웹2.0 기반의 웹서비스 사례연구
- Ajax를 활용한 DashBoard 구현 -

양 희 정

목 차

1. 들어서며
2. 웹2.0 요소기술
3. Ajax 작동원리
4. Ajax 응용분야
5. 맺으며

1. 들어서며

웹2.0은 누구든지 웹에서 제공하는 각종 데이터를 활용하여 다양한 서비스를 생산할 수 있는 '플랫폼으로서의 웹(Web as Platform)' 환경을 의미한다. 즉 이용자 중심의 참여와 개방 그리고 공유에 바탕을 둔 플랫폼으로서의 웹 '패러다임'이라고 정의할 수 있다. 분산화 되고 사용자 중심의 커뮤니티가 핵심인 동적인 열린 공간 웹에서는 정보와 서비스를 일방적으로 수신만 하는 형태가 아닌, 제공되는 응용 프로그램과 데이터를 이용하여 사용자 스스로 새로운 서비스를 창출할 수 있다[1]. 본 글에서는 웹2.0을 구성하는 기술을 정리하고, 구현사례로 Ajax를 활용한 DashBoard 웹서비스의 구축방법을 살펴보고자 한다.

2. 웹2.0 요소기술

2.1 웹2.0 기반 기술환경

1) Next Generation Network & System

웹을 지원하는 시스템 및 네트워크 기반에 있어서 성능, 용량, 가용성, QoS가 최고의 품질을 지원하는 일상생활에 보편화된 인터넷 환경은 웹서비스를 구성하고 있는 사용자와 제공자에게 다양한 컨텐츠 서비스를 보장한다.

2) Semantic Web[2]

시맨틱 웹은 컴퓨터가 정보자원의 뜻을 이해하고, 서로 의사소통을 하며 논리적 추론까지 할 수 있는 '차세대 지능형 웹'이다. 시멘틱 웹 기술은 크게 RDF(Resource Description Framework), RDF Schema와 DAML+OIL, OWL 등의 온톨로지(Ontology[3]) 언어들을 꼽을 수 있으며 이 중에 가장 기초가 되는 것이 RDF다. RDF는 XML의 단점을 보안하기 위해 제시된 기술로 특정 자원에 대한 메타데이터를 설명하는 XML기반의 프레임워크다.

RDF는 자원, 속성, 속성값을 묶어서 하나의 단위로 취급하는 기술로 정보를 구성하는 자원에 대한 좀 더

---

* 한국은행 전산정보국 품질관리반 조사역

1) 누구나 자유롭게 글을 써서 수정해 나가는 사용자 참여로 만들어진 http://www.wikipedia.org는 온라인 백과사전으로, 2001년 미국 비영리 단체인 위키미디어재단을 시작으로 전세계 200여 개 언어로 만들어지고 있으며 한국의 경우 2002년 10월부터 시작

2) World Wide Web을 제시한 팀 버너스리(Tim Berners-Lee)의 1999년도 저서 'Weaving the Web'을 통해 알려진 개념

3) Ontology란 사람이 생활속의 사건이나, 사물등을 경험하면서 자각하는 내재된 공통특징인 '개념'을 컴퓨터에 적용하여 일종의 데이터베이스 형태로 만드는 기술로, Context를 처리하고 표현하기 위해 개념적 의미를 시맨틱 기술에서 도입하여 컴퓨터가 이해하고 처리 할 수 있는 지식으로서의 컨텐츠를 구현함으로써 컴퓨터로 하여금 의미적 내용을 더 잘 이해하고 처리할 수 있도록 해주는 개념과 관계에 대한 명세

세밀한 설명과 관계파악을 쉽게 해준다. 즉 RDF라는 기술을 이용하면 문서에서 사용되는 요소의 의미와 문서 사이의 관련성 표시가 쉬워지고, 컴퓨터 간의 자동화 처리가 간단해 진다.

2.2 웹2.0 기반 문화

1) Blog

웹(web) 로그(log)의 줄임말로, 1997년 미국에서 처음 등장하였다. 일반인들이 자신의 관심사에 따라 일기·칼럼·기사 등을 자유롭게 올릴 수 있을 뿐 아니라, 개인출판·방송·커뮤니티까지 다양한 형태의 1인 미디어다.

이러한 Blog는 RSS(RDF Site Summmary, Rich Site Summary)를 통해 정보의 위치와 내용을 알 수 있으며, 트랙백(TrackBack[4])을 통해 의견을 교환할 수 있어서 이제 단순 '미디어'로만 발전한 플랫폼이 아닌 '다양한 시도'를 수용할 수 있는 도구이다.

2) UCC

UCC(User Created Content)는 사용자가 직접 제작한 콘텐츠를 말한다. 사용자가 상업적인 의도없이 제작한 콘텐츠를 온라인상으로 나타낸 것이다. 디지털 카메라와 개인 미디어의 발달로 최근에는 개인 블로그 뿐 아니라, 포탈에서 비즈니스 차원에서도 활성화 되고 있다.

3) Mashup

매쉬업(Mashup)이란 Open Source 기반의 공개된 웹서비스를 조합하여 만든 웹서비스나 공개 API를 제공하는 업체들에서 데이터를 받아 전혀 다른 새로운 서비스나 융합 애플리케이션을 만들어 내는 것을 말한다. 다수의 정보원으로부터 제공되는 콘텐츠를 조합하여 하나의 서비스로 제공하는 웹사이트 또는 애플리케이션. 팝 뮤직에서 처음 사용되기 시작한 매쉬업은 아티스트나 디스크자키가 두 곡 또는 그 이상의 곡을 섞어 하나의 곡으로 연주하는 것을 의미한다.

4) Crowdsourcing

대중(Crowd)+아웃소싱(outsourcing)의 합성어로 기업에서 협업 아웃소싱의 범위를 개인의 웹서비스에까지 활용하고자 하는 경향을 보이고 있다. 전세계적으로 1084개의 메쉬업이 존재하며, 하루에 약 2.71개의 매쉬업이 생성(2006년 10월 기준)되는 시장에서 개인블로그나 매쉬업은 기업의 새로운 롱테일 마케팅 대상으로 떠오르고 있다.

2.3 웹2.0 기반 웹서비스 요소기술

1) 플랫폼 - SOA, OpenAPI

SOA(Service Oriented Architecture)는 1996년 가트너 그룹에 의해 처음 소개된 개념으로 비즈니스와 사용자의 요구를 보다 쉽게 반영하기 위한 개발 플랫폼이다. 웹서비스는 SOA 개념을 구체화시킨 기술의 환경이다. SOA 개념을 소비자 중심으로 연결 시킨 Open API는 웹 서비스의 개방 지향적인 성격을 잘 나타내고 있는 개념으로, 기업의 API를 외부에 공개한 것으로 일반적으로 웹 서비스 형태로 공개한 것으로, 응용프로그램에서 사용할 수 있도록 운영 체제나 프로그래밍 언어가 제공하는 기능을 제어할 수 있도록 만든 인터페이스다. 2006년 5월, O'Reilly Radar의 컬럼 How To Roll Out An Open API에서 처음으로 등장했다.

2) 웹서비스 기본 구성요소

XML	문서 안에 데이터를 표현하는 마크업 언어
HTTP	클라이언트와 서버 사이에 정보를 전송하는 츠로토콜
SOAP, REST, XML-RPC	HTTP를 이용해서 XML 기반의 메시지를 교환하기 위한 프로토콜
WSDL	웹서비스와 통신할 때 필요한 정보를 담는 XML기반의 서비스 기술포맷
UDDI	웹서비스를 외부에 공개하여 다른 웹서비스의 접근을 허락하는 XML 기반의 레지스트리
Widjet	GUI 단위요소로, Ajax, RAD Toolkit, Dojo Toolkit 등 Javascript 라이브러리로 구성하여 강력한 기능 수행

4) 블로그들 사이의 통신을 위한 매커니즘으로, 어떤 블로거가 자신의 블로그에 다른 블로그의 글에 대한 새로운 코멘트나 감상문을 쓰고 이 두개의 블로그가 트랙백 프로토콜을 지원한다면, 코멘트를 단 블로거는 이 사실을 '트랙백 핑(TrackBack Ping)'을 통해 알릴 수 있다.

3. Ajax 작동원리

3.1 Ajax 개념

Ajax[5]는 'Asynchronous JavaScript and XML'의 줄임말로 JavaScrip를 이용해서 서버에 있는 XML 데이터를 비동기적으로 호출하여 활용하는 기술이다. JavaScrip가 주된 요소인 Ajax는 단순히 특정 '기술' 이상의 '기법'을 의미한다. 웹 어플리케이션이 웹 페이지에 대한 사용자 인터랙션을 효율적으로 처리할 수 있는 수단을 제공한다(사용자 인터랙션이 이루어질 때마다 페이지를 refresh 하거나 전체 페이지를 reload 하는 번거로움을 덜어준다). 이는 또한 데스크톱 어플리케이션 또는 플러그인 기반 어플리케이션의 경우와 유사하게 브라우저를 이용한 RIA(Rich Internet Application[6])를 구현해준다. Ajax 인터랙션은 백그라운드에서 비동기적으로 처리되고, 그 동안 사용자는 페이지에 대한 작업을 계속할 수 있다. 또한 Ajax 어플리케이션은 필요한 데이터만을 웹서버에 요청해서 받은 후 클라이언트에서 데이터에 대한 처리를 할 수 있다. 웹 서버에서 전적으로 처리되던 데이터 처리의 일부분이 클라이언트 쪽에서 처리되므로 웹 브라우저와 웹 서버 사이에 교환되는 데이터 량과 웹서버의 데이터 처리량도 줄어들고 어플리케이션의 응답성이 좋아진다. 다음은 Ajax의 구성요소다.

또한 Ajax 어플리케이션에 별도의 플러그인 소프트웨어나 하드웨어가 필요치 않으며, 플랫폼이나 브라우저[7]에 중립적 특성을 지니고 있다는 점이다. 클라이언트와 적용되는 기술로는 J2EE, .NET, PHP, Ruby, CGI Script 등과 함께 사용할 수 있으며, 개발 생산성을 향상시키 위한 Google AJAXSLT, sarrisa, JSON[8], Ajax.NET 등 브라우저와 서버 프레임워크를 활용할 수 있다.

HTML, CSS	Presentation, with standardized style information
XMLHttpRequest object	Asynchronously retrieves data from web server
DOM[9]	Dynamic display of and interaction with the HTML page
Javascript	Client-side code controls actions (controller pattern)

사실 아직까지 Ajax에 관하여 '뒤로가기'나 북마크 불가와 비동기 접속 및 소스코드 노출에 따른 보안 문제 등은 해결해야 할 과제이다.

3.2 Ajax 작동 절차

Ajax의 작동원리를 이해하기 위해 입력폼에서 자동으로 validate 체크하는 기능을 예로 살펴보도록 한다.

1) Client Event 발생

필드값이 변경되면 폼 엘리먼트가 validate() 함수 호출

```
Birth date: <input type="text" size="10" id="birthDate"
onchange="validate();"/>
```

2) XMLHttpRequest Object 생성

Validate function으로 XMLHttpRequest object 생성

5) 2005년, Adaptive Path의 Jess James Garrett가 에세이 'Ajax : A New Approach to web Application'에서 무거운 클라이언트/데스크톱/어플리케이션과 가벼운 클라이언트/웹/어플리케이션 사이의 간격을 줄이는 방법을 논의하면서 'Ajax'라는 용어를 처음 사용

6) 기존의 웹 어플리케이션 기술이 가진 평면적인 표현과 순차적인 프로세스를 다이나믹한 사용자 인터페이스와 데이터 베이스의 연동을 통해 저렴한 비용으로 하나의 인터페이스에서 모든 프로세스가 처리 가능 하도록 해주는 기술을 말한다. 이 용어는 현재 Adobe와 통합된 Macromedia의 2003년 백서에서 처음 사용된 용어로 특정 제품을 뜻하는 것이 아니라 풍부한 GUI(Graphic User interface)를 제공하는 애플리케이션을 정의하는 단어이다.

7) 예전 브라우저(Internet explorer 5, Safari 1.2, Mozilla 1.0 이전 버전) 사용시 오류 가능성 있음

8) JSON(Java Script Object Notation) : JavaScrip로 쉽게 읽을 수 있는 C언어 계열의 데이터 포맷 (XML 대안)으로 인코딩이 XML 보다 작아 반응속도를 향상시킨 데이터교환 프레임워크

9) Document Object Model : 플랫폼/언어 중립적으로 구조화된 문서를 표현하는 W3C 표준

```
var xmlHttp;
 function createXMLHttpRequest() {
        if (window.ActiveXObject) {
            xmlHttp = new
     ActiveXObject("Microsoft.XMLHTTP");}
        else if (window.XMLHttpRequest) {
            xmlHttp = new
     XMLHttpRequest(); }}

 function validate() {
       createXMLHttpRequest();
       var date =
     document.getElementById("birthDate");
       var url =
 "ValidationServlet?birthDate=" +
                          escape(date.value);
   xmlHttp.open("GET", url, true);
   xmlHttp.onreadystatechange = callback;
   xmlHttp.send(null);}
```

3) XMLHttpRequest object가 서버 호출

사전 정의된 URL에 의해 서버를 호출하게 함

```
xmlHttp.send()
xmlHttp.setRequestHeader("Content-Type",
"application/x-www-form-urlencoded");
```

4) 서버 요청 처리

서버는 XMLHttpRequest도 Http request와 동일하게 처리

```
public class ValidationServlet extends
HttpServlet {
  protected void
      doGet(HttpServletRequest                request,
HttpServletResponse response)
    throws ServletException, IOException {
    PrintWriter out = response.getWriter();
    boolean passed = validateDate(request.
            getParameter("birthDate"));
    response.setContentType("text/xml");
    response.setHeader("Cache-Control",
                       "no-cache");
     String message = "You have entered
                        an invalid date.";
   if (passed) {
      message = "You have entered a
                    valid date.";}
      out.println("<response>");
    out.println("<passed>" +
    Boolean.toString(passed) + "</passed>");
    out.println("<message>" + message +
                          "</message>");
    out.println("</response>");
    out.close(); }}
```

5) 서버의 문서 반환

Content-type 은 일반적으로 text/xml 로 지정

```
response.setContentType("text/xml");
response.setHeader("Cache-Control",
"no-cache");
```

6) XMLHttprequest Object가 callback() 호출

자바스크립트를 사용해서 DOM 문서를 조작 가능하며, 반환된 XML 문서에 대한 객체 구조를 xmlHttp.responseXML에 의해 접근

```
function callback() {
   if (xmlHttp.readyState == 4) {
      if (xmlHttp.status == 200) {
        var mes =
xmlHttp.responseXML.getElementsByTagName("message
")[0].firstChild.data;
        var val =
xmlHttp.responseXML.getElementsByTagName("passed")
[0].firstChild.data;
        setMessage(mes, val);
   }}}
```

7) 브라우저의 문서 갱신

innerHTML를 통해 문서 re-render

```
function setMessage(message, isValid) {
var messageArea =
    document.getElementById("dateMessage");
var fontColor = "red";
    if (isValid == "true") {
       fontColor = "green"; }
    messageArea.innerHTML =
     "<font color=" + fontColor + ">" +
       message + " </font>"; }
```

4. Ajax 응용분야

4.1 Ajax의 이점

1) 사용자 인터페이스 개선 - 콘텐츠 접근강화

콘텐츠 접근성을 강화하는 대표적인 응용분야는 검색어 자동완성(autoComplete) 기능이다. key in 하는 글자마다 완성된 글자를 입력 항목의 리스트로 생성한다. 이 기능은 콘텐츠를 보다 쉽고 편리하게 접근할 수 있도록 만들어 주는 가장 웹2.0에 맞는 기술이라 볼 수 있다. 게다가 한/영 키 전환을 잘못 사용하여 검색 창에 한글을 쓰듯이 영문을 쳤을 때 자동으로 영문을 한글 단어로 바꿔주는 기능은 혁신적인 사용자 인터페이스이자 동시에 콘텐츠 접근성을 강화하여 사용자의 참여를 유도하는 기발한 아이디어라 할 수 있다 이런 것이 웹2.0을 강화하는 Ajax의 진정한 사용 이점이 된다.

2) 소프트웨어 서비스 모델 개선 - 고성능의 웹 어플리케이션

구글은 엑셀과 같은 대표적인 데스크탑 어플리케이션을 웹에 서비스하고 있다. 웹에서 서비스되는 어플리케이션은 브라우저가 있는 모든 곳에서 최신 버전을 자유롭게 이용할 수 있는 장점이 있다. 이런 오피스 제품을 웹에서 이용할 수 있다면 작업내용을 온라인으로 공유할 수도 있을 것이다.

3) 개발 생산성 개선 - 위젯 컴포넌트와 서비스 지향 아키텍처

Ajax를 통해 JavaScript를 통한 RIA 구현에 대한 관심이 높아진 결과 많은 자바스크립트 UI 라이브러리가 생겨났다. Dojo나 Script.aculo.us, Rico, Yahoo UI 라이브러리 등은 쉽게 애니메이션 효과나 드래그 앤 드롭 같은 동적인 효과를 자바스크립트 함수로 제공하고 있다. 좀더 효율적인 개발 방법론으로는 데이터와 프리젠테이션 레이어와의 완벽한 분리를 궁극적인 목표로 삼을 수 있다. Ajax(프리젠테이션 레이어)는 XML(데이터)로 데이터를 받는다. 그러므로 Ajax를 이용한 XML 처리 레이아웃을 출력하는 고성능의 위젯이 다양하게 생산된다면 OpenAPI 같이 XML을 제공하는 서비스 지향 아키텍처와 맞물려 마치 레고블럭을 쌓듯이 다양한 서비스를 만들어 낼 수 있을 것이다.

4.2 Ajax의 활용분야

웹2.0 환경에서 Ajax의 활용으로는 실시간 데이터 검증 및 자동완성과 정렬, 세부데이터 자동 갱신, 서버 자동고지, 사용자 인터페이스 통제, 사용자 의사소통 개선 기능을 활용 할 수 있다. 이 Ajax의 기본 기능으로 DashBoard를구현해 보자. DashBoard는 야후에서 제공하는 뉴스, 주식시세, 날씨, 검색 기능을 OpenAPI로 구성한다.

1) 뉴스

DashBoard에 포함하는 뉴스 URL로는 야후에서 제공하는 RSS[10](http://rss.news.yahoo.com/rss/topstories)를 활용한다.

```
public Collection getNewsItems() {
Document newsXML = getNewsItemsXML();
return parseNewsItems(newsXML);
}
private InputStream
getNewsServiceInputStream() {
InputStream newsInputStream = null;
try {
HttpURLConnection con =
(HttpURLConnection)new
URL(NEWS_URL).openConnection();
con.setDoInput(true);
con.setDoOutput(true);
con.setRequestMethod("GET");
newsInputStream = con.getInputStream();}
catch(Exception e) {
System.out.println("Exception retrieving news: " +
e.toString());}
return newsInputStream;
}
private Document getNewsItemsXML() {
```

10) RSS(Really Simple syndication) : 웹 사이트의 제목이나 요약 등과 같은 메타 데이터를 구조화하여 보여주는 XML기반의 포맷

```
SAXBuilder builder = new SAXBuilder();
InputStream newsInputStream =
getNewsServiceInputStream();
Document news = null;
try {news = builder.build(newsInputStream);}
catch (JDOMException ex) {
System.out.println("Error: " + ex);}
catch (IOException ex) {
System.out.println("Error: ");}
return news;
}
private Collection parseNewsItems(Document news) {
Collection newsItems = new ArrayList();
Element root = news.getRootElement();
List items =
root.getChild("channel").getChildren("item");
Element item = null;
NewsItem newsItem = null;
for(Iterator it = items.iterator(); it.hasNext();) {item =
(Element)it.next();
newsItem = new NewsItem();
newsItem.setGuid(item.getChildText("guid"));
newsItem.setTitle(item.getChildText("title"));
newsItem.setLink(item.getChildText("link"));
newsItem.setDescription(item.getChildText("description")
);
newsItems.add(newsItem);}
return newsItems;}
```

2) 주식시세

(WSDL:www.swanandmokashi.com/HomPage/WebSErvices/StockQuotes.asmx?WSDL)

```
public GetQuotesResponse getStockQuotesFor
      (String ticker) {
  GetQuotes quotes = new GetQuotes(ticker);
  GetQuotesResponse quotesResponse = null;
 try {
     quotesResponse =
     getStockQuotesSoap().getStockQuotes(quotes);
        } catch(java.rmi.RemoteException ex) {
            // TODO handle remote exception
        }
  Quote[] quotesArray =
   quotesResponse.getGetQuotesResult().getQuote();
  Quote quote = null;
  for(int i = 0; i < quotesArray.length; i++) {
     quote = quotesArray[i]; }
 return quotesResponse;
}
private StockQuotesSoap getStockQuotesSoap() {
      StockQuotesSoap stockQuotesSoap = null;
 try {
     stockQuotesSoap = new
      StockQuotesLocator().getStockQuotesSoap();
     } catch (ServiceException ex) {
       ex.printStackTrace();}
 return stockQuotesSoap;}
```

3) 날씨

입력되어 있는 우편번호에 해당하는 지역의 날씨가 자동으로 갱신되면서 Display.

(WSDL:www.webservicex.net/WeatherForecast.asmx?WSDL)

```
public WeatherForecasts getForecastFor
      (String zipCode) {
      WeatherForecasts forecasts = null;
 try {
     forecasts = getWeatherForecastSoap()
.getWeatherByZipCode(zipCode);
     WeatherData[] dataArray =
          forecasts.getDetails().getWeatherData();
     WeatherData data = null;
          SimpleDateFormat parser = new
SimpleDateFormat("EEEE, MMMM dd, yyyy");
          SimpleDateFormat formatter = new
SimpleDateFormat("EEE. MM/dd");
   for(int i = 0; i < dataArray.length; i++) {
      data = dataArray[i];
      try {
data.setDay(formatter.format(parser.parse(data.getDay())));}
      catch(Exception e) {
System.out.println("\n\n\nParsingException:"+e);}}}
      catch(java.rmi.RemoteException ex) {
          // TODO handle remote exception}
        return forecasts;
}
private WeatherForecastSoap getWeatherForecastSoap() {
   WeatherForecastSoap weatherForecastSoap=null;
   try {
       weatherForecastSoap = new
WeatherForecastLocator().getWeatherForecastSoap();
        }
        catch (ServiceException ex) {
            ex.printStackTrace();}
 return weatherForecastSoap;}
```

4) 검색

검색조건에 대한 자동완성 기능을 수행하여 검색어 목록이 생성되고 보다 쉽게 검색을 도와주는 고성능의 검색기를 장착할 수 있다.

```
private static final String YAHOO_SEARCH_URL
="http://api.search.yahoo.com/WebSearchService/V1/we
bSearch" + "?appid=thunderboltsoftware"
            + "&type=all"
            + "&results=10"
            + "&query=";

private InputStream getResultsInputStream(String
searchTerm) throws Exception {
  String url = YAHOO_SEARCH_URL + searchTerm;
  System.out.println("url: " + url);
  HttpURLConnection con =
(HttpURLConnection)newURL(url).openConnection();
        con.setDoInput(true);
        con.setDoOutput(true);
        con.setRequestMethod("GET");
        return con.getInputStream();
}
public Collection search(String searchTerm) {
  Document resultsXML =
      getSearchResultsXML(searchTerm);
  return parseResults(resultsXML);
}
private Document getSearchResultsXML(String
searchTerm) {
    SAXBuilder builder = new SAXBuilder();
    Document news = null;
 try {
     InputStream searchResultsInputStream =
           getResultsInputStream(searchTerm);
     news=builder.build(searchResultsInputStream);}
  catch (JDOMException ex) {
     System.out.println("Error:"+ex.toString());}
  catch (IOException ex) {
     System.out.println("Error:"+ex.toString());}
  catch(Exception ex) {
     System.out.println("Exception:"+ex.toString());}
 return news;
}
private Collection parseResults(Document results)
{ Collection searchResults = new ArrayList();
   Namespace ns =
       Namespace.getNamespace("urn:yahoo:srch");
   Element root = results.getRootElement();
   List items = root.getChildren("Result", ns);
  System.out.println("res size:"+items.size());
  Element item = null;
  SearchResult result = null;
for(Iterator it = items.iterator(); it.hasNext();) {
item = (Element)it.next();
result = new SearchResult();
result.setTitle(item.getChildText("Title", ns));
result.setSummary(item.getChildText("Summary", ns));
result.setUrl(item.getChildText("Url", ns));
searchResults.add(result); }
return searchResults; }
```

5. 맺으며

웹2.0 환경에서 서비스 제공자는 모두에게 개방된 열린 공간을 제공하고, 이용자는 쉽고 빠르게 지식과 정보를 스스로 생산하고 공유하면서 이른바 '참여의 웹', '생활화된 웹' 등의 웹2.0 트렌드가 형성되었다. 본 사례와 같이 웹2.0 기술을 이용하여 쉽게 정보와 지식을 '생산'·'공유'·'소비'하는 열린 인터넷의 확장을 기대해 본다.

참 고 문 헌

[1] 김대현, "웹2.0 기술의 발전과 이를 활용한 당행 정보시스템 개발전략", 2007

[2] 윤희철, "Semantic Web", 2007

[3] 이윤복, "Ajax 이론", 2007

[4] 이종필 & 주현식, "Webservice", 2007

[5] 박지강, "당신은 웹2.0 개발자입니까?", 2007

[6] Ryan Asleson & Nathaniel T.Schutta, 박형일 역, "Foundations of Ajax", 2006(source download: http://www.apress.com/book/)

[7] Mochio Umeda, 이우광 역, "웹진화론(WEB Shinkaron)", 2006

[8] Hirosh Ogawa & Yasunari Goto, 권민 역, "WEB2.0 Innovation", 2006

[9] Mark Volkmann, "Asynchronous JavaScript and XML(Ajax)", 2006

[10] Philip McCarthy, "Ajax for Java developers:Build dynamic Java applications", 2005

[11] Greg Murray, "자바 개발자를 위한 Ajax FAQ"

[12] Greg Murray, "자바 기술을 이용한 Ajax의 활용"

◐ 저 자 소 개 ◑

양 희 정(Hee-Jung Yang)

1994년 숭실대학교 공과대학 전자계산학과 졸업 (공학사)

2004년 11월~현재 한국은행 전산정보국 품질관리반 조사역

E-mail: guibee@bok.or.kr