Open API와 Ajax를 이용한 다국어 메타검색 서비스의 모델링 및 구현†

(Modeling and Implementation of Multilingual Meta-search Service using Open APIs and Ajax)

김 선 진*, 강 신 재**
(Seon-Jin Kim, Sin-Jae Kang)

요 약 자바스크립트 기반의 Ajax는 ActiveX 기술의 대안으로 주목받고 있는데, 대부분의 웹 브라우저에서 기본으로 지원되고, 비동기 상호작용을 통한 빠른 속도와 뛰어난 트래픽 절감 효과, 그리고 화려한 인터페이스 등의 장점들로 인해, 현재 국내 대형 포털 사이트들에서도 이 기술을 접목시켜 기존의 서비스를 재오픈 하는 추세이기도 하다. 본 연구에서는 이러한 Ajax 기술과 국내외 주요 사이트에서 제공하는 오픈 API들을 이용하여 다국어 메타검색 시스템을 모델링하고 구현하였다. 사용자로부터 입력받은 한국어 질의어를 구글 번역 API를 이용하여 전 세계 54개국 언어 중 하나의 언어로 번역한 후, 대표적인 소셜 웹 사이트(Flickr, Youtube, Daum, Naver 등)의 정보를 통합 검색한다. 검색된 결과는 Ajax 기술을 통해 화면의 일부분만 동적 로딩하여 빠른 속도로 출력해주는 동시에, 불필요한 정보의 중복 전송을 방지하여 서버의 트래픽과 패킷당 통신 요금을 절감하는 효과를 가져왔다.

핵심주제어 : 오픈 API, Ajax, 교차언어 정보검색, 매쉬업

Abstract Ajax based on Java Script receives attention as an alternative to ActiveX technology. Most portal sites in korea show a tendency to reopen existing services by combining the technology, because it supports most web browsers, and has the advantages of such a brilliant interface, excellent speed, and traffic reduction through asynchronous interaction. This paper modeled and implemented a multilingual meta-search service using the Ajax and open APIs provided by international famous sites. First, a Korean query is translated into one of the language of 54 countries around the world by Google translation API, and then the translated result is used to search the information of the social web sites such as Flickr, Youtube, Daum, and Naver. Searched results are displayed fast by dynamic loading of portion of the screen using Ajax. Our system can reduce server traffic and per-packet communications charges by preventing redundant transmission of unnecessary information.

Key Words : Open API, Ajax, Cross-language Information Retrieval, Mashup

† 이 논문은 2006학년도 대구대학교 학술연구비 지원에 의한 논문임
* 대구대학교 컴퓨터·IT공학부 학사과정
** 대구대학교 컴퓨터·IT공학부 교수 (교신저자)

1. 서 론

최근의 웹은 '웹 2.0'이라는 개념이 등장하면서 새로운 변화를 가져왔다. 과거의 웹(또는 웹 1.0)이 일방적인 정보 제공의 형태였다면, 웹 2.0은 참여와 개방성을 통해 사용자들이 스스로 정보를 창조하고 공유하는 구조라 할 수 있다. 웹 서비스 사업자에 의존하던 방식에서 벗어나 사용자 스스로 콘텐츠에 꼬리표(tag)를 붙여 타인과 공유하게 되는 것이다. 이것은 대중들이 정보를 분류한다고 하여 'Folk'와 'Taxonomy'를 합쳐 'Folksonomy'라는 말을 만들어 내기도 했다.[1)]

사실 웹 2.0 개념이 등장하기 이전부터 개발자들은 사용자의 참여를 이끌어낸 다양한 아이디어와 기술들을 접목시켜 왔다. 유비쿼터스 환경의 증대와 뛰어난 인터넷 인프라는 사용자들을 온라인으로 참여하게 만들었고, 이를 통해 사용자들 스스로가 더 많은 것을 원하게 만들었던 것이다. XML, RSS, Ajax, 오픈 API와 같은 기술들이 현재 웹 2.0에서 대표적인 기술이라 할 수 있다.

국내외 유수 인터넷 기업들은 자사의 콘텐츠 검색 기술을 일반 사용자들도 쉽게 구현하고 사용할 수 있게 외부에 API 형태로 공개하기 시작했는데, 이를 '오픈 API'라고 한다. 이는 기존의 서비스가 좀 더 활성화됨으로써 기업 입장에서는 배너광고 수익을 창출할 수 있고, 다양한 플랫폼에서 자사의 콘텐츠를 사용하게 되어 정보의 유입 채널이 더욱더 풍부해질 뿐만 아니라, 개발자들이 스스로 공개된 API들을 조합해 새로운 매쉬업(mashup) 서비스를 창출해 낼 수 있게 한다[1]. 매쉬업도 대표적인 웹 2.0 기술 중 하나인데, 두 개 이상의 기술이나 서비스를 융합하여 새로운 서비스를 만들어 내는 것을 말한다[2].

머지않은 미래에 꼭 필요한 웹 서비스를 추정해 보면, 여러 나라의 언어로 작성된 방대한 양의 웹 데이터에서 사용자가 원하는 정보를 효율적으로 검색, 가공, 활용할 수 있는 서비스의 개발은 필연적이라고 할 수 있다. 따라서 국외 정보도 국내 정보처럼 쉽고 편리하게 검색할 수 있는 검색엔진에 대한 사용자들의 수요가 많다고 판단된다. 다국어 정보검색(Multilingual Information Retrieval)의 정의는 서로 다른 언어로 이루어진 정보들로부터 원하는 정보를 검색하는 것을 말한다. 이로써 사용자가 언어에 구애받지 않고 여러 언어의 문서를 검색해서 원하는 정보를 얻을 수 있다. 예를 들어 한국어 질의어로 검색하면 한국어 문서뿐만 아니라 일본어, 중국어, 영어 문서를 모두 사용자에게 제시해 줄 수 있도록 하는 것이다[3-4].

본 연구에서는 사용자의 모국어로 질의어를 입력하여 외국어로 되어 있는 정보를 통합 검색할 수 있는 다국어 메타검색 시스템을 모델링하고 구현하는 것을 목표로 한다.

구글 번역 API 서비스를 이용하여 다국어 번역 기능을 구현하고, 대표적인 웹 2.0 사이트를 통합 검색할 수 있는 메타 검색엔진을 구현하였다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구에 대해 살펴보고, 3장에서는 제안한 시스템을 설계하고 인터페이스를 제시한다. 마지막으로 4장에서는 결론 및 향후 과제에 대해서 기술한다.

2. 관련 연구

대표적인 인터넷 기업들은 검색, 쇼핑, 지도, 카페글 등 자신들의 정보 자산에 접속할 수 있는 오픈 API를 무상으로 제공하고 있다. 국외에서는 구글(Google), 야후(Yahoo), 아마존(Amazon), 이베이(eBAY) 등 수 많은 대규모 업체에서 웹(Web) API를 제공하고 있고, 국내에서는 네이버(Naver)와 다음(Daum)에서 제공하는 오픈 API가 가장 널리 사용된다. 대부분의 웹 API는 REST (Representational State Transfer) 또는 SOAP (Simple Object Access Protocol) 방식으로 제공되며, XML(eXtensible Markup Language)과 JSON(JavaScript Object Notation) 형식의 데이터를 전달한다[5].

Ajax는 2005년 2월 Jesse James Garrett에 의해 처음 사용되어, 많은 웹 개발자들의 주목을

1) http://en.wikipedia.org/wiki/Folksonomy

받았다[6]. 이는 새로운 기술이 아닌 다양한 기존 웹 기술들의 조합체로써, 비동기 통신을 통한 빠른 속도와 뛰어난 트래픽 절감 효과, 그리고 화려한 인터페이스 등의 장점을 지닌다. 또한 대부분의 웹 브라우저에서 기본으로 지원되기 때문에 별도의 ActiveX와 같은 플러그인 설치가 필요 없다.

일반 웹 어플리케이션 모델은 클라이언트가 요청을 보낸 후 서버는 그 응답을 처리하고, 요청에 대한 응답의 전체페이지 데이터를 전송하면, 클라이언트는 응답이 도착하기까지 다른 작업을 하지 않고 기다려야만 했다. 반면, Ajax는 XMLHttp Request를 이용하여 백그라운드에서 서버와의 통신을 수행하며, 요청에 대한 응답 XML 데이터를 전체페이지가 아닌 필요한 문서 속성의 데이터에 대해서만 동적으로 교체할 수 있다는 것이 가장 큰 차이점이다[7].

이러한 장점들로 인해 현재 국내 대형 포털 사이트들에서도 속속 이 기술을 접목시켜 기존의 서비스를 재오픈 하는 추세이다. 다음(Daum)의 한메일 Express를 시작으로 네이버(Naver) 메일, 네이버 지도, 네이트, 싸이월드 선물가게 등이 그 예이다. 지금은 거의 모든 사이트에서 보편화 된 '검색어 자동완성기능' 또한 대표적인 Ajax 기술이라고 할 수 있다.

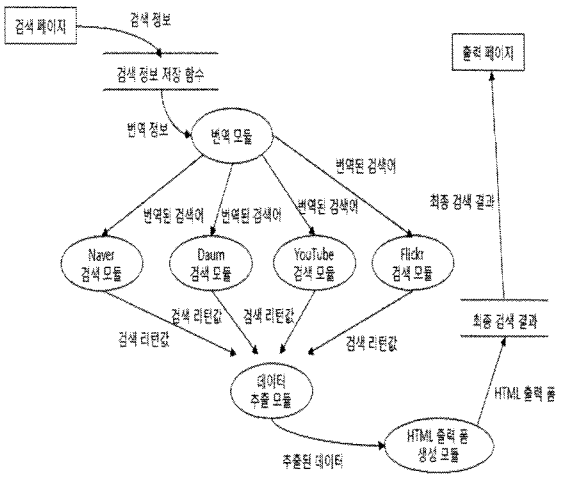
1994년 미국 라이코스 사가 최초의 상용 웹 검색 포탈 서비스를 시작한 이후로 Infoseek, Inktomi, Altavista, Teoma 등 여러 인터넷 비즈니스 회사에서 자사의 웹 사이트를 통해 웹 검색 서비스를 제공하였다. 이런 초기 서비스를 거쳐 현재는 Google, Yahoo, MSN 등의 포탈에서 제공하는 웹 검색 서비스를 통해 전 세계의 다양한 언어로 작성된 웹 페이지를 검색할 수 있다[8]. 이에 따라 방대한 양의 정보로부터 적합한 정보를 검색해 내는 문제는 중요한 문제로 부각되고 있는데[9], 이를 위해 그 동안 다양한 연구가 수행되어 왔다. 그 중, 메타 검색엔진은 다른 검색엔진으로부터 검색자의 질의어에 따른 검색 내용을 취합한 후 검색자에게 보여주기 때문에 검색자는 다양한 검색 결과를 얻을 수 있고, 기존의 검색엔진에서 질의어에 대한 결과를 종합하여 결과를 보여주기 때문에 내부적으로 데이터를 저장할 공간이 필요하지 않다는 장점이 있다[10].

다국어 정보검색을 위해서는 기본적으로 질의어 변환 기술이 요구된다. 이 과정에서 가장 어려운 문제는 질의어의 중의성 해소와 고유명사, 명사구, 미등록어의 변환이다[11]. 고품질의 질의어 변환을 위해서는 사전, 온톨로지, 병렬 코퍼스 등과 같은 자연언어 자원이 필요하다. 이러한 자연언어 자원을 대량 구축하려면 많은 비용이 들 뿐 아니라, 확보하기 어려운 자원을 필요로 하거나 특정 영역에서만 우수한 성능을 발휘하는 등의 단점도 있다[12]. 본 논문에서는 이러한 단점들을 극복하고, 비교적 손쉽게 다국어 정보검색을 구현하기 위해 오픈 API를 사용하였다.

3. 다국어 메타검색 서비스

이 장에서는 다국어 메타검색 시스템을 모델링하고 구현된 인터페이스를 제시한다.

그림 1은 본 시스템의 구성도이다. 사용자가 검색 페이지에서 질의어를 입력하면 검색 모듈을 통해 번역 및 검색을 수행하여 그 결과를 JSON이나 XML 형식으로 응답받아 데이터 추출 모듈로 보낸다. 데이터 추출 모듈은 응답받은 내용에서 출력에 필요한 요소들만을 추출하여 HTML 출력 폼 생성 모듈로 다시 넘겨주고, HTML 출력 폼 생성 모듈은 이 요소들에 HTML

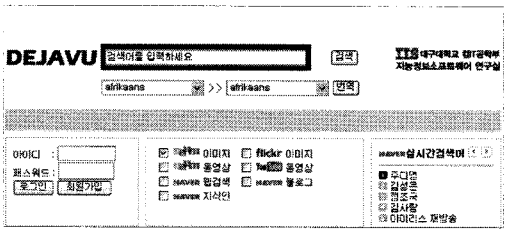


<그림 1> 시스템 구성도

태그를 입혀 최종 검색 결과를 생성한 뒤 출력 페이지로 전송한다. 마지막으로 출력 페이지는 최종 검색 결과를 Ajax 기술에 기반하여 동적으로 출력하게 되는데, 이 모든 과정은 Ajax 기법을 사용하지 않는 방법에 비해 매우 빠른 시간 안에 이루어질 수 있다.

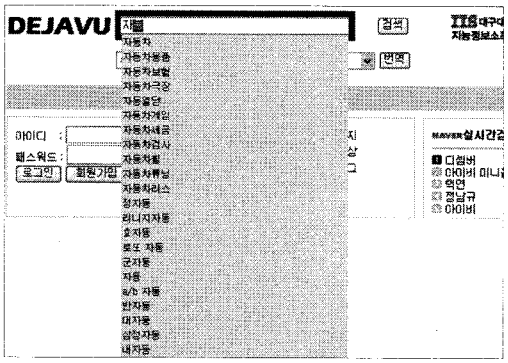
3.1 검색 인터페이스 구성

메타검색 인터페이스의 초기화면은 다음과 같이 구성된다.



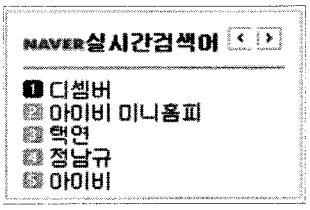
<그림 2> 검색 인터페이스

질의어 입력 시 유용한 기능으로는 네이버(Naver) 검색어 데이터베이스를 이용하여 Ajax를 이용해 검색어 자동완성 기능을 구현하였다. 그림 3은 '자동차'를 검색하기 위해 사용자가 '자동'까지 입력하면 그에 해당하는 추천 검색어들을 JSON 형식으로 받아와 자동으로 완성하여 보여줌으로써, 사용자가 질의어를 마우스로 편리하게 선택해 입력할 수 있도록 구현한 화면이다.



<그림 3> 검색어 자동완성 기능

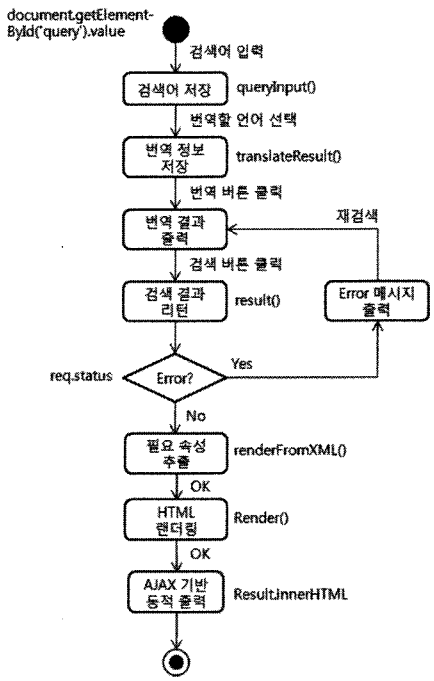
검색 화면의 우측 편에는 그림 4와 같이 실시간 급상승 검색어를 표시해줌으로써 사용자들이 최신 동향을 쉽게 파악할 수 있도록 하였다. 이 기능은 네이버(Naver) 급상승 검색어 오픈 API를 이용하여 구현하였다.



<그림 4> 실시간 검색어

3.2 번역 및 검색 모듈

질의어 입력 후 번역 및 검색 모듈은 다음과 같은 순서로 이루어진다.



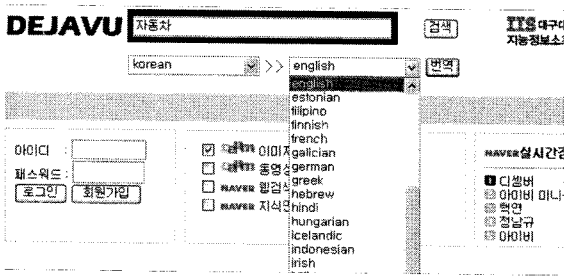
<그림 5> 번역 및 검색 모듈

번역된 질의대역어를 이용하여 대표적인 소셜 웹 사이트인 플리커(Flickr)[2]와 유튜브(YouTube)[3], 그리고 국내의 다음(Daum)[4]과 네이버(Naver)[5]

2) http://www.flickr.com
3) http://www.youtube.com

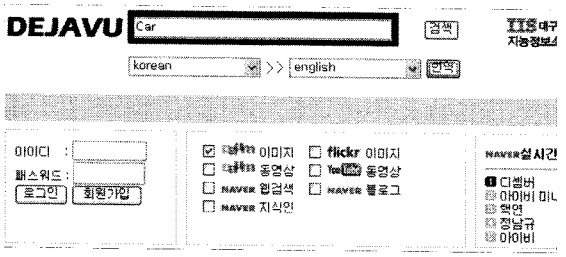
의 오픈 API를 이용하여 검색을 하게 된다.

다음은 영문사이트 검색을 위해 한국어 질의어를 영문으로 번역하는 과정을 보여준다. 한국어를 영어로 번역하기 위해 그림 6과 같이 번역 대상 언어를 영어로 설정하고 번역 버튼을 클릭한다. 만약 질의어의 대역어로 여러 개가 존재하는 경우에는 사용자가 선택할 수 있게 하였다.



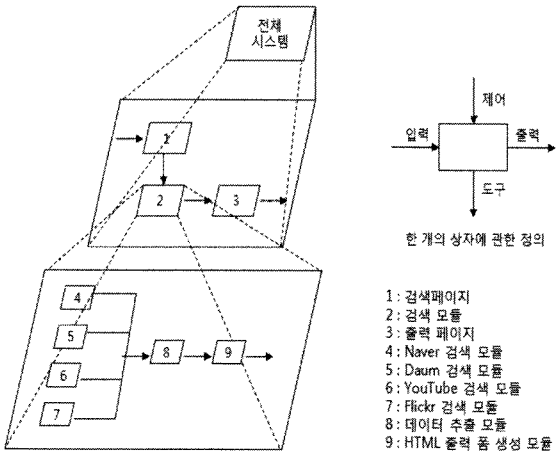
<그림 6> 번역 언어 설정 화면

본 시스템의 번역모듈은 구글 번역 오픈 API[6]를 이용해서 구현하였는데, 전 세계 54개국의 언어를 대상으로 상호 번역할 수 있고, 단어뿐만 아니라 간단한 문장까지도 번역이 가능하다. 번역이 완료된 화면은 다음과 같다.



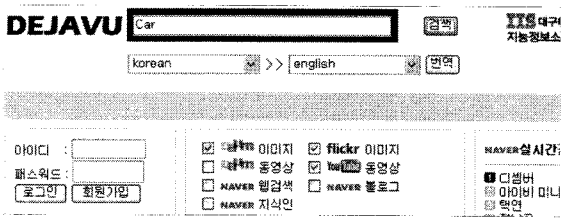
<그림 7> 번역이 완료된 화면

검색 모듈은 그림 8의 아래 부분 그림에서 보는 것과 같이 각 사이트별 검색 모듈이 따로 존재한다. 이는 각각의 API들이 요구하는 질의 경로와 파라미터, 오류메시지 등이 모두 상이하기 때문인데, 검색과 출력 및 랜더링 과정에 대한 Ajax 기반의 체계적인 제어와 관리를 위해서 각 모듈별로 설계, 구현한 것이다.



<그림 8> 검색 모듈 구조

사용자의 선택에 따라 검색할 사이트를 선택할 수 있게 하였는데, 그림 9와 같이 체크박스를 통해 검색하고자 하는 사이트를 선택한 뒤 검색 버튼을 클릭하면 된다.



<그림 9> 검색 사이트 선택 화면

검색 버튼이 클릭되면 미리 등록되어 있던 자바스크립트 이벤트 핸들러가 작동하게 되며, 질의어와 체크박스 정보를 인지하여 검색 과정을

```
NaverSearch.prototype = {
  init: function() {
    // '검색' 버튼을 눌렀을 때 검색어 입력 요소를 구함
    this.queryInput =
      document.getElementById(this.options.queryInput);

    // 검색 버튼을 구하고 이벤트 등록
    var searchBtn =
      document.getElementById(this.options.searchButton);

    // '검색' 버튼을 눌렀을 때 doClickOnSearchBtn 함수 호출
    ajax.Event.addListener(searchBtn, "click",
      ajax.Event.bindAsListener(this.doClickOnSearchBtn, this));
```

<그림 10> 검색 관련 이벤트 함수

4) http://www.daum.net
5) http://www.naver.com
6) http://code.google.com/intl/ko-KR/apis/ajaxlanguage/documentation/

순차적으로 수행하게 된다. 아래 그림 10은 이 과정을 자바스크립트로 코딩한 검색 이벤트 함수의 일부분이다.

3.3 데이터 추출 및 랜더링 모듈

검색 모듈은 JSON이나 XML 형태로 검색 결과를 리턴받는다. 아래 그림 11은 'Daum 이미지 검색 API'를 이용해 XML 형식으로 리턴받은 검색 결과의 한 예이다.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <channel>
    <title>Search image Daum Open API</title>
    <desc>Daum Open API search result</desc>
    <totalCount>34079</totalCount>
    <result>10</result>
    <sort>1</sort>
    <q>car</q>
    <pageno>1</pageno>
  - <item>
      <author>UyBo</author>
      <title>car</title>
      <link>http://blog.daum.net/tosselpha/5480502</link>
      <image />
      <thumbnail>http://image02.search.daum-img.net/03/2.c
      <width>940</width>
      <height>705</height>
      <pubDate>20090323232641</pubDate>
      <cp>7</cp>
    </item>
```

<그림 11> XML 형식으로 리턴받은 검색 결과

데이터 추출 모듈은 이 리턴값에서 제목, 썸네일 URL, 링크 주소, 요약 설명 등을 파싱하여 추출하고, 이렇게 추출된 정보를 랜더링 모듈이 HTML 태그를 입혀 최종 검색 결과를 생성한 후 출력 모듈로 내보낸다. 아래 그림 12는 이 과정을 자바스크립트로 코딩한 소스의 일부분이다.

```
renderFromDaumImgXML: function(xmlDoc) {
    this.result.innerHTML = "";
    var items = xmlDoc.getElementsByTagName("item");

    for (var i = 0 ; i < items.length ; i++) {
        var item = items.item(i);

        var titleNode = item.getElementsByTagName("title").item(0);
        var title = (titleNode.firstChild) ? titleNode.firstChild.nodeValue : "";

        var linkNode = item.getElementsByTagName("link").item(0);
        var link = (linkNode.firstChild) ? linkNode.firstChild.nodeValue : "";

        var descNode = item.getElementsByTagName("thumbnail").item(0);
        var desc = (descNode.firstChild) ? descNode.firstChild.nodeValue : "";

        var titleDiv = document.createElement("div");
        var titleHtml = '<hr/><p align="center"><img src="' + desc + '" width="120"
        titleDiv.className = "itemTitle";
        titleDiv.innerHTML = titleHtml;

        var descDiv = document.createElement("div");
        descDiv.className = "itemDesc";
        descDiv.innerHTML = '<p align="center"><a href="' + link + '" target="_new"

        this.result.appendChild(titleDiv);
        this.result.appendChild(descDiv);
    }

    var btnDiv = document.createElement("div");
    btnDiv.className = "itemBtn";
```

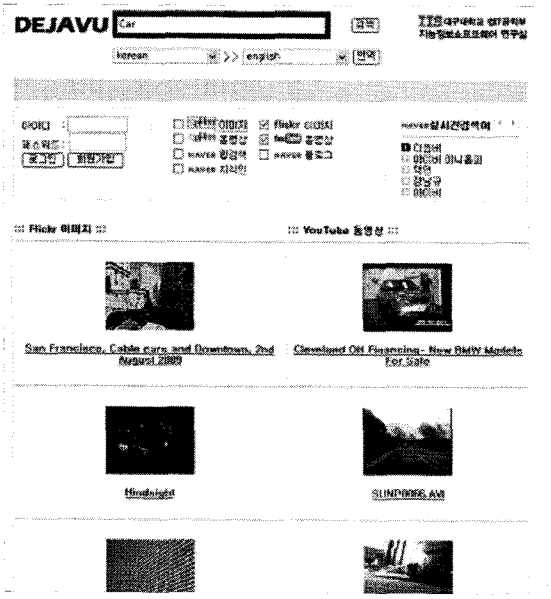
<그림 12> 데이터 추출 및 랜더링 모듈

3.4 출력 인터페이스 구성

출력 모듈은 랜더링 모듈로부터 받은 최종 검색 결과를 Ajax 모듈을 통해 그림 13과 같이 동적으로 출력하게 된다.

<그림 13> 3개 사이트 검색결과 예시

더 이상 필요 없어진 검색 결과는 체크박스를 해제함과 동시에 그림 14와 같이 화면에서 사라지게 되고, 동일한 질의어에 대한 다른 사이트의 검색 결과도 보고 싶을 경우에는 체크박스에 추가로 체크만 해주면 그림 15와 같이 현재 화면에 동적으로 추가된다.



<그림 14> Daum 이미지 검색결과 체크 해제

<그림 15> Daum 동영상 검색결과 체크 추가

위 과정 역시 자바스크립트 이벤트 핸들러에 의해 동작하며, 체크박스를 클릭하면 아래의 함수가 호출된다.

이 함수는 체크박스가 선택되었다면 해당 항목에 대한 startSearch() 함수를 실행해서 검색하도록 하고, 선택되지 않은 상태라면 closeResultView() 함수를 호출하여 해당 검색 영역을 화면에서 삭제한다. 따라서 체크박스를 선택하고 해제하는 행위만으로도 동적인 검색

```
doCheck: function(e) {
    var event = window.event || e;
    var target = ajax.Event.getTarget(event);
    var type = null;
    if (target == this.kinCheckBox) {
        type = 'kin';
        if (target.checked) {
            if (this.resultView[type] == null && this.searched) {
                this.startSearch(type);
            }
        } else {
            if (this.resultView[type] != null && this.searched) {
                this.closeResultView(type);
            }
        }
    } else if (target == this.blogCheckBox) {
        type = 'blog';
        if (target.checked) {
            if (this.resultView[type] == null && this.searched) {
                this.startSearch(type);
            }
        } else {
            if (this.resultView[type] != null && this.searched) {
                this.closeResultView(type);
```

<그림 16> doCheck() 함수

결과 화면 제어가 가능하다.

4. 결론 및 향후 연구과제

본 논문에서는 오픈 API와 Ajax를 이용하여 다국어 메타검색 서비스를 모델링하고 구현하였다.

웹 2.0 환경에서 무엇보다 중요한 것은 아직 구현되지 않는 새로운 서비스를 매쉬업을 이용한 독창적인 아이디어로 신속하게 구현하는 것이다. 이러한 측면에서 본 논문은 차세대 웹에서 반드시 필요한 다국어 메타검색 서비스를 단순히 모델링하여 아이디어만 제시한 것이 아니라, 오픈 API와 Ajax를 이용하여 직접 구현함으로써 그 가능성을 입증하였다.

구글 번역 오픈 API로 질의어를 번역한 후, 소셜 웹 사이트의 오픈 API들을 이용하여 검색을 수행하고, 검색된 결과는 JSON이나 XML 형태로 반환받아 추출과 랜더링 과정을 거쳐 완성된 HTML 결과값을 생성하고, 이는 Ajax 모듈을 통해 동적으로 화면에 출력되는 과정을 거친다.

향후 연구로는 Ajax 기술을 좀 더 극대화시켜 드래그 앤 드롭을 통해 검색 결과물을 사용자가 자유자재로 화면에 재배치하는 기능과, 개

인화 서비스 개념을 도입하여 각 사용자의 성향에 맞는 검색 결과를 제시해주는 모델을 구현하고자 한다. 이를 위해서는 사용자 검색 이력을 수집하여 분석한 뒤, 향후 검색 시 이를 참조하여 더 나은 양질의 검색 결과를 얻는 방안을 모색하고 있다.

참 고 문 헌

[1] 오창훈, "Open API를 활용한 매쉬업 가이드", 2009.

[2] Manes, Jack M., "Library 2.0 Theory: Web 2.0 and Its Implications for Libraries," Webology, Vol.3, No.2, June 2006.

[3] 황세찬 외, "Open API를 활용한 다국어 정보검색 시스템 모델링에 관한 연구", 한국산업정보학회 춘계학술대회 논문집, pp. 129-132, 2009.

[4] 최용석, 최기선, "과도한 지식을 요구하지 않는 공통기반축에 의한 용어 번역과 한영 교차정보검색에의 응용", 한국 인지과학회 논문지, 제14권, 제1호, pp. 29-40, 2003.

[5] Bizer, Christian, "The RDF Book Mashup : From Web APIs to a Web of Data", In SFSW, 2007.

[6] 권훈, 곽호영, "RSS와 Ajax를 이용한 맞춤형 실시간 정보 서비스 모델 설계", 한국콘텐츠학회 논문집, 제5권, 제1호, pp. 25-28, 2007.

[7] 유성수, 노봉남, "AJAX 기술과 보안", 한국정보과학회 논문집, 제33권, 제2호(C), pp. 621-625, 2006.

[8] 임성채, 안준선, "역파일에 기반한 웹 검색 엔진의 랭킹 시스템 구현", 한국정보과학회 논문집 Vol.34, No2(C), pp. 35-40, 2007.

[9] 이경하, 이규철, "키워드 질의를 이용한 순위화된 웹 서비스 검색 기법", 한국전자거래학회지 논문집, 제13권, 제2호, pp. 213-233, 2008.

[10] 이인근, 손세호, 권순학, "지식기반 의미 메타 검색엔진", 한국지능시스템학회 논문집, 제14권 제6호, pp. 737-744, 2004.

[11] D. A. Evans, "The state of the art in CLIR", Panel presentation, 9th Search Engines Conference, Netherlands, 2004.

[12] 최용석, 서충원, 신사임, 김재호, 최기선, "한영 질의어 변환을 위한 공통 중간개념 구축", 한국정보과학회 언어공학연구회 논문집, pp. 422-427, 2001.

김 선 진 (Seon-Jin Kim)

- 비회원
- 2004년~현재 : 대구대학교 컴퓨터·IT공학부 학사과정
- 관심분야 : Web2.0, Mashup, Ajax, Recommendation System, Semantic Web

강 신 재 (Sin-Jae Kang)

- 종신회원
- 1995년 : 경북대학교 컴퓨터공학과 (공학사)
- 1997년 : 포항공과대학교 (POSTECH) 컴퓨터공학과 (공학석사)
- 2002년 : 포항공과대학교(POSTECH) 컴퓨터공학과 (공학박사)
- 1997년~1998년 : SK Telecom 정보기술연구원 연구원
- 2007년 : 오스트리아 University of Innsbruck, DERI 연구소 방문교수
- 2002년~현재 : 대구대학교 컴퓨터·IT공학부 부교수
- 관심분야 : Semantic Web, Social Web, Ontology Matching, Natural Language Processing