore,

by introducing the contents management system of enterprise portal, lots of information property could be effectively managed and the communication functions like E-mail and messenger makes to hold new news and knowledge resources in common.

By introducing the new IT strategy to minimize the complexity and maintenance cost of the enterprise IT infra and to maximize productivity and flexibility of business, SOA(Service Oriented Architecture) rises as a new paradigm at the viewpoint of business. So Web services appeared as an alternative in order to solve the obstacle factors to integrate services between enterprises. Web service is a new form of Web application. The integrated services between the intra- or extra- heterogeneous applications could accept an offer while user could not feel such as calling a method in the local computer.

Therefore this thesis handled the communication between the portal to support the web service of SOA and the portlet by using WSRP protocol, and built all portlet by observing the JSR168 standard for the comparability with other portal frames. By appling the JSR168 and WSRP based portal framework in this paper, the common work space could be offered by one access point inside an enterprise and the business renovation could be accomplished easily by reusing and integrating the many program components which was made by lots of times and costs.

Web service-based portal framework system, which was suggested in this paper supports the personal and cooperated environments and offers integrated management environment of business resources. But, because the

knowledge management and transaction are impossible under this kind of environment, portlet ontology to classify and to manage the information of portlet in the portal system could be constructed. In case of appling the portlet ontology to portal system, an organic mutuality application which is based on the meaning treatment between the portal and each portlet is possible, and knowledge based portlet handling support is possible. If the ontology, which was applied in a portal system, is introduced to the business, the development cost will be reduced by the reuse of the existing resources. And competitive power and technical expertise could be ensured by furnishing an advanced information service.

제 1 장 서론

인터넷을 기반으로 한 e-비즈니스가 일반화되면서 기업의 정보 시스템은 B2B 전자상거래 등을 위해 이질적인 어플리케이션과 플랫폼 통합이 절실히 요구되고 있다. 이에 따라 기업 내외부에 산재한 이질적인 정보 자원을 통합하여, 기업의 생산성 제고와 업무 유연성을 극대화 할 수 있는 새로운 기업 정보 시스템으로 EIP(Enterprise Information Portal) 개념이 대두하게 되었다. EIP는 단순 애플리케이션이나 플랫폼의 통합뿐만 아니라, 그룹웨어, KMS(knowledge management system), ERP(enterprise resource planning), CRM(customer relationship management) 등 기업의 핵심 정보 자원 시스템에서부터 B2C(Business to Consumer), B2B(Business to Business) 등 각종 전자상거래에 이르기까지 기업 내외부의 모든 업무 프로세스와 관련 정보를 인터넷 기반의 단일 접점으로 처리할 수 있도록 통합하는 새로운 개념의 비즈니스 플랫폼이다[5,19,20].

EIP는 1998년 시스코가 CEC(Cisco Employee Connection)라는 기업 정보 포털 시스템을 통해 연간 5,000만 달러의 비용을 절감하게 된 후, 기업 정보 시스템의 새로운 표준으로 자리를 잡게 되었다. EIP를 통해 기업 내외부의 각종 정보 시스템에 분산되어있는 기업 정보 자산과 업무 처리를 통합하여 각자의 역할과 업무에 적합한 개인화된 지식 정보 시스템을 구축할 수 있었다. 기업은 EIP를 활용하여 고객과 협력사에까지 각종 기업 정보 자산을 맞춤형으로 통합 제공할 수 있었다. 즉, EIP는 기업과 구성원 간 정보 자산의 공유와 업무 프로세스의 통합을 실현하는 차세대 기업 정보 시스템이다.

기업에서 EIP 시스템을 도입하는 경우, 기존 시스템과 신규 시스템 간의 원만한 통합(seamless integration)이 이루어져야 한다. 이질적인 시스템의 통합을 위한 방법으로 랩퍼(wrapper)등을 이용한 방법, 즉 시스템 측면으로 이야기하면 각기

별도 시스템으로 구성된 기존시스템과 신규시스템의 의사소통을 하기 위한 특별한 코드로 된 랩퍼들을 구축하는 부분에서 프로젝트의 소요비용 거의 대부분을 차지하게 된다고 할 수 있다. 이러한 문제는 결국 표준화된 인터페이스의 구축이라는 해결 과제를 제시하게 되며, 이에 본 논문에서는 XML 기반 표준 인터페이스를 제공하는 웹서비스를 EIP 시스템에 탑재 구현함으로써 이러한 문제점을 해결하고자 하는 방향을 제시한다.

웹서비스는 로컬 컴퓨터상에 있는 메소드를 호출하는 것과 같이 사용자가 느끼지 못할 정도로 자연스럽게 내부 또는 외부의 이질적인 어플리케이션간의 통합 서비스를 제공받을 수 있는 장점이 있다. 새로운 비즈니스 파트너간의 시스템과의 통합도 자동적으로 이루어지게 된다. 또한 다양한 통신 방식에 의해 구성된 분산 시스템에 대한 접근성이 뛰어나고, 소프트웨어 통합을 자동화함으로써 상호 연결된 작업을 기존에 비해 훨씬 빠르고, 유연하며, 효율적으로 처리할 수 있다.

이에 본 논문에서는 포털 시스템의 최소 단위 어플리케이션인 포틀릿을 웹서비스 기반으로 랩핑하여 포털 시스템에서 활용할 수 있도록 웹서비스의 포틀릿으로 구현하는 방법과 웹서비스를 포틀릿 형태로 활용하여 타 시스템에서 활용할 수 있도록 웹서비스 API로의 변환을 구현한다. 이에 본 논문에서는 포털 시스템의 가장 큰 취약점인 기존 자원들의 재사용을 가능하게 하고, 개발 비용을 절감하며, 또한 새로운 비즈니스 프로세스에 대한 접근을 보다 빠르고, 유연하게 대처할 수 있는 포털시스템을 구성함에 그 목적이 있다.

제 1 절 연구 배경

웹은 현대 생활을 살아가는 우리들에게 선택이 아닌 필수적인 존재가 되었다. 네트워크 환경이 발달하면서 컴퓨팅 자원, 정보, 각종 서비스들이 모두 인터넷을 중심으로 연결되고 언제 어디서나 원하는 정보를 검색, 저장, 공유 등 기존 환경에서 하던 작업들을 수행할 수 있게 되었다.

이처럼 현재의 웹은 방대한 정보를 지니기는 하지만 근본적으로 사용자가 원하는 정보를 활용하고 가공할 수 있는 방법은 제공하지 않는다. 이러한 문제점을 해결하기 위한 관점에서 출발하여 웹의 개념이 점점 확장되어가고 있다. 현재의 웹은 단순한 정보 인프라로서의 단계를 넘어 사용자 환경에 맞는 UI와 업무 그리고 정보를 관리할 수 있는 개인화(Personalization) 환경을 제공한다. 이와 더불어 비즈니스 활동을 지원하기 위한 프로세스 공유와 정보의 공유 등의 협업(Collaboration) 환경을 제공할 수 있는 포털(Portal) 개념이 등장하게 되었다[5].

이러한 개념을 기반으로 정부, 기업, 기관 등의 재래적인 통합정보시스템은 EII(Enterprise Information Integration)와 EAI(Enterprise Application Integration)[2,3,5]를 거쳐 컨텐츠 관리(content management)와 협업(collaboration)을 핵심기능으로 하는 Enterprise Information Portal(EIP)로 진화하고 있다. EIP는 차세대 전사적 정보화 기술의 화두로써 통합정보시스템의 거점 기술이라고 할 수 있다. 현재의 EIP 시스템 시장은 IBM의 WebSphere, Microsoft의 SharePoint, BEA의 Web Logic 등이 국내·외 시장에서 선점 경쟁을 하고 있다. 그러나 이러한 EIP 시스템들은 기존 시스템과 신규 시스템의 통합에 대한 관계 정의나 의미처리가 불가능하기 때문에, 정보자산관리, 협업지원, 개인화 등과 같은 포털의 필수 기능들을 지원하는데 문제점이 있다. 이에 따라, 차츰 기업 정보 자산과 응용 시스템 통합을 기본 목적으로 하는 정보 시스템 형태는 크게

2가지 기술을 기반으로 변모하고 있다[5,7].

첫째, JSR168[1,14], WSRP[11,12,15]와 같은 포털 시스템(portal system) 표준을 기반으로 한 서비스 통합 시스템이다. 엔터프라이즈 포털(enterprise portal)이 증가하면서, 포틀릿(portlet)을 위한 다양한 API들과 원격 비주얼 컴포넌트들을 호출하기 위한 다양한 메커니즘이 개발되고 있다. 그러나 포털 시스템이 차세대 핵심 응용 시스템임에도 불구하고, 이에 대한 기술 기반이 미흡할 뿐만 아니라 호환되지 않는 다양한 인터페이스는 어플리케이션 공급자(application provider), 포털 소비자(portal consumers) 및 포털 서버에게 정보 시스템의 통합 시 호환성 문제점을 안겨 주고 있다. 개발 업체의 고유 기술들을 이용하여 기존 정보 시스템들이 구축되었기 때문에, 포털 시스템간의 포털 컴포넌트, 즉 포틀릿의 재사용에 어려움이 있다[1,14,15].

둘째, 현재 웹 기술은 정보 표현과 단순 정보 제공 위주의 정적인 기능을 지닌 웹 페이지에서 실제 서비스를 수행함으로써 사용자나 기타 응용 시스템에 동적인 기능을 제공할 수 있는 웹 서비스(Web Services)[8,10]로 진화하고 있다. 그러나 현재의 웹 서비스 기술은 기능적인 처리에서는 어느 정도 상용화 수준에 도달하였지만, 의미기반의 지능적인 처리를 하는데 한계가 있다. 이에 따라, 온토로지(ontology)가 핵심 기술이 되는 시맨틱 웹 서비스가 차세대 웹 기술로 부각이 되고 있으며, W3C, OASIS 등 표준화 기관에서 시맨틱 웹 서비스 실현을 위한 표준이 집중적으로 연구되고 있다.

제 2 절 연구 목적 및 필요성

본 논문은 기업 활동에 필요한 기업 내·외적 자산은 물론 사용자 간의 협업체계, 커뮤니케이션 및 제반 온라인을 통한 거래 등 광범위한 모든 자원 즉, 컨텐츠 관리와 분석 및 제공 기능을 웹서비스 기반에서 이루어 질 수 있는 EIP 시스템에 대해

서 연구한다.

현재의 EIP 시스템들은 기업 내 이기종간의 시스템 통합에 대한 표준 인터페이스의 정의에 있어 난점을 보유하고 있으므로, 정보자산관리, 협업지원, 개인화 등의 핵심 기능을 지원하는데 문제점이 있다. 이러한 문제는 결국 표준화된 인터페이스의 구축이라는 해결 과제를 제시한다.

이에 본 논문에서는 XML 기반 표준 인터페이스를 제공하는 웹서비스를 EIP 시스템에 탑재 구현함으로써 기업 내 자산의 통합에 있어 보다 우월한 시스템을 구현하고자 한다. 웹 서비스는 새로운 시스템을 구축하는 것이 아니라 기존에 존재하고 있는 시스템을 통합하여 운영할 수 있는 새로운 패러다임의 한 축을 구성하며, 또한 단순히 통합을 위한 표준 인터페이스의 구현을 위한 도구뿐만이 아니라, 하나의 독립된 웹 어플리케이션을 구현하는 단위로의 역할도 수행한다. 즉, 웹서비스를 포털시스템에 도입함으로써 웹서비스로 구현된 컴포넌트의 조합으로, 개발자들은 새로운 웹서비스 컴포넌트를 개발함과 더불어 기존의 웹서비스에서 필요한 웹서비스의 조합으로 새로운 웹 어플리케이션을 조립해 낼 수 있으며, 컴포넌트의 재활용을 극대화 할 수 있는 기반을 제시한다.

1. 연구 목적

현재의 웹은 방대한 정보 자원의 관리가 어렵고, 목적에 따라 무분별하게 구축되었던 기업 어플리케이션으로 인한 업무의 비효율성 증가, 기존 시스템의 유지관리 비용이 급증하는 여러 문제점들이 제기되었다. 이에 따라 어플리케이션간의 통합과 다양한 시스템 내부에 존재하는 정보에 대한 단일한 접근과 관리를 가능하게 하고 개인화에 따른 차별화된 인터페이스 기능을 제공할 수 있는 포털 개념이 등장하게 되었다. 이러한 개념을 기반으로 하여, 차세대 전산적 정보화 기술의 화두가 되고 있는 EIP 기술이 대두되고 있다.

EIP는 기업 활동 및 핵심 역량에 필요한 내·외적 자산은 물론 협업체계, 커뮤니케이션 및 온라인을 통해 거래되어지는 모든 자원들을 통합관리하고, 분석 및 제공 할 수 있는 시스템이다. 즉, 기업 내·외부 정보를 통합하여 이용자가 단일 게이트웨이를 통해 중요 의사 결정을 내리는데 필요한 개인 맞춤형 서비스를 제공하는 어플리케이션으로 비즈니스 파트너들이 누구나, 언제, 어디서나 다양한 도구를 활용하여 하나의 표준화된 인터페이스로 개인화된 지식과 정보 등을 수집하고 협업 환경의 업무 처리를 할 수 있는 솔루션과 관련한 인프라를 의미한다.

EIP 기술은 특별한 솔루션을 의미하는 것이 아니라 기존에 널리 사용하고 있는 IT 기술의 총 집합체이다. 또한 메타 기능을 지원하고 결과를 통합 그룹화 할 수 있는 강력한 검색엔진을 기반으로 데이터 자동 필터링 기능과 사용자 프로파일을 자동 추적하는 개인화 기능이 기본이다. 또한 임의의 정보 자원에 있어서 통일된 인터페이스를 통해 보여주게 된다. 한 번의 인증으로 여러 업무를 동시에 수행 할 수 있는 환경 제공과 사용자 수준별 보안기능, 데이터 보호 및 무결성 유지 그리고 협업을 위한 막강한 각종 도구를 제공한다. 하지만 막강한 기능을 제공함에도 불구하고 차세대 웹의 궁극적 형태인 의미처리와 각 서비스의 합성 기능들은 지원되지 못한다. 또한 각 포털시스템 간의 상호 운용성을 체계적으로 표현할 수가 없고, 관리 기능도 부실한 상황이다.

이에 본 논문에서는 웹 서비스 기반의 포틀릿을 개발하고 포털 시스템과 포틀릿 간의 상호 운용성을 체계적으로 표현하고자 한다. 그리고 원격지에 존재하는 포틀릿을 위한 WSRP 표준을 적용하여 원격지에 존재하는 웹 서비스를 포틀릿으로 변환하고, 또한 포틀릿을 웹 서비스로 변환할 수 있는 환경을 제공한다. 또한 새로운 서비스 및 새로운 기능을 제공할 수 있는 포틀릿 합성 기능을 제공한다.

2. 연구의 필요성

기업 및 단체의 핵심 역량을 강화하기 위하여 무엇보다도 개별 구성원의 역량 강화가 절실히 요구되며 이를 위해 업무의 통합을 통한 효율성 확보와 사용자 중심의 업무 환경이 중요하게 되었다. 이러한 맥락에서 EIP는 기업의 활동에 필요한 기업 내·외적 자원은 물론 사용자 간의 협업 체계, 커뮤니케이션 및 온라인 거래 등에 이르는 광범위한 모든 자원을 통합하여 관리, 분석, 제공할 수 있는 시스템이며, 궁극적으로 구성원의 능력과 업무의 효율성을 극대화시키기 위한 사용자 중심의 대화형 시스템 환경을 목표로 한다. 또한 자원의 접근 방법을 단일화된 통합 환경을 제공하여야 한다. 이러한 기능을 제공하기 위해 다음과 같은 핵심 기능을 포함하여야 한다.

- Aggregation(집합성)은 기업 내 정보가 포털을 통해 수집되기 때문에 다양한 정보의 자원들이 포털에 집합되어 있음을 의미한다.
- Integration(통합성)은 서로 공간적으로 떨어져 있을 뿐만 아니라 조직화되지 않은 정보 자원들과 어플리케이션들이 조직화되어 통합되어 있음을 의미한다.
- Presentation(표현성)을 통해 도표나 그래프등과 같은 정보의 시각화를 통해 포털 시스템내의 정보 자원을 사용자가 원하는 다양한 형태로 쉽게 표현하여 사용자가 쉽게 이해할 수 있어야 한다.
- Access(접근성)은 포털상의 정보를 어떻게 접근할 것이냐의 관점에 따른 특성이다. 이는 핸드폰이나 전화등과 같은 다양한 접근매체를 통하여 포털상의 정보에 접근할 수 있는 다양한 접근방식을 보장해야 함을 의미한다. 최근에는 핸드폰의 사용이 일반화됨에 따라 핸드폰을 이용한 모바일 뱅킹이 기업포털과 연계하여 핫이슈가 되고 있다.
- Personalization(개인화)을 통해 사용자가 포털에 접근하였을 때 포털의 모든 정보를 필요로 하지는 않으므로 사용자의 업무 또는 목적에 맞는 정보만을 선택하여 사용자 자신이 필요로 하는 기능들로만 구성된 환경으로 커스터마이징 할 수 있는 개인화 기능을 제공해야 한다.

▪Administration(관리) 기능을 통해 포털상의 모든 사용자가 정보를 관리할 수 있는 기능을 제공해야 한다.

이러한 기능들을 포함하는 엔터프라이즈 포털이 증가하면서, 다양한 포틀릿들을 지원하기 위한 포털 프레임워크와 인터페이스들이 주요 업체들에 의해 개발 및 발표되었다. 그러나 포틀릿의 재사용 및 호환성의 문제로 인해 상호 운용성을 위한 인터페이스들은 어플리케이션 공급자, 포털 소비자 및 포털 서버 벤더들에게 많은 문제점을 표출하였다. 이러한 문제점을 해결하기 위해 포틀릿 표준이 개발되었다. 포털 시스템 상에서 표준으로는 JSR(Java Specification Request)168 자바 포틀릿 API와 WSRP (Web Services for Remote Portlets) 표준이 제안되었다. 표준은 포털과 사용자가 시각적인 인터페이스를 통해 웹 서비스들 간의 상호 운용뿐만 아니라, 포틀릿 및 포털 간의 상호 운용성을 제공한다. 이에 본 논문에서는 WSRP와 JSR168 API 표준을 활용한 웹 서비스와 포틀릿간의 상호 운용성의 구현에 연구의 필요성을 두었다. 또한, 웹서비스를 시스템에 활용하여 구성하게 되면 비즈니스 프로세스를 기본적인 표준 빌딩 블록단위로 분할하여 이를 IT 프로세스와 유연하게 일치시키는 특징이 존재하게 된다. 그에 따라 시스템의 생성 블록으로서의 재사용 컴포넌트로 시스템 구축에 실제적이고 가용적인 방법을 활용할 수 있다. 포틀릿이 웹서비스의 단위로 개발되어 장착되게 되면 모듈화 된 재사용 컴포넌트를 개발 활용할 수 있으며, 이에 따라 포틀릿 합성의 유효성을 극대화하는 방안이 필요하다.

포털이 보유해야할 핵심 기능을 보유하고, 해당 핵심기능이 웹서비스 기반으로 구성되고, 또한 비즈니스 프로세스의 신규 또는 기존 비즈니스 프로세스의 통합 등을 처리할 수 있도록 웹서비스를 지원하는 포털 프레임워크의 필요성이 증대하게 되었다. 이러한 포털시스템을 도입할 기업의 경우 해당 포털 프레임워크 상에서 기업에 맞는 서비스를 개발하거나, 통합, 조립함으로써 기업 내 자산을 보다 손쉽게 개발, 설치, 관리 및 변경할 수 있게 된다.

이러한 기술이 정착화 되려면, 이에 따라 웹상에서 어플리케이션의 트랜잭션 처리와 안정성 및 퍼포먼스가 검증된 어플리케이션 서버 상에서 운용되어야 하며, 웹서비스를 지원할 수 있는 WSRP 어댑터를 장착한 포털 프레임워크가 필요하게 되었다. 이에 본 논문에서는 WSRP 어댑터를 장착하고, 기존의 포털의 기능을 보유한 포털 프레임워크를 설계 및 구현한다.

제 3 절 연구 내용 및 구성

본 논문에서 구현하는 포털 프레임워크는 어플리케이션 서버 상에서 운용되도록 구성한다. 포털 프레임워크는 웹서비스를 지원할 수 있도록 WSRP 어뎁터를 장착하고 있으며, 이는 외부의 포틀릿과 WSRP를 통한 통신을 함으로써 포털의 사용자는 본인의 시스템에서 사용하는 것과 같은 효과를 누릴 수 있도록 구성한다. 또한 포털 프레임워크로써 기본적으로 갖추어야 될 기능을 보유한 Content Management , Portal System Management, Collaboration Management, Personalization Manag-ement등의 모듈을 탑재할 수 있도록 설계한다. 이러한 모듈들은 JSR168 기반의 포틀릿을 통하여 서비스 받게 한다. 또한 포털 프레임워크상의 모든 API 들은 웹서비스 기반으로 작성하게 되며, 이러한 서비스를 웹서비스 API의 형태로 제공함으로써 외부의 타 시스템에서도 해당 포틀릿이나, 웹서비스를 웹서비스 기반 포털 프레임워크 상에서 자유로이 사용할 수 있도록 설계한다.

1. EIP의 핵심 기능 연구

포털의 개념 및 진화과정을 통해 포털의 핵심 기능을 논해볼 때 일반적인 기능들로 대략 다음과 같은 6가지 기능이 도출 된다.

Aggregation(집합성), Integration(통합성), Presentation(표현성), Access(접근성), Personalization(개인화), Administration(관리) 등의 기능을 제공하여야

한다.

포털 시스템의 종류에는 수평적(Horizontal) 포털과 수직적(Vertical) 포털 시스템이 있다. 수평적 포털은 앞에서 설명한 대략 6가지 포털이 제공해야 하는 핵심 기능들을 제공해 주는 시스템 구성요소를 말한다. 수직적 포털은 수평적 포털 기반위에 구축하려는 포털의 성격 및 기업의 사업 목적에 따라 응용이 될 수 있는 특화된 시스템 또는 포틀릿들을 의미한다. 이에 본 논문에서는 포털 시스템에서 필수적인 기능들을 연구 및 정의한다.

2. 웹 서비스와 포틀릿의 상호 운용성 연구

엔터프라이즈 포털이 증가하면서, 이른바 포틀릿이라 불리는 포털 컴포넌트를 위한 다양한 API들이 여러 벤더들에 의해 발표되었다. 또한 원격 비주얼 컴포넌트들을 호출하기 위한 다양한 메커니즘들도 다수의 벤더들에 의해 소개되고 있다. 그러나 호환이 어려운 인터페이스들은 어플리케이션 공급자, 포털 소비자 및 포털 서버 벤더들에게 많은 문제점을 표출하였다. 포털 시스템 상에서 이러한 문제를 해결하기 위해, JSR168 자바 포틀릿 API와 Web Services for Remote Portals(WSRP) 표준이 제안되었다. 이러한 표준은 포털과 사용자가 시각적인 인터페이스를 통해 웹 서비스들 간의 상호 운용뿐만 아니라, 포틀릿 및 포털 간의 상호 운용성을 제공한다. 또한 적절하게 이들 표준을 사용한다면, 어플리케이션 공급자나, 포털 소비자들은 특정 엔터프라이즈 포털 제품에 구애받지 않고, 포틀릿이나 사용자가 시각적인 인터페이스를 통해 웹 서비스를 구축할 수 있다. 이에 본 논문에서는 웹 서비스와 포틀릿, 포틀릿과 포틀릿 간의 상호운용성을 극대화하고, 자원의 효율적 사용 방안을 연구한다.

3. 웹 서비스 기반 포털 프레임워크 설계 및 구축

본 논문에서는 포틀릿의 JSR168 표준에 맞추어 포틀릿의 기본적인 환경설정 내용을 XML로 구현함으로써 포틀릿의 추가 및 삭제, 분류화, 개인화 등을 손쉽게 이용할 수 있는 합성 방안에 대해 연구한다.

본 논문에서 포털 프레임워크로써 기본적으로 갖추어야 할 기능을 보유한 Content Management, Portal System Management, Collaboration Management, Personalization Management등의 모듈을 탑재하여 구성된다.

이러한 모듈들은 JSR168 기반의 포틀릿(Portlet)을 통하여 서비스 되게 된다. 또한 포털 프레임워크상의 모든 API 들은 웹서비스 기반으로 작성하게 되며, 이러한 서비스를 웹서비스 API(Operation) 의 형태로 제공함으로써 외부의 타 시스템에서도 해당 포틀릿이나, 포털프레임 등을 자유로이 사용할 수 있도록 연구 및 설계한다.

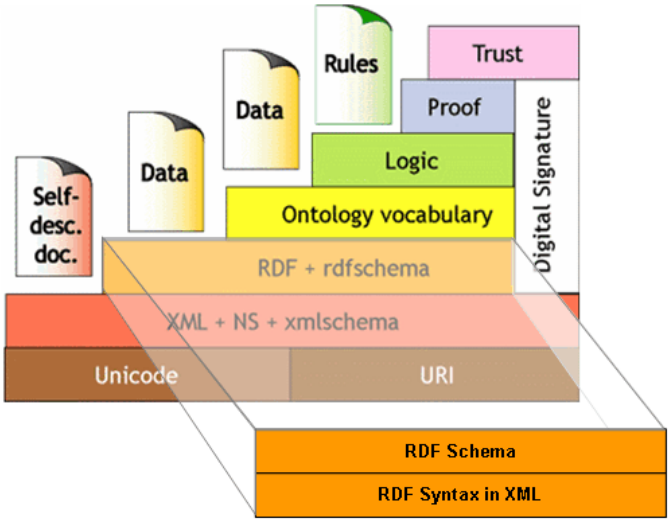
논문의 구성은 다음과 같다. 2장에서는 본 연구의 기반이 되는 온토로지의 개념과 개발 방법론에 대해 알아보고 온토로지 내용을 기반으로 하는 시맨틱 웹과 웹서비스에 대해 개념과 핵심 기술을 알아본다. 3장에서는 본 연구의 핵심이 되는 포털 시스템의 개념을 정의하고, 그 구조 분석을 통하여 기능 분석을 한다. 또한 포털 시스템의 종류를 조사하여 각각의 특징들을 기술하고, EIP 시스템의 기능을 살펴보도록 한다. 또한 3장에서 분석된 연구내용을 기반으로 하여 포털 시스템의 요소 기술을 정의하고, 포틀릿 개념 및 특징과 기능을 기술한다. 그리고 웹 서비스와 포틀릿의 상호 운용성을 제안하기 위한 WSRP와 JSR168 표준 기술을 기반으로 포틀릿과 웹 서비스의 변환 연구를 제안한다. 4장에서는 포털 프레임워크 아키텍처를 구성하고 제안하고 포틀릿의 합성 환경 분석을 기반으로 포털 프레임워크의 기본 화면을 설계 및 구현한다. 5장에서는 포틀릿들의 주요 기능을 설계하고 구현한다. 또한 포털 시스템의 관리 기능을 구현과 이에 따른 데모 이미지를 서술한다. 마지막으로 6장에서는 연구의 결론을 기술한다.

제 2 장 온토로지의 개념 및 시멘틱 웹과 웹 서비스

웹을 이용하여 단순히 어떠한 일을 할 수 있는지의 여부보다는 사용자의 목적을 얼마만큼 만족시킬 수 있는지의 여부가 더욱 중요하다. 현재의 웹은 사실상 사용자의 목적을 만족시켜주는 것보다는 정보의 외형적 표현방법에 치중하고 있다. 현재의 웹에서 사용자는 대부분의 경우 원하는 정보를 획득하여 작업하기 위하여 정보를 검색하고 확인하는 과정을 반복적으로 직접 수행해야 한다.

지금까지 HTML을 사용하여 작성한 수많은 자료들은 사실상 정보를 잘 포장하여, 웹을 보는 "사람"이 읽고 작업하기 편하도록 만들어졌다. 즉, 정보의 내용보다는 사람이 보기 편하도록 외형을 꾸며 제시하는데 중점을 두었다. 차세대 웹에서는 사람뿐만 아니라 사람에게 임무를 부여받은 자동화된 프로그램, "에이전트(agent)"가 사람을 대신하여 웹상의 정보를 읽고 작업하고 나아가 이를 가공하여 새로운 정보를 만들어 낸다. 흡사 가족휴가를 계획하기 위하여 웹상에 있는 여행정보를 사용자가 일일이 직접 찾아서 비행기와 호텔을 예약하는 대신에 여행 대행사(travel agent)에게 대략적 휴가일정과 개인적 선호도만을 말해주면 세부일정과 여행에 필요한 예약이 이루어지는 것과 같은 원리이다.

이러한 여행 대행사 업무를 웹상에 있는 정보를 이용하여 스스로 진행할 수 있는 자동화 프로그램, 즉, 에이전트를 실현하기 위해서는 웹상의 정보가 사람 눈이 아니라 컴퓨터 프로그램이 이해할 수 있는 "의미(semantic)"를 가지고 있어야 한다. 이러한 의미를 표준화된 방법으로 체계적으로 표현하는 것이 시맨틱 웹(Semantic Web)의 주된 목적이다. 의미 정보가 추가되면 웹의 응용범위는 비교할 수 없을 정도로 확장 가능하다. (그림 1)은 Tim Berners-Lee가 제안한 시맨틱 웹 구현을 위한 계층구조이다.



(그림 1) 시맨틱 웹 계층구조

시맨틱 웹은 웹이 제공하고 있는 정보를 잘 정의된 온토로지(ontology)를 기반으로 표현함으로써, 물리적 또는 논리적으로 분산되어 있는 어플리케이션 간의 상호운용성을 제공할 수 있다. 다시 말해서, 시맨틱 웹에서는 서로 다른 데이터 구조를 갖고 있는 어플리케이션들도 온토로지를 통해 상대방의 정보를 이해하고 처리가 가능하다. 웹을 창시한 Tim Berners-Lee는 시맨틱 웹을 잘 정의된 의미를 지닌 정보를 제공하여 컴퓨터와 인간의 협력을 더욱 향상시키는 웹의 확장이라고 정의하고 있다.

이 계층구조의 가장 하위레벨은 웹 프로토콜에서 자원을 지칭하기 위한 주소지정 방법인 URI가 기반이 되고, 이를 기반으로 XML과 Namespace, RDF(S), 온토로지 등의 연구가 진행되고 있다. 그 상위 계층인 Logic 파트에서는 온토로지 상에 기술된 rule을 기반으로 요청된 의미정보들의 관계(relation)를 도출하고, 도출된 정보들 간의 관계를 바탕으로 추론(inference)하여 새로운 의미정보를 산출해 낸다. Proof 계층에서는 하위 계층인 Logic 계층에서의 추론이 옳은지 그른지를 증명하는 단계이며, Trust 계층은 보안에 관련된 사항과 Proof 계층에서 증명된 결과가 과연 신뢰할 수 있는지 없는지를 판단하는 단계이다. 이를 위해서 증명된 결

과에 정보에 대한 등급을 부여하거나 Digital Signature등의 기술을 사용할 수 있다. Trust 계층은 시맨틱 웹의 이상을 실현하기 위한 가장 최상위의 단계이다.

시맨틱 웹에서 추구하는 이상은 웹상에 존재하는 문서들이 단순히 디스플레이만을 위해 사용되는 것이 아니라 컴퓨터 판독이 용이하게 표현되어 있고 데이터간의 연관관계가 잘 정의되어 있어 서로 컴퓨터 프로그램이나 자율적 에이전트가 손쉽게 접근하여 문서의 의미에 적합하게 처리할 수 있게 되어 다양한 응용영역에서 자동화되고 데이터의 공유를 이루는 것이다. 이러한 이상을 실현하기 위해서 요구되는 핵심 요소가 온토로지(Ontology)이다. 철학에서 존재의 본질과 유형에 관한 이론을 의미하는 온토로지는 인공지능이나 웹 관련 연구에서는 기본적으로 개념의 체계적 규정을 말하며 일반적으로 용어와 용어들 간의 관계를 표현하는 컴퓨터 판독이 가능한 공식규정을 의미한다. 시맨틱 웹에서도 온토로지는 특정 도메인에 관한 지식용어들의 집합으로서 이들 용어뿐만 아니라 용어들 간의 의미적 연결 관계(relation)와 간단한 추론 규칙들(axioms)을 포함한다.

제 1 절 온토로지의 개념

온토로지에 대한 정의는 다양하다. Tom Gruber는 온토로지를 “공유된(shared) 개념화(conceptualization)에 대한 형식화(formal)되고 명시적인 명세(explicit specifi -cation)” 라고 정의하였다. 이 정의를 살펴보면 다음과 같은 네 가지 용어가 복합되어 있다.

■개념화(Conceptualization)

대상으로 삼고 있는 도메인에서 일어나는 현상에 연관된 개념들을 파악하기 위한 추상적 모델을 말한다.

■명시적(Explicit)

개념의 사용 유형과 사용된 유형의 제약조건은 명시적으로 정의된다.

■형식화(Formal)

컴퓨터가 이해할 수 있어야 한다는 것을 말한다.

■공유된(Shared)

온토로지는 합의된 지식을 나타내므로 어느 개인에게만 국한되는 것이 아니라 그룹 구성원이 모두 동의하는 개념임을 의미한다.

온토로지는 어플리케이션 사이에서 웹 기반 지식처리, 공유 및 재사용을 가능하게 하는 중요한 역할과 함께 사람과 어플리케이션 사이에서 공통된 주제의 의사소통을 제공한다. 검색엔진을 예로 들면, 찾으려고 하는 키워드와 의미적으로 유사하지만 구조적으로 다른 단어들을 가진 웹 페이지들을 찾아주는 역할을 한다.

온토로지는 용어(terms)와 관계(relations)들로 구성된 사전으로서 특정 도메인에 관련된 단어들을 계층적 구조로 표현하고 추가적으로 이를 확장할 수 있는 추론 규칙을 포함한다. 온토로지의 역할 중 하나는 서로 다른 데이터베이스가 같은 개념에 대해서 서로 다른 단어나 식별자를 사용할 경우에 이를 해결해주는 데 있다. 예를 들어, 주소를 포함하는 두 데이터베이스에서 postal code와 zip code는 같은 것을 의미하다. 이 두 데이터베이스의 정보를 비교하거나 통합하려는 프로그램이 있다면 이 두 단어가 같은 것을 지칭한다는 사실을 알아야 하며 이것이 바로 온토로지를 통해서 이루어진다. 온토로지는 웹 기반의 지식 처리나 응용 프로그램 사이의 지식 공유, 재사용을 가능하게 하는 아주 중요한 요소이다.

제 2 절 온토로지 모델링

온토로지 기반의 의미 정보 모델링은 반복적인 과정을 거쳐 관련 도메인의 어플리케이션이나 문제해결 방법에 온토로지를 적용하면서 또는 해당도메인의 전문가와 상의를 통해 세부적인 사항들을 채워가며 평가하고 수정함으로써 이루어진다.

이 과정에서 온토로지 설계자는 장단점과 다른 해결책의 보완을 고려하여 모델링 방식을 결정해야 한다. 현재 알려진 온토로지 모델링 방법론으로는 Methodology, OTK Method 등 다양하다. 온토로지를 개발하는데 있어서 정석이란 없다. 도메인 모델링과 마찬가지로 온토로지 구축은 고려중에 있는 어플리케이션의 성격과 그 어플리케이션에서 예측되는 확장성에 달려있다. 여러 온토로지 모델링 기법을 분석 및 종합하여 다음과 같이 7단계로 나누어 온토로지를 구축 할 수 있다.

■ 1단계, 온토로지가 적용될 도메인(domain)과 적용 범위(scope)를 결정한다. 온토로지 모델링은 구축될 온토로지가 어느 도메인에서 적용될 것인지를 결정하는 것으로부터 시작된다. 도메인을 결정하지 않고 모델링 한다는 것은 무리이며 자칫 잘못하면 온토로지 구축시 방향성을 잃을 수도 있다. 온토로지가 적용될 도메인을 결정된 후에는 도메인의 특성과 온토로지의 사용목적을 고려하여 온토로지가 적용될 범위(scope)를 정한다. 정확한 범위를 정하지 않고 온토로지 모델링에 들어가면, 온토로지 구축시 일관성을 잃게 되고, 시간이 지날수록 온토로지의 크기가 커져 유지보수 및 평가하는데 문제점을 발생할 수 있다.

■ 2단계, Competency Question 문서 작성한다. 온토로지가 이용되는 동안 요청되는 여러 질의에 대해 응답할 수 있도록 가능한 한 다양한 질의들을 나열함으로써 온토로지 내에 기술되어야 할 내용의 세부 사항들을 정립하는데 도움이 되는 Competency Question 문서를 만든다.

■ 3단계, 도메인 내에서 중요한 의미를 지니는 용어(terms)들을 나열한다. 2단계에서 작성한 Competency Question 문서에 나타난 여러 용어들을 나열해 보면서 온토로지 내에 기술될 여러 클래스(class)들과 속성(property)들을 도출한다.

■ 4단계, 3단계에서 도출된 여러 용어들을 기초로 온토로지 내에 기술될 클래스들을 추출한다. 여기서 클래스는 유사한 속성들을 지닌 요소들의 집합을 의미한다.

또한 이렇게 추출된 클래스는 분류 계층(taxonomic hierarchy)을 구성하는 중요 요소가 된다.

■ 5단계, 3단계에서 도출된 여러 용어들을 기초로 온토로지 내에 기술될 속성들을 추출한다. 여기서 속성(property)은 클래스에 대한 부가적 의미 정보를 정의하고, 다른 의미 정보와의 관계(relation)를 기술한다.

■ 6단계, 5단계에서 추출된 속성에 대한 제한조건(constraint)들을 기술한다. 이에 대한 예로써 "어떤 속성은 그 타입으로'string' 값만 올 수 있고, 정의된 값 중에서 반드시 1개의 값을 가져야 한다. 기본값으로 'default'라는 문자열을 가진다."등이 될 수 있다.

■ 7단계, 클래스에 대한 인스턴스(instance)들을 생성한다.

위에서 기술한 바와 같이 온토로지 기반의 의미 모델링은 다루고자 하는 도메인에서 사용되는 용어들을 정의하고 그들 사이의 관계를 규정하여 그 과정을 구현하는 것을 의미한다. 이 과정에 포함되는 것으로 개념들의 클래스화, 그 개념들의 관계를 클래스의 계층적인 구조로 정립, 클래스들의 속성과 그 속성에 존재하는 다양성과 제한요소 정의, 마지막으로 인스턴스들의 생성을 들 수 있다. 이를 통해 얻어진 공동의 단어들과 공유된 이해를 바탕으로 상호운용시 의사소통의 일관성이 가능해 질 수 있다.

제 3 절 온토로지 기술 언어

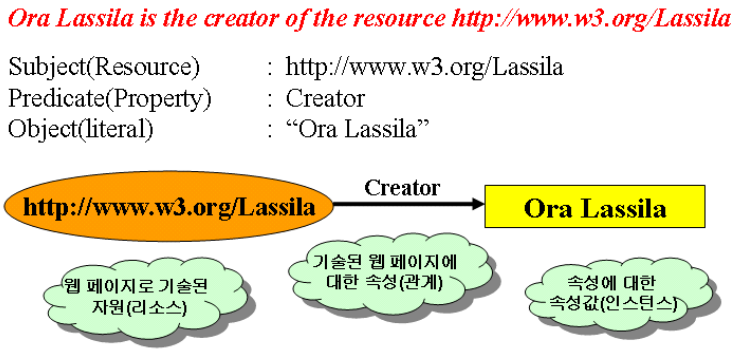
서로 상이한 메타데이터 도메인 간의 의미적 연결에 사용자 그룹사이의 합의가 전제된 XML 스키마만으로는 메타데이터 요소들의 의미와 타 요소들과의 관계를 기

계 가독형으로 표현하는 작업은 불가능하다. 이러한 한계를 극복하기 위해 W3C의 주도로 RDF가 제정되었다.

RDF는 다양한 메타 데이터 사이의 연결을 위해 의미(semantics), 구조(structure) 및 구문(syntax)에 대한 공통적인 규칙을 지원한다. 이러한 메커니즘을 통해 기계가 이해할 수 있는 정보자원을 교환하는 메타 데이터 사이의 상호운용을 지원한다. RDF는 다양한 메타 데이터의 의미를 정의하지 않고 메타 데이터에 필요한 데이터 요소를 정의해 사용할 수 있게 만든다. RDF는 지식 표현분야의 영향을 받아 웹 자원을 기계가 이해할 수 있고, 자동적으로 기술할 수 있도록 설계되었다.

XML은 트리 구조 형태로 고정되게 설계되어 노드가 문서에 포함되어 인덱스 되기 때문에 메타데이터를 표현하기 위한 유연성이 부족하다. 반면 RDF에서 노드는 인덱스 되지 않고 URI를 갖는 자원이기 때문에 메타데이터를 표현하기 위한 유연성을 지원할 수 있다.

XML 스키마는 생성 가능한 XML 문서의 집합을 제약하는 문법적 해석이 주된 기능인 반면, RDF는 의미해석에 이용된다. 다시 말해, XML 스키마는 XML 문서의 구조를 모델링하여 사용하는 데이터 모델링 언어이고, RDF는 지식을 모델링하기 위한 메타데이터 모델링 언어이다. 따라서 메타데이터가 데이터로 인코딩될 때 XML 구문은 매우 유용하지만, 순수한 XML로 메타데이터를 모델링할 때는 의미적 상호운용성에 한계가 있다. 즉, 웹 정보자원을 표현하는데 있어 XML이 구문적 계층에서 데이터의 구조를 정의하는데 이용되고, RDF가 의미적 계층에서 지식표현을 위한 기능을 하는 것이다. 따라서 두 언어는 배타적이기보다 상호 보완적인 특성을 갖고 있다. 그러나 RDF는 속성(property)과 클래스(class)의 명확한 정의, 클래스와 클래스 간의 관계, 속성과 속성 간의 관계를 규정하는 방법은 제공하지 않는다. (그림 2)은 RDF에 대한 데이터 모델을 나타낸다.



(그림 2) RDF 데이터 모델

이에 대한 보완책으로 RDF 스키마는 메타데이터의 속성에 관한 정의, 속성의 적용 도메인과 속성 값에 대한 제어방법 및 클래스 간의 관계 등을 사람이 이해하고 컴퓨터가 처리 가능한 형태로 표현하는 기능을 갖추었다. RDF와 RDF 스키마는 웹 자원 지식화의 가능성을 시사하는 메타데이터 스키마 정의의 근간이다. RDF와 XML 구문을 이용하여 메타데이터를 표현하고, RDF 스키마는 메타데이터의 무결성을 보증하기 위해 구문에 관계없이 메타데이터의 자원과 특성에 대한 유효성을 검증한다.

RDF 스키마는 자원의 특성을 기술하기 위해 사용될 수 있는 자원의 집합이고, 웹 자원을 기술하기 위해 이용된 메타데이터의 구조를 해당 어플리케이션에서 사용되는 특정한 어휘로 정의한다. 메타데이터 스키마와 같은 응용스키마 정의에 사용될 수 있도록 Class, Resource, Property등의 클래스와 이들을 인스턴스화 하여 새로운 자원을 정의할 수 있는 형 체계를 제공하고 있다. 그러나 RDF를 보완하기 위해 나온 RDF 스키마도 동의요소 처리, 역관계(inverseOf), 합집합 관계(unionOf), 교집합 관계(intersectionOf) 등과 같은 의미정보 처리를 위한 주요 관계의 지원이 미비하여 의미정보를 표현하는 데는 한계가 있다. 이에 RDF와 RDF 스키마의 모델링 요소를 확장하고 강화하기 위해서 DAML+OIL이 설계되었다.

DAML+OIL은 미국의 DARPA와 유럽의 IST 프로젝트에 의해 공동으로 개발된 언어로 DAML의 문법적 특성을 상속받고 있다. DAML+OIL은 웹 자원에 대한 시맨틱 마크업 언어이며 W3C의 RDF와 RDF 스키마 표준에 기반을 두고 이들을 확장한 프레임 기반의 온토로지 표현 언어이다.

OWL은 DAML+OIL에 기반을 둔 온토로지 구축 경험을 토대로 개념의 일관성을 확보하여 Class와 Property의 개념 및 그들 사이의 관계를 보다 명료하게 정의한 온토로지 언어이다. 다시 말해서, OWL은 시맨틱 웹 형성의 기저인 온토로지 개발에 필요한 언어 구조를 재정립한 결과로 볼 수 있다. OWL은 웹 온토로지와 연관된 지식 베이스(knowledge bases), 즉, 추론 시스템에 축적된 논리적 명제(assertions)들을 정의하는 언어로서, 이러한 명제는 하나의 온토로지에 근거하거나 또는 OWL이 명시하는 방식에 따라 수집한 다수의 분산형 온토로지에 근거할 수 있다. OWL 명제는 도메인 구성 요소들에 관한 사실과 구성 요소들 간의 관계를 기술하며, 이러한 명제의 집합으로 이루어진 온토로지는 구문적으로 정의되지 않은 사실의 논리적 유추가 가능하다.

제 4 절 시맨틱 웹과 웹 서비스

1990년대의 가장 큰 정보통신분야의 변화로서 아마도 많은 사람들이 웹(web)의 출현을 들 것으로 생각된다. 1990초부터 보급되기 시작한 웹은 불과 10여 년의 기간 동안 학술분야뿐만 아니라 우리의 일상생활에도 지대한 영향을 미치게 되었다. 식사 시간이 되면 오늘 구내식당에서 어떤 음식이 제공되는지 몇 번의 클릭으로 웹에서 간단히 확인할 수 있으며, 다음 주 출장을 위한 왕복 기차표를 예매하고 바로 신용카드로 결재하는 일들은 어느덧 자연스러운 일상이 되어 대수롭지 않은 일이 되었다. 가정의 전화처럼 웹이 없는 생활을 상상하기 어렵게 된 것이다.

웹의 사용자 수와 정보의 양이 증가하면서 사용자의 이용수준과 요구사항도 다양하고 점차 복잡해지고 있다. 가히 정보의 홍수라 할 만큼 정보의 양이 증가함에 따라 목적에 부합하는 필요한 정보만을 효율적으로 추출하고 가공하여 정확하고 적절한 정보를 생성하는 문제의 중요성이 부각되고 있다. 웹을 이용하여 단순히 어떠한 일을 할 수 있는지의 여부보다는 사용자의 목적을 얼마만큼 효과적으로 만족시킬 수 있는가 하는 것이 더욱 중요해지고 있는 것이다. 현재의 웹은 사실상 사용자의 목적을 만족시켜주는 것보다는 정보의 외형적 표현방법에 치중하고 있다. 즉, 사용자의 클릭을 통한 웹 네비게이션과 검색엔진을 이용한 정보소재 파악이 주 이용수단이다. 사용자는 대부분의 경우 원하는 정보를 획득하여 작업하기 위하여 정보를 검색하고 클릭하고 확인하는 과정을 반복적으로 직접 수행하여야 한다. 예를 들어 가족휴가를 보내기 위하여 여행지 정보와 예약, 가격, 결제정보 등을 각각 직접 찾아 검토하고 이를 바탕으로 계획을 세우게 된다. 이러한 과정은 많은 시간과 노력을 요구하는 지루한 작업이 될 수 있다. 지금까지 HTML을 사용하여 작성한 수많은 자료들은 사실상 정보를 잘 포장하여, 웹을 보는 "사람"이 읽고 작업하기 편하도록 만들어졌다. 즉, 정보의 내용보다는 사람이 보기 편하도록 외형을 꾸며 제시하는데 중점을 두었다.

차세대 웹에서는 사람뿐만이 아니라 사람에게 임무를 부여받은 자동화된 프로그램, 즉, "기계"가 사람을 대신하여 웹상의 정보를 읽고 작업하고 나아가 이를 가공하여 새로운 정보를 만들어 낼 수 있도록 정보의 내용을 강조한다. 이러한 기계를 흔히 에이전트(agent) 또는 대행자라고 한다. 흡사 가족휴가를 계획하기 위하여 웹상에 있는 여행정보를 일일이 직접 찾아서 비행기와 호텔을 예약하는 대신에 여행 대행사(Travel Agent)에게 대략적 휴가일정과 개인적 선호도만을 말해주면 세부일정과 여행에 필요한 예약이 이루어지는 것과 같은 원리이다. 이러한 여행 대행사 업무를 웹상에 있는 정보를 이용하여 스스로 진행할 수 있는 자동화 프로그램, 즉, 에이전트를 실현하기 위해서는 웹상의 정보가 사람 눈이 아니라 컴퓨터 프로그램이 이해하기 용이한 "의미" 즉, "시맨틱(semantic)"을 가지고 있어

야 한다. 이러한 의미를 표준화된 방법으로 체계적으로 표현하고자 하는 것이 시맨틱 웹의 주된 목적이다.

의미정보가 추가되면 웹의 응용범위는 비교할 수 없을 정도로 확대될 수 있다. 간단한 정보검색을 예로 들어보면 현재와 같은 검색어의 포함유무에 의한 검색이 아니라 "개념"을 이용한 검색과 이를 이용한 작업지시도 가능해질 것이다. 현재 이러한 정보가공은 많은 사람이 많은 시간을 들여서 행하여야 하는 어려운 작업이다. 일상의 예로 팩스로 전송된 주소록과 엑셀과 같은 스프레드시트(spread sheet)로 전달된 주소록을 비교하면, 팩스 자료는 사람을 위한 자료로서 컴퓨터 처리를 위해서는 사람의 손을 거쳐 다시 가공되어야 한다. 반면 스프레드시트 자료는 컴퓨터 프로그램, 즉 에이전트에 의하여 표나 차트와 같은 다른 정보형태로 손쉽게 변환되거나 처리할 수 있다. 스프레드시트 자료에는 컴퓨터 프로그램이 이해할 수 있도록 자료의 의미가 포함되어 있기 때문에 컴퓨터 프로그램에 의해서 손쉽게 처리될 수 있는 것이다.

최근 차세대 웹 기술의 주역으로 기대되는 시맨틱 웹에 대한 국내외적 관심은 학술적 연구와 병행하여 전자상거래를 비롯한 다양한 분야의 응용연구로 확대되고 있다. 최근 산업자원부에서도 제 2의 정보기술 혁명을 촉진할 차세대 지능형 웹 기술로서의 시맨틱 웹의 중요성과 이에 대한 체계적 대응의 필요성을 표명한 바 있다.

제 5 절 시맨틱 웹의 개발배경

오늘날의 웹이 만들어진 것은 인터넷의 수많은 사람들의 노력의 결과이지만, 웹은 처음에는 전 세계 대학과 연구소에 흩어진 물리학자들 간의 공동연구에 필요한 즉각적 정보교환 방안으로 1989년 팀 버너스-리(Tim Berners-Lee)에 의해 창시되었다. 1991년 일반에 공개된 후 인터넷 대중화에 결정적 역할을 하여 이제는 전

세계적인 사용자를 가지고 있다. 웹은 가히 혁명이라 할 만큼 인터넷 발전에 지대한 영향을 가져왔으며 지금도 WWW 컨소시엄(World Wide Web Consortium 또는 W3C)을 주축으로 발전하고 있다. W3C는 1994년 10월 팀 버너스-리의 주도로 MIT 대학에서, 웹이 처음 만들어진 CERN 연구소와 공동으로 설립되어 웹의 발전에 필요한 기술과 표준의 발전을 도모하고 있다. 웹 문서작성에 필요한 HTML과 XML 등의 규격도 바로 W3C의 산물이다.

이러한 웹 창시와 발전을 주도한 팀 버너스-리가 웹의 다음 발전단계를 지칭하며 새롭게 제안한 시맨틱 웹(Semantic Web: http://www.semanticweb.org/)은 제2의 정보기술 혁명을 촉진할 가장 유망한 차세대 웹 기술이라고 할 수 있다. 시맨틱 웹은 웹상에 존재하는 정보들을 사람뿐만 아니라 컴퓨터 프로그램 같은 기계들이 해독하고 작업하기 용이하게 표현하고 이들 정보간의 유기성까지 체계적으로 표현하여 다양한 응용영역에서 자동화되고 유기적으로 통합화된 정보공유 체제를 이루는 것을 목표로 한다.

시맨틱 웹을 구성하는 핵심 기술요소로는 (1) 웹상의 자원의 성질과 다른 자원과의 관계를 표현하기 위한 방안으로서 RDF(Resource Description Framework)와 같은 자원 서술방안에 관한 기술과, (2) 웹상의 정보를 단순한 데이터 차원에서 처리하여 사람이 의미를 부여하는 현재의 상태에서, 정보 생성단계에서부터 의미를 부여하여 정보가 지식(knowledge)으로서의 가치를 지닌 상태로 향상시킬 수 있는 지식의 체계적 표현방안인 온토로지(ontology)에 관한 기술, 그리고 (3) 이들 자원에 관한 메타정보와 지식체계인 온토로지를 이용하여 추론하고 사용자 대신에 유용한 일을 비교적 자율적으로 실행할 수 있는 컴퓨터 프로그램인 에이전트(agent) 기술로 분류할 수 있다. 이들 요소기술들은 모두 기존의 Unicode를 지원하는 XML을 기반으로 한다.

제 6 절 시맨틱 웹의 구성

시맨틱 웹(semantic web)은 웹상의 정보에 컴퓨터가 쉽게 해석할 수 있는 잘 정의된 의미(semantic)를 부여하여 사람과 컴퓨터간의 협동 작업을 원활하게 하기 위해서 제안되었다. 현재 웹상의 문서는 사람이 읽고 해석하기에 편리하도록 설계되어 있기 때문에 컴퓨터가 문서의 내용을 읽어 그 내용에 적합하게 처리하기에는 어려움이 많다. 시맨틱 웹에서는 자연어 위주의 기존 웹 문서와 달리 컴퓨터가 해석하기 쉽도록 의미를 부여한 계층을 가지고 있기 때문에 자동화된 에이전트나 정교한 검색엔진들이 부여된 의미를 이용하여 고수준의 자동화와 지능화를 이룰 수 있게 된다.

1. 자원서술

의미부여를 위해서는 데이터뿐만 아니라 데이터에 대한 추론을 규정하는 규칙(rule)을 표현할 수 있는 언어가 필요하다. 시맨틱 웹을 위한 의미부여 언어로서 이미 사용 중인 XML(eXtensible Markup Language)과 RDF(Resource Description Framework)를 이용한다. XML은 컴퓨터 판독(computer-readable)이 용이한 문서작성 규정이다. 마크업(markup)이란 문서내용의 역할을 지정하는 추가 정보를 지정한다는 것을 뜻하며 문서에 논리적 구조를 부여하고 정보자체를 기술하는 역할을 한다. 마크업은 HTML에서 사용된 것과 유사한 태그(tag)를 써서 이루어진다. 반면에 HTML과는 달리 XML에서는 사용자가 필요한 태그(tag)를 직접 만들어 웹 문서에 사용할 수 있다. 사용자는 임의의 태그를 지정하여 문서를 구조화 할 수 있으며 컴퓨터 프로그램이나 스크립트(script)는 이렇게 사용자가 정의한 태그를 이용하여 다양한 작업을 수행할 수 있게 된다. XML의 이러한 확장성으로 인하여 XML은 언어를 표현하는 언어역할인 메타언어(metalanguage)의 기능을 갖는다.

XML은 임의의 문서구조를 지정할 수 있는 반면에 문서구조가 갖는 의미에 대해서는 아무것도 정의하지 않으며 따라서 표현한 문서의 의미를 해석하는 역할은 하지 않는다. RDF는 이러한 문서의 의미를 지정하는 역할을 한다. RDF는 기본적으로 세 개의 정보를 지닌 쌍들을 정의한다. 이 세 개의 정보는 일반문장의 주어, 동사, 목적어에 해당하는 것으로서 사람이나 웹 문서 등 특정대상(object)이 특정속성(attribute)에 대하여 특정 값(value)을 가지고 있는 것을 표현한다. RDF는 XML이 메타언어 기능을 사용하여 XML로 표현될 수 있다. 주어에 해당되는 대상과 목적어에 해당되는 값, 그리고 동사에 해당되는 속성들은 모두 URI로 지정될 수 있다. 따라서 사용자는 새로운 개념이나 동사를 URI를 써서 손쉽게 정의할 수 있는 것이다. RDF만을 가지고는 시맨틱 웹의 목적을 만족하기에는 부족한 면이 있으며 DAML+OIL(DARPA Agent Markup Language + Ontology Inference Layer) 언어와 같은 확장된 기능의 의미부여 언어 등이 활발히 연구되고 있다. RDF 모델 및 문법규격은 현재 W3C 추천(recommendation) 상태이다.

2. 지식서술

시맨틱 웹에서 추구하는 이상은 웹상에 존재하는 문서들이 단순히 디스플레이만을 위해 사용되는 것이 아니라 컴퓨터 판독이 용이하게 표현되어 있고 데이터간의 연관관계가 잘 정의되어 있어 서로 독립적인 컴퓨터 프로그램이나 자율적 에이전트가 손쉽게 접근하여 문서의 의미에 적합하게 처리할 수 있게 되어 다양한 응용영역에서 자동화되고 통합화된 데이터의 공유를 이루는 것이다. 이러한 이상 실현을 위해서 한 가지 더 요구되는 것이 있다. 앞의 그림에서 RDF층 위에 존재하는 온토로지(ontology)가 그것이다.

철학에서는 존재의 본질과 유형에 관한 이론을 의미하는 온토로지는, 인공지능이나 웹 관련 연구에서는 기본적으로 개념의 체계적 규정을 말하며 일반적으로 용어와 용어들 간의 관계를 표현하는 컴퓨터 판독이 가능한 공식규정을 의미한다.

시맨틱 웹에서도 온토로지는 특정주제에 관한 지식용어들의 집합으로서 이들 용어 뿐만 아니라 용어들 간의 의미적 연결 관계와 간단한 추론 규칙을 포함한다.
시맨틱 웹에서는 몇몇의 복잡한 대규모 온토로지 대신에 다수의 소규모 온토로지와 이들 간의 상호 참조로 이루어진 웹을 추구한다. 이러한 소규모의 온토로지는 온토로지 저장소(repository)에 보관되어 검색될 수 있으며, 필요한 온토로지를 가져와 확장하거나 특화(subclass)하여 사용할 수 있게 된다. 또한 온토로지를 웹상의 정보검색에 이용한 온토로지 기반검색(ontology-based search)은 새로운 차원의 선택도를 갖는 정보검색을 가능하게 할 것이다.

시맨틱 웹의 등장은 특히 지능형 시스템의 활성화에 크게 기여할 것으로 기대된다. 지능형 시스템의 활용에 장애가 되었던 요소 중에서 지식획득 및 지식공학, 지식표현 문제를 들 수 있다. 시맨틱 웹의 활성화는 이 두 가지 문제에 근본적인 해결책을 줄 수 있다. 웹상의 수많은 사용자가 제공하는 수많은 정보를 자동화된 컴퓨터 프로그램인 에이전트가 처리할 수 있는 형태로 비용을 들이지 않고 이용하게 되면 지식획득의 노력을 줄일 수 있으며 지식표현 문제 또한 에이전트 간에 확장성과 호환성을 유지하는 공유가능한 표현법의 제공으로 해결될 수 있을 것으로 기대된다.

3. 통합운용

시맨틱 웹의 등장은 특히 자동화된 자율적 컴퓨터 프로그램이라고 할 수 있는 에이전트(agent) 시스템의 활성화에 결정적 역할을 할 것으로 기대된다. 웹과 같은 분산 환경의 증대와 사용자 수의 급속한 증가는 분산 환경과 자율적 기능을 전제로 하고 있는 에이전트 기술의 본격적인 실용화를 가져오고 있다.
에이전트 기술은 웹 기반 소프트웨어 시스템 개발기간을 단축시키고 웹 환경에 적응하고 새로운 다양한 수요의 확장을 포용하는 강건하고 융통성 있는 소프트웨어 시스템 개발의 필수요소가 되고 있다. 시맨틱 웹 환경에서 자원서술을 위한

RDF나 지식서술을 위한 온토로지가 비교적 정적인 구조를 가지고 있는 반면에 에이전트는 이러한 자원정보와 지식을 바탕으로 사용자의 요구에 맞게 정보를 추출하고 가공하여 제공하는 동적인 역할을 담당하게 된다. 그러므로 에이전트 기술과 시맨틱 웹의 이러한 상호보완적 상승작용을 통한 에이전트 기술의 활성화와 시맨틱 웹의 실현이 이루어 질 것으로 기대된다.

제 7 절 시맨틱 웹의 활용

시맨틱 웹의 활용분야는 현재 웹과 관련된 모든 분야라고 할 수 있지만 이 절에서는 전자상거래를 한 가지 예로 제시하고자 한다. 전자상거래 분야에서는 거래당사자 간에 필요한 다양한 표준을 바탕으로 한 전자상거래 프레임워크를 위한 체계적 노력이 진행되고 있다. RDF 자원서술, 온토로지 기반 지식서술, 에이전트 중심 통합처리로 대변되는 시맨틱 웹 기술은 전자상거래 프레임워크와 직접, 간접적으로 밀접한 관계를 가지고 있다. 특히 전자상거래 프레임워크(framework)나 전자카탈로그 또는 상품분류 코드 등의 전자상거래 관련 작업은 XML을 기반으로 하고 있으며 기본적으로 시맨틱 웹에서 추구하는 온토로지 연구목표와 유사한 목표를 가지고 있다. 즉, 상품으로 대변되는 “개념”들의 체계적 정리가 전자상거래에도 필요한 것이다. ISO/IEC13250:2000 Topic Maps 표준도 이러한 노력의 일환이라고 할 수 있다.

전자상거래 프레임워크는 컴퓨터 및 컴퓨터 망을 이용하여, 한 기업체의 비즈니스 경계를 넘어 거래 상대자와의 상거래를 가능하게 하는 기능을 규정하는 일반적 골격을 의미한다. ebXML은 XML을 기반으로 하는 프레임워크며 위의 전자상거래 과정의 많은 부분들은 XML을 이용한다. 이러한 전자상거래 분야의 표준화 및 이를 바탕으로 한 전자상거래 프레임워크의 개발은 W3C, ebXML, UN/CEFACT 등의 컨소시엄이나 기관이 주도하여 위에서부터 아래로 보급하는 형태를 가지고 있다. 반면에

시맨틱 웹은 DAML에서 추구하는 바와 같이 위에서부터 아래로의 주도적 작업이 아닌 아래서부터 위로 진행되는 온토로지의 보급을 지향한다고 할 수 있다. 이러한 방향은 흡사 웹의 발전과 유사하다고 할 수 있다. 웹상에 정보를 제공하는 개개인이나 기관은 단지 정보를 표현하는 방법인 HTML을 이용하여 정보를 표현하였고 이러한 표준을 기반으로 한 웹에 의해서 정보의 전달과 공유 등의 활용이 가능하였다. 시맨틱 웹의 온토로지도 같은 맥락으로 지식을 표현하는 기본 표준만을 제시하고 각각의 정보 제공자가 이를 기반으로 지식을 "편집"하여 제시하고 결과적으로 시맨틱 웹에서 추구하는 데이터 단계보다 상위 단계인 지식단계에서의 정보 활용이 자연스럽게 이루어지리라는 것이다. 이렇게 제시된 웹상의 지식들은 다른 사용자가 그대로 활용하거나 또는 보완하고 변형하여 새로운 지식을 창출하는 밑거름이 될 수 있는 것이다. 지식의 생성과 창출이 다시 말하자면 아래(정보 생성자)로부터 점차 위(다른 정보 생성자 또는 소비자)로 확산되는 형태를 갖게 되는 것이다.

전자상거래는 정의상 거래 당사자 간의 상호 작용 있게 된다. 거래 당사자가 개인 또는 기업이나 정부 등의 기관 등으로 다를 수 있으나 웹상에 존재하는 거래 당사자에 관련된 다양한 정보도 전자상거래에 이용하는 것이 필수적이다. 현재의 웹을 대상으로 한 데이터마이닝도 바로 이러한 정보를 이용하기 위한 목적이다. 시맨틱 웹에서는 이러한 데이터마이닝이 자동화된 컴퓨터 프로그램인 에이전트에 의하여 자연스럽게 이루어질 수 있게 되어 거래자의 편리성이 대폭 증대되고, 이에 따라 거래의 실현 가능성도 함께 증대되어 전자상거래와 e-비즈니스가 획기적으로 증대될 것으로 기대된다. 또한 전자상거래에서 추구하는 표준화와 전자상거래 프레임워크의 개발 노력 또한 시맨틱 웹과 공용할 수 있는 영역을 확대하는 것이 상호 보완적 발전에도 유리할 것이라고 여겨진다.

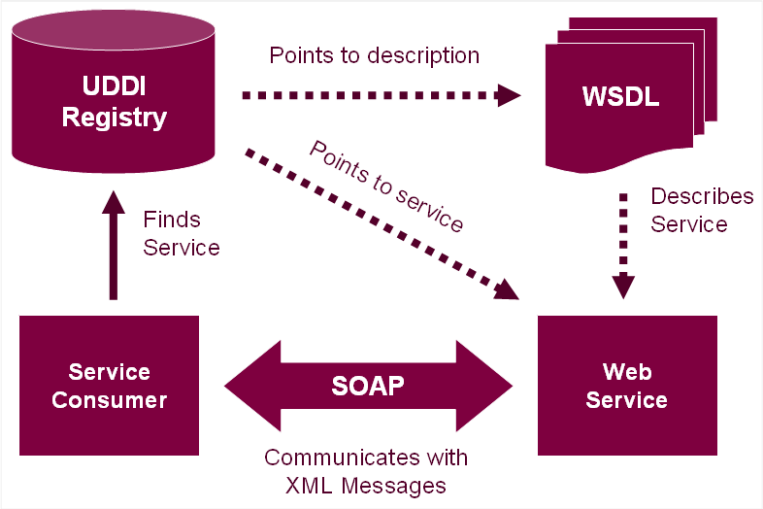
제 8 절 웹 서비스의 개요

현재의 컴퓨팅 환경은 이기종이 복합적으로 존재하고 다양한 종류의 어플리케이션, 프로토콜, 포맷들이 혼재하는 매우 복잡한 것이기 때문에 이들을 하나로 통합하기 위해서는 어느 정도 정형화된 규약이 필요하다. 더불어 보안 문제가 지속적으로 제기되는 환경에서 방화벽을 경유해 데이터와 메시지를 전달할 방법도 찾아야 한다. 또한 경우에 따라서는 원격지에 있는 서비스 객체나 API를 사용할 방법도 강구해야 한다. 그리고 어떤 서비스가 어떤 서버에 위치하는지를 알 수 있는 디렉터리 서비스도 매우 중요한 요소이다.

웹 서비스는 표준 인터넷 프로토콜을 사용하여 접속할 수 있는 프로그램화가 가능한 어플리케이션 로직이다. 이는 컴포넌트 기반 개발기법과 웹의 장점만을 활용하며, 컴포넌트처럼 웹 서비스는 서비스의 구현 결과에 대한 걱정 없이 재사용이 가능한 블랙박스 기능을 가지고 있다. 웹 서비스는 특정 객체를 위한 프로토콜이 아닌, HTTP나 FTP와 같이 언제 어디서나 사용할 수 있는 웹 프로토콜이나 데이터 포맷을 사용하여 접속한다. 또한 웹 서비스의 인터페이스는 전적으로 웹 서비스가 수용하고 생성할 수 있는 메시지로 정의된다. 사용자가 웹 서비스 인터페이스가 가능한 메시지를 생성하고 활용할 수만 있다면, 웹 서비스는 모든 프로그래밍 언어와 플랫폼에서 구현될 수 있다.

웹 서비스의 이용 절차는 먼저 서비스 제공자는 공개할 서비스들을 UDDI 레지스트리에 등록한다. UDDI 레지스트리는 웹 서비스에 대한 정보와 이 서비스에 대한 추가 정보의 저장소를 의미한다. 클라이언트가 자신이 이용하고자 하는 웹 서비스를 UDDI에서 검색하면, UDDI는 검색된 웹 서비스의 위치 정보를 클라이언트에 반환한다. 이후 클라이언트는 반환된 서비스의 위치정보를 참조하여 WSDL 파일을 요청한다. 이 요청에 대해 서비스 제공자는 서비스의 상세 정보가 기술되어 있는 WSDL 파일을 서비스 클라이언트에게 반환한다. 이러한 일련의 과정을 거친 후에

클라이언트가 자신이 이용하고자 하는 웹 서비스를 SOAP 메시징을 통해 요청하면, 서비스 제공자는 요청된 서비스를 처리한 후 결과를 SOAP 메시징을 통해 서비스 클라이언트로 반환한다. (그림 3)는 웹 서비스 이용 과정을 보여주고 있다.



(그림 3) 웹 서비스 이용 절차

웹 서비스는 컴포넌트와 같이 서비스 구현 방법을 고려하지 않고도 다시 사용할 수 있는 블랙박스 기능을 제공하며, 제공된 서비스를 설명하는 잘 정의된 인터페이스나 계약서도 제공한다. 서비스를 프로그램적인 인터페이스로 제공하는 경우 핵심적인 이점은 사용자 인터페이스의 설계나 사용자가 원하는 성능의 추가를 염려하지 않고 핵심 성능 개발에만 집중할 수 있다.

제 9 절 웹 서비스의 표준기술

Web Service 는 소위 말하는 XML형태의 RPC 프로토콜이다. 물론 XML-RPC, XMOP(XML Metadata Object Persistence)과 같은 표준이 있기는 하지만, XML-RPC, XMOP는 널리 받아들여지지 못했다. 이때 SOAP (Simple Object Access Protocol)이

새롭게 등장했다. SOAP 0.9 스펙은 기존의 COM을 기초로 하여 Microsoft사를 주축으로 1999년 처음 세상에 등장했으며 1999년 11월 1.0을 그리고 2000년 4월 1.1 버전이 그리고 현재는 1.2 버전(http://www.w3.org/TR/2001/WD-soap12-20010709/)이 발표되었다. 뿐만 아니라 관련된 프로토콜들이 속속 등장했으며 그 중 가장 주목할 것들은 바로 SOAP과 함께 WSDL(Web Service Description Language), DISCO(Web Services Discovery), UDDI(Universal Description, Discovery and Integration) 같은 것들이 있다.
이후 Web Service는 W3C표준 절차를 밟고 있을 뿐 아니라, IBM, Oracle, SUN, Borland 같은 주요 벤더들에게 폭넓게 지지를 받고 있으며 각 벤더들의 주요 개발툴들은 이 Web Service를 수용하기 위해서 끊임없이 노력하고 있다.

Web Service는 향후 IT분야에 큰 변화를 초래할 것이다. 이것은 마치 FTP 같은 파일 전송 프로토콜에서 시작한 인터넷이 RPC(Remote Procedure Protocol)로 클라이언트/서버 환경의 중요 인프라로 떠올랐듯이, Web Service는 HTML 기반의 페이지 서비스로써의 웹을 진정한 분산 어플리케이션의 중요 인프라로 만들어 줄 것이다.

1. 웹 서비스의 데이터 표준, XML

XML은 웹상에서 구조화된 문서를 전송 가능하도록 설계된 표준화된 텍스트 형식이다. 차세대의 웹 환경을 논할 때 빼놓을 수 없는 것이 XML이다. 사실상 XML이 이후의 모든 웹 서비스에서 핵심이라 할 수 있다. XML을 사용함으로써 얻게 되는 가장 큰 이점은 데이터를 주고받는 형태가 표준화된다는 것이다. XML을 지원하는 서버라면 서로 주고받는 데이터는 읽을 수 있는 형태의 텍스트로 전달되고 이 텍스트는 정의된 태그에 의해서 해석될 수 있으므로 개발과 디버깅에 드는 시간을 대폭 단축할 수 있다. 더불어 방화벽 안쪽에 설치된 서버라고 하더라도 대개 웹 서비스를 위한 포트는 오픈되어 있으므로 데이터를 요청해서 가져오는 것이 가능

하다.

앞으로는 XML을 지원하는 어플리케이션들이 점점 많아지고 또 이 어플리케이션들은 서로가 XML 포맷으로 데이터를 주고받으면서 최종 사용자가 받는 서비스의 질이 향상될 것이다. 특히 서비스 제작자들이 현재의 웹 인프라를 이용해 다른 서비스에 접근해 데이터를 받아올 수 있다는 사실은 대단한 가능성을 내포하고 있다.

XML은 데이터의 표현과 구조가 분리되어 있기 때문에, 데이터 자체는 웹 사이트들 간에 손쉽게 조직화되고 프로그래밍 가능하며 편집되고 교환될 수 있도록 하기 위하여 제한을 받지 않는다. XML은 인터넷 시대를 위한 공통 언어이다. 마치 웹이 사용자들과 어플리케이션 간에 의사소통에 혁신을 몰고 온 것처럼, XML은 어플리케이션이 각각 다른 어플리케이션의 교환을 가능하게 한다. 서비스 제작자들이 XML에 친숙해짐에 따라, 단순히 데이터를 전달하기 위한 수단으로서 XML을 사용하는 단계를 넘어서고 있다.

2. 웹 서비스의 데이터 전송 표준, SOAP

네트워크 환경에서 프로그램은 원격 객체와 서비스를 이용하고 상호 작용함으로써 그 확장성을 무한히 향상시킬 수 있다. 이러한 분산 어플리케이션을 실현하는 중요한 기술로서 RPC가 있다. 지금까지의 여러 RPC 기술들은 플랫폼 호환이나 보안 측면에서 구현상 제약이 많았다. 이에 XML의 유연성과 HTTP의 대중성으로부터 SOAP이 제안되었다. SOAP(Simple Object Access Protocol)은 HTTP를 보완하는 차세대 인터넷 프로토콜이다. SOAP은 인터넷이 기업 컴퓨팅의 핵심 플랫폼으로 자리잡아가고 있는 상황에서 인터넷을 통한 원격 어플리케이션 활용을 원활하게 지원하기 위한 목적으로 만들어 졌다.

원격 어플리케이션과 통신하는 데는 분산 객체 프로토콜이 유용하다. 현재 DCOM, 객체관리 그룹에서 정의한 IIOP 등 다양한 분산객체 프로토콜들이 사용되고

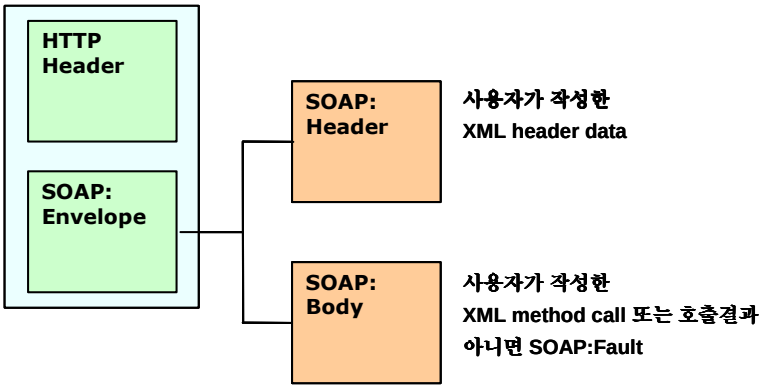
있다. 이 프로토콜은 모두 클라이언트로 하여금 서버 어플리케이션에 대한 RPC를 요청하고 그에 대한 응답을 수신할 수 있게 한다. 방화벽에 막히는 분산 객체 프로토콜은 인트라넷에서 사용하면 매우 잘 작동한다. 하지만 이 프로토콜을 인터넷 공중망에서 사용하면 약간의 문제가 발행한다. 인터넷에 연결된 서버들은 모두 잠재적으로 어떤 인터넷 사용자도 접근이 가능하다. 그래서 보안에 문제가 생길 가능성이 크다. 이런 문제를 해결하기 위하여 대부분의 기관들은 접근 가능한 웹 서버에 방화벽을 설치하여 위험스러운 접근을 통제한다.

TCP/IP 아키텍처의 경우, 널리 사용되고 있는 프로토콜들은 각기 고유한 포트 번호를 할당받으며 이 프로토콜을 사용하여 만들어진 요청은 해당 포트 번호를 운반한다. 예를 들어, HTTP는 포트 80번, FTP는 포트 21번을 할당받는다. 대부분의 방화벽은 특정한 프로토콜이 사용하는 포트로 전송된 모든 트래픽을 거부함으로써 특정 프로토콜을 차단한다.

HTTP 프로토콜은 매우 단순하다. 그것은 주로 브라우저로부터 서버로 전송되는 몇 가지 기본적인 단어들, 예컨대 GET, POST 등으로 구성된다. 각 단어의 뒤에 다른 정보들이 이어질 수 있다. 그 정보들은 모두 단순한 문자열의 형태로 전송되는 것이 전형적이다. SOAP은 이 가운데 어떤 것도 변경하지 않으며 HTTP 자체에 어떤 추가할 것도 요구하지 않는다. 그 대신에 SOAP은 요청 및 응답 메시지의 포맷을 정의하는데 XML을 사용하고 정상적인 HTTP POST 명령을 이용하여 이 정보를 전송하도록 허락한다. 때문에 모든 SOAP 트래픽은 포트 80번을 통하여 실행된다. 이것은 SOAP가 인터넷상에서 어떤 웹 서버와도 작동할 수 있음을 의미한다. SOAP은 방화벽이 문제가 되지 않는다. SOAP를 설계할 때 일차적으로 목표로 삼은 것 가운데 하나가 HTTP, 방화벽, 프록시 등 인터넷의 기존 인프라의 상위에서 효과적으로 사용될 수 있음을 보증하는 것이다.

SOAP은 XML을 요청(request)/응답(response) 메시지의 내용을 XML로써 표현하고, 전송 프로토콜로서 HTTP를 채택하였다. SOAP 메시지는 크게 HTTP상에서 XML 메시지를 전달하기 위해 부가적으로 필요한 정보를 포함하고 있는 HTTP Header 파트와 파라미터, 데이터타입, 반환값 등 실제 전달되어야 할 값들을 포함하고 있는

SOAP Envelope 파트로 구분할 수 있다. SOAP Envelope 파트는 (그림 5)와 같이 다시 클라이언트가 메시지를 처리하는데 필요한 부가적인 처리 정보를 지니고 있는 SOAP Header 부분과 실질적인 메시지를 지니고 있는 SOAP Body 부분으로 구분된다.



(그림 4) SOAP 메시지 구조

(그림 4)는 SOAP 메시지의 전체적인 구조를 보여주고 있다. 분산 객체 프로토콜은 원격 어플리케이션을 활용하는데 효과적인 수단이지만 오늘날의 인터넷 환경에서 그대로 사용하기 위해서는 인프라의 변경이 필요하다. SOAP은 이러한 문제를 해결해 보안상의 문제없이 기존의 인프라를 그대로 활용 가능하다.

3. 웹 서비스에 대한 위치 정보 서비스, UDDI

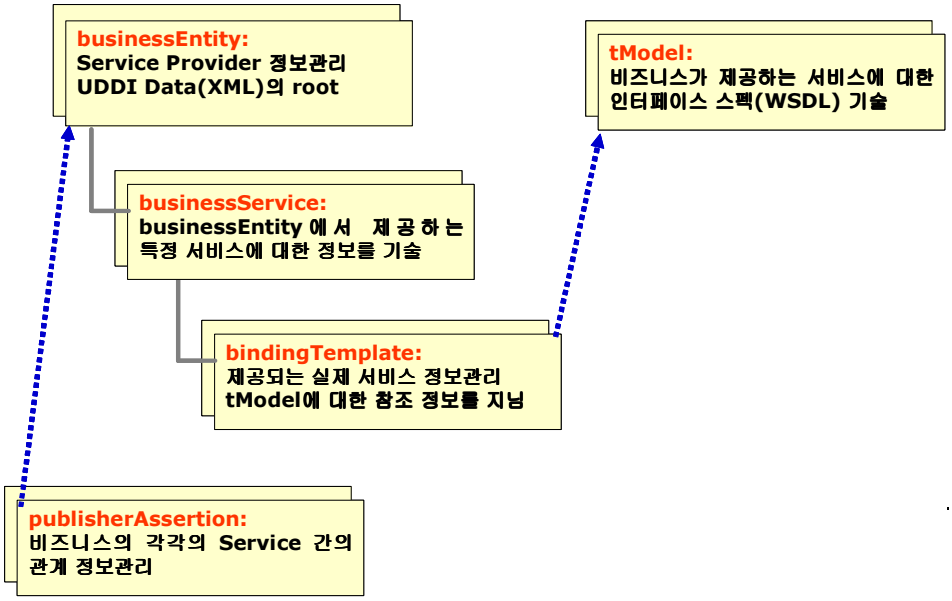
UDDI(Universal Description, Discovery & Integration)는 기업으로 하여금 자신이 서비스의 제공자임을 드러내어, 가용한 제품과 서비스 설명을 레지스트리에 공개한다. UDDI를 사용하면 협력업체에 의해 제공된 프로그래밍이 가능한 웹 서비스의 정보를 찾을 수 있다. 또한 이 레지스트리를 사용하여 모든 조직에서 제공할 웹 서비스에 대한 정보를 게시한다. 프로그래머의 인터페이스는 XML, SOAP 및 HTTP에 프로그래머가 프로그램에서 직접 레지스트리와 상호작용 할 수 있게 하는 UDDI 레지스트리 사양의 일부로 정의되어 있다.

이외에도, UDDI 레지스트리의 인스턴스 및 프로그램 간의 상호작용을 관리하려면 정의된 SOAP 기반 메시징 인터페이스의 의미와 특정 XML 데이터 구조에 대한 지식이 필요하다. 이러한 필수 지식을 신속하고 정확하게 익힌다면 UDDI와의 상호작용을 실현할 수 있다. 즉, 서비스 등록의 부가 가치 활동을 시작하기 위해 이러한 지식을 배워야 하는 것은 웹 서비스를 사용하여 실제 업무 문제를 해결하기 위해 넘어야 할 장벽이다.

UDDI는 표준 인터넷을 통하여 글로벌 레지스트리에 저장된 정보를 공유하도록 구성되어 있으며, 이러한 방식을 이용하여 전 세계의 원격지 구성 요소를 간단하면서도 동적으로 사용이 가능하다. 또한 UDDI는 웹 서비스와 통합되어 있기 때문에 웹 서비스의 설명과 발견하는 기능이 연동되어 있다. 이것은 마치 전 세계 모든 기업의 상호와 전화번호를 기록한 거대한 전화번호부와 같은 역할을 한다. 그래서 이 UDDI를 통하여 개발자는 자신이 원하는 기능을 제공하는 서비스 제공자를 발견하고 사용할 수 있다.

UDDI를 사용하는 주목적은 다른 사용자에게 자신의 존재를 알리고 업무와의 상호작용에 사용할 수 있는 웹 서비스를 제공했음을 알리는 것이다. 이러한 서비스의 예로 구매 주문, 송장, 발송 통지 등의 업무 문서를 위한 서비스를 들 수 있다. 제공한 서비스를 관리하는 데 사용하는 도구 및 제품에 따라 UDDI 레지스트리를 사용하여 해당 서비스에 대한 기술 정보를 광고할 수 있다.

UDDI를 사용하면 서비스 제작자들은 자신들이 만든 서비스가 어떤 기능을 갖는지를 묘사하고 등록할 수 있으며 이 서비스를 사용하는 측에서는 어느 곳에 어떤 서비스가 있는지를 지정된 프로토콜을 이용해 매우 쉽게 찾아낼 수 있다. 현재는 각 서비스에 대한 정보를 얻어내는 방법이 통일되어 있지 않고 각 플랫폼, 시스템마다 다르게 되어 있기 때문에 어플리케이션에서 사전 정보 없이 원하는 서비스를 찾아내기가 매우 어렵다. UDDI를 사용하게 되면 어플리케이션은 각 디렉터리 서버가 지원하는 안에서 어떤 서비스가 가능한 지를 바로 질의해서 얻어낸다.(그림 5)는 간단한 UDDI 데이터 구조를 보여주고 있다. 아래의 다섯 요소가 UDDI의 핵심적인 구성 요소이다.



(그림 5) UDDI 데이터 구조

앞으로는 각 어플리케이션에서 UDDI에 접속해 지능적으로 정보를 수집하고 이를 사용자에게 종합하여 반환하는 방식이 일반적인 형태가 될 것으로 전망된다. 이러한 UDDI 디렉터리는 앞으로 더욱 활발해 질 B2B 전자상거래에서 없어서는 안 될 중요한 요소이다.

4. 웹 서비스 기술 언어, WSDL

WSDL(Web Services Description Language)은 문서 지향적 또는 프로시저 지향적인 정보를 포함한 메시지에서 작동하는 종점 집합으로서의 네트워크 서비스를 설명하는 XML 형식이다. 이러한 작동 및 메시지에 대해 대략적으로 설명한 다음, 구체적인 네트워크 프로토콜 및 메시지 형식으로 연결하여 종점에 대해 정의한다. 관련된 구체적인 종점은 추상 종점(서비스)과 결합되어 있다. WSDL은 메시지를 받거나 보내는 순서규칙과 같이 서비스의 동작을 표현하는 방법과 서비스를 작성하

는 방법을 포함한다. 서비스의 합성은 모든 형식을 지원해야 하지만 런타임에 교환되고 바운드되는 서비스 참조와 함께 전달되는 참조를 허용해야 한다.

WSDL은 웹 서비스를 기술하는 일종의 스크립트 언어이다. WSDL은 XML 포맷으로 구성되고 HTTP를 통해서 전달될 수 있으며 다르게 말한다면 인터페이스를 정의하는 IDL(Interface Definition Language)에 해당한다. 즉 이 서비스가 어떤 메소드, 어떤 속성을 가지며, 어떤 인수로 호출해야 하고 어떤 방식의 반환값을 제공하는지를 알려주는 것이다. 이 내용을 알게 되면 클라이언트에서는 알아낸 인터페이스 규약에 맞추어 호출하고 서비스를 사용할 수 있게 된다.

제 3 장 포털 시스템의 개념 및 핵심기술

인터넷의 기능이 고도화 되고 방대한 정보 자원의 저장고로 확산되기 시작하자, 인터넷 서비스 사용자들은 점차적으로 곳곳에 산재되어 있는 인터넷 서비스를 찾아다니며 이용하는 것 보다 효율적인 정보 접근을 보장하는 인터넷 관문이 필요하게 되었다. 이러한 사용자의 요구 사항을 충족시키기 위하여 인터넷 포털 사이트들이 개설되었고, 나아가서 사용자의 서비스 이용 성향을 파악하여 검색 서비스, 쇼핑 서비스, 주식 서비스 등을 사용자가 원하는 정보 자원을 맞춤형으로 일괄 지원하는 개인화(personalization) 기능을 탑재한 포털 시스템이 등장하게 되었다 [5,6,7,18,19,20].

본 장에서는 기업, 기관 등 조직의 정보 자원을 효율적으로 관리하고 개인화 기능 등 다양한 정보 서비스를 지원하는 EIP 시스템의 종류와 기능을 분석하여 웹서비스 기반의 EIP 프레임워크가 갖추어야 요구 사항을 도출한다.

제 1 절 포털 시스템의 구조

포털 시스템은 기업 정보 자산의 효율적인 관리를 위한 단일화된 통합 환경을 구축하고, 사용자에게는 개인화 등 정보 자산의 효율적인 관리를 통하여 업무 효율을 극대화할 수 있는 기능을 제공한다. 기업이나 조직의 환경이 동적으로 급변하게 됨에 따라, EIP 시스템은 조직의 생존을 위한 필요 요건이 되고 있다.

본 절에서는 포털 시스템의 개념을 정의하고, 포털 시스템의 종류와 분류를 기술한다.

1. 포털 시스템의 개념 및 정의

일반적으로 포털(portal)이란 가상적인 공간의 정보자원이나 서비스에 접근하기 위한 입구(entry point)를 의미한다. 인터넷상의 대표적인 포털로는 Yahoo, AOL, MSN등이 있다. 이들 포털은 거대한 인터넷 정보 자원을 내용별로 조직화하여 사용자가 원하는 다양한 정보 서비스를 제공한다. 최신 뉴스서비스, 주식시세 정보서비스 그리고 날씨 정보 서비스 등 이런 특정 분야의 정보 서비스를 일괄적으로 제공받을 수 있다. 일반적으로, 포털 내에는 다양한 정보들이 망라되어 있고, 이를 기반으로 사용자에게 정보 접근을 위한 다양한 정보 서비스를 제공하고 있다.

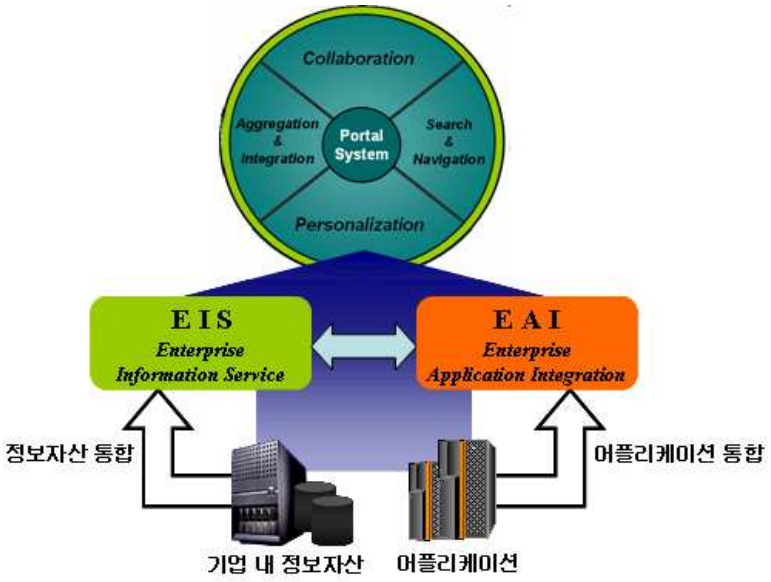
인터넷 포털의 유용성이 증대되고 사용자의 다양한 요구가 표출되어 짐에 따라, 인터넷 포털이 제공하는 기능에도 커다란 변화를 가져 왔다. 사용자마다 서비스 이용 성향이 다르고, 관심분야도 다르기 때문에, 사용자가 인터넷 포털에서 제공하는 모든 서비스들을 필요로 하지 않으며 정보 이용정도에서도 편차가 심하다. 쇼핑을 주목적으로 하는 사용자는 쇼핑 섹션에서 많은 시간을 할애하며, 정보검색을 주목적으로 하는 사용자는 정보검색 섹션에 이용정도가 집중된다. 이렇기 때문에 포털 시스템은 각각의 사용자별로 맞춤 정보 서비스를 제공하는 개인화(personalization) 기능을 제공하고 있다. 개인화 기능을 이용하여 포털 사용자는 자신이 원하는 정보만으로 구성되는 개인화 된 포털 사이트를 구축할 수 있다. 포털은 정보 접근을 위한 관문일 뿐만 아니라, 다른 사람들과 메일을 송수신하거나, 게시판을 통해 정보 자원을 공유하는 등 사용자가 정보 생활에 필요로 하는 제반 기능을 제공하는 가상공간 (virtual space)이 되었다.

반면에, 기업이나 조직의 포털 시스템을 위한 EIP라는 용어가 처음으로 등장한 것은 1998년 메릴 린치(Merrill Lynch)에서 발간한 보고서에서 다음과 같은 EIP 정의가 등장하면서 일반화 되었다. EIP(Enterprise Information Portal)는 기업 내외부의 정보 자원을 통합하여 사용자가 단일 게이트웨이(gateway)를 통해 일괄적으로 업무를 처리하고 중요한 의사결정을 내리는데 필요한 맞춤 정보를 제공할 수 있게 해주는 어플리케이션이다[3].

EIP는 기업의 내부 및 외부에 산재해 있는 정보를 결합, 관리, 분석하고 배포하는 소프트웨어 어플리케이션의 집합으로 경영 정보, 기업의 정보 자산, 데이터웨어하우스(data warehouse)와 데이터마트(data mart) 그리고 정보 자원 관리 어플리케이션들이 포함된다. EIP를 한마디로 "기업 내외에 산재한 정보를 단일 웹 인터페이스 (Web interface)를 통해 사용자에게 제공하는 시스템"으로 정의할 수 있으며, 기능적으로는 의사결정(decision processing) 지원 엔터프라이즈 포털과 협업(collaborative processing) 기반 엔터프라이즈 포털로 구분하기도 한다 [3,5,19,20].

의사결정 지원 엔터프라이즈 포털은 경영정보 툴과 분석도구를 이용하여 기업 내외부의 정형화된 데이터를 분석하여 경영 전략에 필요한 각종 리포트를 생성, 제공하는데 초점을 둔다. 이에 비해 협업기반 엔터프라이즈 포털은 그룹웨어나 워크플로우(workflow) 도구를 사용하여 상대적으로 비정형화된 정보를 조직화하고 공유할 수 있도록 지원하는 것이다. 일반적으로 EIP에서는 이러한 구분이 명확하지 않으며, 의사결정 지원 엔터프라이즈 포털과 협업기반 엔터프라이즈 포털이 서로 밀접하게 연관되어 있기 때문에 점차 하나의 EIP 시스템으로 통합되고 있다.

이상과 같은 내용들을 종합하여 EIP 시스템의 중심 개념을 (그림 6)에 요약하였다. EIP 시스템은 정형 또는 비정형 정보 자원의 수집과 통합, 정보 자원의 검색 및 탐색 등의 다양한 정보 자원 관리 기능, 사용자 중심의 개인화 포털, 사용자간의 정보 자원의 공유와 협업 등을 지원하는 기업의 정보 자원과 응용을 통합한 시스템이라 할 수 있다.



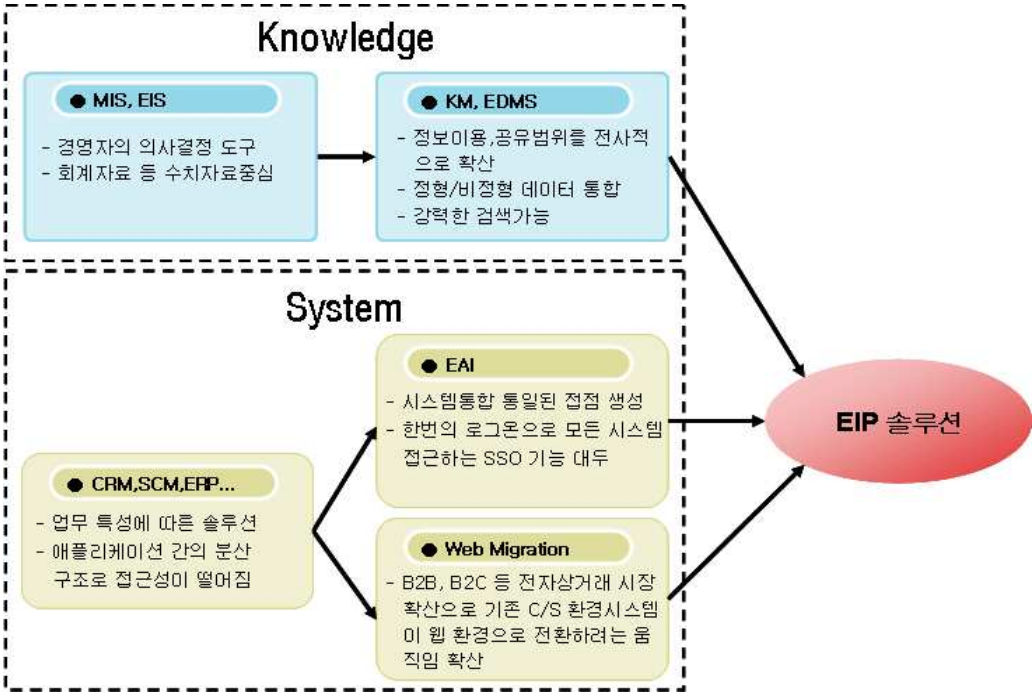
(그림 6) 엔터프라이즈 정보 포털(EIP) 시스템의 개념

2. 포털 시스템의 종류

인터넷 포털은 초창기에 단순히 검색 서비스만을 제공하였으나, 메일, 쇼핑, 뉴스, 커뮤니티 등의 다양한 기능이 추가되어진 메가 포털(mega portal), 인터넷 포털(Internet portal) 또는 웹 포털(Web portal)등의 형태로 발전하게 되었다. 인터넷 포털의 대표적인 예로써 Yahoo, MSN, AOL, 네이버 등이 있다. 인터넷 포털은 사용자들에게 인터넷 공간으로 들어가는 중요한 접근점으로 인식되고 있으며, 아직도 크게 각광을 받고 있다.

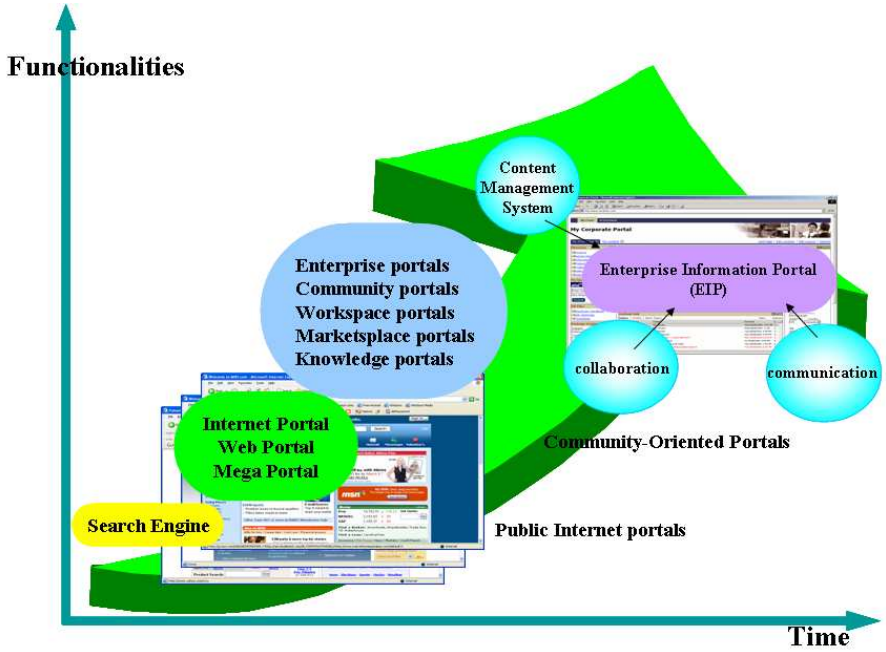
포털 시스템들의 개념은 다양한 정보 자원과 애플리케이션들이 산재해 있는 기업과 기관으로 확장되어 조직의 정보 자원 관리와 응용을 위한 기반 시스템으로 발전하게 되었다. EP시스템(Enterprise Portal System)은 기업 내외부에 산재되어 있는 업무 관련 정보를 효율적으로 통합하는 기업 정보 시스템인 EIS와 업무 처리 애플리케이션을 통합하여 관리할 수 있는 기업 애플리케이션 통합의 EAI의 형태로

발전하였으며, 급변하는 정보 환경에 대응할 수 EIP 형태로 진화되었다. (그림 7)은 EIP 시스템의 기능 변천 과정을 보여 주고 있다.



(그림 7) EIP 기능의 발전

포털 시스템은 인터넷과 같은 대규모 컴퓨터 네트워크가 급속하게 보급되고 웹 기술이 비약적으로 진보함에 따라 필연적으로 나타난 현상이다. 컴퓨터 네트워크와 웹기반 분산 처리 등 기반 기술의 지원에 따라, 정보의 공유와 유통을 혁신하여 기업의 정보 자산 관리의 효율성을 제고할 수 있게 되었고, 사용자간의 다양한 통신 형태를 개발하여 협업 등 현실 세계의 기업 활동을 가상 기업 공간까지 확장할 수 있게 되었다. 포털 시스템은 글로벌 네트워크 시대의 대표적인 정보 시스템의 형태라 할 것이다. 포털 시스템의 변천 과정을 (그림 8)에 도시하였다.



(그림 8) 포털 시스템의 발전 과정

포털 시스템은 형태와 기능에 따라 여러 형태로 분류할 수가 있다. 특히, EIP는 지원하는 서비스와 기능이 적용되는 비즈니스 상황에 따라서 B2B, B2C, B2E, B2G 등으로 다양하게 나타나기 때문에 그 형태를 일률적으로 분류하는 것은 쉽지 않다. 델파이 그룹(Delphi Group)에서는 정보 자원의 형태와 사용자 커뮤니티의 다양성 측면에서 다음 4가지 형태로 분류하고 있다.

n **Publishing Portal**

다양한 관심사를 가진 대규모의 사용자 커뮤니티를 대상으로 한 포털로서 온라인 검색(online search)과 정보 유통과 전달을 주목적으로 하는 전형적인 브로드캐스팅(broadcasting) 모델이다.

n **Commercial Portal**

많은 전문 사용자 커뮤니티에 특정 영역의 컨텐츠를 제공하는 포털로서 가장 일

반적인 형태의 온라인 커뮤니티 포털이다. 커스터마이즈된 사용자 인터페이스(user interface)를 제공하기는 하지만 광범위한 사용자를 대상으로 단순한 형태의 컨텐츠를 제공하는 모델이다. 이러한 형태의 포털은 하나의 화면에 웹정보를 모아서 제공하기 때문에 '채널(channel)'로 불리기도 한다.

n **Personal Portal**

특정 개인을 위해 특별히 정제된 정보를 제공하는 모델이다. Commercial Portal과 같이 상대적으로 좁은 영역의 컨텐츠를 제공하지만 특정 사용자를 위해 보다 더 개인화된 정보서비스를 제공한다.

n **Corporate Portal**

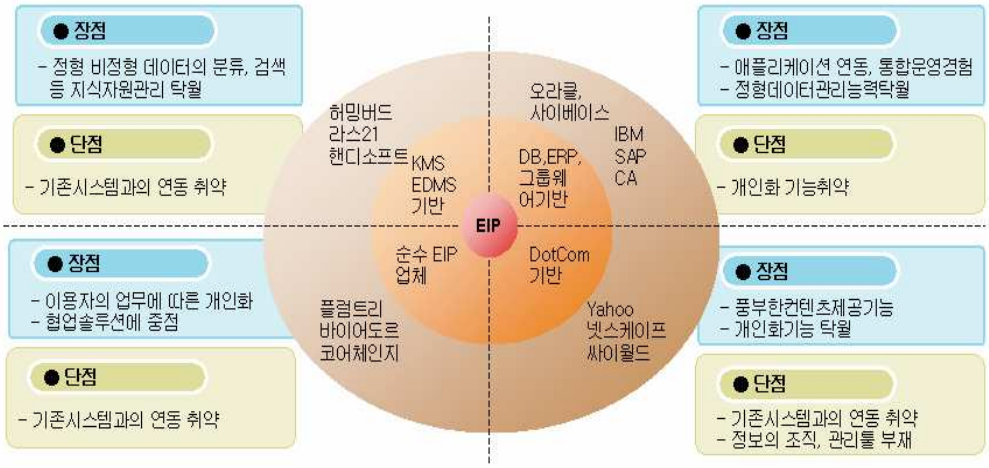
기업이나 기관의 사용자등 상대적으로 좁은 범위의 사용자에게 풍부한 콘텐츠를 제공하는 모델이다. Corporate Portal은 일반적으로 조직 단위의 인트라넷(intranet) 어플리케이션으로 구축되며, 조직의 운영 목적을 달성하는데 필요한 제반 기능을 제공한다.

포털 시스템에는 다양한 형태가 존재하지만, 분산된 정보 자원을 통합하여 혁신적인 정보 공유 및 유통 체계를 구축하여 사용자에게 단일화된 정보 접근점을 제공하고, 협업 등의 부가적인 비즈니스 서비스를 제공하는 기업 또는 기관의 게이트웨이 역할을 수행한다.

3. 상용 EIP 시스템 동향 분석

포털 시스템을 제공하거나 보유하고 있는 업체는 기업용 솔루션을 개발해온 업체들이 대부분이다. 기존 기업용 정보 시스템이 EIP로 진화함에 따라, 보유하고 있던 기업 정보 시스템의 기능에 EIP의 핵심 기능을 추가하여 확장하고 있는 추세이다. 따라서 업체 자신이 속한 업종 군에 따라 포털시스템이 추구하는 방향이나

개념이 다를 수밖에 없으며, 기존의 기업 정보 시스템에 따라 EIP 형태도 결정된다. 이러한 포털시스템은 업종에 따라 (그림 9)와 같이 구분할 수 있다.



(그림 9) 업종에 따른 EIP의 구분

n **기존의 e비즈니스 솔루션을 보유한 업체**

IBM, 오라클, 사이베이스 등이 여기에 해당하며, 해당 업체에 속한 포털시스템은 비즈니스 애플리케이션과의 연동, 시스템 통합, 정형 데이터의 관리능력을 앞세워 기업 내 애플리케이션간의 효율적인 통합을 실현하고 있다. 이러한 시스템은 개인화의 기능이나, 지식 관리 영역 측면의 기능이 미약한 단점이 있다. 해당 솔루션을 보유한 업체에서는 EIP의 핵심을 B2B, B2C, B2E등의 영역에서 비즈니스 데이터 트랜잭션에 초점을 맞추어 개념을 설명하고 있다[2].

n **KM 기반의 EIP 솔루션을 보유한 업체**

허밍버드, 라스21등을 들 수 있으며, 정보 공유와 이를 통한 협업 체계 구축을 EIP의 목표로 하고 있다. 이들 업체의 솔루션은 기업 내의 정보 자원을 디지털화하여 저장, 검색하는 기능은 뛰어나지만 각 애플리케이션간의 데이터 연동이 취약한 단점이 있다[2].

n **순수 EIP 엔진 기반의 EIP 솔루션을 보유한 업체**

현재 시장 점유율 1, 2위를 지키고 있는 플럼트리(Plumtree) 와 바이어도르(Viador)가 대표적이며, 개개의 업무 정의에 맞게 포털을 개인화하는데 강점을 지니고 있다. 이들 포털 시스템은 기존 e-비즈니스 솔루션을 보유한 업체에 비하여 비즈니스 애플리케이션의 통합과 관련한 기술이 미약하지만, 순수 EIP 엔진의 관점에서 기업의 정보 자원을 개개인의 업무 특성에 맞도록 통합 관리해 준다는 점에 강점이 있다[2,3,5].

n **순수 닷컴 엔진 기반의 EIP 솔루션을 보유한 업체**

이들 포털 시스템은 EIP 개념이 존재하기 전부터 운영해 온 인터넷 포털의 기술을 기반으로 뛰어난 개인화 서비스 운영 경험과 풍부한 컨텐츠에 강점이 있다. 이들 포털 시스템의 경우, 기존의 SI 업체와 연계하여 포털 시스템을 구축하는 전략을 세우고 있다[2,3,5].

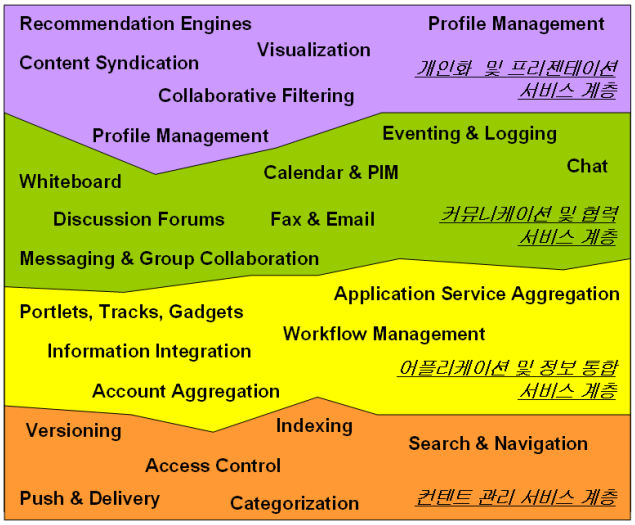
제 2 절 기존 포털 시스템의 기능 분석

포털 시스템의 기능을 효과적으로 분석하기 위하여 수평 포털(horizontal portal)과 수직 포털(vertical portal)로 나누어 고찰한다. 수평 포털은 포털 시스템의 인프라에 해당하는 기능을 제공하고, 수직 포털은 응용 영역별로 특화된 서비스 기능을 제공한다.

1. 수직 포털의 기능 분석

수직 포털은 수평 포털 기반위에 구축하려는 응용 영역에서 포털 시스템의 성격 및 목적에 따라 요구되는 특화된 시스템 또는 서비스들, 예를 들면, 정보검색 서

비스, 쇼핑 서비스, 이메일 서비스 등의 기능을 주로 지원한다. 수직 포털에서 주로 제공하는 기능을 요약 정리하여 계층적으로 도시한 것이 (그림 10)이다 [5,19,20].



(그림 10) 수직 서비스 계층 구조

최하위 계층은 컨텐츠 관리, 즉 정보 자산 또는 지식 자산을 효율적으로 관리하기 위한 서비스를 제공하는 컨텐츠 서비스 계층(content service layer)이다. 이 계층에서는 정보의 분류(categorization), 정보에 대한 접근 제어기능(access control), 인덱싱(indexing) 기능, 정보 검색 및 탐색기능, 버전 관리 기능 등이 제공된다. 컨텐츠 서비스 계층을 기반으로 사용자들이 서로 협력하여 업무를 처리하고 연관 정보를 효율적으로 공유할 수 있다. 또한 기업 내 다양한 정보 자산과 어플리케이션들을 통합하는 통합 서비스 계층(aggregation service layer)의 구현이 가능해 진다. 서비스 통합을 확고히 제공해 주기위해, Single Sign-On (SSO), 포틀릿(portlet) 관련 지원, 데이터 통합 기능 등이 제공되고 있다.

포털 사용자들 간의 원활한 의사소통과 통신, 협업을 위한 각종 정보 서비스 기능을 제공하는 협력 서비스 계층(collaboration service layer)이 필요해 진다.

이 계층에는 일정관리, 이메일 및 팩스 서비스, 채팅 및 메신저 서비스, 토의 및 포럼 기능 등이 제공된다. 수직 포털의 최상위에는 통합된 서비스나 정보 자원 중에서 사용자가 필요한 개체들만 선택적으로 선정, 조합할 수 있는 개인화 서비스 계층(personalization service layer)이 구현되고 있다. 개인화 서비스 제공하기 위해 사용자의 프로필(profile) 관리, 사용자가 추출한 서비스의 필터링(filtering), 추천 엔진(recommendation engines), 비즈니스 규칙 엔진(rules engines)등이 필요하며, 정보 자원의 다양한 시각화(visualization) 기능을 지원하는 사용자 인터페이스가 요구된다.

2. 수평 포털의 기능 분석

수평 포털은 포털시스템의 기반이 되는 인프라 시스템을 의미하며, 일반적으로 다음 6가지의 기본 기능을 지원하여야 한다[5,9,19,20].

n **집합 기능(aggregation)**

기업의 내외부에 산재되어 있는 정보원으로부터 애플리케이션, 컨텐츠, 서비스 그리고 전자상거래 등의 비즈니스 프로세스 등 다양한 형태의 정보 자원을 수집하여 포털 시스템에서 일괄적으로 지원하여야 한다. 기업 포털 시스템은 이질적인 정보 자산들을 결합할 수 있는 표준화된 컨테이너(container)를 제공하여야 한다.

n **통합 기능(integration)**

공간적으로 분산되어 있거나, 이질적인 구조를 갖고 있는 다양한 정보 자원을 효과적으로 통합할 수 있는 기능을 제공하여야 한다. 이질적인 정보 자원을 통합하기 위하여, 표준화된 정보 구조를 설정하고 랩퍼(wrapper) 또는 중재자(mediator)등의 미들웨어를 이용하거나 표준 메시지 프로토콜 등을 응용한다.
통합은 정형, 비정형 정보 자원뿐만 아니라, 비즈니스 프로세스까지 그 대상으로 하고 있다.

n **표현 기능(presentation)**

도표나 그래프등과 같은 정보 시각화 도구를 활용하여, 포털 시스템내의 정보 자원을 사용자가 원하는 다양한 형태로 표현하여 쉽게 이해할 수 있어야 한다. 포털 시스템에서는 XSLT 처리 기능이나 다양한 시각화 기능이 제공되어야 한다.

n **접근성 보장 기능(access)**

다양한 통신 매체가 보편화됨에 따라, 포털 시스템은 정보는 인트라넷 또는 인터넷 이외에도 여러 채널을 통하여 접근할 수 있는 길이 열렸다. 즉, 핸드폰, PDA, 전화등과 같은 다양한 통신 매체를 통하여 포털 시스템의 정보 자원에 접근할 수 있는 다양한 접근 방식을 보장해야 한다. 향후 유비쿼터스 서비스 제공에서도 포털 시스템의 역할을 할 수 있도록 하여야 한다. 또한, 접근성 보장은 보안 및 보호 문제와 밀접한 관계가 있으며, 사용자 관리와 SSO 기능도 접근성 보장에 포함된다.

n **개인화 기능(personalization)**

개인화 기능은 포털 시스템의 가용성을 제고해 주는 핵심 기능이다. 포털 사용자는 필요로 하는 정보 자원과 기능만 포털 시스템을 재구성하여 정보 관리의 효율성을 극대화 할 수 있다. 개인화 기능은 사용자 프로파일에 의한 사용자 관리와 접근성 보장 기능등과 상호 관계가 있다.

n **시스템 관리 기능(administration)**

포털 관리자는 포털 시스템의 보장을 위한 최적화, 통신망 및 저장 장치 등 하드웨어 요소의 최적 운영, 정보 자산과 서비스 저장고 운영, 사용자의 권한 설정 및 적합한 자원 할당, 보안 등급 설정 등 포털 시스템의 효율적인 관리를 위한 제반 도구를 제공하여야 한다. 관리 도구에서는 실시간 모니터링, 사용자 친화적인 인터페이스 등이 제공되어야 효율적으로 관리 목표를 달성할 수 있다.

제 3 절 EIP 시스템의 기능 분석

전술한 포털 시스템의 기능 분석을 토대로 하여 EIP 시스템이 제공하여야 할 핵심 기능을 요약 분석한다. EIP 시스템의 핵심 기능은 시스템 아키텍처 개발에 필요한 컴포넌트를 지정하기 때문에, 시스템 구축의 요구 사항 분석 단계에 해당한다. 본 논문에서는 이러한 핵심 기능을 구현하는 EIP 시스템 설계, 구현한다.

■ 개인화 기능(personalization)

개인화 기능은 EIP 시스템이 유용성(usability)를 제고하는 핵심 요소 중 하나이다. 사용자는 서비스 저장고에서 제공되는 다양한 서비스를 drag-and-drop 형식으로 조합하여 자신에 적합한 시스템을 구성할 수 있어야 한다. EIP 포털 시스템은 서비스 컴포넌트를 자유자재로 합성할 수 있는 기능을 제공하여 한다.

■ 협업 기능(collaboration)

협업은 개인 또는 조직의 업무 효율의 증진시킬 수 있는 유용한 도구이다. EIP 시스템은 필수적으로 협업 기능을 제공하여야 하며, 이를 위해서는 정보 공유, 비즈니스 프로세스의 통합 등 협업 환경을 지원하여야 한다.

■ 검색 기능(search)

EIP 시스템에서는 정형/비정형 문서와 컨텐츠, 서비스, 비즈니스 프로세스, 애플리케이션 등 다양한 정보 자원의 생성, 공유, 배포 등을 지원하므로, 이를 효율적으로 관리할 수 있어야 한다. 이러한 정보 자산 관리의 핵심 기능이 검색이라 할 수 있다. 단순 키워드 검색에서부터 의미기반 검색에 이르기까지 다양한 수준의 검색 방법이 지원될 필요가 있다. 효율적인 검색을 위해 정보 자원의 저장 구조, 인덱싱 기법 등이 해결되어야 한다.

■ 통합 관리 기능(system integration)

기업의 내외부 이질적인 정보 자원의 통합은 EIP 시스템이 당면한 중요 과제이다. EIP 시스템은 형태적 이질성뿐만 아니라 의미적 이질성까지의 통합을 요구하고 있다. 현재의 대부분의 EIP 시스템은 형태적 이질성 통합에 초점을 두고 있으며, 향후 온토로지(ontology)등 의미 기반 처리가 활성화 되면 의미적 이질성 통합도 가능할 것이다.

본 논문에서는 정보 자원의 형태적 이질성을 통합에 목표를 두고 있으며, 온토로지를 응용한 의미적 통합 방안을 간단한 사례를 통하여 제시한다.

■ **시스템 보안 및 관리 기능(system Security & management)**

사용자의 수준과 정보 자원의 중요도에 따른 보안 등급을 설정하여, 사용자와 정보 자원의 활용 상황을 실시간으로 모니터링하고 관리하여야 한다. 시스템의 보안과 관리 기능은 포털 시스템의 성능과 사용자의 업무 생산성을 저하시키지 않고 정보 자원을 확실하게 보호하고 관리할 수 있는 기능을 제공해야 한다.

■ **출판 및 분리 기능(publishing & distribution)**

정보 자원의 생성과 배포를 효율적으로 지원하여야 한다. 정보 기술에 대한 전문 지식이 없는 사용자도 컨텐츠를 쉽게 가공 제작 및 관리 할 수 있어야 하며, 효과적인 컨텐츠의 분배 및 출판 등의 기능을 제공해야 한다.

■ **커뮤니티 기능(community)**

사용자들을 업무 단위 또는 특정 목적 단위로 커뮤니티를 형성하여 관리하는 기능이 재공 되어야 한다. 커뮤니티 또는 포럼의 운영에 필요한 전자 게시판, 메일, 채팅, 블로그 등의 커뮤니티 관련 기능이 커뮤니티 생성과 동시에 포털 시스템에서 제공해야 한다.

■ **SSO 기능(Single Sign On)**

SSO는 보안과 사용자의 편의성을 위한 적극적인 방책이다. 또한 포털 시스템이

기업 또는 기관의 정보 자원을 접근하는 단일 게이트웨이 역할을 할 수 있도록 하여 준다. SSO는 한 번의 로그인만으로서 포털 시스템 전체 정보 자원의 사용에 대한 인증을 획득할 수 있다.

위에서 논한 EIP 시스템의 핵심 기능을 분석하여 보면, 다음과 같이 4가지 영역으로 구분할 수 있다. 위에서 서술한 핵심 기능 외에, 4가지 영역별로 영역의 목적에 적합한 새로운 기능을 도출하는데 도움이 된다. 또한, EIP 시스템 아키텍처를 서브시스템 단위로 구성하는데도 도움이 된다.

n **프리젠테이션 서비스(presentation service) 영역**

EIP 시스템에서 제공되는 각종 서비스 기능들을 활용하여 사용자의 요구에 적합한 정보를 제공하는 기능을 담당하는 영역이다.

n **이용자 서비스(user service) 영역**

EIP 시스템의 각 서비스 모듈 및 응용 소프트웨어가 운용되는 영역으로 실질적으로 서비스가 실행되는 영역이다. 일반적으로 EIP 시스템의 서비스 모듈은 포틀릿 형태로 구현된다.

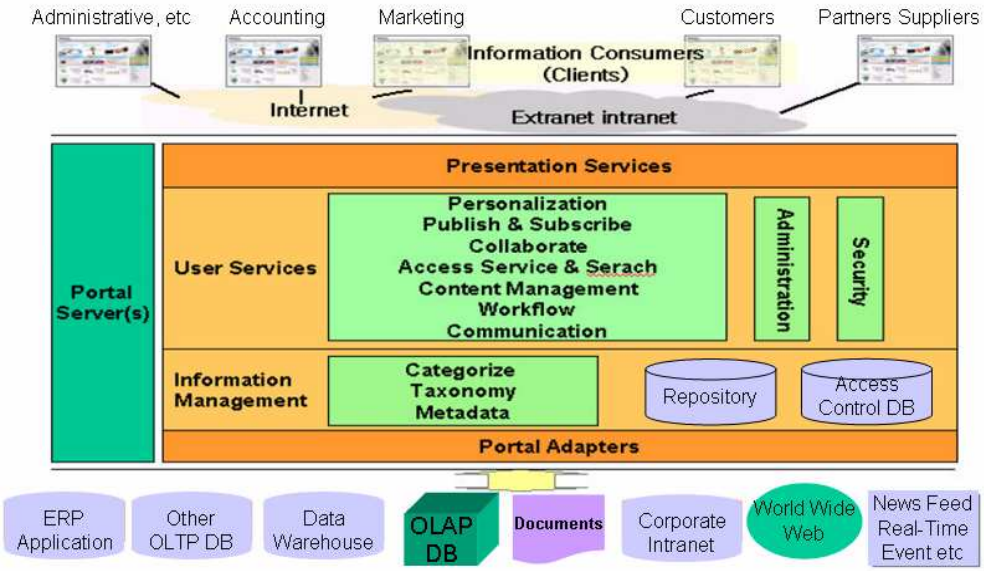
n **정보관리(information management) 영역**

EIP 시스템의 정보 자산, 사용자, 보안 정책, 시스템 구성 요소 등 제반 요소를 관리하는 영역이다. EIP 시스템의 성능과 밀접한 관계가 있다.

n **포털 어댑터(portal adapter) 영역**

기존의 데이터베이스 및 응용 어플리케이션과의 연결을 담당하는 중재자 역할을 하는 미들웨어 영역이다. 포털 어댑터는 기존 시스템을 랩핑 또는 캡슐화하여 EIP 시스템과 독립적으로 운영될 수 있도록 하여야 한다.

이상에서 분석한 EIP 시스템의 핵심 기능과 서비스 영역을 요약 정리하면 (그림 11)로 도시할 수 있다. (그림 11)는 포털 시스템을 설계 및 구현하는데 개념 설계도로 활용한다.



(그림 11) 포털 시스템의 구조

제 4 절 포틀릿의 개념과 기능

포털 시스템을 구성하는 기본 단위는 포틀릿(portlet)이다. 포털 시스템은 포틀릿간의 상호 메시지 교환으로 동작하게 되며, 포털 시스템을 위한 메시지 버스, 컨테이너, 중재자 등의 기반 인프라를 제공하여야 한다.

포틀릿을 비롯하여 EIP 시스템의 요소 기술에 대하여 분석하고 또한 이기종 또는 포털 벤더간의 상호 호환 운영을 위하여 분산 시스템의 기본 구성단위로 인식되고, 차세대 시스템 구축 기술인 SOA(service-oriented architecture)의 요소 기술로 인정받고 있는 웹서비스를 포틀릿으로 구현함으로써 필요한 포틀릿과 웹서비스의 상호 운용 구조에 대해서 서술한다.

앞에서 살펴본 바와 같이 EIP의 경우 다양한 서비스로 이루어지게 되며, 독립적으로 서비스만을 전문적으로 개발하는 업체가 등장하게 되었다. 이러한 업체들은 EIP 업체가 요구하는 규격에 맞추어 애플리케이션 서비스를 개발하여 배포하게 됨에 따라 애플리케이션 서비스간의 호환성 문제가 야기됨에 따라 어플리케이션 서비스 개발자, EIP 시스템 업체, 포털 사용자간에 심각한 문제가 제기되었다, 이러한 문제점을 극복하기 위한 방안으로 자바 포틀릿 API (Java Portlet API) 표준이 제안되었다. 이 표준은 포털 시스템과 사용자가 시각적인 인터페이스를 통해 포틀릿 및 포털 시스템간의 상호 운용성을 제공할 목적으로 개발되었다. 또한 자바 포틀릿 API는 어플리케이션 서비스 개발자나, 포털 소비자들이 특정 EIP 시스템에 구애받지 않고, 포틀릿을 조합, 합성할 수 있는 방법을 제공하고 있다[1,6,14,15].

포틀릿은 사용자가 요청하는 컨텐츠를 동적으로 생성하고 제공할 수 있도록 구현된 자바 스펙의 한 부분이며, 포틀릿은 포틀릿 컨테이너(Portlet container)에 의해 관리되어지는 일종의 웹 컴포넌트라고 볼 수 있다. 포틀릿은 이전의 자바에서 사용된 서블릿(servlet)과 같은 웹 컴포넌트이지만, 포털 시스템과 같은 웹 어플리케이션에 삽입되어(Plug-in) 실행될 수 있는 특별한 구조 가짐으로써 서블릿과 다른 포탈에 최적화된 특징을 보유하고 있다.

포틀릿은 웹 어플리케이션에서 보유하고 있는 다양하고 복잡한 컨텍스트(Context)에 따라 호출된 포틀릿에 해당되는 포틀릿 인스턴스를 생성하여 제공하도록 설계되어졌다. 예를 들면, 사용자의 신상정보를 제공하는 포틀릿에 대하여, 사용자별로 활용하는 포틀릿은 신상정보 포틀릿을 활용하게 될 것이다. 그러나 사용자가 요청(request)하는 컨텍스트(Context)에 따른 제공되는 컨텐트는 사용자별로 틀릴 것이다. 이때 포틀릿은 사용자별 요청에 따라 포틀릿을 생성하는 구조가 아니라 포틀릿의 컨텐트에 따른 인스턴스를 생성하여 제공하도록 구성됨으로써 서블릿과의 성능의 이점을 제공하고 있다.

포틀릿에 의해 생성되어 제공되는 모든 내용(Content)은 Markup 언어(HTML,

XHTML, XML)로 구성되게 되며, 이를 마크업 조각(Makeup Fragment)이라고 칭한다. 이러한 포틀릿을 구현할 때 사용되는 프로그래밍 패턴은 모델-뷰-컨트롤러 패턴(Model-View- Controller pattern)에 따라 구현될 수 있으며, 이를 구현하기 위한 자바 기술로는 모델과 컨트롤러의 구현은 자바클래스로 구현하며, 뷰를 구성하는 기술에는 JSP, XSLT 등을 활용한 템플릿의 형태로 구성된다[1,14,15].

포틀릿의 기능적인 구성은 다음 기능을 기본적으로 보유한다.

(1) 사용자 프로파일 정보에 대한 접근.

(2) 포틀릿 컨텐트의 뷰어(Viewer).

(3) 포틀릿 윈도우의 상태제어(최소화, 최대화, 포틀릿 위치변경 등).

(4) 동적인 컨텐트를 구성하기 위한 데이터 및 메시지의 접근 기능[1,14].

이러한 포틀릿을 구현하기 위해서 본 논문에서는 Java Portlet Specification Version 1.0에 의거하여 EIP 시스템의 포틀릿을 설계하고 구현할 것이다. 앞에서 언급했듯이 포틀릿은 포틀릿 컨테이너에 의해 관리되어지며, 포틀릿은 자바기반의 웹 컴포넌트를 의미한다. 포틀릿 컨테이너는 런타임 환경에서 요청이 있을 때 포틀릿과 프로바이드를 실행하게 된다[1,9,14,15].

(1) Java Portlet Specification, Java2 Platform, Enterprise Edition의 연관성

포틀릿 API 1.0은 J2EE 1.3에 기반한다. 포틀릿 컨테이너는 J2EE 스펙에 맞는 실행 환경을 지원해야 한다.

(2) Java Portlet Specification과 Servlet Specification의 연관성

포틀릿은 서블릿의 개념, 이름과 행동의 유사한 기능성 때문에, Servlet Specification 2.3에 정의와 비슷하다고 볼 수 있다. 그러나 관점이나 기능적으로 제공해야 하는 기능 및 제한되어져야 하는 기능으로 인하여 서블릿과 구분된 별도의 구성요소로 구분하여 정의하게 되었으며, 따라서 포틀릿 API는 서블릿 API와 유사한 기능이 다수 존재한다.

<표 1> 포틀릿과 서블릿의 차이점

비교사항	내 용
유사점	▪ Portlets는 웹 컴포넌트를 구현하는 자바 기술. ▪ Portlets는 특수화된 컨테이너에 의해 관리. ▪ Portlets는 동적인 내용을 생성. ▪ Portlets lifecycle는 컨테이너에 의해 관리. ▪ Portlets는 request/response 패러다임을 통해 웹 이용자와 상호작용.
차이점	▪ 포틀릿은 마크업 조각(Makeup Fragment)으로 구성. ▪ 포털이 포틀릿 마크업 조각의 집합에 따라 하나의 포털 페이지로 구성. ▪ 포틀릿은 URL에 직접 바운드 되지 않음. ▪ 웹 클라이언트들은 포털 시스템을 통해 포틀릿과 상호작용. ▪ 포틀릿은 액션 요청과 요청들의 랜더링과 같은 더 많은 재정의된 요청 핸들링을 가지고 있음. ▪ 포틀릿은 포틀릿이 수행되는 기능과 포털 페이지 안에서 실제 상황의 양을 나타내는 미리 정의된 portlet mode들과 window state 들을 가지고 있음. ▪ 포틀릿은 하나의 포털페이지에 여러 번 존재 가능. (인스턴스로 생성됨)
독창성	▪ 지속적인 환경설정과 맞춤화된 데이터의 접근과 저장에 대한 도구들을 가지고 있음. ▪ 포틀릿은 사용자 프로파일 정보에 접근 가능. ▪ 포틀릿은 컨텐츠 안에서 하이퍼링크를 생성하기 위한 URL Rewrite 기능을 가지고 있다. 이것은 포털 서버로 하여금 페이지 조각들에서의 액션들과 agnostic 링크생성을 가능하게 해줌. ▪ 포틀릿은 두 가지 다른 범위에서 포틀릿 세션 안에 일시적인 데이터 저장 가능
제한기능	▪ response의 character set encoding 설정. ▪ response의 HTTP header 설정. ▪ 포털에서 클라이언트의 request URL.

제 5 절 WSRP의 구조와 동작

포틀릿을 구성하는 포틀릿 API는 포탈 내부의 자원에 대한 표준을 정의함으로써, 다양한 형태의 포틀릿에 대한 표준화를 이루었다. 수많은 엔터프라이즈 포탈이 증가하게 되면서, 이러한 포탈 내부의 표준에 대한 방식은 정의되고, 표준화되어 서로 다른 벤더의 제품과 호환되는 장점을 제공하게 되었다. 시간이 지남에 따라 컨텐츠 측면에서의 통일화가 필요로 하게 되었다. 즉, 내부의 자원의 표준화에서 외부 자원의 표준화와 호환, 재사용을 제공하기 위하여 제정된 규정이 WSRP이다. WSRP는 특정 서비스를 보유하지 않는 포털 사이트들이 해당 서비스를 이용할 수 있는 방법을 제시한다. WSRP는 포탈 또는 응용프로그램 소유자들이 제 3자로부터 가져온 웹서비스를 포탈 페이지, 즉 포틀릿의 일부로서 쉽게 포함할 수 있도록 해주며, 포틀릿은 웹서비스 제공 측의 서버에서 동적으로 갱신된 대화식 콘텐츠와 서비스를 표시한다[8,11,12,15,22].

이에 본 절에서는 EIP 시스템에서 필요한 WSRP의 정의 및 기능을 기술하고, 이를 기반으로 구현한다.

1. WSRP의 정의

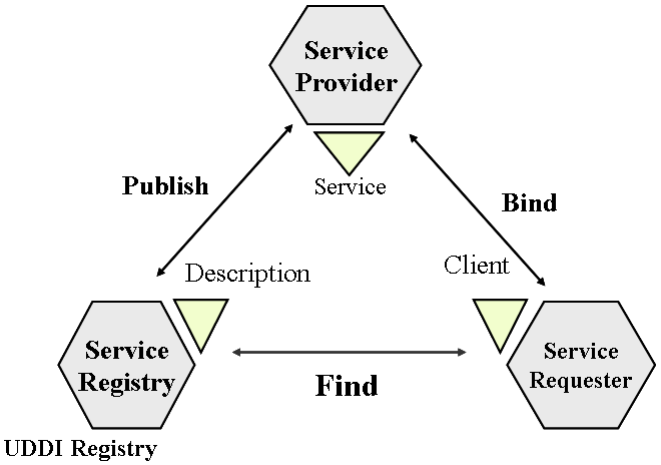
WSRP(Web Services for Remote Portlets)는 UDDI나 서비스 제공자 측에서 제공하는 웹서비스를 포탈에 장착할 필요가 있을 때 해당 서비스를 포틀릿으로 구현한 것을 의미한다. 일반적으로 웹 서비스를 이용하는 절차는 다음의 과정에 의하여 활용되어진다.

- 웹서비스의 등록 : 서비스 제공자는 공개할 서비스들을 UDDI 레지스트리에 등록한다. UDDI 레지스트리는 웹 서비스에 대한 정보와 이 서비스에 대한 추가 정보의 저장소를 의미한다.

n 서비스 요청자의 웹서비스 검색 : 클라이언트가 자신이 이용하고자 하는 웹 서비스를 UDDI에서 검색하면, UDDI는 검색된 웹 서비스의 위치 정보를 클라이언트에 반환한다.

n WSDL 파일의 참조 : 클라이언트는 반환된 서비스의 위치정보를 참조하여 WSDL 파일을 요청한다. 이 요청에 대해 서비스 제공자는 서비스의 상세 정보가 기술되어 있는 WSDL 파일을 서비스 클라이언트에게 반환한다.

n SOAP 메시징을 통한 서비스의 요청 : 이용하고자 하는 웹 서비스를 SOAP 메시징을 통해 요청하면, 서비스 제공자는 요청된 서비스를 처리한 후 결과를 SOAP 메시징을 통해 서비스 클라이언트로 반환한다.



(그림 12) 웹서비스 이용절차

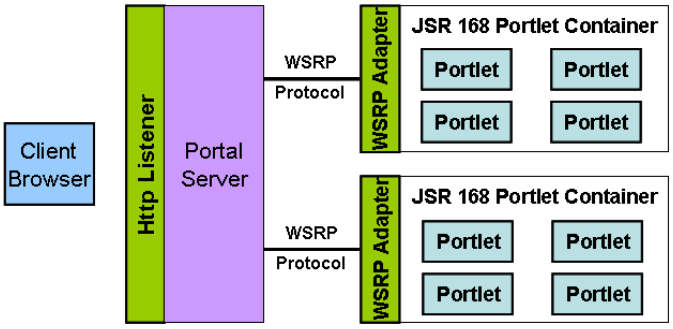
위와 같은 절차에 의해 활용되어지는 웹서비스를 본 논문에서 구현 하고자 하는 EIP 프레임 상에서는 두 가지 방향에서 정의하여 활용하고자 한다.

가. 서비스 요청자로의 EIP 프레임워크

UDDI에 등록되어있는 기존의 웹서비스를 EIP 프레임에 장착하여 서비스를 이용하는 서비스.

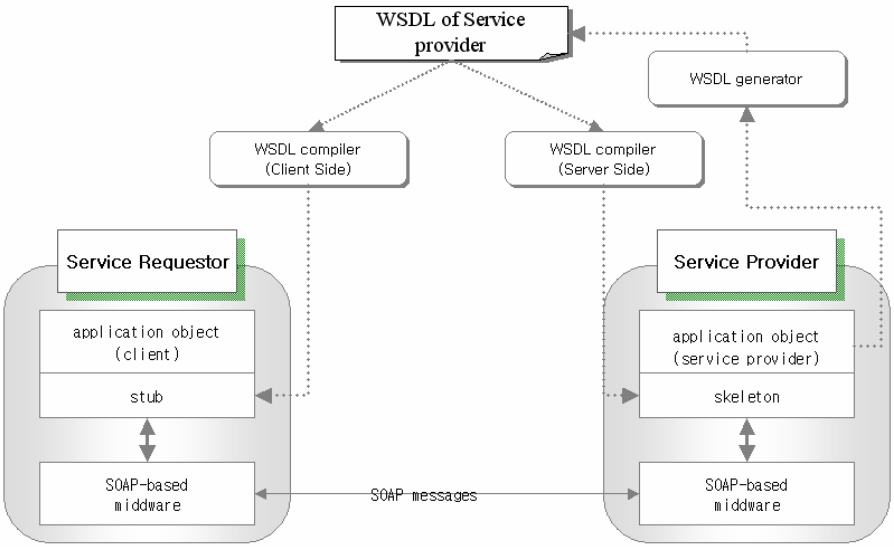
나. 서비스 제공자로의 EIP 프레임워크

EIP 프레임 상에 웹서비스가 서비스 제공자가 되어 해당 웹서비스를 타 시스템에서 활용 가능하도록 하는 서비스이다. 이와 같이 EIP 프레임워크에서는 두 가지 관점에서의 즉, 서비스 요청자와 서비스 제공자로의 역할을 수행하여야 하며, 그에 대한 포털서버와 웹서비스를 보유한 포틀릿간의 상호 운용성에 대한 그림은 다음과 같다.



(그림 13) JSR168과 WSRP의 상호 운용 방식

이러한 상호 운용 방식에 의거하여 웹서비스를 활용할 수 있도록 웹서비스의 내용을 기술하는 언어를 WSDL이라고 한다.



(그림 14) WSDL의 활용

WSDL은 제공하고자 하는 웹서비스의 문서 지향적 또는 프로시저 지향적인 정보를 포함한 메시지를 표시하는 정보의 집합으로서의 해당 웹서비스에 대한 URI에 대한 내용과 필요로 하는 정보를 메시지 형태로 구성하여 설명하는 XML 형식이다. 이러한 작동 및 메시지에 대해 대략적으로 설명한 다음, 구체적인 네트워크 프로토콜 및 메시지 형식으로 연결하게 되는 URI에 대해 정의한다. WSDL은 메시지를 받거나 보내는 순서 규칙과 같이 서비스의 동작을 표현하는 방법과 서비스를 작성하는 방법을 포함한다. 즉, 클라이언트에서 해당 서비스를 이용할 때 필요한 모든 정보를 표시하며, 이러한 정보는 실행 시에(Run-time) 바운드되게 된다.

WSDL은 웹 서비스를 기술하는 일종의 스크립트 언어이다. WSDL은 XML 포맷으로 구성되고 HTTP를 통해서 전달될 수 있으며 다르게 말한다면 인터페이스를 정의하는 IDL(Interface Definition Language)에 해당된다. 즉 이 서비스가 어떤 메소드, 어떤 속성을 가지며, 어떤 인수로 호출해야 하고 어떤 방식의 반환값을 제공하는지를 알려주는 것이다. 이 내용을 알게 되면 클라이언트에서는 알아낸 인터페이스 규약에 맞추어 호출하고 서비스를 사용할 수 있게 된다.

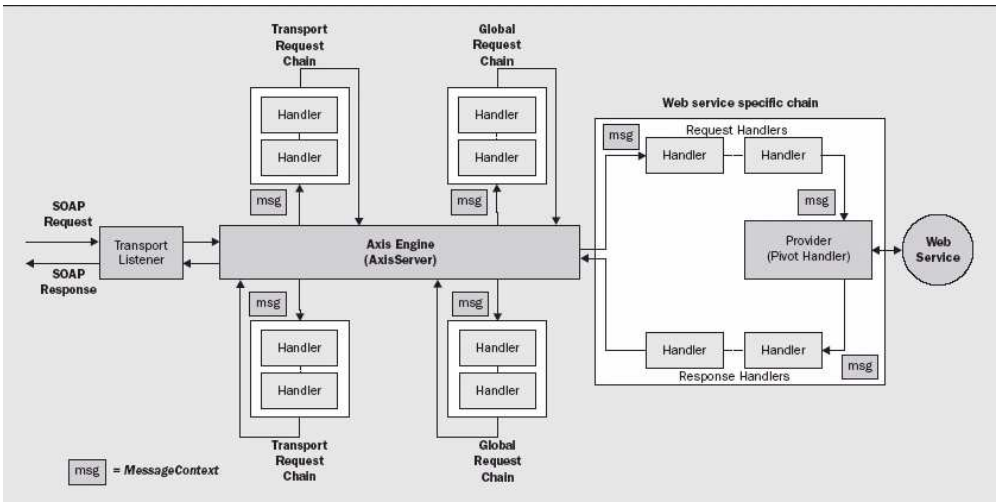
2. WSRP의 구현

본 논문에서는 이러한 WSDL 파일을 활용하고 응용할 수 있는 기반을 위하여 AXIS 엔진을 활용하였다. Apache Axis는 SOAP(Simple Object Access Protocol)의 오픈소스 구현이다. 기본적으로 SOAP는 컴퓨터들 간의 자료전송을 위한 표준 프로토콜을 제공한다. Axis는 아래 세 가지 스펙을 만족하도록 구현되어져 있다.

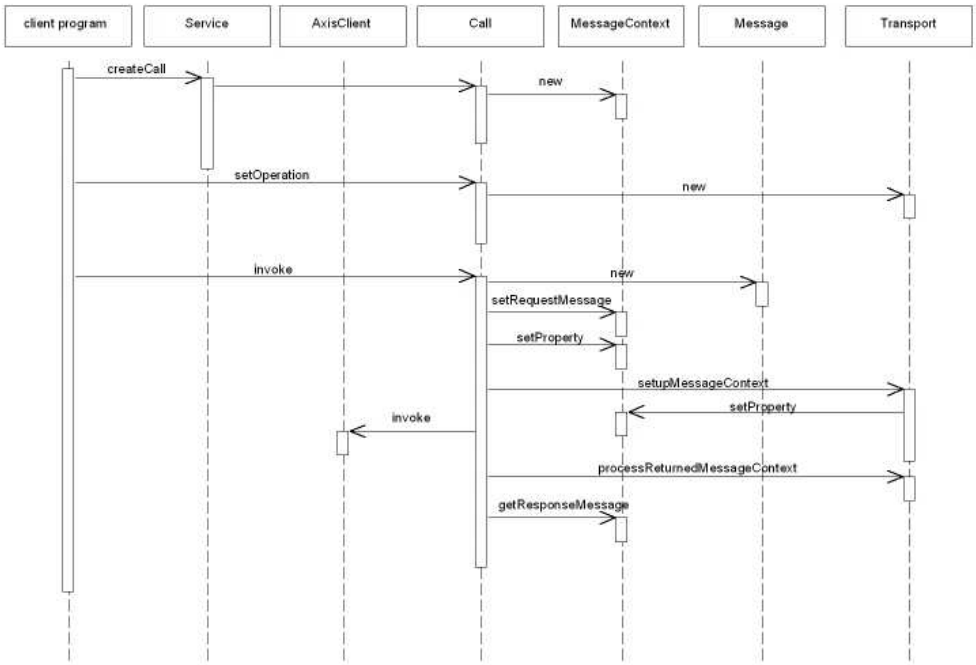
- SOAP (Simple Object Access Protocol)
- WSDL (Web Service Description Language)
- WSDD (Web Service Deployment Descriptor)

초기 웹서비스를 지원하는 엔진들은 SOAP 메시지가 도착하면 이를 단순히 BackEnd 객체에 연결을 해주는 브로커의 역할만을 진행하게 되었다[10]. 따라서 객체가 호출되기 전에 취급하여야 하는 전처리가 쉽지 않게 되었다. AXIS의 경우 이를 핸들러라는 것을 두어 해결함으로써 현재 존재하고 있는 Web Service 엔진 중에 최고의 기능을 보유하고 있다고 하여도 과언이 아닐 것이다. 이에 본 논문에서는 EIP의 웹서비스를 담당하는 컨테이너로 Axis Engine을 활용함으로써 웹서비스를 구현 적용할 수 있는 프레임을 구현하였다.

Axis엔진의 클라이언트와 메시지 처리방식 등의 상호 운용은 다음 그림과 같다.



(그림 15) Axis 엔진의 상호 운용 방식



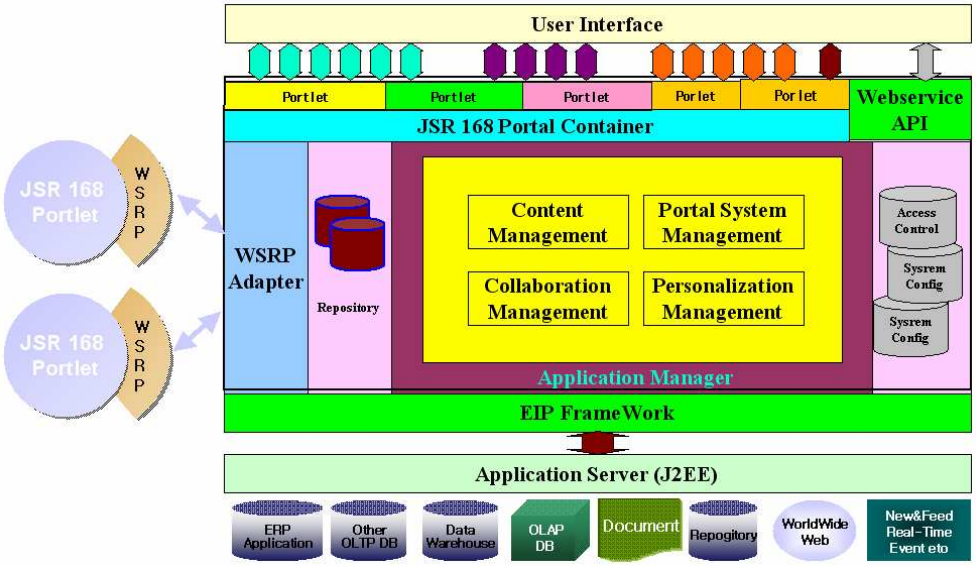
(그림 16) Axis 엔진에 대한 클라이언트 호출 Sequence Diagram

제 4 장 포털 프레임워크 아키텍처 설계 및 구현

본 장에서는 앞 장에서 논의한 EIP 시스템의 기본 기능과 요소 기술을 바탕으로 포털 프레임워크 아키텍처를 설계 구현한다. 본 논문에서 구현한 포털 프레임 아키텍처는 포틀릿과 웹서비스의 상호 연동이 가능한 시스템으로 공개 소프트웨어를 활용하여 구현한다.

제 1 절 웹서비스 기반 EIP 프레임워크 아키텍처

EIP 시스템의 핵심 기능과 요소 기술에 대하여 상세히 분석하였다. 본 논문에서는 분산 시스템의 단위 요소인 웹서비스와 상호 운용이 가능한 EIP 시스템을 구현하는데 목적이 있다. 일반적으로 EIP는 포틀릿을 중심으로 구축되기 때문에, 웹서비스와 상호 호환되는 EIP 시스템을 구축하기 위해서는 웹서비스와 포틀릿간의 상호 운용성을 보장하는 변화 서브시스템이 필요하다. 즉, 웹서비스를 포틀릿화하고, 포틀릿을 웹서비스화 하는 서브시스템이 요구된다. 변환서브시스템을 활용하면, EIP 시스템도 웹서비스만으로 구성된 분산 시스템의 형태로 활용할 수도 있고, 원격지 웹서비스를 지역 포틀릿으로 간주하여 EIP 시스템의 일부로 사용할 수 있다. 포틀릿과 웹서비스의 상호 운용성의 시스템의 성능 및 기능을 제고하는 중요한 요소이며, 서비스 컴포넌트의 재사용으로 시스템 구축의 효율성을 증대시킬 수 있다. 상호 변환 서브시스템을 갖은 EIP 시스템 프레임워크 아키텍처는 (그림 17)과 같다.



(그림 17) EIP 프레임워크 아키텍처

기업 내의 모든 애플리케이션은 J2EE 기반의 Application Server를 통하여 통합한다. 포털 프레임워크는 원격 포틀릿을 지원할 수 있도록 WSRP 어댑터를 장착하고 있으며, 이를 이용하여 원격지 포틀릿을 지역 포틀릿처럼 사용할 수 있다. 웹서비스와 연동이 가능하도록 웹서비스 API를 장착하였다. 웹서비스 API에서는 웹서비스와 포틀릿간의 상호 운용성을 제공하는 한다. EIP 시스템이 다른 시스템내의 웹서비스를 포틀릿 형태로 호출할 수 있고, 다른 시스템에서 EIP 시스템의 포틀릿을 웹서비스 형태로 호출할 수도 있다.

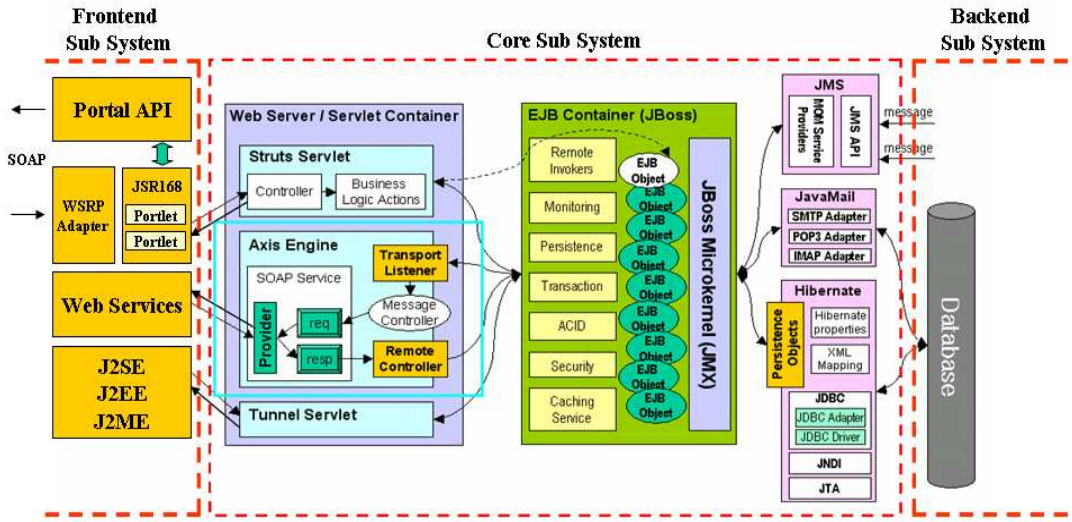
EIP 포털 시스템이 기본적으로 제공하여야 할 컨텐츠 관리(content management), 포털 시스템 관리(portal system management), 협업 관리(collaboration manag-ement), 개인화 관리(personalization management)등의 기능은 JSR168 표준 포틀릿 형태로 제공된다. 일반적으로 독립적인 포틀릿 개발 업체가 전문화된 표준 포틀릿을 제공하고 있기 때문에, 포틀릿 컴포넌트를 plug-and-play 형태로 활용할 수 있다. EIP 프레임워크를 활용하면 필요에 따라 해당 포틀릿 컴포넌트만을 추가로 탑재하면 되기 때문에 EIP 시스템의 기능을 용이하게 확장할 수 있다.

제 2 절 웹서비스 기반 EIP 프레임워크 구현 방안

본 논문에서 구현한 EIP 포털 프레임워크는 HTML/XML, JSP를 이용하여 구현되는 iframe 형식의 포틀릿, 웹서비스 형식의 포틀릿, J2EE, J2SE, J2ME를 이용하여 구현된 순수 Java 포틀릿 형식 등 세 가지 형식의 포틀릿을 지원한다.

단순히 웹 서비스만을 지원하는 것이 아니라 내부적으로 웹 베이스에서 작동되는 모든 기타 타 시스템과의 연동을 위한 아키텍처와 엔터프라이즈 환경을 위한 자바의 EJB 환경을 지원하고 처리하기 위한 EJB Container등을 포함한 총체적인 아키텍처를 기반으로 구성되어야 한다[8,13,22].

(그림 18)은 본 논문에서 구현한 웹서비스 기반 포털 프레임워크의 시스템 내부 구성도이다. 포틀릿과 웹서비스의 상호 연동을 지원하기 위하여 포틀릿 컨테이너, 웹 컨테이너, EJB 컨테이너가 포함되어 있으며, 기반 시스템 구현에는 공개 소프트웨어를 활용하여 구현하였다.



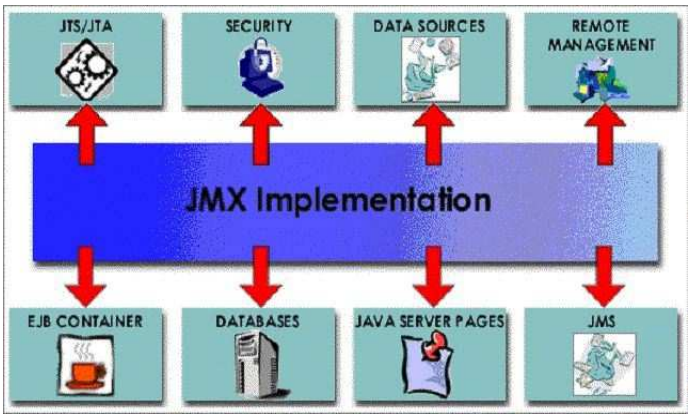
(그림 18) 웹 서비스 기반 EIP 프레임워크 구현 시스템

EIP 프레임워크 구현은 EIP시스템의 포탈의 핵심 기능을 의미하는 사용자 관리, 기타 Portlet을 구성하고 있는 비즈니스 로직을 포함하는 Beans, EJB Object를 보유하고 있는 EJB Container를 가지는 핵심 서브시스템(Core Sub System), EIP의 사용자 유저 인터페이스와 비즈니스 객체를 외부로 노출하는 기능을 하는 전단부 서브시스템(Frontend Sub System), 그리고 데이터의 저장소와의 연결을 위한 Persistence기능을 위한 후단부 서브시스템(Backend Sub System)의 3부분으로 나누어 구현한다.

1. 핵심 서브시스템

EIP 시스템은 기업이나 기관의 정보 자원을 접근하는 게이트웨이로서 통합된 비즈니스 서비스를 제공하기 때문에 각종 애플리케이션을 운용할 수 있는 애플리케이션 서버를 기반으로 하여야 한다. 많은 상용 애플리케이션 서버들이 개발되어 있지만, 대부분의 제품이 고가이고 업체 고유의 비표준적 특성을 내장하고 있다. 때문에 본 논문에서는 순수 자바로 구현되어진 J2EE기반의 애플리케이션 서버를 기반의 EIP 시스템을 구축하기로 하고, 대표적인 공개 소프트웨어 프로젝트로 진행되고 있는 JBOSS 3.0을 애플리케이션 서버로 선정하였다. JBOSS는 순수 자바로 구현된 표준과 호환 되는 엔터프라이즈 JavaBeans 애플리케이션 서버로써, 기본 EJB 컨테이너와 MX 인프라인 JBossServer, JMS 메시징을 위한 JBoss MQ, 전자우편을 위한 JBossMail, JTA/JTS트랜잭션을 위한 JBossTX, JAAS 기반의 보안을 위한 JBossSX, JCA 연결을 위한 JBossCX, CMP 지속을 위한 JBossCMP등을 제공한다.

본 논문에서는 여러 Enterprise Object를 효율적으로 관리하기 위하여 J2EE 스텍을 사용하기 위하여 JBOSS3.0의 JXM(Java Management Extension)을 채택하여 탑재하였다.



(그림 19) JBOSS3.0 JMX Implementation

2. 전단부 서브시스템

EIP의 사용자 유저 인터페이스와 비즈니스 객체를 외부로 노출하는 기능을 하는 전단부 서비스 시스템은 EIP 시스템에 Portlet으로 구현되는 여러 비즈니스 기능에 대한 사용자 유저 인터페이스와의 통신 역할을 담당한다.

가. Portal API

Portal API 는 포탈시스템을 이루는 자바 API 인터페이스들의 모음으로써 포탈의 주요기능인 권한관리 및 기타 시스템적인 측면의 기능을 라이브러리화 하여 제공한다. EIP 자체시스템을 구현한 인터페이스이다. 또한 JMS 시스템에서 사용되는 메시지를 주고받는 표준 인터페이스의 역할도 수행한다.

나. JSR168

Portal을 구성하는 단위인 포틀릿을 구현하기 위하여 자바 진영에서 제정한 포탈 표준을 준수하도록 구현한 JSR 168 표준스펙을 지원한다.

다. WSRP Adapter

여러 다른 Provider에서 제공하는 웹서비스를 활용하기 위한 WSRP Adapter를 제공하며, WSRP Adapter는 JSR168 표준에 의거한 Portlet의 형태로 변환되어 사용된다.

라. J2SE, J2EE, J2ME

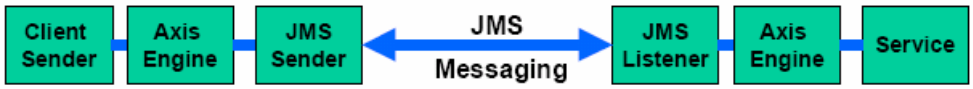
기존 자바 애플리케이션과 EIP 시스템과의 통합과 분산처리를 위한 자바 기술을 의미하며, 웹서비스를 제공하지 않는 시스템과의 통합을 위하여 Turnel Servlet을 통한 시스템의 통신을 제공한다.

3. 후단부 서브시스템

후단부 시스템은 데이터베이스와 엔터프라이즈 환경 상에서의 메시지 교환 등 영속적인 접촉이 필요한 데이터 커넥션, 데이터 구조, 메시지 구조 및 처리방법에 대한 사항에 대한 기술에 대한 내용이다.

가. JMS(Java Message Service)

엔터프라이즈 환경에서 기업이 내외부의 유관 기관과 메시지를 교환하는 아주 중요한 일이며, 메시지 서비스 시스템은 이러한 일을 총괄하게 된다. 여기서 메시지 서비스란 웹서비스 또는 포틀릿간의 표준 프로토콜을 활용한 데이터 통신을 의미한다. 본 논문에서는 대표적인 메시지 서비스 시스템인 JMS를 이용한다. 본 논문에서 JMS는 웹서비스 기반의 SOAP 메시지 처리 기능을 제공하게 된다. 즉, HTTP 상에서 Axis Engine을 통해 전송되어지는 메시지를 정규화하고 효율적으로 동기화할 수 있는 기반을 제공한다. (그림 20)에 JMS의 메시지 처리 기능을 간략하게 도시하였다.



(그림 20) HTTP 상에서 JMS SOAP 메시지

나. Hibernate

객체 모델링을 관계형 데이터 모델(relational data model)로 변환할 경우에는 객체와 데이터 간에 개념적 불일치가 존재한다. 예를 들면, 사용자의 프로파일을 정의하는 데이터모델의 경우 사용자의 별명이나, 관계형 데이터 모델의 경우 패스워드를 보관하는 시스템 사용에 대한 관계형 데이터모델과 사용자의 기본사항정보를 관리하는 관계형 모델 등 두개의 데이터 모델이 발생하게 되나, 객체 모델링의 경우 사용자 프로파일에 대한 단일 객체로 발생하게 되는 객체와 관계형 데이터 모델간의 불일치가 발생하게 된다. 또한 데이터베이스 벤더별로 처리되어지는 SQL의 종류가 틀려지게 되는 경우가 발생하게 되어 이를 해결하기위한 방안으로 Hibernate를 도입하여 처리하였다. Hibernate는 객체를 관계형 데이터베이스로 매핑하여 저장하는 ORM(Object-Relational Mapping) 도구로써 이를 활용하면 다음과 같은 이점이 있다.

- 관계형 데이터베이스 모델을 사용하는 데이터베이스의 벤더별로 insert, delete, update SQL 문을 객체 생성, 객체 저장, 객체 삭제 등으로 대체하도록 해 준다.
- 임의의 객체 검색을 위해서는 SQL과 유사한 Hibernate Query Language(HQL)을 사용한다.
- 이 과정이 모두 투명하다. 사용자가 바라보는 것은 java data class들과 xml mapping(*.hbm.xml)파일 뿐이다.
- Performance 향상을 위해 hbm.xml파일과 data class들을 가지고 CGLIB 라이브러리를 이용해 동적으로 Proxy 코드를 만들어 낸다. 그러므로 Reflection이 매번

일어나지 않는다.

- 일반 java class로 data구조를 구성하고, 이것에 대해 단지 hbm.xml을 만들어 줌으로써 persistency를 해결할 수 있다.
- hbm.xml을 DDL로 바꾸어 주는 Ant Task를 제공하기 때문에, 좀더 편리하게 초기 RDB 테이블을 만들 수 있다. 이 DDL에는 table create statement와 foreign key등의 constraint 관련 statement가 포함되어 있다.
- 각종 DB 제품에 따라 서로 다른 SQL과 Tuning 방법을 모두 반영한 SQL이 만들어 지기 때문에, 특정 RDB 제품에 대한 의존성(DB Product Lock-In)이 사라진다.

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD//EN"
"http://hibernate.sourceforge.net/hibernate-mapping-2.0.dtd">
<hibernate-mapping>
        <class name="org.objectworld.app.model.Menu" table="T_MENU" discriminator-value="Menu">
   <id name="id" type="long" column="MENU_ID" unsaved-value="0">
   <generator class="hilo"/>  </id>
   <discriminator column="SUBCLASS" type="string"/>
   <property name="displayName" column="DISPLAY_NAME" type="string"/>
   <many-to-one name="parent" column="PARENT_MENU_ID"/>
   <subclass name="org.objectworld.app.model.BranchMenu" discriminator-value      ="BranchMenu">
       <list name="childMenus" lazy="true" cascade="all">
           <key column="PARENT_MENU_ID"/>
           <index column="IDX"/>
           <one-to-many class="org.objectworld.app.model.Menu"/>
       </list>
     (....중략....)
               </subclass>
               <!-- org.objectworld.app.model.TerminalMenu -->
               <subclass                              name="org.objectworld.app.model.TerminalMenu"
discriminator-value="TerminalMenu">
                       <many-to-one            name="content"            column="TARGET_CONTENT_ID"
cascade="save-update"/>
               </subclass>
        </class>
        <!-- org.objectworld.app.model.Content root -->
        <class name="org.objectworld.app.model.Content" table="T_CONTENT">
               <id name="contentId" type="long" column="CONTENT_ID">
                       <generator class="hilo"/>
               </id>
```

```
                <property name="displayName" column="DISPLAY_NAME" type="string"/>
<one-to-one name="menuLink" class="org.objectworld.app.model.TerminalMenu" property-ref="content"/>
                <set name="variants" inverse="true" lazy="true">
                        <key column="CONTENT_VARIANT_ID"/>
                        <one-to-many
                          class="org.objectworld.app.model.ContentVariant"/>
                </set>
        </class>
        ...
</hibernate-mapping>
```

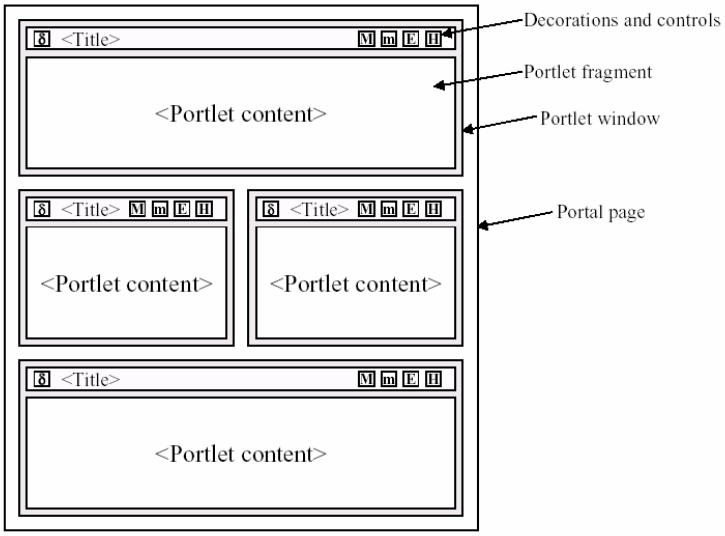
(그림 21) Hibernate Mapping DTD

제 3 절 포틀릿 합성 방법 분석

포틀릿은 Java 기반의 웹 구성요소로서 포틀릿 컨테이너를 통하여 관리된다. 포틀릿 컨테이너에 등록된 포틀릿들을 플러그-인(Plug-In) 가능한 사용자 인터페이스를 보유한 포틀릿 윈도우 형태로 합성하며, 포털 시스템은 이러한 다수의 포틀릿 윈도우를 조합하여 포털페이지를 완성한다.

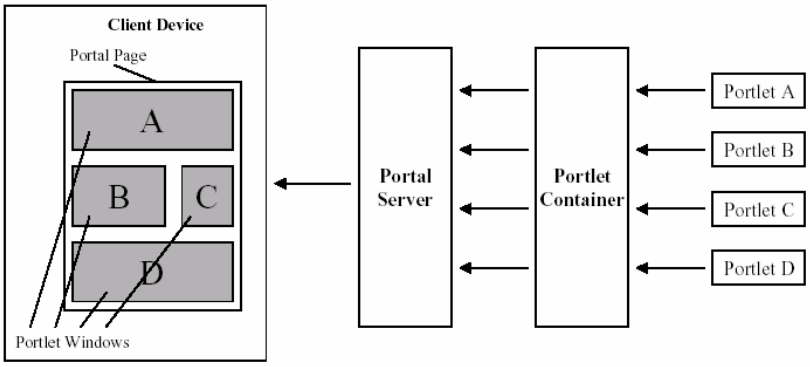
본 절에서는 포틀릿 합성을 위한 분석과 포틀릿 운영 절차를 소개한다. 또한 포틀릿 컨테이너 운용 절차를 소개, 포틀릿 합성 구현과 포털 프레임워크 기본화면을 설계하고 주요 포틀릿 설계와 기능을 제시한다.

포털 페이지는 다음과 같이 포틀릿이 여러 개로 묶여 하나의 페이지로 구성되어질 수 있다. 즉, 포틀릿은 내용은 HTML이나 XML과 같은 마크업 언어(Makeup Language)로 렌더링 될 수 있으며 이렇게 생성된 포틀릿의 내용을 포틀릿 프레그먼트(Fragment)라고 칭한다. 또한 포틀릿의 사용자 인터페이스는 title, Window State 버튼과 다른 장치(decoration)들을 추가함으로써 그 형태를 이루게 되며, 이렇게 만들어진 조각은 포틀릿 윈도우로 불려진다. 그런 다음 포털은 각각의 포틀릿 윈도우들을 완전한 문서 즉, 포털 페이지로 만들기 위해 결합하고 이를 포틀릿의 합성이라고 한다[1,14,15].



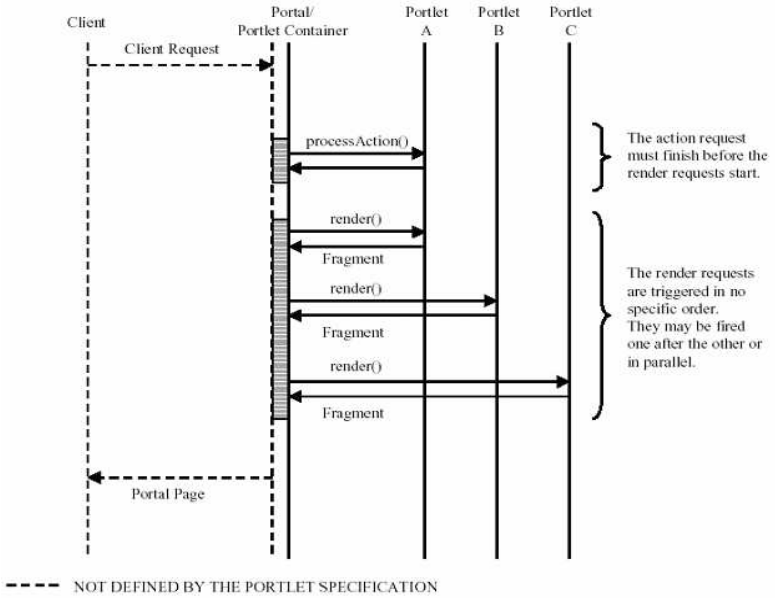
(그림 22) 포털 페이지의 구성

포틀릿들은 포틀릿 컨테이너를 통하여 관리된다. 포틀릿 컨테이너는 포틀릿들에 의해 렌더링 된 컨텐츠를 받는다. 또한 포틀릿 컨테이너는 포틀릿에게 런타임 환경을 제공하게 된다. 포털은 포틀릿에서 요청을 실행할지 여부를 포틀릿 컨테이너에게 전송하고 해당 요청에 대한 실행을 포틀릿을 통해 실행한 후 그 결과를 포탈에 전송하여 합성한다.



(그림 23) 포털 페이지의 생성

포틀릿 객체가 적절하게 초기화된 후에, 포틀릿 컨테이너는 클라이언트 요구를 다루기 위해 포틀릿을 호출할 수도 있다. 포틀릿 인터페이스는 Request 핸들링을 하여 processAction method와 그 render method의 두 메소드를 정의하여 구성하게 된다. 포털/포틀릿 컨테이너가 포틀릿의 processAction 메소드를 호출할 때, 포틀릿 요구는 action request라고 지칭하며, 포털/포틀릿 컨테이너가 포틀릿의 render 메소드를 호출할 때, portlet 요구는 render request라고 지칭한다. 일반적으로, 클라이언트 요구는 포틀릿에 의해 새로 만들어진 URL에 의해 연결되며, 이 URLs을 포틀릿 URLs 이라 부른다. 포틀릿 URL은 특별한 포틀릿에 전송되게 된다. 포틀릿 URLs는 action URL이나 render URL 두 가지 유형으로 구분되게 된다. 이 URLs에 의거하여 포틀릿 객체가 클라이언트의 호출과 응답을 적절하게 배치 운영되게 된다. (그림 24)는 포틀릿 객체의 클라이언트에서 호출함으로써 발생되는 Request 핸들링에 관한 Sequence Diagram 이다.



(그림 24) Portlet의 Request Handling Sequence

제 4 절 포틀릿 합성 방법의 구현

포틀릿 합성은 포틀릿의 구성을 관리하는 Portlet Management와 연관되어 구성되어진다. Portlet Management는 여러 다양한 서비스를 제공하는 포틀릿들을 성격에 맞도록 분류화(Categorization)하고, 포틀릿들을 사용자가 쉽게 이용할 수 있게 하는데 그 목적이 있다. 따라서 포틀릿을 합성하여 포털 페이지를 합성한다는 의미는 Portlet Management에서 설정된 이용 권한 문제 및 포틀릿의 분류등과 관련이 있으므로 포틀릿의 합성과 관리를 동시에 논의한다. 본 논문에서는 포틀릿의 JSR168 표준에 맞추어 포틀릿의 기본적인 환경설정 내용을 XML로 구현함으로써 포틀릿의 추가 및 삭제, 분류화, 개인화 등을 손쉽게 이용할 수 있도록 설계 구현하였다.

<표 2> Portlet.xml 파일

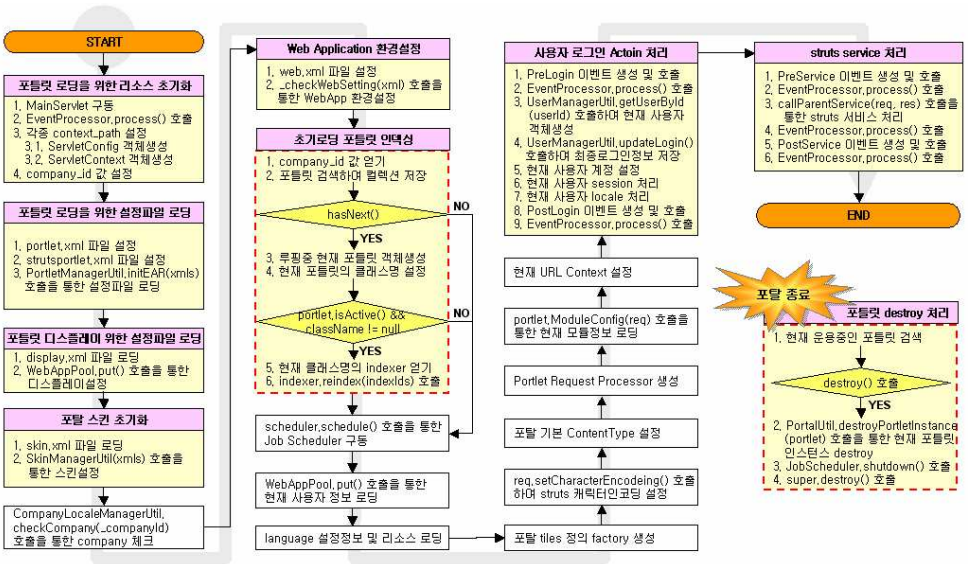
파일명	Portlet.xml
용도	현재 포털에서 지원되는 포틀릿 등록정보 파일 현재 운용중인 포틀릿의 상세정보를 파악하고자할 때 이용
구성샘플	<?xml version="1.0"?> <portlet-app xmlns= "http://java.sun.com/xml/ns/portlet/portlet-app_1_0.xsd" version="1.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://java.sun.com/xml/ns/portlet/portlet-app_1_0.xsd http://java.sun.com/xml/ns/portlet/portlet-app_1_0.xsd"> <portlet> <portlet-name>2</portlet-name> <display-name>My Account</display-name> <portlet-class>com.liferay.portlet.StrutsPortlet</portlet-class> <init-param> <name>view-action</name> <value>/my_account/view</value> </init-param> <expiration-cache>0</expiration-cache> <supports> <mime-type>text/html</mime-type> </supports> <resource-bundle>com.liferay.portlet.StrutsResourceBundle</resource-bundle> <security-role-ref> <role-name>Guest</role-name> </security-role-ref> <security-role-ref> <role-name>Power User</role-name> </security-role-ref> <security-role-ref> <role-name>User</role-name> </security-role-ref> </portlet> </portlet-app>

<표 3> strutsPortlet.xml 파일

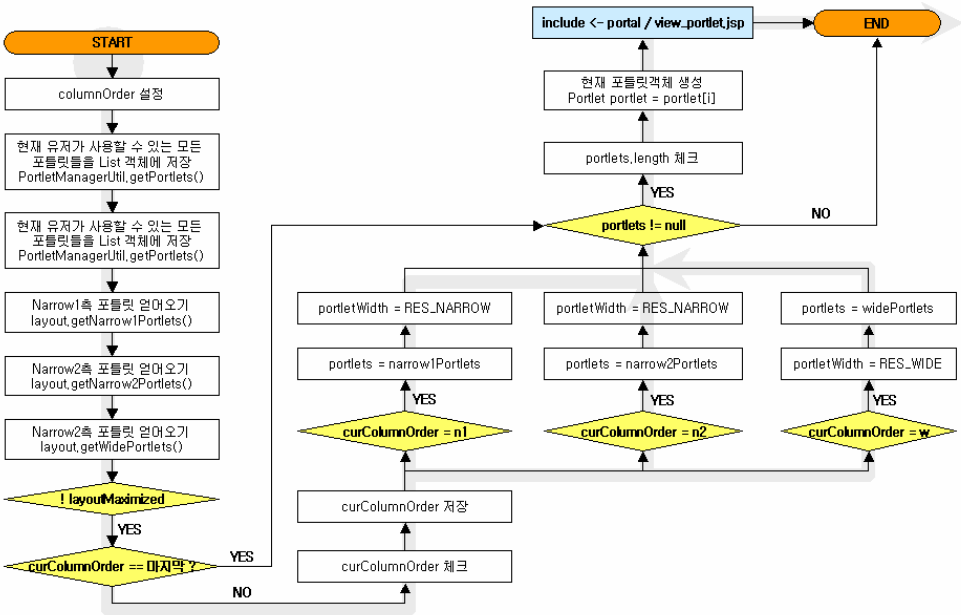
파일명	strutsPortlet.xml
용도	현재 포털에서 지원되는 포틀릿의 struts-path와 적용컬럼을 설정하는 파일 화면의 Narrow 속성에 따라 화면에 보여지는 영역이 결정되게됨.
구성샘플	<?xml version="1.0"?> <!DOCTYPE portlets PUBLIC "-//Liferay//DTD PORTLET 2.2.0//EN" "http://www.umtec.com/dtd/strutsPortlet _2_2_0.dtd"> <portlets> <portlet id="2" struts-path="my_account" /> <portlet id="3" struts-path="search" narrow="true" /> <portlet id="4" struts-path="weather" narrow="true" /> </portlets>

<표 4> Display.xml 파일

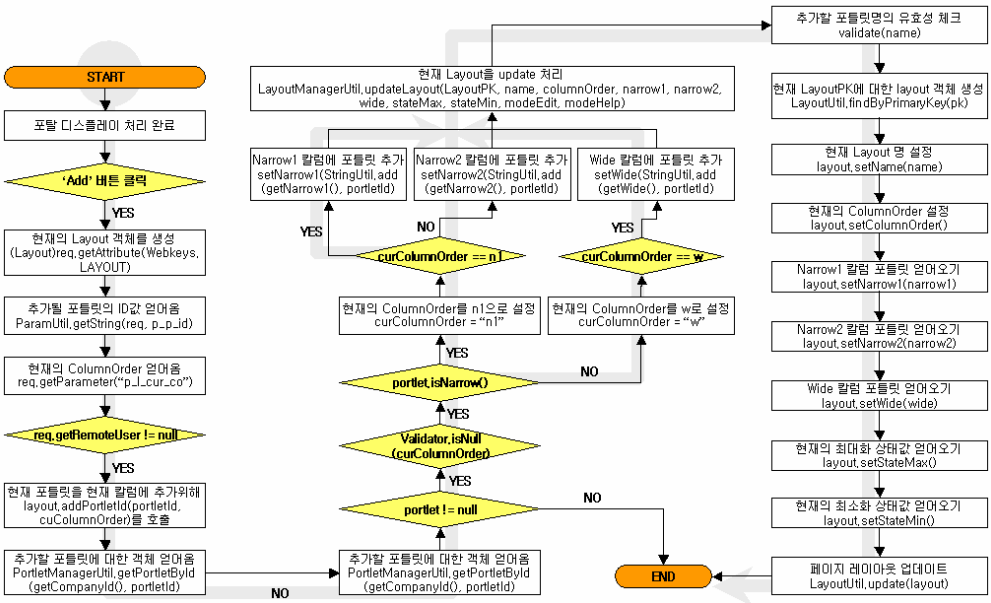
파일명	Display.xml
용도	현재 EIP에서 지원하는 포틀릿들을 분류화(Categorization)한다. 포틀릿들은 Category.%분류명%으로 분류한다.
구성샘플	<?xml version="1.0"?> <!DOCTYPE display PUBLIC "-//Liferay//DTD DISPLAY 2.0.0//EN" "http://www.umtec.com/dtd/display_2_0_0.dtd"> <display> <category name="category.christianity"> <portlet id="7" /> <portlet id="29" /> <portlet id="51" status="new" /> <portlet id="46" /> <portlet id="21" /> <portlet id="44" /> <portlet id="32" /> <portlet id="45" /> </category> <category name="category.community"> <portlet id="10" /> <portlet id="33" status="new" /> <portlet id="8" /> <portlet id="52" status="alpha" /> <portlet id="11" /> <portlet id="22" status="alpha" /> <portlet id="1" /> <portlet id="42" status="alpha" /> <portlet id="19" /> <portlet id="24" /> </category> </display>



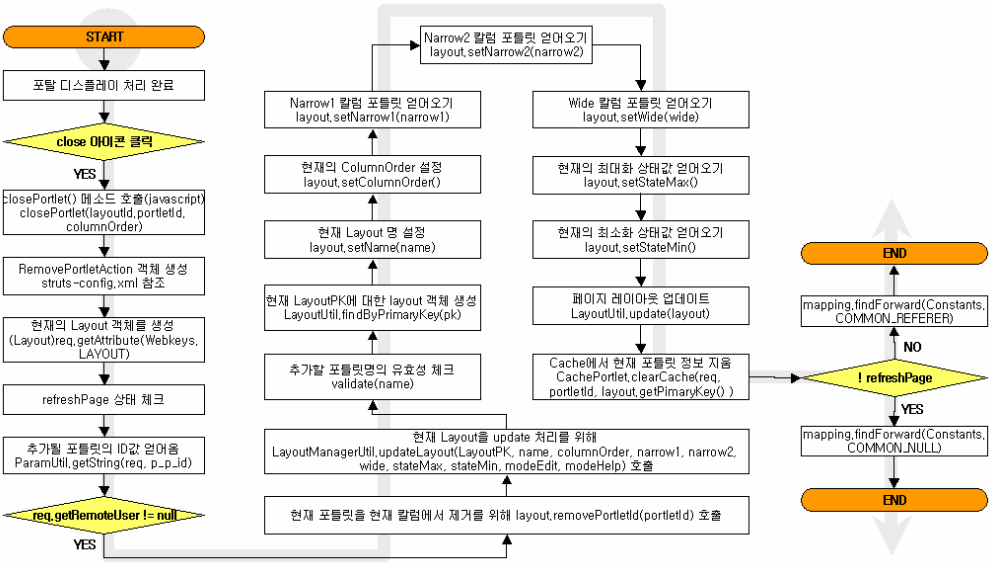
(그림 25) 포틀릿과 포털 시스템 상호작용절차 Activity Diagram



(그림 26) 포털페이지 상에 포틀릿 디스플레이절차 Activity Diagram



(그림 27) 포털페이지 상에 포틀릿 추가절차 Activity Diagram

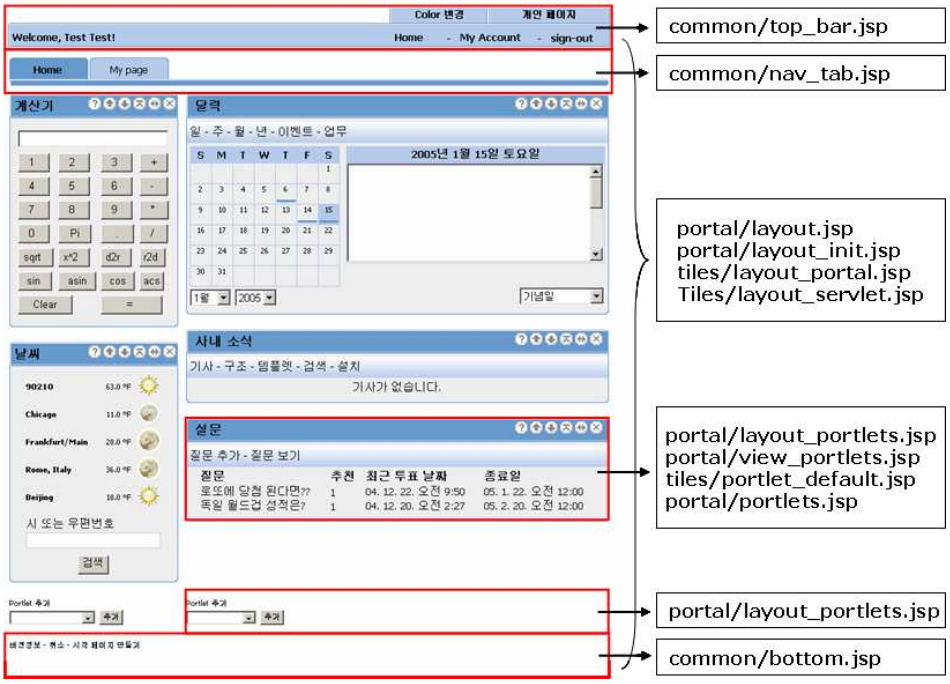


(그림 28) 포털페이지 상에 포틀릿 삭제절차 Activity Diagram

제 5 절 포털 프레임워크 구현 및 기본 화면설계

앞장에서 언급한 바와 같이 포털 페이지는 포틀릿들이 합성됨으로써 하나의 포털페이지가 생성되게 된다. 본장에서는 이러한 EIP 시스템 상에서 공통적으로 사용하고 있는 리소스를 분석하고 EIP 메인 페이지 즉, 화면상의 포틀릿을 포함하고 있는 포털의 메인화면을 구성하고 구현한다.

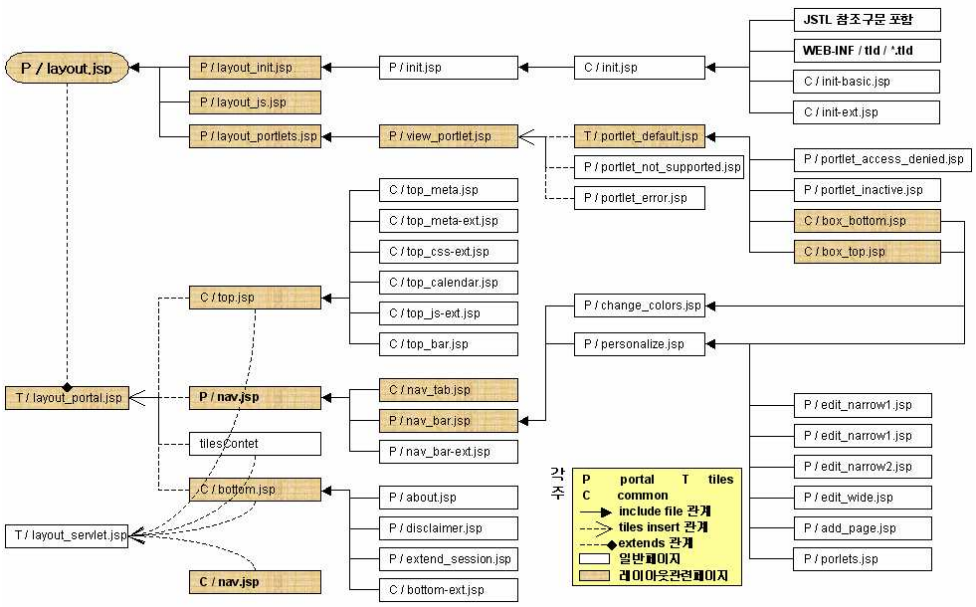
모든 화면의 기본적인 동작 메커니즘은 MVC(Model, View, Controller)방법에 의거한 Struts 프레임워크 기반으로 설계하여 구현한다. MainLayout을 설정할 때 포틀릿 박스를 그려내게 되며, 이때 포틀릿 박스는 JSR168 표준에 맞도록 프로그래밍 한다.



(그림 29) 화면 설계 및 파일 연관도

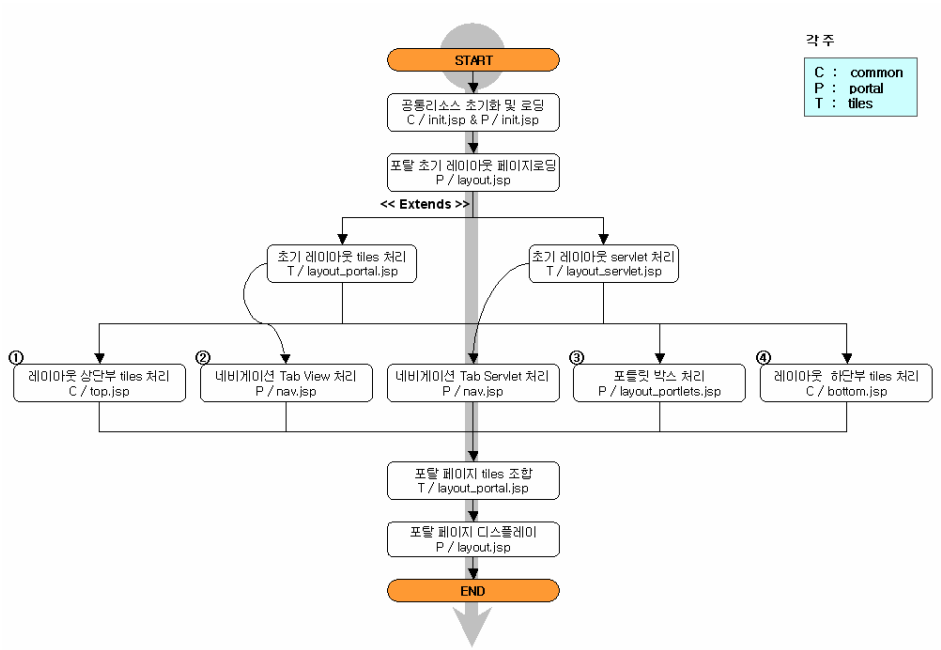
아래 그림은 파일의 효율적 관리를 할 수 있는 네비게이션 기능을 나타낸다. 포털 시스템의 파일은 포털 페이지를 구성하는 페이지를 나타내며, 페이지마다 관계

를 가짐으로써 포털 페이지 및 포틀릿들을 효율적으로 관리하고 생성 및 삭제가 가능하다.

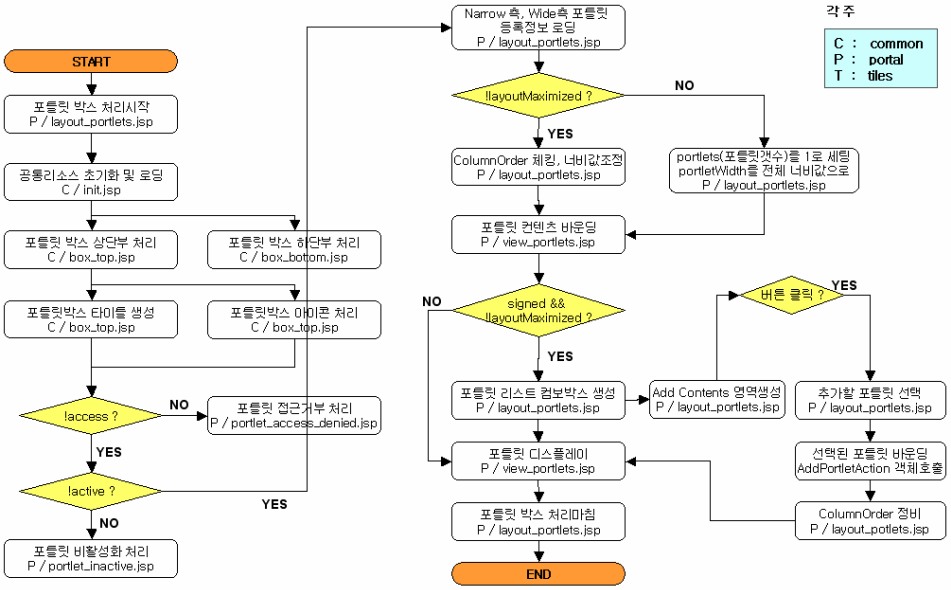


(그림 30) 파일 네비게이션

(그림 31)은 포털 페이지 구성 시 각각의 포틀릿 윈도우의 레이아웃의 정의와 전체 레이아웃의 정의를 통해 일관성 있게 구현하고자 하였다. 이는 정적으로 사용자가 레이아웃을 지정하는 것이 아닌 수시로 사용자에 맞게 편집 및 수정이 가능하다. 이로서 언제든지 사용자에게 최적의 환경을 제공하게 되며, 작업의 효율성 및 협업의 능률을 극대화 할 수 있는 기능을 제공한다.



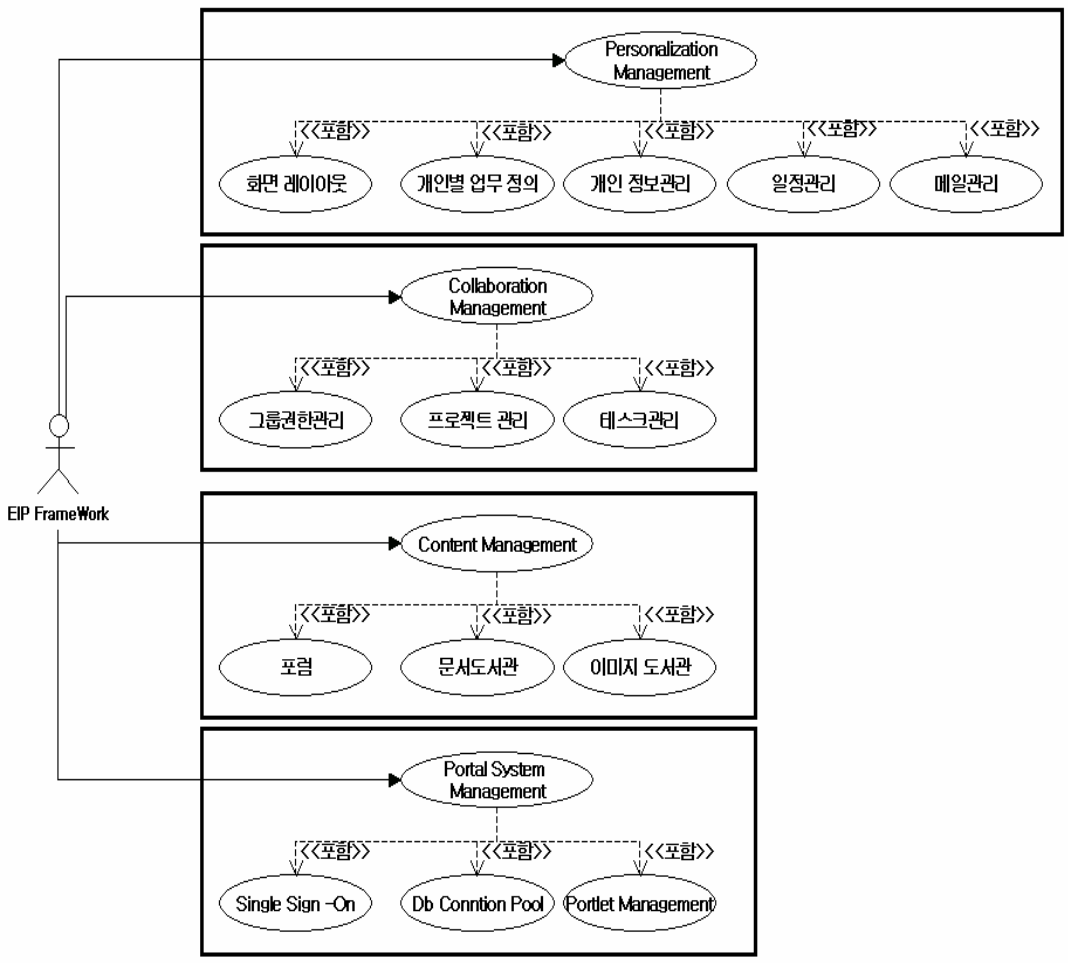
(그림 31) 전체 레이아웃 처리 Activity Diagram



(그림 32) 포틀릿 박스 Activity Diagram

제 5 장 포틀릿 설계 및 구현

포틀릿은 포틀릿에 따라 사용자의 목적과 용도에 따라 관리되고 분류되어 질 수 있다. 따라서 본 논문에서는 EIP 프레임워크의 기능을 다음 (그림 33)과 같이 Personalization Management, Collaboration Management, Content Management, Portal System Management로 구분하여 구현하고자 한다.



(그림 33) EIP 프레임의 유즈케이스

1. Personalization Management

EIP 사용자의 개인화 기능을 보유한 영역이다. 개인화 영역의 구성은 사용자에 맞는 화면 구성기능과 개인의 업무를 관리하고 정의하는 기능, 개인의 신상정보 및 기타 관련 정보의 관리 기능, 일정관리, 메일관리 등의 기능을 포함한다.

<표 5> 개인화 관리 기능

기 능	설 명
화면레이아웃	개인별 화면의 배치 개인별 화면 색상 변환
개인별업무정의	개인별 업무 포틀릿의 설정 개인별 탭의 설정
개인정보관리	개인 신상정보관리
일정관리	개인별 일정관리 협업의 테스크 관리와의 연동 칼렌더의 제공
메일관리	개인 이메일의 관리 이메일 공지의 뿌림

2. Collaboration Management

협업을 위한 기능 정의 영역이다. 이를 지원하기 위해 여러 가지 방법을 통하여 정보와 프로세스를 공유함으로서 협업 환경을 지원한다.

<표 6> 협업 관리 기능

기능	설명
그룹권한관리	그룹별 권한 부여 포틀릿별 사용권한부여
프로젝트관리	그룹별, 개인별 프로젝트 관리기능 프로젝트 생성, 진행, 종료.
테스크관리	개인일정과 프로젝트와의 연동된 테스크 관리

3. Content Management

컨텐츠를 효율적으로 관리하기 위한 영역이다. 이는 포털 시스템내의 여러 커뮤니티 기능과 출판 및 분리 기능 등 다루어지는 여러 컨텐츠들을 다양한 방법으로 관리한다.

<표 7> 컨텐트 관리 기능

기능	설명
포럼	포럼, 게시판, 공지의 기능 부여
문서관리	각종 다양한 문서 자료의 관리 문서의 버전별 관리 버전별 히스토리 관리 개인별, 그룹별, 팀별 권한 부여
이미지관리	이미지 앨범의 기능 부여 개인별, 그룹별, 팀별 이미지 관리

4. Portal System Management

포털 시스템관리 영역으로 포털 시스템의 기본 아키텍처를 관리하는 영역으로써 좁은 의미의 포털 프레임워크 영역으로 정의할 수 있는 영역이다.

<표 8> 포털 시스템 관리 기능

기능	설명
Single Sign-On	Single Sign-On 기능부여
DB Connection Pool	DB Connection Pool의 지원 EIP의 성능에 밀접한 연관 다양한 데이터베이스 연동 가능
Cache Service	EIP의 퍼포먼스를 위한 Cache 서비스 지원

제 1 절 포팅 시스템 관리 기능의 구현

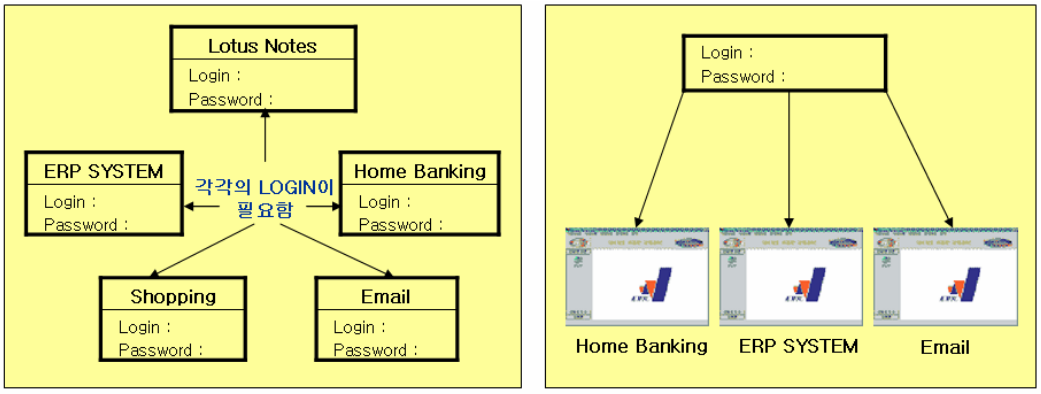
해당 영역에서는 EIP 시스템에 전반적으로 영향을 미치는 시스템적인 영역으로써 SSO(Single Sign On)과 Cache Service 등으로 구성되어 있다.

1. SSO(Single Sign On)

사용자가 시스템에 한 번의 로그인으로, 기업 내의 허가된 모든 자원에 접근할 수 있도록 지원하는 시스템을 의미한다. 한 번의 로그인만으로 기업의 각종 업무

시스템 및 인터넷 서비스에 접속 가능한 보안솔루션이다. SSO를 지원함으로써 얻을 수 있는 장점은 다음과 같다. 해당영역은 개인화 영역의 개인 신상정보 설정과 연관되어 구현되어 진다.

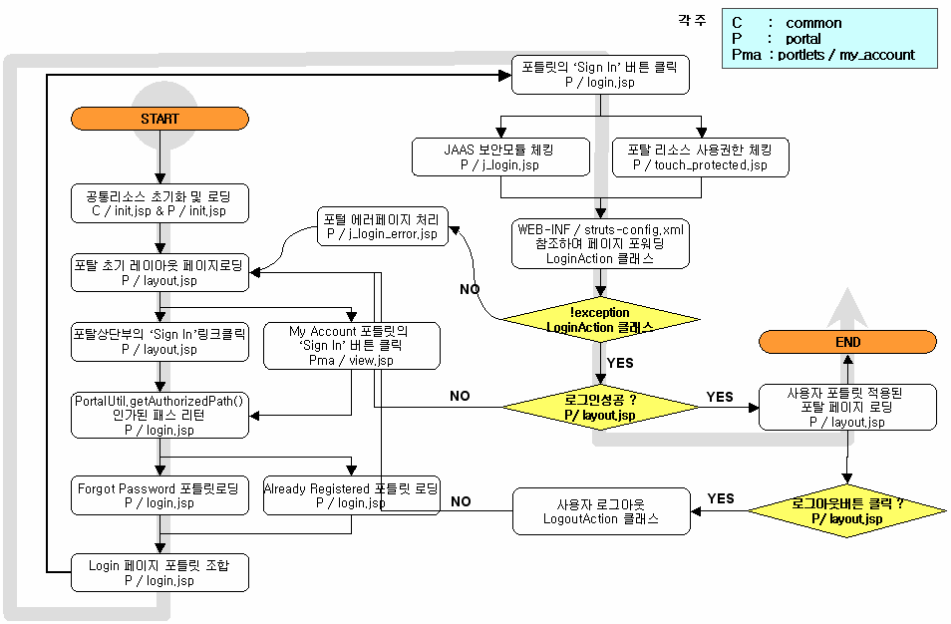
- 사용자 편의성 및 보안성 증가
- 사용자 업무 관리 단일화 및 이동의 편리성 증대
- 추가서비스 개발 비용 감소
- 패스워드 관리 담당자의 업무감소
- 어플리케이션 및 시스템 관리자의 업무 감소



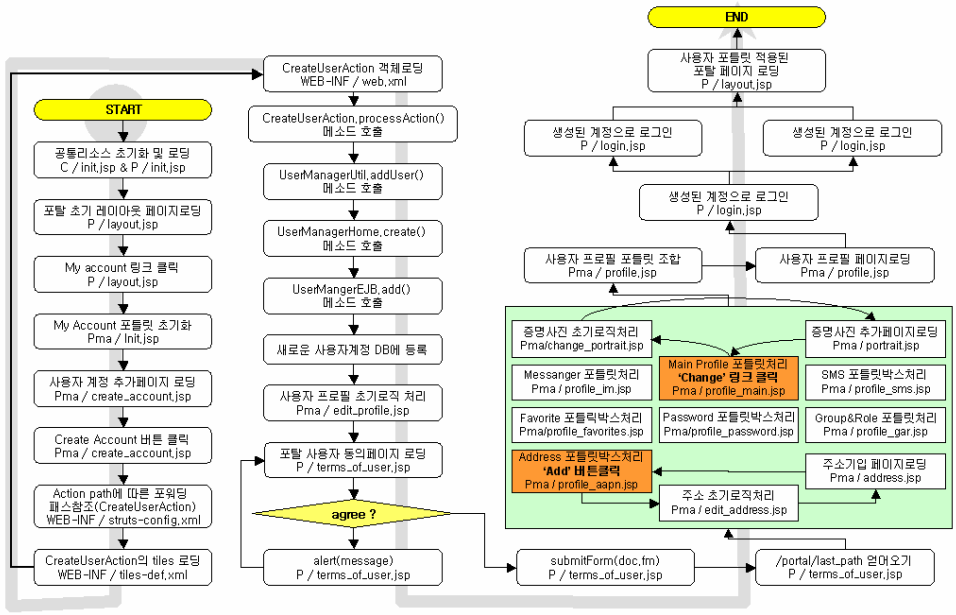
< 기존의 Login 화면 > < SSO Login 화면 >

(그림 34) SSO가 적용된 상태도

아래 (그림 35)는 본 포털 시스템에서 SSO를 적용한 상태의 그림이다. 한 번의 로그인 절차를 거쳐 포털 시스템 내의 자원들을 사용할 수 있다. 이는 관리자의 업무 부담을 줄일 수 있으며, 작업의 효율성을 높일 수 있다. 또한 보안적 측면에서도 관리체계의 단일화를 이룸으로써 더욱더 보안측면이 강화될 수 있다.



(그림 35) SSO가 적용된 로그인 절차 및 EIP 리소스 사용절차 Activity Diagram

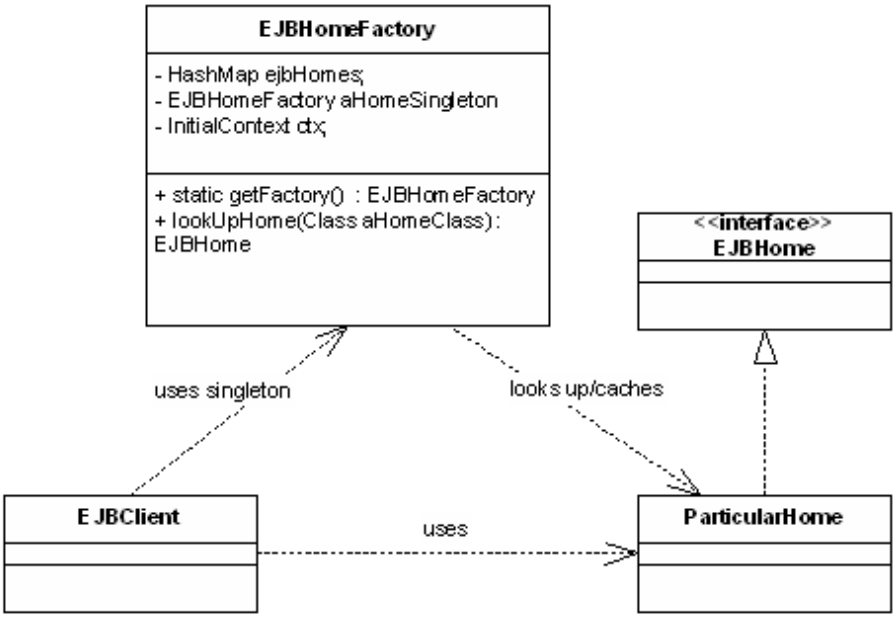


(그림 36) Account 생성 Activity Diagram

2. Cache Service

EIP의 경우 여러 다양한 애플리케이션의 집합이므로 애플리케이션의 성능에 특히 민감하다고 할 수 있다. 따라서 EIP 시스템을 자바기반의 시스템으로 구축 시, 특히 EJB를 사용할 경우 EJB 클라이언트가 EJBhome 객체를 Lookup해야 하는데 이때, 동일한 home 객체를 여러 번 Lookup하게 되는데 이는 EIP의 성능을 떨어뜨리는데 특히 많은 퍼포먼스를 소비한다. 특히나 웹서비스 시스템으로 서비스를 제작할 때 신경을 써야 하는 부분이 바로 퍼포먼스 측면이므로 이에 EJBhome 객체를 캐시 서비스해 그 낭비를 최소화 할 수 있다.

캐시 서비스를 구축하는 방법은 Lookup하는 상세한 부분을 추상화하고, 클라이언트가 EJBhome을 단 한번만 Lookup하게 만들도록 객체를 설계하고 구현하도록 한다. 즉, 재사용 가능한 EJBhome을 Factory 기법으로 패턴화시켜 클라이언트 애플리케이션이 실행되는 동안 EJBhome을 캐시하도록 하는 방법을 활용한다.



(그림 37) EJBhome의 Factory 패턴화 클래스 Diagram

```
import javax.ejb.*;
import java.rmi.*;
import java.util.*;
import javax.naming.*;
/** EJB Home Factory, maintains a simple hashmap cache of EJBHomes**/
public class EJBHomeFactory
{
    private Map ejbHomes;
    private static EJBHomeFactory aFactorySingleton;
    Context ctx;
    private EJBHomeFactory() throws NamingException
    {
        //Initialize context with System property Info
        ctx = new InitialContext();
        this.ejbHomes = Collections.synchronizedMap(new HashMap());
    }
    /*** Returns the singleton instance of the EJBHomeFactory***/
    public static EJBHomeFactory getFactory() throws NamingException
    {
        if ( EJBHomeFactory.aFactorySingleton == null )
        {
            EJBHomeFactory.aFactorySingleton = new EJBHomeFactory();
        }
        return EJBHomeFactory.aFactorySingleton;
         }
    /*** Lookup and cache an EJBHome object ***/
    public EJBHome lookUpHome(Class homeClass) throws NamingException
    {
        EJBHome anEJBHome = (EJBHome) this.ejbHomes.get(homeClass);
        if(anEJBHome == null)
        {
            anEJBHome = (EJBHome) javax.rmi.PortableRemoteObject.narrow (
            ctx.lookup (homeClass.getName()), homeClass);
                    this.ejbHomes.put(homeClass, anEJBHome);
        }
        return anEJBHome;
    }
}//EJBHomeFactory
```

(그림 38) EJBhome의 Factory 패턴화 예제

제 2 절 개인화 관리 기능의 구현

개인별로 화면을 제어하거나 개인별 일정 관리 및 개인의 정보를 관리하는 기능을 설계한다. 이에 본 논문에서는 개인화 관리 도구의 요구사항을 통해 화면 구성과 필수적인 개인정보 설정 부분들을 고려하여 구현한다.

1. 개인 정보 설정, 화면 레이아웃

(그림 39) 개인별 프로파일 설정 화면

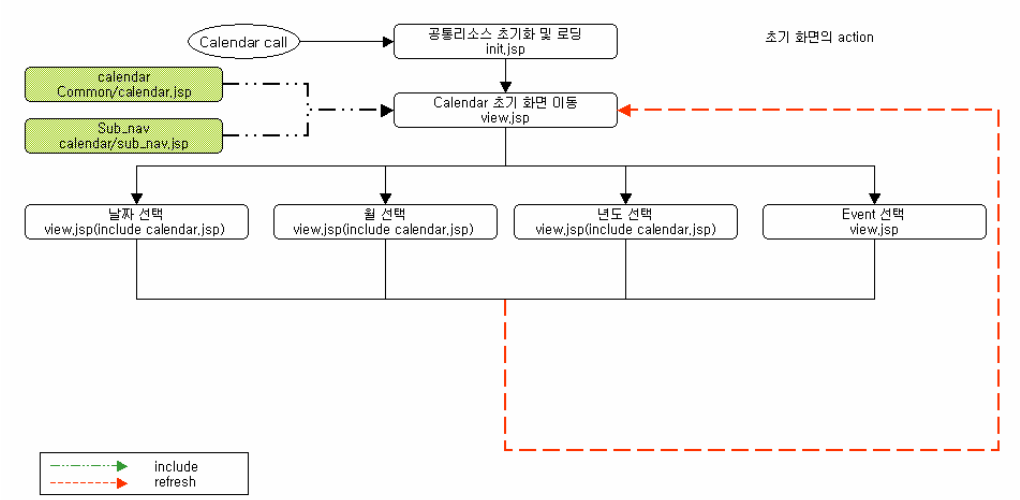
해당 기능들은 구현 설명된 SSO 기능과 Portlet Management 의 기능이 복합적으로 아우러져 구현되어지고, 이미 앞 절에서 어떤 형태로 구현되는지 Activity Diagram을 구현 설명하였으므로 Activity Diagram은 생략하고 구현된 화면위주로 설명한다.

(그림 40) 개인별 페이지 환경설정 화면

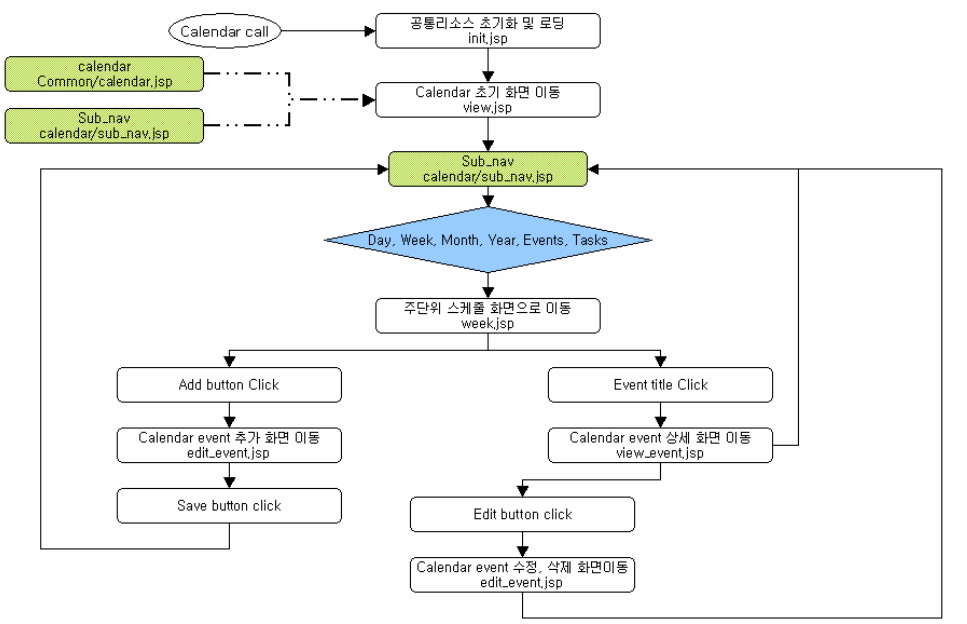
2. 개인 일정관리

개인의 일정관리 영역은 해당 사용자의 개인별 스케줄을 설정하며 Day, Week, month, year등의 영역별로 관리 · 구현되며 하나의 포틀릿의 형태로 구현한다.

(그림 41) 개인 일정관리 메인화면



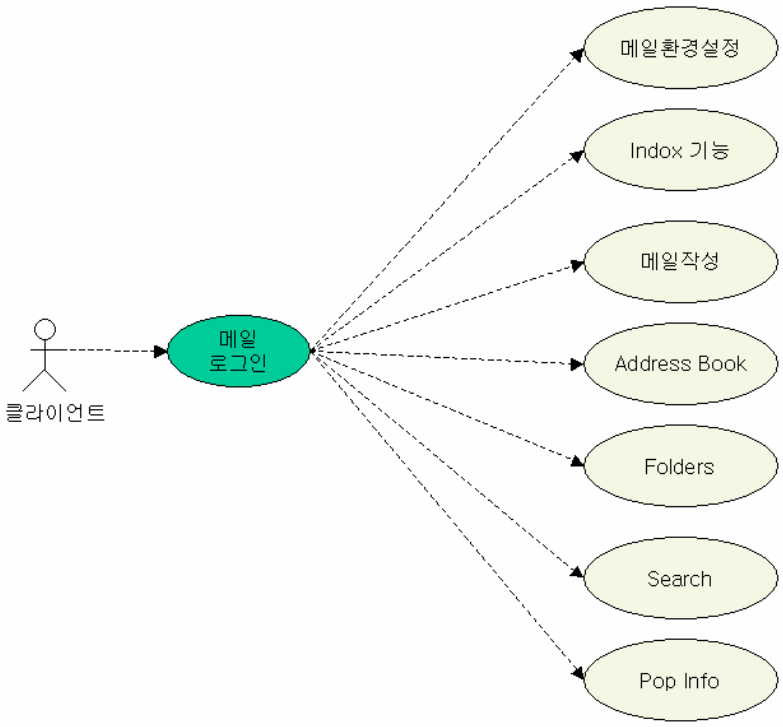
(그림 42) 일정관리 초기화면 Activity Diagram



(그림 43) 일정관리 작업별 Activity Diagram

3. 메일관리

메일의 경우 Collaboration Management의 협업기능으로 구분하기도 하며, 본 논문에서는 개인의 메일을 취급하는 형태로 취급하여 메일의 기능을 구분한다. 본 논문에서 메일의 구현으로 javamail의 IMAP과 SMTP를 기준으로 제작하였다. 아래의 그림은 사용자가 메일을 로그인 했을 때 설정, 관리 및 사용할 수 있는 기능들을 나타낸다.



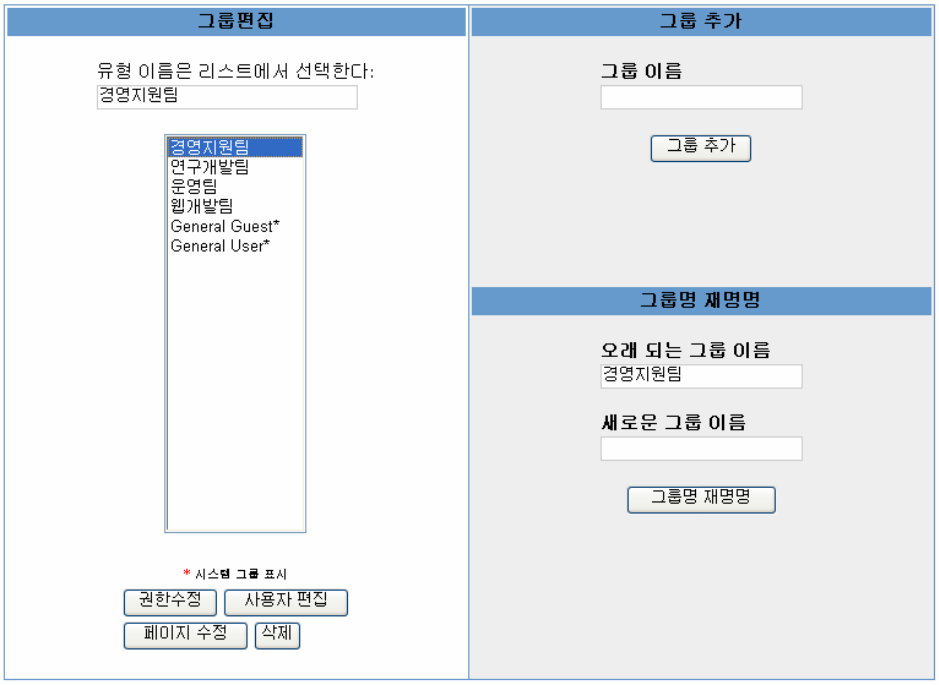
(그림 44) 메일 UseCase Diagram

(그림 45) 메일 포틀릿 메인화면

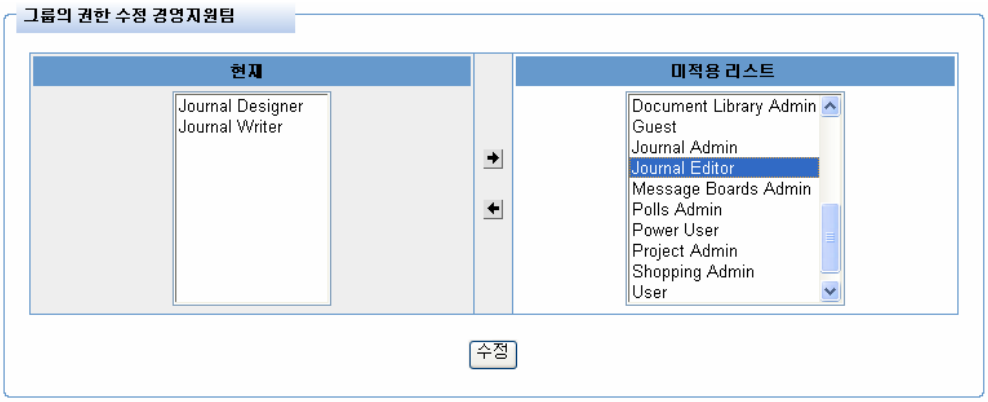
제 3 절 협업 관리 기능의 구현

Collaboration Management에서는 개인 사용자의 권한 및 그룹 지정이 되어야 하며 개인별 관리가 가능하여야 한다. 또한 모든 권한은 포틀릿별로 권한이 적용되게 된다.

EIP에서의 협업기능의 경우 대상이 되는 회사별로 제공되는 기능이 각각 틀리기 때문에 프레임워크의 구조에서는 포틀릿에 대하여 그룹핑하여 사용자별로 제공할 수 있는 기반을 제공하는 역할을 하여야 하며, 협업에 관련된 서비스가 포틀릿으로 제공될 경우 사용자의 그룹 권한별로 포틀릿을 이용할 수 있는 권한을 부여할 수 있는 체제가 갖추어져야 한다.



(그림 46) 그룹의 편집 및 추가



(그림 47) 그룹별 권한의 배정

(그림 48) 포틀릿의 그룹 배정

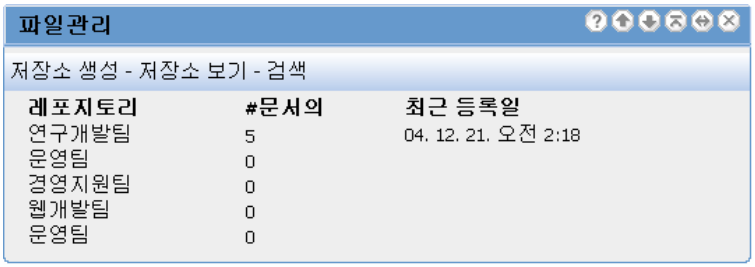
제 4 절 콘텐츠 관리 기능의 구현

Content Management는 회사 내의 정보 자원 등을 효율적으로 관리할 수 있는 체제를 갖추는 역할을 하는 영역이다. 일반적으로 사용자나 그룹별로 자원을 관리할 수 있는 체계가 갖추어져야 한다. 본 논문에서 제공하는 Content Management 기능은 파일관리와 이미지 관리, 포럼 등을 들 수가 있다.

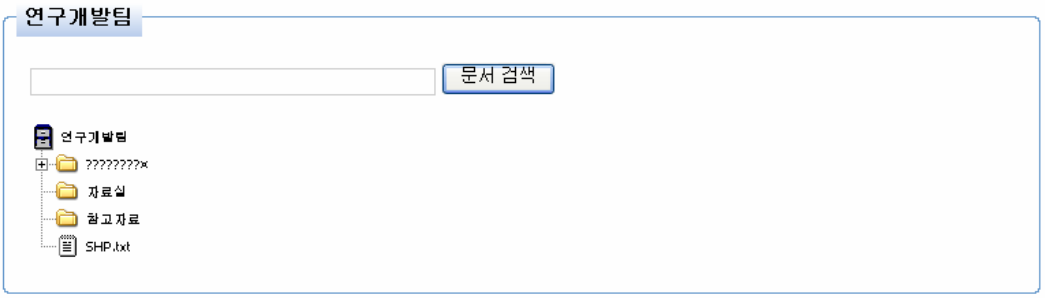
1. 파일관리

파일관리는 기업 내에 존재하는 문서 및 파일을 관리하는 기능을 가진 포틀릿으

로 문서를 저장할 수 있는 저장소(Repository)를 지정하고 문서를 저장 관리할 수 있는 기능을 보유한다.



(그림 49) 파일관리 포틀릿 메인화면



(그림 50) 저장소 안의 내용보기

(그림 51) 저장소(Repository)의 생성

저장소의 생성 시 해당 저장소를 접근할 수 있는 권한을 읽기 권한과 쓰기 권한을 가진 그룹을 지정하게 함으로써 Collaboration Management의 기능을 활용할 수 있도록 구현하였다.

(그림 52) 문서의 관리

모든 문서 및 파일은 버전별로 히스토리를 관리할 수 있는 기능을 보유하도록 작업을 지원함으로써 협업 기능의 효율성 기원과 기능성을 강화하였다.

2. 이미지 갤러리

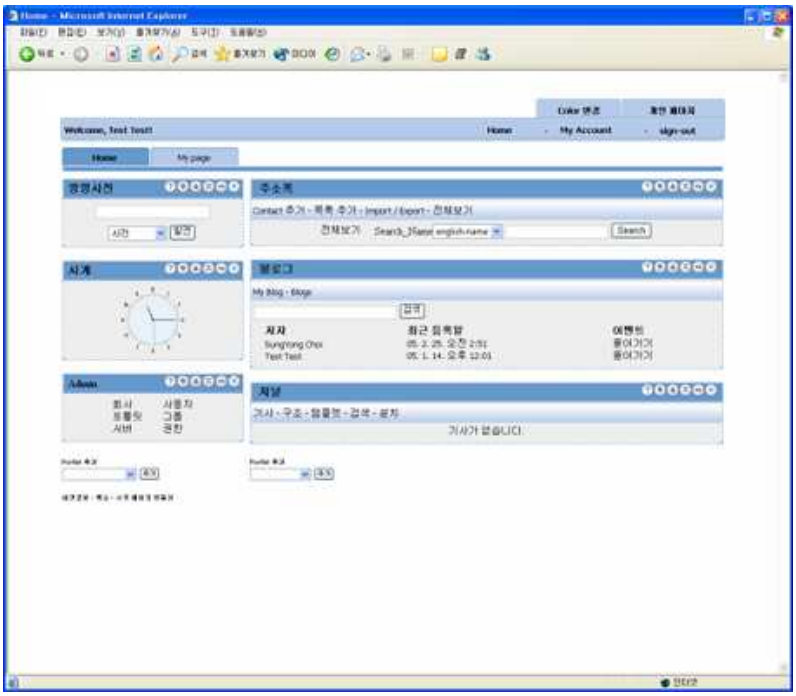
사내의 이미지 및 동영상 등을 관리할 수 있는 기능을 보유한 포틀릿으로 문서 정보와는 다른 멀티미디어 정보의 관리를 할 수 있는 기능을 보유한 포틀릿이다.

(그림 53) 이미지 갤러리 포틀릿 메인화면

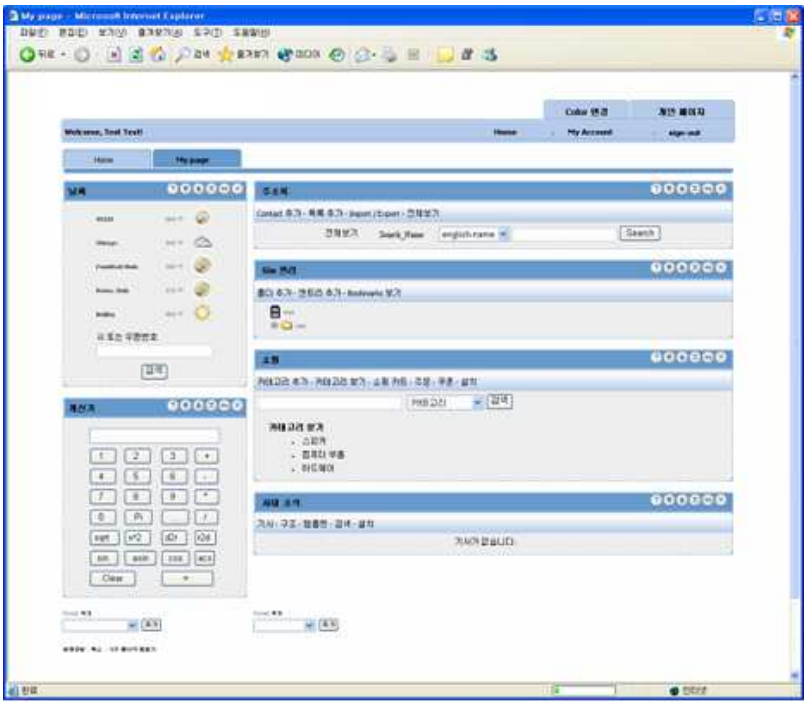
(그림 54) 이미지 갤러리 내용보기

제 5 절 포털 프레임워크의 서비스

구현된 포털 프레임워크 상의 포틀릿을 사용자가 배치하고 활용하는 화면의 구성은 다음과 같다. 보이는 구성화면은 정적으로 보일 수 있으나 실제 구현된 화면은 사용자가 동적으로 구성할 수 있으며, 개인화된 화면으로 구성할 수 있다. 또한 협업의 기능을 활용한 작업 지원도 가능하도록 사용자가 설계할 수 있다. 한번 구성된 화면은 사용자가 다시 바꾸지 않는다면 지속적으로 유지되며, 포틀릿 하나하나의 정보는 시시각각 갱신되게 된다. 단 협업지원을 위한 포틀릿이나 또는 실시간 데이터가 갱신되어야 하는 포틀릿의 경우에는 제한한다.



(그림 55) 로그인 후 첫 Home 탭의 구성



(그림 56) 사용자 개인이 설정한 Mypage

제 6 장 결론

기업은 기본적으로 수많은 정보들과 어플리케이션들을 바탕으로 다양한 업무 처리를 한다. 예를 들면, 병원의 경우 입원, 퇴원, 진료등과 같은 업무를 처리하는데, 데이터베이스를 구축하여 정보의 통합을 시도하였고, 각각의 어플리케이션을 구축하여 전체 업무의 통합을 시도하였다. 그러나 방대한 데이터의 효율적인 관리 정책의 미비와 각각의 업무들이 개별적으로 취급되어 이를 달성하기에는 너무 많은 어려움이 있었다. 더군다나 이 경우 시간적, 인적, 물질적으로 적지 않은 비용이 들었고, 작업의 효율에도 적지 않은 문제점을 갖고 있었다. 이에 업무에 연관된 정보들을 통합하여 체계적으로 관리하는 시스템인 EIS(Enterprise Information Service)와 진료, 검사, 입원과 같은 실제 업무와 관련된 어플리케이션들을 통합하여 놓은 시스템인 EAI(Enterprise Application Integration)이 자연스럽게 제안되었으며, 이러한 정보의 통합을 위한 EIS와 어플리케이션의 통합을 위한 EAI가 융합된 것이 현재의 엔터프라이즈 포털(enterprise portal)이다.

엔터프라이즈 포털을 통하여 직원 및 고객은 기업 내 대부분의 업무 수행이 가능하다. 이렇듯 포털을 이용하는 기업의 경우 직원-직원, 직원-고객, 직원-파트너 간의 협업을 통해서 업무 처리를 가능하게 하며 이를 통해 경영혁신 및 기업 내 생산성 향상에 많은 기여를 할 수 다. 이후 더 발전하여 엔터프라이즈 포털에 컨텐츠(content) 관리 시스템을 도입하여 기업 내의 수많은 정보 자산들을 효율적으로 관리할 수 있을 뿐만 아니라 엔터프라이즈 포털을 통해서 직원이나 고객이 들어오면 상호간의 이메일, 메신저와 같은 커뮤니케이션 기능을 통해 새로운 소식이나 지식자원들의 공유도 가능하게 한다.

이러한 기업 내의 엔터프라이즈 포털은 비즈니스 관점에서 IT 아키텍처를 구성하는 서비스 지향 아키텍처(SOA)가 새로운 패러다임으로 부상하고 있다. 이는 SOA

가 기업 IT 인프라의 복잡성 및 유지비용을 최소화하고, 기업의 생산성과 유연성을 극대화 할 수 있는 새로운 IT전략으로 대두되고 있기 때문이다. SOA는 변화하는 비즈니스 우선순위에 대처하기 위해 비즈니스 프로세스기반 IT 인프라의 요소를 재사용하고 통합할 수 있는 안전한 컴포넌트로 다루는 유연성을 제공한다. SOA는 기업이 내부 애플리케이션을 외부 타 기업과 통합할 수 있는 표준 소프트웨어 요소이자 비즈니스 프로세스의 집합체이다. SOA는 비즈니스 프로세스를 기본적인 표준 빌딩 블록 단위로 분할하여, 이를 IT 프로세스와 유연하게 일치시키는 특징이 있다. 따라서 기업내부는 물론 협력사 및 고객과 표준화된 e-비즈니스 환경을 구축할 수 있게 된다. 이러한 SOA 아키텍처를 구현하기 위한 기술적 방법으로 웹 서비스가 표준으로 자리 매김을 하고 있다.

이에 웹 서비스가 기업 내 또는 기업 간 통합 서비스의 장애요인을 해결할 수 있는 대안으로 등장하였다. 이러한 웹 서비스(Web Services)는 웹 어플리케이션의 새로운 형태이다. 로컬 컴퓨터상에 있는 메소드를 호출하는 것과 같이 사용자가 느끼지 못할 정도로 자연스럽게 내부 또는 외부의 이질적인 어플리케이션간의 통합 서비스를 제공받을 수 있다. 새로운 비즈니스 파트너간의 시스템과의 통합도 자동적으로 이루어지게 된다. 또한 다양한 통신 방식에 의해 구성된 분산 시스템에 대한 접근성이 뛰어나고, 소프트웨어 통합을 자동화함으로써 상호 연결된 작업을 기존에 비해 훨씬 빠르고, 유연하며, 효율적으로 처리할 수 있다. 특히, 웹 서비스는 새로운 시스템을 구축하는 것이 아니라 기존에 존재하고 있는 시스템을 통합하여 운영할 수 있다.

또한 웹 서비스는 프로그래밍 가능한 캡슐화 된 언어이다. 다른 어플리케이션 혹은 웹 서비스로부터 호출될 수 있는 캡슐화 된 기능이 있으며, 그 인터페이스와 사용방법, 기능에 대한 설명이 외부에 공개되어 있어 정확한 코딩을 통해 누구나 이용할 수 있는 어플리케이션이 되는 것이다. 즉, 사용자는 복잡한 구현 대신에 간단한 인터페이스만을 구현하면 웹 서비스를 바로 사용하는 것이 가능하다. 일종

의 컴포넌트 기반 어플리케이션이라고 할 수 있다. 웹 서비스는 웹에 존재하는 컴포넌트이며 개발자들은 새로운 웹 서비스 컴포넌트를 개발할 뿐만 아니라 필요한 웹 서비스를 찾아서 자신이 만든 컴포넌트와 조립하는 과정을 거쳐 새로운 어플리케이션을 개발할 수 있다.

이에 본 논문에서는 서비스지향 아키텍처를 구성하는 웹서비스를 지원할 포털과 포틀릿들 간의 통신을 WSRP 프로토콜을 이용하여 처리하였으며, 또한 모든 포틀릿들은 다른 포털 프레임과의 호환성을 위하여 JSR168 표준을 준수하여 모든 포틀릿들을 구축하였다.

본 논문에서 구축된 JSR168과 WSRP 기반의 포털 프레임워크를 활용함으로써, 기업이나 조직에서의 단일 엑세스 포인트를 제공하여 정보의 관리와 협력(Collaboration) 등으로 가상공간에서의 공통적인 작업공간(Workspace)를 제공할 수 있으며, 많은 시간과 비용을 들여 제작한 여러 프로그램 컴포넌트들을 손쉽게 재사용하고 통합함으로써 기업 내 조직의 업무 혁신을 달성할 수 있다.

본 논문에서 제시한 웹 서비스 기반 포털 프레임워크 시스템은 개인화 및 협업 환경을 지원하고, 회사 자원의 통합관리를 할 수 있는 환경을 제공한다. 이에 본 논문에서 제시한 웹서비스 기반 포털 프레임워크와 더불어 포털 시스템 상에서 포틀릿의 정보를 분류 및 관리하기 위해서 포틀릿 온토로지를 구축할 수 있다. 포틀릿 온토로지를 포털 시스템에 적용할 경우 포털과 각각의 포틀릿 간의 의미처리 기반의 유기적인 상호운용이 가능해지며, 지식기반 포틀릿 처리 지원이 가능해진다.

이렇듯 온토로지를 적용한 포털 시스템을 기업 내에 도입할 경우, 기존 자원들의 재사용을 통해 개발비용을 절감할 수 있으며, 고수준의 정보 서비스를 제공함으로써 연구개발 기술력 신장과 경쟁력을 확보할 수 있을 것이다. 이와 관련하여 시맨틱 웹서비스와도 연계 가능한 연구도 필요할 것으로 생각된다.

참 고 문 헌

[1] Java Portlet Specification Version 1.0 2003.8.7 Alejandro Abdelnur, Stefan Hepper

[2] EIP 시장동향과 사례분석. IT Brief, 2003

[3] EIP(Enterprise Information Portal) 솔루션 도입을 위한 평가기준 개발에 대한 연구, 2000, 박성준, 최강림, 전미현, 이성준

[4] META Delta (2004) 2004 megaPortal

[5] Enterprise Portal의 개념, 솔루션 및 적용사례, e-Buszgroup, 박정서

[6] IBM WebSphere Portal Server Architecture Specification 4.x, Carol Jones

[7] IBM WebSphere Portal Server Functional Specif ication 4.x, Martin Scott Nicklous

[8] WebServices Conceptual Architecture, WebServices Architetcture Team, IBM SoftwareGroup, 2001

[9] WebSphere Portal Server 1.2 Technical Whitepaper, IBM Software Group, 2001

[10] Simple Object Access Protocol (SOAP) 1.1, Don Box, David Ehnebuske, Gopal Kakivaya

[11] http://www.oasis-open.org/committees/download.php/3343/oasis-200304-wsrp-specification-1.0.pdf

[12] http://www.oasis-open.org/committees/wsrp/#documents

[13] http://ws.apache.org/axis/java/user-guide.html#WhatAxisCanSendViaSOAPWithRestrictedInteroperability

[14] Java Community Process(JCP), JSR168: Portlet Specification, http://jcp.org/en/jsr/detail?id=168

[15] OASIS Web Services for Remote Portlets TC, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrp

[16] Enterprise Integration: An Architecture for Enterprise Application and Systems Integration, Wiley; 1st edition, February 1, 2002, Fred A. Cummins

[17] Knowledge Management Metrics Development:A Technical Approach, White Paper No. Ten, http://www.dkms.com/white_papers.htm, June 25, 1998

[18] Executive Information Systems, Inc., Portal Progress and Enterprise Content Management, January 9, 2003, Joseph M. Firestone, Ph.D.

[19] "Enterprise Information Portals," Merrill Lynch, Inc., NY, November 16, 1998, Christopher C. Shilakes and Julie Tylman

[20] "Enterprise Information Portals:Realizing The Vision Of Information At Your Fingertips,' " AViador, Inc. White Paper, San Mateo, CA, January 1999, Viador

[21] "The Semantic Web," Scientific American, Vol.284, No.5, May 2001, pp.34-43., Tim Berners Lee, J. Hendler, and O. Lassilla

[22] Microsoft MSDN, XML Web Service, http://msdn.microsoft.com/library/default.asp?url=/nhp/Default.asp?contentid=28000442

[23] "Web Services Framework", W3C Workshop on Web Services, San Jose, USA, 2001., IBM Corp, Microsoft Corp

[24] "A Runtime System for Interactive Web Services", Amsterdam, Netherlands: 1999, Claus Brabrand, Anders Møller, Anders B. Sandholm, Michael I. Schwartzbach

[25] "Interactive Web Services ", Architecture White Paper, 2003, Web Collage

[26] "Building Interactive Web Services with WSIA & WSRP", Web Service Journal, December 2002, Elion Reshef

[27] "SW개발 패러다임의 변화 '웹서비스' - 웹서비스 발전방향과 전망", 경영

과 컴퓨터, 2003년 9월, 박범대
[28] “시맨틱 웹의 온토로지 언어”, 정보과학회지, 제21권, 제3호, pp. 18-27, 2003, 이재호
[29] “시맨틱 웹에서의 온토로지 공학”, 정보과학회지, 제21권, 제3호, pp.28-35, 2003, 양정진
[30] “의미기반 정보검색”, 정보과학회지, 제19권, 제10호, pp. 7-18, 2001, 장명길 외
[31] “한국어정보처리와 온토로지”, 2004 한국어정보처리연구회 동계 튜토리얼 자료집, 옥철영
[32] “질의응답시스템을 위한 백과사전 기반 지식베이스와 온토로지”, 제15회 한글 및 한국어 정보처리학술대회 자료집, pp. 177-183, 2003, 최호섭, 옥철영, 김창환, 왕지현, 장명길
[33] “온토로지를 이용한 인터넷웹 검색에 관한 실험적 연구”, 정보관리학회지, 제20권, 제1호, pp. 417-455, 2003, 김현희, 안태경
[34] “시맨틱웹을 위한 온토로지 언어와 구현사례 연구”, 정보관리연구, 제34권, 제3호, pp. 87-109, 2003, 정도헌
[35] "The Semantic Web", Scientific American, 2001, Berners-Lee, T., Hendler, J., Lassila