범용 위치 기반 웹 서비스 시스템

김용욱, 이준우, 나연묵
단국대학교 전자컴퓨터공학과
e-mail : cncel81@gmail.com

Generalized Location-based Web Service System

KIM YongUk, Joon-Woo Lee, Yunmook Nah
Dept. of Electronics and Computer Engineering, Dankook University

요 약

위치 기반 웹 서비스 연구는 여러 단말기와 인터넷 포탈의 로컬 서비스에서 이루어져 왔다. 개별 단말기에서 이루어진 작업과 로컬 서비스의 구현은 한정적인 영역으로 제한되어 있다. 인터넷 포탈의 로컬 서비스는 지리정보를 인지할 수 없고 정보의 제공자와 유형이 한정적이다. 휴대 단말기를 위한 컨텐츠는 휴대 단말기를 위한 어플리케이션에 종속적이다. 포탈과 단말기의 제약을 넘어 범용적으로 활용하려면 우선 웹 서비스의 범위를 확장시켜 위치 기반의 컨텐츠를 처리할 수 있게 해야 한다. 본 논문은 범용화된 위치 기반 웹 컨텐츠로 확장을 위해 기반 기술을 정리하며 그 구조와 구현을 다룬다. 이 시스템을 통해 위치 기반 웹 서비스 시스템을 일반화 시킬 수 있다.

1. 서론

위치 기반 웹 서비스 연구는 휴대 단말기, 웹 브라우저, 휴대 전화 등의 다양한 응용에서 요구한다. 휴대 단말기는 GPS 혹은 GeoIP[1] 등의 기술을 이용하여 정보에 위치 정보를 붙이고 있다. 웹 브라우저는 RIA 기술을 통해 위치 정보를 서버에 전달한다[2]. 개별 단말기는 위치 기반의 컨텐츠를 생산하고 있다. 야후의 플리커[3] 서비스와 구글의 구글 어스[4] 서비스는 개별 단말기가 제공하는 컨텐츠의 위치 정보나 사용자의 GPS 나 IP 정보를 통해 정보를 관리한다.

여러 장비에 의해 발생된 지역 정보는 제한된 영역에 사용된다. 아이폰이 사진마다 첨부한 GPS 좌표는 플리커 서비스 외엔 범용으로 사용되기 어렵다. 음성이나 동영상에도 지역 정보는 첨부되어 있다. 멀티미디어 컨텐츠에 포함된 지역 정보 역시 특정 서비스나 단말기에서만 활용이 가능하다. 웹 문서가 어디에서 생성된 정보인지 확인을 하려면 웹 포탈에서는 네이버 지도[5]의 포스트 맵[6] 서비스, 로컬 서비스 등을 이용해야 한다. 그 밖의 서비스에서는 사용할 수 없다. 그나마 네이버의 포스트 맵은 사용자에 의해 생산된 정보를 제공한다. 나머지 로컬 서비스는 컨텐츠 프로바이더가 제공하는 정보만 응용할 수 있다.

모든 장비에서 사용할 수 있는 범용화된 위치 기반 웹 컨텐츠를 위한 서비스 시스템을 만드는 것이 본 논문의 목표이다. 범용화된 위치 기반 웹 컨텐츠는 컨텐츠와 컨텐츠의 메타데이터, 컨텐츠의 위치의 집합으로 확장되어야 한다. 전통적인 웹 컨텐츠는 구조적 문서와 메타데이터로 구성된다. XTHML / HTML [7][8] 등 의 규약으로 구조적 문서를 기록한다. 문서의 주소 등의 데이터는 메타데이터가 된다. 위치 기반 웹 컨텐츠에선 여기에 추가적으로 위치 정보가 포함해야 한다.

전통적인 웹 서비스는 주소를 통한 접근이 있다. 주소는 절대 혹은 상대 주소가 있다. 하이퍼링크를 통한 연결을 통하는 방법도 있다. 한 문서에 포함된 하이퍼링크를 통해 다른 문서로 접근한다. 검색 엔진을 통한 키워드 검색도 있다.

위치 기반 웹 컨텐츠는 컨텐츠와 그 컨텐츠의 위치를 표현하는 주소 이외에 위치 정보를 포함하고 있고 이 위치 정보를 이용하는 질의가 필요하다.

본 논문에서는 범용화된 위치 기반 웹 컨텐츠를 제공하기 위한 컨텐츠 컨테이너, 그 컨텐츠를 사용하기 위한 클라이언트, 위치 정보 기반 웹 컨텐츠를 질의하기 위한 기술을 다룬다. 본 논문의 구성은 다음과 같다. 서론에 이어 제 2 장에서 위치 기반 웹 서비스에 이뤄진 관련 연구를 소개한다. 제 3 장에서 위치 기반 웹 서비스를 위한 컴포넌트들을 상세히 다루며, 제 4 장에서 결론을 맺는다.

2. 관련연구

2.1 인터넷 포탈의 로컬 서비스

네이버, 다음 등의 인터넷 포탈은 로컬 서비스라는 위치 기반의 웹 컨텐츠를 제공한다. 포털의 로컬 서비스는 컨텐츠 프로바이어가 제공하는 웹 컨텐츠를 자사가 제공하는 지도 서비스에 통합하여 특정 지역에 관한 키워드 질의를 사용자가 요청했을 경우에 로컬 서비스에서 검색하여 그 지역의 지도와 함께 결과

This work was supported by the Korea Science and Engineering Foundation (KOSEF) grant funded by the Korea government (MOST) (No. R01-2007-000-20958-0).

를 제공한다. 출력된 결과물은 지도와 인근 업체명과 전화번호로 한정되며 하이퍼미디어 문서가 아니다. 로컬 서비스에 위치 정보와 업체명 전화번호를 제공하는 것은 컨텐츠 프로바이더에 의해 유상으로 이루어지며 하이퍼미디어 문서가 아니기 때문에 한정적인 분량과 일정한 양식의 컨텐츠를 제공한다.

제한된 컨텐츠 프로바이더에서 벗어난 포탈 서비스도 일부 존재한다. 네이버 포스트 맵 서비스는 지도 위에 컨텐츠를 포스트 잇의 형태로 제공하는 서비스다. 기존의 로컬 서비스가 컨텐츠 프로바이더가 제공하는 컨텐츠 만을 제공했던 것에 비해 네이버 계정을 가진 사용자가 자유롭게 추가할 수 있다. 하지만 네이버 포스트 맵 이외의 컨텐츠는 열람되지 않으며 포스트 맵의 데이터는 단순한 텍스트로 되어 있다는 점에서 인터넷 포탈의 로컬 서비스의 한계는 명확하다.

2.2 Geolocation API Specification

W3C 의 Gelocation API Specication[9]은 위치 정보를 전달하기 위한 표준 API 규약이다. 규약은 navigator 객체와 point 객체를 포함하고 있다. navigator 객체는 몇 개의 메서드를 가지고 있다. 각 메서드들은 콜백 함수를 받고 위치정보를 point 객체에 담아 콜백 함수에 전달한다

point 객체는 위치 정보와 측정한 시간을 함께 담고 있다. 위치 정보는 시간에 따라 변하는 요소이기 때문에 어느 시기에 어느 위치인지가 중요한 변수가 된다. 위치 정보는 WGS84[10] 규약에 맞추어 실수형 자료로 저장하며 위도, 경도, 높이 등의 정보와 함께 방향과 속도 정보를 포함한다.

Geolocation 을 지원하는 도구로서 파이어폭스 지오드[11]와 구글 기어스[12]를 예로 들 수 있다. 지오드는 무선 랜을 통해 근접한 무선 네트워크 신호를 파악하여 사용자의 PC 의 위치를 역 추적한다. 구글 기어스는 이 방법 이외에 IP 나 GPS 를 통한 위치 추적을 지원한다. 지오드와 기어스는 추적한 위치를 Geolocation API Specification 으로 사용할 수 있도록 제공한다.

Geolocation API Spec 을 쓰는 도구는 여러 웹 어플리케이션과 결합한다. 구글 맵스의 마이 맵스, 라스트미니트, 럼블 등의 서비스는 지오드나 기어스의 위치 정보를 적극적으로 활용한다. 마이 맵스에 접속하면 사용자의 위치의 지도가 보인다. 라스트미니트[13]는 가장 가까운 음식점을 안내한다. 럼블[14]은 소셜 네트워크 서비스로 친구가 추천한 인근 지역의 컨텐츠를 소개한다.

Geolocation API Specification 은 자바스크립트의 사용이 필수적이다. 웹 컨텐츠는 XHTML / HTML 과 CSS[15] 만으로 구성이 가능하다. 텍스트 위주의 웹 컨텐츠가 자바스크립트 코드를 통해 RIA 를 갖는 것은 비효율적이며 불합리하다. Geolocation API Specification 은 웹 어플리케이션을 위해 위치 정보 시스템을 확장하지만 웹 컨텐츠 전반에 걸쳐 지원하기는 힘든 특성이 있다.

2.3 야후 파이어이글

야후 파이어이글[16]은 위치 정보 서비스를 위한 플랫폼이다. 여러 단말기에서 사용자의 위치 정보를 갱신하고 수신할 수 있는 API 를 제공하며 통합적인 위치 정보 관리 센터를 제공한다. 사용자의 휴대 전화가 전파에 의해 측정한 좌표는 야후 파이어이글의 서버에 전송되며 사용자가 가진 노트북에서 그 좌표를 활용할 수 있다. 한 장비의 사용을 중단했을 경우 다른 장비를 통해 정보를 갱신한다. 사용자가 휴대전화를 끊고 GPS 가 달린 자전거를 탈 경우 자전거가 위치 정보를 파이어이글을 통해 전달 할 수 있다.

서버에 전송된 위치 정보는 다양한 어플리케이션에서 활용할 수 있다. 사용자가 허가한 어플리케이션에서 파이어이글 서버에 접속해 정보를 질의한다. 휴대전화를 들고 달리는 사용자의 위치를 사용자의 운동량을 관리해주는 나이키 플러스 사이트에서 관리할 수 있다.

3. 범용 위치 기반 웹 서비스 시스템

범용 위치 기반 웹 서비스 시스템은 두 가지 상황을 처리하는 것을 우선했다.

표 1. 범용 위치 기반 웹 서비스의 상황

1. 사용자가 URI 를 접근 했을 때
2. 문서 내의 하이퍼링크를 참조 할 때

사용자가 해당 URI 로 직접 접근할 때 그 컨텐츠가 어떤 위치 정보와 연관이 있는지 웹 컨텐츠와 함께 지도를 보여준다. 문서 내에 포함된 하이퍼링크의 문서가 위치 정보를 포함하는 문서인 경우 하이퍼링크에 대해서도 위치 정보에 맞추어 컨텐츠와 함께 지도를 제공한다.

3.1 시스템의 구조

다음은 범용 위치 기반 웹 서비스 시스템의 구조이다.

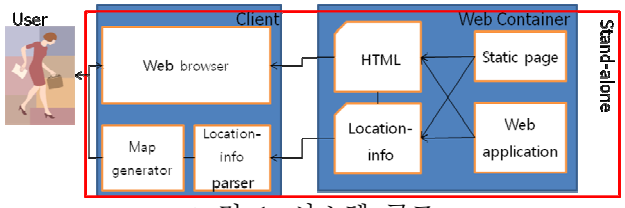

그림 1. 시스템 구조

범용 위치 기반 웹 서비스 시스템은 크게 유저, 클라이언트, 웹 컨테이너로 구성된다. 웹 컨테이너는 웹 서버를 확장시킨 형태로 클라이언트에게 제공할 컨텐츠는 구조적인 문서와 위치 정보이다. 구조적인 문서는 XHTML / HTML 문서와 CSS 문서로 구성된다. 구조적인 문서와 위치 정보는 생성 방법에 따라 두 가지 모듈에 의존하게 되는데 스테틱 페이지를 위한 모듈과 웹 어플리케이션을 위한 모듈에 의존된다. 정적

페이지는 가공의 작업을 거의 거치지 않은 채 출력되며 단순한 웹 서버 작업이면 충분하다. 구조적 문서와 지역 정보를 정적 파일로 보관하고 클라이언트의 요청에 맞추어 제공하면 된다.

클라이언트의 구조는 웹 브라우저와 맵 제너레이터, 지역 정보 파서로 나눈다. 구조적인 문서의 변경은 최소한으로 한정했기 때문에 전통적인 웹 브라우저가 해석하는데 문제가 없다. 구조적인 문서는 기존의 웹 브라우저가 해석하며 지역정보에 기반한 문서를 맵 제너레이터와 지역 정보 파서가 다룬다. 지역 정보 파서가 지역 정보 파일을 해석한 후 맵 제너레이터에게 전달한다. 맵 제너레이터는 해석된 정보에 맞추어서 지도를 생성하여 지도와 웹 컨텐츠를 함께 사용자에게 전달한다.

원칙적인 구조는 위와 같지만 웹 브라우저와 지역 정보 파서, 맵 제너레이터는 완벽하게 분리 될 수 없는 상황도 있다. 사용자에게 웹 컨텐츠와 지도로 표현되는 컨텐츠가 통합이 되어 나타나는 것이 더 효과적인 상황이 있다. 이런 경우에는 완전히 다른 층으로 구현하는 것 보다 유연하게 대처하는 것이 더 나았다.

3.2 웹 컨테이너

웹 컨테이너의 구조를 자세하게 보면 다음과 같다.

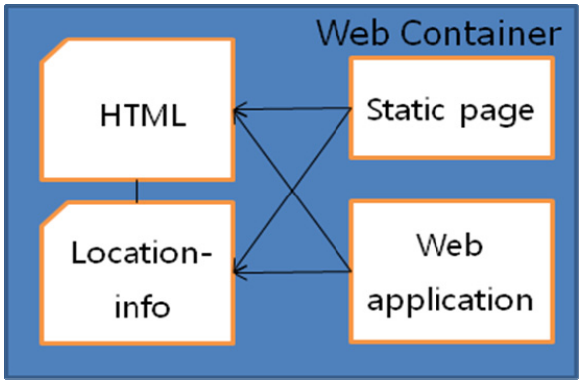

그림 2. 웹 컨테이너 구조

웹 컨테이너는 위치 기반의 웹 컨텐츠를 클라이언트에게 제공하는 모듈이다. 이 모듈은 정적인 페이지와 동적인 페이지에 대응하는 부분이 별개로 필요하다. 두 유형의 데이터는 별도의 방법으로 다루어진다. 정적인 페이지의 처리는 간단하다. 웹 서버가 정적인 문서 제공하는 방법과 동일하게 처리할 수 있다.

지역 정보 문서를 어떻게 구조적인 문서에 포함시키 위해 link 마크업 태그를 다음과 같이 확장해야 한다[17].

표 2. link 마크업 태그로 지리정보 추가

```
<link href="http://your.domain/foo.geo" rel="self"
type="application/geoweb" />
```

link 마크업 태그는 구조적 문서에 다른 문서를 첨부하기 위해 사용되는 XHTML / HTML 마크업 태그이다. 이 마크업 태그를 지역 정보 문서를 첨부하는데 확장하여 사용한다. 지역정보를 위한 문서의 타입을 "application/geoweb"으로 정하였다.

웹 컨텐츠에 대해 다음의 작업을 수행하여 지역정보 문서를 얻는다.

표 3. link 마크업 지리 정보 찾기

```
find_locations: function(uri)
   tags = parse_uri (uri)
   for each t = tags do
      if is_link(t) then
         if is_location_info(t) then
            add_vector(v, find_location(t))
         end if
      end if
   next t
   return v

find_location: function(t)
   i = itertator(t)
   set_namespace('georss')
   match(i, 'where')
   set_namespace('gml');
   match(i, 'point')
   return get(i,'pos')
```

find_location 을 통해 해석되는 지역정보는 다음의 형식이다.

표 4. 지역정보 문서

```
<georss:where>
   <gml:Point>
      <gml:pos>45.256 -71.92</gml:pos>
   </gml:Point>
</georss:where>
```

정적 페이지는 파일 포맷이 달라지는 점을 제외하면 처리 과정의 변화가 거의 없다. 반면에 동적 페이지의 경우에는 웹 어플리케이션에서 데이타베이스에 저장했던 자료에 맞추어 구조적 문서와 지역 정보를 동적으로 생성해야 한다. 데이터베이스 스키마에는 지역 정보를 위한 WGS 필드가 추가된다.

동적 페이지를 위해 웹 컨테이너가 응답하는 순서는 다음과 같다. 클라이언트의 URI 입력을 받는다. Rewrite 모듈을 이용하여 웹 어플리케이션의 콘트롤러로 정보를 보낸다. 해당되는 동적 페이지가 존재하는 패턴이면 웹 컨테이너는 데이터베이스에서 컨텐츠와 지역정보를 가져와서 반환한다.

표 5. 동적 페이지를 위한 응답

```
process_dynamic_geoweb: function(uri, post_data)
   callback = do_rewrite_mudule(uri, post_data)
   do_serve(callback, uri, post_data)

callback_proto: function(uri, post_data)
   action = get_action(uri);
   if exist(action) do
      db = get_database(action)
```

```
    end if
    if exist_location_info(action, db) do
        li = get_location_info(database)
        do(action, li, database)
    else
        do(action, database)
    end if
```

3.3 웹 클라이언트

웹 클라이언트의 구조는 다음과 같다.

그림 3. 클라이언트 구조

웹 클라이언트는 모질라 파이어폭스를 확장하여 구현하였다. 맵 생성기와 위치 정보 파서의 모듈은 C 언어와 자바스크립트 언어를 사용하였으며 전체적인 UI는 XUL 언어를 통해 구현하였다.

컨테이너가 제공하는 구조적 문서와 위치 정보를 웹 클라이언트는 수신하는데 구조적 문서는 웹 브라우저가 파싱해서 사용자에게 제공한다. 위치 정보는 위치 정보 파서가 해석한 후 맵 생성기에 전달한다. 파싱은 구조적 문서에서 link 마크업 태그를 먼저 정규식으로 검색한 후 해당 문서를 읽어와서 진행했다. 해당 문서는 GeoRSS 문서의 형태로 되어 있으며 이 문서에서 위치 정보의 좌표를 해석한다.

해석한 위치 정보를 기준으로 맵 생성을 하는데 다음, 구글, 네이버에서 제공하는 지도의 Open API 를 사용하여 지도 정보를 생성하였다. 생성된 정보는 모질라 파이어 폭스가 XUL 인터페이스를 통해 사용자에게 보여준다.

표 6. 웹 클라이언트의 동작

```
client: function(uri)
    c = get_content(uri)
    if (exist_location_info(c)) then
        m = generate_map(get_location_info(c))
        append_content(m);
    end if
    for each t=tags do
        if (match(t, 'a')) then
            if (exist_location_info(t)) then
                m = generate_map(get_location_info(t))
                append_popup(t, m)
            end if
        end if
    next t
```

3.4 구현

메모리 1 기가 바이트의 펜티엄 4 윈도우 XP 환경에 클라이언트를 위치했다. 메모리 1 기가 바이트, 펜티엄 4 듀얼 프로세스, 우분투 리눅스 환경의 컴퓨터에 웹 컨테이너를 배치했다. 웹 컨테이너 제작을 위해 아파치 2 서버, rewrite 모듈, php 모듈 등의 도구와 PHP 언어를 사용했고 클라이언트는 모질라 파이어폭스와 XUL 기술, 자바스크립트 언어를 사용하였다. 사용한 기술과 도구는 플랫폼 독립적이다. 서로 다른 사양과 환경의 컴퓨터 환경이지만 다른 환경에서도 동일하게 작업을 할 수 있거나 최소한의 수정으로 작업할 수 있을 것이라 추정한다.

4. 결론

기존의 포탈 기반의 서비스와 모바일 서비스들이 범용적인 위치 기반의 컨텐츠를 제공하기 어려웠기 때문에 본 연구에서 웹 컨테이너와 웹 클라이언트를 확장한 구조를 제안하고 구현하였다. 본 논문의 구현을 이용하여 지역에 기반한 컨텐츠를 효과적으로 제공할 수 있다.

본 논문에서 제안한 구조는 모질라의 파이어폭스에서 구현되었지만 다른 웹 브라우저에서도 성공적으로 구현하여 위치 기반의 웹 서비스를 제공할 수 있고 사용자 질의 만족도를 향상 시킬 수 있다.

웹 컨테이너와 웹 클라이언트 구조이기 때문에 지역에 관한 질의와 인근에 위치한 컨텐츠를 찾기 힘든 점이 문제가 있다. 현재 이 문제를 해결하기 위해 웹 컨테이너, 웹 클라이언트 구조에 검색엔진을 추가하는 연구를 진행 중이다.

참고문헌

[1] GeoIP, http://www.maxmind.com/app/ip-location
[2] Firefox 3.1 for developers, https://developer.mozilla.org/Ko/Firefox_3.1_for_developers
[3] Flikr, http://www.flickr.com/
[4] Google 어스, http://earth.google.com
[5] 네이버 지도, http://maps.naver.com/
[6] 네이버 포스트 맵, http://map.naver.com/postmap/
[7] XHTML Specification, http://www.w3.org/TR/xhtml11/
[8] HTML 4.01 Specification, http://www.w3.org/TR/REC-html40/
[9] Geolocation API Specification, http://dev.w3.org/geo/api/spec-source.html
[10] US National Imagery and Mapping Agency, "National Imagery and Mapping Agency Technical Report 8350.2, Third Edition," 1994
[11] Mozilla Firefox Geode, http://labs.mozilla.com/2008/10/introducing-geode/
[12] Google Gears, http://gears.google.com/
[13] LastMinute.com, http://www.lastminute.com/
[14] Rumble, http://www.rummble.com/
[15] CSS2 Specification, http://www.w3.org/TR/CSS2/
[16] Yahoo FireEagle, http://fireeagle.yahoo.net
[17] 김용욱 외, "GeoRSS 와 핑백을 이용한 지오웹 컨텐츠 서비스 시스템," 한국정보처리학회, 2008