Journal of KIIT. Vol. 17, No. 5, pp. 1-12, May 31, 2019. pISSN 1598-8619, eISSN 2093-7571
http://dx.doi.org/10.14801/jkiit.2019.17.5.1

위치기반 시맨틱 검색을 위한 반응형 웹 애플리케이션 구현

이수형*, 이용주**

Implementation of Responsive Web Application for Location-based Semantic Search

Suhyoung Lee*, Yongju Lee**

이 논문은 2016년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임 (No. 2016R1D1B02008553).

요 약

기존의 Open API들과는 달리 링크드 데이터는 웹 자체를 하나의 거대한 지식베이스로 만들어 수준 높은 SPARQL 질의를 수행할 수 있으며, 다양한 데이터셋으로부터 서로 다른 정보들을 매쉬업하여 하나의 새로운 콘텐츠를 효율적으로 생성할 수 있다. 본 논문에서는 위치 기반 시맨틱 검색을 위한 반응형 웹 애플리케이션을 구현하였다. 링크드 데이터인 DBpedia와 Google에서 제공하는 GoogleMap API를 매쉬업하고 검색된 개체들에 대한 세부적인 정보를 확인할 수 있는 시맨틱 브라우저 기능을 제공한다. 본 시스템은 반응형 웹 디자인 사상을 적용하여 PC 및 모바일 등 다양한 접속 환경에서 동일하게 사용할 수 있다. 본 논문에서 구현된 시스템은 기능이 유사한 현존 시스템들과 기능 스펙을 비교하였다. 비교 결과 시맨틱 사용, 링크드 기반 브라우저, 매쉬업 기능 등 다양한 측면에서 본 시스템의 우수성을 보여준다.

Abstract

Unlike existing Open APIs, Linked Data are made as a huge intelligent base to perform high-level SPARQL queries, and it is possible to create efficiently a new content by mashuping different information from various datasets. This paper implements a responsive web application for location-based semantic search. We mashup DBpedia, a kind of Linked Data, and GoogleMap API provided by Google, and provide a semantic browser function to confirm detail information regarding retrieved objects. Our system can be used in various access environments such as PC and mobile by applying responsive web design idea. The system implemented in this paper compares functional specifications with existing systems with similar functions. The comparison results show the superiority of our system in various aspects such as using semantic, linked-based browser, and mashup function.

Keywords

linked data, responsive web application, location-based semantic search, Google map, Node.js, SPARQL

* 경북대학교 IT대학 컴퓨터학부
- ORCID: https://orcid.org/0000-0002-6584-6328
** 경북대학교 IT대학 컴퓨터학부 교수(교신저자)
- ORCID: https://orcid.org/0000-0002-1705-4967

· Received: Jan. 27, 2019, Revised: Mar. 27, 2019, Accepted: Mar. 30, 2019
· Corresponding Author: Yongju Lee
School of Computer Science and Engineering, Kyungpook National University, 80, Daehak-ro, Buk-gu, Daegu 702-701, Korea,
Tel.: +82-53-950-7285, Email: yongju@knu.ac.kr

Ⅰ. 서 론

최근 정부 각 부처에서는 그들이 보유한 데이터의 효율적인 공개와 재사용 증진을 꾀하고 있으며, 정부 3.0 공공정보 개방운동의 일환으로 Open API 구축을 활발히 추진하고 있다[1]. 공공기관 외에도 구글, 야후, 이베이, 아마존과 같은 기업들 역시 몇 년 전부터 웹 2.0의 대표적 산물인 Open API를 이용하여 그들의 자원 및 서비스를 외부로 공개하기 시작하였다. 이러한 Open API들은 두 가지 이상의 서로 다른 자원을 섞어서 완전히 새로운 가치의 컨텐츠를 만드는 매쉬업(Mashup) 개발을 촉진시켰으나[2], Open API는 다음과 같은 문제점을 가지고 있다.

첫째, Open API는 REST, SOAP, JavaScript, XML-RPC 등 다양한 프로토콜로 서비스가 제공되고 있으며, 각 API들은 우리가 알고 있는 표준을 따르지 않고 다른 엑세스 메커니즘에 의존하고 있을 뿐만 아니라 서로 다른 포맷으로 데이터를 제공하고 있다[3]. 둘째, Open API는 이들을 구성하고 있는 각 아이템들에게 글로벌하고 유일한 ID (identification)를 할당하지 못함에 따라 API들 사이에 데이터 링크를 설정할 수가 없다. 셋째, Open API는 중복된 데이터들을 분리된 사일로(silos) 속으로 방치할 수밖에 없었으며, 매쉬업 개발자들은 그들의 애플리케이션 개발을 위해 특정 Open API에 접속하여 그들의 요구에 맞는 메소드를 찾아 선택해야 한다[4]. 넷째, Open API를 통한 서비스 방식은 연결된 서비스가 아니라 고립된 개별적 서비스였고, 몇 개의 API 기능에 대한 매쉬업이지 데이터 자체가 연결된 완전히 통합된 모습은 가질 수가 없었다.

본 연구에서는 이러한 Open API의 단점을 극복하기 위해 웹 상에 존재하는 데이터 사이의 의미적 연결을 통해 데이터를 구조화하는 링크드 데이터(Linked Data) 활용 전략을 제안한다. 현재 링크드 데이터에 대한 핵심 이슈는 링크드 데이터를 기반으로 한 응용 애플리케이션의 개발이다[5]. 하지만 링크드 데이터 구축에 관한 국내 연구는 최근 몇 개 수행된 적은 있으나[6]-[8] 이를 활용한 링크드 기반 시맨틱 웹 애플리케이션 개발에 관한 논문들은 거의 찾아보기 힘든 상황이다. 링크드 데이터 형식을 이용하여 정보를 서로 연결함으로써 얻을 수 있는 장점은 다음과 같으며, 이는 참고문헌 [9]를 참조하여 재정리한 것이다.

- 데이터의 재사용이 가능하다. 내가 만든 데이터가 아니더라도 외부에 개방된 링크드 데이터를 연결하면 웹을 하나의 거대한 지식베이스처럼 사용할 수 있다.
- 데이터의 중복을 감소할 수 있다. 링크드 데이터를 통해 공개된 데이터를 이용하면 자신이 원하는 데이터가 이미 존재하는지, 있다면 어디에 존재하는지를 알 수 있으므로 시스템의 사일로 문제에 의해 발생될 수 있는 불필요한 데이터 중복 문제를 해결할 수 있다.
- 데이터의 상호운용성을 극대화할 수 있다. 웹 표준인 RDF(Resource Description Framework) 형태로 발행되므로 마치 웹을 하나의 거대한 글로벌 데이터베이스처럼 질의하고 이용할 수 있다. 이를 통해 상호운용성을 높이고 데이터 통합을 쉽게 할 수 있다.
- URI, RDF, SPARQL 등 표준에 의존한다. 실제 RDF에 대해서는 SPARQL로 질의하는데, 질의를 할 수 있는 곳(즉, Endpoint)에 대한 네임스페이스만 언급해 주면 한꺼번에 여러 곳에 동시에 질의를 보낼 수 있다. 또한 이를 응용 프로그램에서 이용할 수 있으므로 Open API에 비해 데이터 접근을 더욱 구체화할 수 있어 데이터 지향의 매쉬업을 할 수 있다.
- 데이터 네트워크화가 가속화된다. 데이터 링크를 통해 런타임으로 새로운 데이터의 발견이 가능하고, 웹 상에 새로운 데이터 소스가 계속 생산됨에 따라 데이터들이 새로운 가치를 더할 수 있다. 즉, 정보를 개방해서 연결하면 할수록 그만큼 네트워크 효과는 더 증대한다.

이러한 목표를 달성하기 위해 본 논문은 다음과 같이 구성되어 있다. 2장에서 관련 연구를 살펴보고, 3장에서 링크드 데이터 기반 반응형 웹 애플리케이션을 구현한다. 4장에서 기능 비교 분석을 수행하고, 5장에서 결론을 내린다.

II. 관련 연구

2.1 링크드 데이터

위키피디아(Wikipedia)에서 링크드 데이터는 “웹 상에 존재하는 데이터를 개별 URI로 식별하고, 각 URI에 링크 정보를 부여함으로써 상호 연결된 웹을 지향하는 모델이다”라고 정의하고 있다[10]. 기술적인 측면에서 링크드 데이터의 중심이 되는 아이디어는 HTTP와 URI를 활용하는 것이다. URI를 이용하여 웹 문서들을 식별하는 것뿐만 아니라 임의의 실세계 객체들을 식별할 수 있다. 이러한 링크드 데이터를 구축하기 위해서는 비구조적인 데이터를 구조화하여 시맨틱 웹에 적합한 형태로 표현해야 한다. 여기서 자원을 구조화한다는 것은 데이터를 RDF 형태로 표현하는 것이다. RDF는 정보의 자원과 속성과의 관계를 주어(Subject), 술어(Predicate), 목적어(Object)의 트리플(Triple) 구조로 표현하여 인터넷 상의 자원이 어떤 값을 갖는지 서술하는 형식을 갖는다. 여기서 정보의 자원은 URI를 통해 고유 식별자를 가지고 여러 가지 속성들을 이용하고자 자원들 간의 링크를 생성한다.

2009년 TED 컨퍼런스에서 팀 버너스리는 다음과 같은 링크드 데이터 4가지 구축 원칙을 제시하였는데[11], 이 원칙은 표준화된 기술을 이용하여 서로 다른 소스에서 나온 데이터를 링크하는 가이드라인이 되고 있다. (1) URI 사용: 개별 객체에 각각 URI를 부여한다. (2) HTTP URI의 사용: 객체에 접근할 수 있는 HTTP URI를 사용한다. (3) RDF나 SPARQL 표준 포맷의 사용: 어떤 사람이 URI를 볼 때 표준 RDF나 SPARQL을 사용해서 유용한 정보를 제공해야 한다. (4) 링크정보 부여: RDF에 포함되어 있는 다른 정보자원에 접근할 수 있어야 한다. 각 자원들을 URI로 구분하고 서로 연결하여 사용자들에게 다양한 정보를 제공할 수 있도록 링크를 제공한다.

그림 1은 링크드 데이터 구축의 추세를 보여주고 있다[12]. 하나의 원은 하나의 데이터세트를 표현하고, 각 데이터세트들의 연결을 선으로 표현한 그림이다. 하나의 데이터세트 안에는 수많은 양의 데이터들이 포함되어 있다.

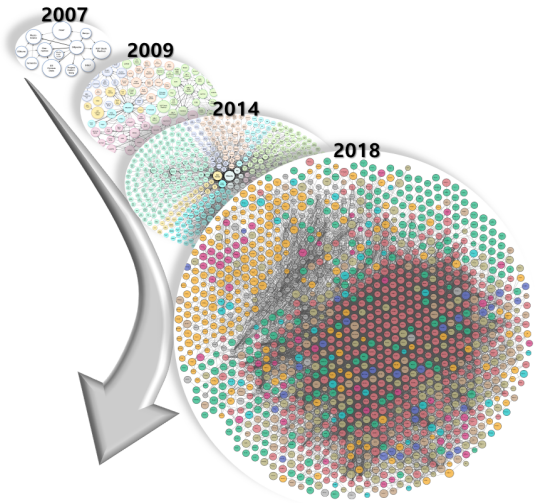


그림 1. 링크드 오픈 데이터 구축 추세
Fig. 1. Trend of building linked open data

이러한 데이터들은 2007년 12개의 데이터세트를 시작으로, 2009년엔 93개, 2014년엔 570개, 그리고 2019년 1월 현재 1,234개의 데이터세트로 그 규모가 기하급수적으로 늘어나고 있다. 각 데이터세트들은 방송, 미디어, 지리정보, 정부, 출판, 의학, 소셜 웹 등 다양한 분야에서 다방면에 활용되고 있다.

2.2 DBpedia

온라인 백과사전인 Wikipedia를 관리하는 비영리 재단인 Wikimedia에 의해 2007년 시작된 프로젝트 중 하나로, DBpedia[13]는 Wikipedia의 데이터를 추출, 가공하여 구조화된 정보를 링크드 데이터 형식으로 제공하는 것이다. DBpedia에서 제공하는 SPARQL Endpoint를 이용하여 복잡한 질의를 할 수 있고, 이를 통하여 웹 상의 데이터를 Wikipedia의 데이터와 연결하여 활용할 수 있다.

그림 2는 DBpedia의 데이터 제공 구조이다. 일반 웹브라우저에서 데이터에 접근하는 경우 HTML를 이용한 일반적인 형태의 웹 데이터로 제공받을 수 있으며, 이 외에도 사용자의 요청에 따라 RDF 형태로 제공 받을 수도 있다. 또한 SPARQL Endpoint도 가지고 있기 때문에 적절한 질의문을 Endpoint에 질의하면 SPARQL 질의를 통하여 특정 개체들에 관한 전체 데이터가 아닌 원하는 부분을 지정하여 특정 데이터만 효율적으로 검색할 수 있다.

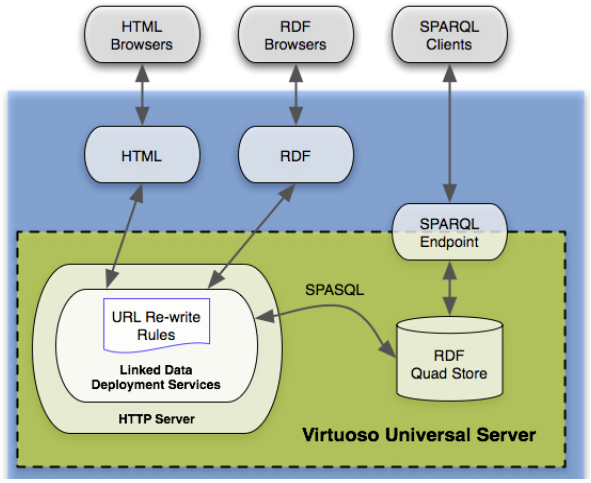


그림 2. DBpedia 데이터 제공 구조
Fig. 2. Structure for providing DBpedia data

DBpedia는 온라인 백과사전을 기반으로 생성된 데이터이기 때문에 다양한 분야에 걸쳐 방대한 양의 데이터를 가지고 있다. 또한 Wikipedia의 내용이 바뀔 경우 DBpedia의 내용도 함께 변경되기 때문에 꾸준하게 데이터가 추가/발전되고 있다. DBpedia의 영어 버전은 4백 50만여 가지의 개체에 관하여 125개의 다양한 언어로 된 각종 정보를 가지고 있다. 144만 5천개의 사람, 73만 5천개의 장소 등을 가지고 있으며, 5천만 개 이상의 외부 링크가 있다. 이는 DBpedia가 링크드 데이터 내에서 상호 연결을 위한 중요한 허브 역할을 하도록 지원한다.

현재 영어, 독일어, 일본어 등 다양한 언어로 DBpedia를 구축하고 있으며 이것을 활용한 오픈 소스 애플리케이션 툴들이 쏟아져 나오고 있다. 이는 DBpedia가 접근하기 쉽고, 비교적 정확하며, 무엇보다도 시간이 지날수록 정보가 발전되는 장점이 있기 때문이다. 본 논문에서도 이 DBpedia를 중점적으로 활용한다.

III. 반응형 웹 애플리케이션 구현

본 장에서는 링크드 데이터의 일종인 DBpedia를 사용하여 사용자 위치정보를 기반으로 한 시맨틱 웹 검색 애플리케이션 구현에 관한 내용을 설명한다. DBpedia 데이터 중에서 지형지물이나 사건정보 등은 위치정보를 포함한다. 이들 위치정보와 사용자의 현재 위치정보를 융합하여 사용자 주변에 있는 개체들을 탐색·활용할 수 있다.

본 연구에서 구현한 애플리케이션은 시맨틱 웹 데이터와 Google에서 제공하는 GoogleMap API를 매쉬업한 반응형 웹 애플리케이션 형태로 구성된다. 즉, 사용자 주변에 있는 개체들에 대한 시맨틱 정보를 보다 쉽게 검색하고 활용할 수 있도록 지도 위에 표시한다. 위치 기반 검색에서 요구되는 사용자의 위치정보는 HTML5에 포함되어있는 Geolocation API를 통하여 획득된다.

구현된 웹 애플리케이션은 서버/클라이언트 형태로 개발되었으며 서버측은 Node.js[14] 프레임워크를 사용하였다. 클라이언트의 경우 반응형 웹 디자인[15]을 적용하여 PC, 스마트폰, 태블릿 등 접속하는 디스플레이 종류에 따라 화면의 크기가 자동으로 변하도록 만들었다. 또한 웹 페이지 내용을 수정할 경우 하나의 페이지만 수정하면 PC와 모바일 등 다양한 디바이스에서 동일하게 반영된다.

3.1 개발 환경

위치 기반 반응형 웹 애플리케이션의 개발 환경은 표 1과 같다. 운영체제로 Windows 10 Pro를 사용하였고 개발 언어로는 JavaScript와 HTML5를 사용하였다. 서버 구현을 위한 플랫폼은 Node.js 8.9.3 버전을 이용하였고 개발 도구로는 통합 개발 환경인 Eclipse 4.7.2버전을 사용하였다.

표 1. 개발 환경
Table 1. Developing environments

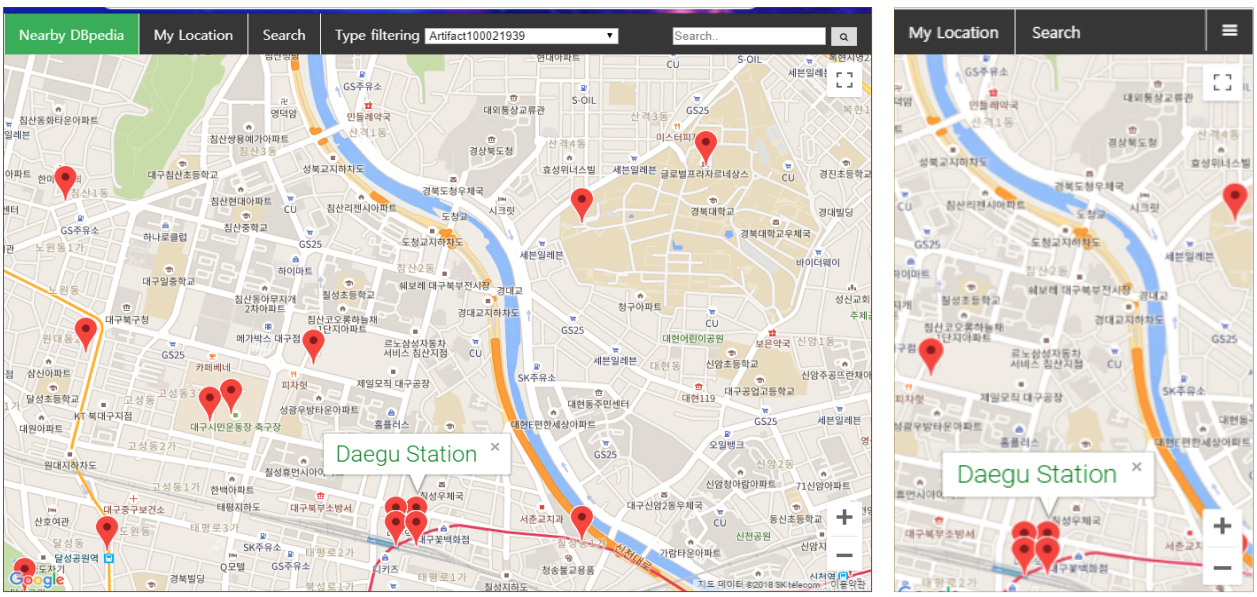
Operating System	Window 10 Pro (64bit)
Server Platform	Node.js v8.9.3
Client Language	Java Script, HTML5
Library	Google APIs v23.0.0 express v4.13.1 ejs v2.4.1 sparql v0.1.3
Development Tool	Eclipse v4.7.2
Test Environments	Chrome v68.0 Chrome Mobile Emulate Samsung Galaxy S8+

애플리케이션 테스트를 위해 PC 환경에서는 Chrome 브라우저를 이용하였으며, 모바일 환경에서는 Google Chrome 개발도구에 있는 Mobile Emulate 기능과 실제 모바일 기기인 Samsung Galaxy S8+를 이용하였다. Google Chrome의 Mobile Emulate 기능을 이용하면 기기의 크기, 해상도 등을 임의로 설정할 수 있고, 미리 정의되어 있는 실제 존재하는 제품의 사양과 동일한 값들을 불러와 테스트해 볼 수 있다. 이를 이용하여 다양한 기기를 구비하지 못하여도 손쉽게 다양한 환경을 시험해 볼 수 있다. 그렇다 하더라고 실제 기기와는 차이가 있을 수 있으므로 보유하고 있는 스마트폰을 사용하여 테스트를 진행하였다.

3.2 메뉴 구성

본 애플리케이션은 반응형 웹 디자인을 적용하여 스마트폰 같은 모바일 기기 혹은 태블릿이나 노트북, 데스크톱 등 다양한 환경에서 모든 기능을 동일하게 활용할 수 있다. 그림 3은 웹 서비스 접속 첫 화면으로 각각 PC와 모바일 환경에서의 접속 모습이다. 두 화면 모두 비슷한 모습으로 구성되어 있으며, 넓은 부분을 지도로 구성하여 사용자 위치정보를 활용한 각종 결과물을 시각적으로 볼 수 있도록 표현하였다. 화면의 상단에는 사용자로부터 명령을 입력받을 수 있는 몇 개의 버튼들이 나열되어 있는데, 해당 기능들은 아래에 배치된 지도와 함께 상호작용을 하도록 구현되어 있다. 각 기능들을 살펴보면 My Location, Search, Type Filtering 기능이 있고 마지막으로 상단 메뉴의 오른쪽에는 키워드 검색을 할 수 있는 기능도 갖추었다.

My Location은 사용자 위치정보를 기반으로 화면에 표시되는 지도의 중심 좌표를 사용자의 위치로 이동시킨다. 최초 접속 시 지도의 중심을 사용자 위치를 중심으로 표시하지만, 이후 사용자의 조작에 따라 지도의 위치 변경도 가능하고, 이를 최초 접속했을 때와 같이 다시 사용자의 위치 중심으로 되돌릴 수도 있다. Search는 읽어 들인 위치정보를 기반으로 시맨틱 검색을 실행하여 DBpedia 개체들을 검색하는 기능이다. 검색에 관한 자세한 내용은 3.4절에서 설명한다. 일련의 위치 기반 시맨틱 검색 과정을 통해 검색된 개체들은 그림 3과 같이 지도에 마커 형태로 표시된다. 이들 마커에는 라벨을 포함하여 사용자가 구분하기 쉽도록 구성하였다.



(a) PC 화면
(a) PC screen

(b) 모바일 화면
(b) Mobile screen

그림 3. 웹 애플리케이션의 초기 화면
Fig. 3. Initial screen of web application

세 번째에 있는 Type Filtering은 Search 기능을 이용하여 검색된 개체 목록들을 분석하여 개체들의 특성을 나타내는 rdf:type 값들을 취합하여 동적인 드롭다운 형태의 목록으로 구성하게 한다. 개체들이 가진 속성들을 중복되지 않게 목록으로 표현함으로써 사용자가 이 목록 중 하나를 선택할 수 있다. 이 목록에서 특정 값을 선택하면, 해당 속성을 가지는 개체들만 지도에 표시되도록 구현하였다.

그림 4는 이 Type Filtering기능을 사용하여 'Station' 속성을 가지는 개체들만 표시한 모습이다. 검색된 개체들 중 지하철, 기차역 등 'Station' 속성을 가진 개체들만 표시되어 있는 모습으로 지하철 노선과 유사하게 몇 개의 선 형태의 모습을 볼 수 있다.

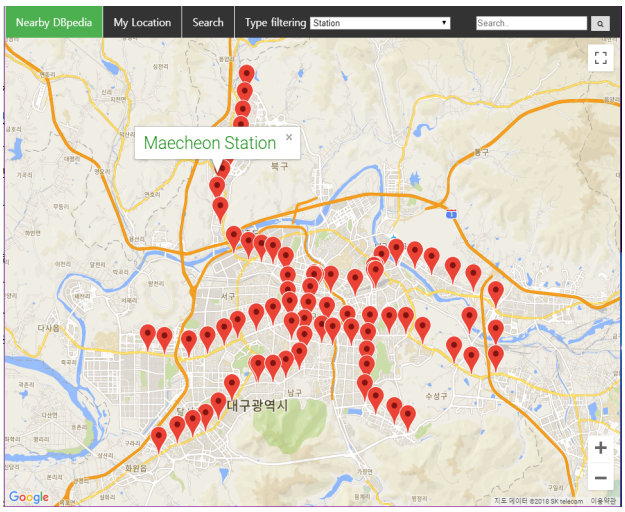


그림 4. Type Filtering 기능
Fig. 4. Type filtering function

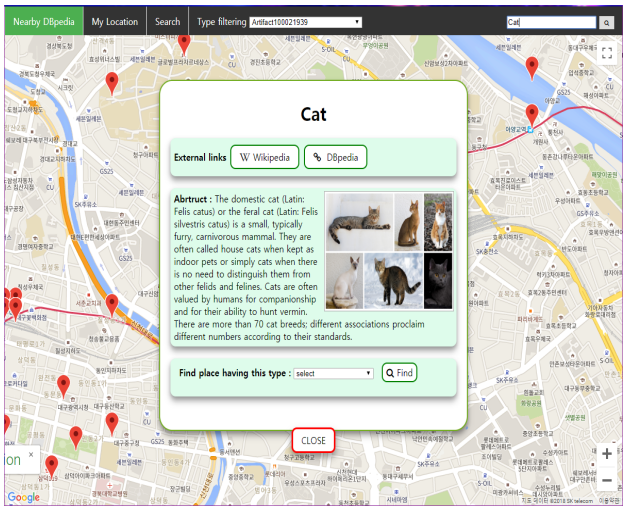


그림 5. 키워드 기반 검색 기능
Fig. 5. Keyword-based retrieval function

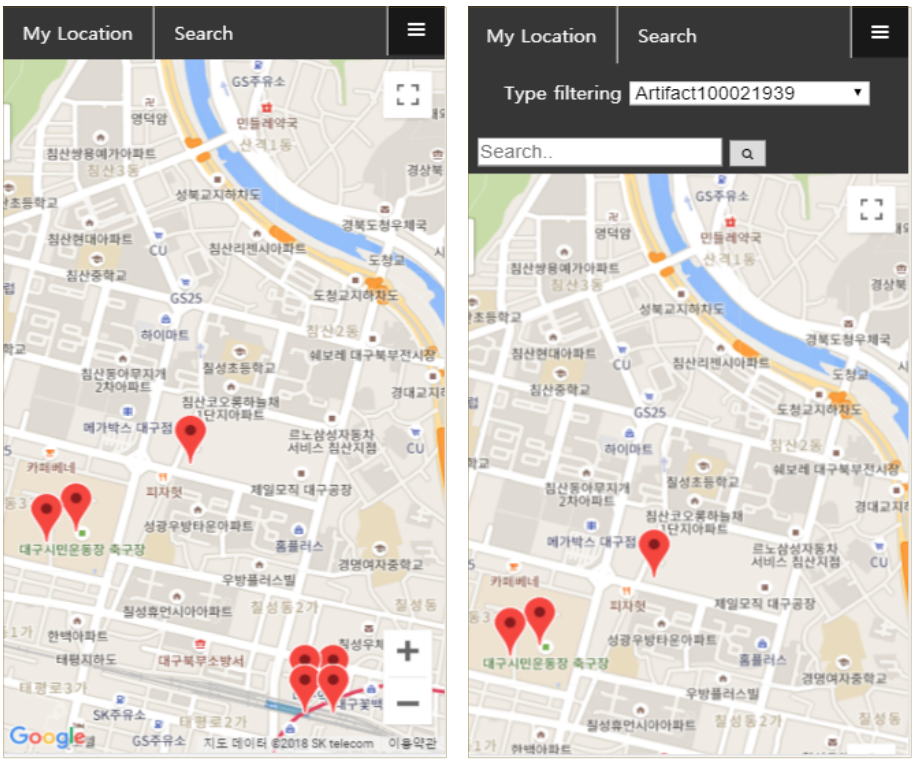


그림 6. 추가 메뉴 숨기기/보이기 기능
Fig. 6. Hiding/show function for additional menu

PC 버전을 기준으로 상단 메뉴의 가장 오른쪽, 모바일 화면에서는 숨겨진 메뉴를 클릭했을 때에 나타나는 키워드 검색 부분을 이용하면 문자 입력에 의한 검색을 할 수 있다. 이 기능을 통하여 웹 애플리케이션 사용 도중에 추가로 궁금한 사항이 생겼을 때, 위치 기반 검색이 아닌 일반적인 텍스트 입력을 통한 검색 또한 가능하도록 구성하였다. 그림 5는 'Cat'을 입력하여 세부정보 창을 표시한 모습이다. 'Cat' 개체는 위치정보를 가지고 있지 않기 때문에 위치 기반 검색으로는 찾을 수가 없다. 이처럼 텍스트 입력을 통하여 위치 기반으로는 찾을 수 없는 개체도 찾을 수 있도록 구성하였다.

모바일 버전의 최초 접속 시 넓은 지도를 제공하기 위하여 My Location과 Search 버튼을 제외한 나머지 기능들은 숨겨져 있다. 이러한 추가적인 기능은 모바일 버전에서만 표시되도록 설정되어 있는 ☰ 형태의 추가 메뉴 숨기기/보이기 버튼을 이용하면 추가적인 메뉴를 표시하거나 이를 감출 수가 있다. 그림 6은 모바일 환경에서 최초 접속 시 일부 메뉴가 숨겨져 있는 화면과 버튼을 눌러 숨겨져 있던 기능들이 표시되어 있는 모습을 보여주고 있다.

3.3 시스템 구조

시스템 구조는 사용자가 웹 브라우저를 통해 정보를 요청하는 클라이언트 및 이를 처리하는 서버로 구성된다. 서버 개발을 위해 사용된 Node.js 프레임워크는 Google의 Chrome V8 JavaScript 엔진을 기반으로 한 비동기 방식 JavaScript 런타임으로, 서버 사이드 애플리케이션 오픈 소스 개발 도구로써 Non-blocking I/O와 단일 스레드 이벤트 루프를 통한 높은 처리 성능을 가지는 것이 특징이다. 또한 Node.js는 모델-뷰-컨트롤러(Model-View-Controller) 패턴을 이용하여 유연한 개발을 효율적으로 지원한다.

Node.js는 npm이라고 불리는 라이브러리 패키지 관리자를 이용하여 다양한 라이브러리를 편리하게 관리하고 사용할 수 있다. 본 시스템 구현에는 구글 지도를 이용하기 위한 Google APIs, Node.js의 기본적인 기능을 제공해주는 express 프레임워크, 템플릿 엔진인 ejs, SPARQL 질의문을 편리하게 처리할 수 있는 sparql 라이브러리 등을 사용한다.

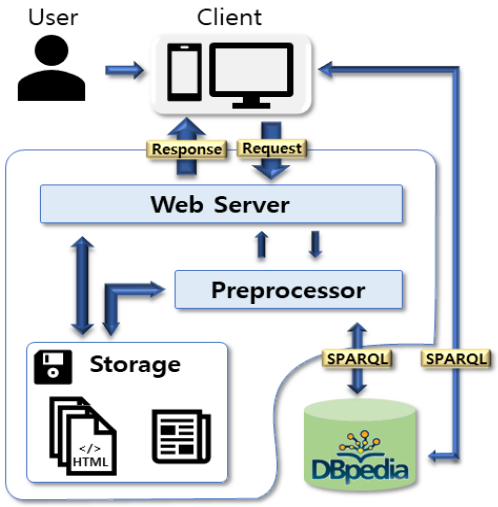


그림 7. 시스템 구조
Fig. 7. System architecture

그림 7은 전체적인 시스템 구조를 보여주고 있다. 서버에는 사용자의 요청을 처리하는 Web Server가 있으며, 사용자 요청에 따라 Preprocessor를 통하여 서버 내부 데이터를 활용하거나, 필요한 경우 DBpedia 데이터를 검색하여 융합된 결과를 조합하여 사용자에게 전송한다. 클라이언트 측에서는 크게 두 부분으로 나누어지는데, 먼저 첫 화면에서 위치 기반 시맨틱 검색을 실행하여 지도에 표시해주는 부분과 지도에 표시된 마커를 조작하여 해당 개체에 대한 상세 정보를 조회하는 부분으로 구성된다. 최초 접속 시 가장 먼저 사용자의 위치정보 사용 가능 여부를 확인한다. 이 정보의 사용이 불가능한 경우 사용자에게 위치정보 사용 불가능을 알리는 메시지를 보낸다. 사용자의 위치정보가 사용 불가할 경우 기능이 수동으로 전환되어 검색 버튼을 클릭할 때 원하는 위치정보를 사용할 수 없는 상황에서도 위치 기반 검색이 가능하도록 하였다.

3.4 위치 기반 검색 프로세스

위치 기반 서비스를 제공하기 위해선 사용자의 위치 파악이 우선 필요하다. 사용자 위치정보 획득에는 HTML5가 제공하는 Geolocation API를 이용하여 위치정보를 확인한다. HTML5의 Geolocation API는 GPS나 Wifi, IP 주소 등 다양한 방법으로 위치정보 획득을 시도한다. 또한 이런 위치정보는 개인정보 보호와 연관하여 보안 연결(HTTPS)을 요구한다.

```
IF 위치정보 사용 가능?
  THEN
      페이지 지도의 중심위치를 사용자 위치로 이동.
      사용자 위치정보를 이용하여 DBpedia 검색
          SPARQL 쿼리 실행.
      쿼리 결과를 바탕으로 지도에 마커를 표시.
  ELSE
      사용자에게 위치정보 사용 불가능 통보.
IF Search 명령 입력
  THEN
      현재 지도 중심좌표를 기준으로 DBpedia 검색
          SPARQL 쿼리 실행.
      쿼리 결과를 바탕으로 지도에 마커를 표시.
  ELSE
IF Type Filtering 명령 입력
  THEN
      Search 기능을 이용하여 type 목록 구성.
      기존의 마커 제거.
      선택된 타입 값을 바탕으로 같은 속성를 보유한
          마커 새로 표시.
  ELSE
```

그림 8. 위치 기반 검색 알고리즘
Fig. 8. Location-based retrieval algorithm

그림 8은 위치 기반 검색을 실행하고 데이터를 확인하는 전체적인 과정을 보여주고 있다. 위치 기반 검색 프로세스는 가장 먼저 사용자의 위치정보 사용 가능 여부를 먼저 확인하여 사용이 불가능 할 경우 이를 사용자에게 간단한 메시지로 알려준다. 위치정보 사용이 가능한 경우, 획득한 위치정보를 이용한 질의문을 DBpedia에서 제공하는 SPARQL Endpoint에 질의하여 위치 기반 검색을 실행한다. 이 검색 결과를 바탕으로 지도에 마커를 표시한 후 사용자에게 결과를 제공한다. 또한 최초 사용자 위치 기반 검색을 실행한 후, 지도의 위치를 임의의 위치로 이동한 후 수동으로 검색 명령을 실행할 때도 사용자의 위치 정보가 아닌 새롭게 설정된 지도의 중심 좌표를 기준으로 검색되도록 구현하였다. 이를 통하여 사용자의 주변뿐만 아니라 원하는 장소로 지도를 이동하여 해당 지역의 데이터를 확인할 수 있다.

위의 검색 과정에서 위치 기반 검색에 사용되는 SPARQL 질의문의 내용은 그림 9와 같다.

```
SELECT ?res ?label ?lat ?lng
(group_concat(distinct ?type;separator= "," ) as ?types)
WHERE {
      ?res rdfs:label ?label .
      ?res geo:lat ?lat .
      ?res geo:long ?lng .
      ?res rdf:type ?type .
FILTER (
      ?lng > (lng - SearchRange) AND
      ?lng > (lng + SearchRange) AND
      ?lat > (lat - SearchRange) AND
      ?lat > (lat + SearchRange) ) .
FILTER regex(lang(?label), "en" ) .
}
```

그림 9. 위치 기반 SPARQL 질의문
Fig. 9. Location-based SPARQL query statement

질의문의 내용을 살펴보면 일반적으로 가장 먼저 나오게 되는 PREFIX 부분이 생략되어 있는 모습을 볼 수 있는데, 이는 DBpedia의 SPARQL Endpoint가 사용자의 편의를 위하여 DBpedia에서 자주 사용되는 접두어들은 입력하지 않아도 이들 PREFIX를 자동으로 처리해 주기 때문이다. 질의문의 구성은 가장 먼저 SELECT 절에서 출력할 변수로 개체의 URI 주소를 지정하는 ?res, 라벨을 지정하는 ?label, 위도와 경도를 지정하는 ?lat와 ?lng, 그리고 마지막으로 개체의 속성인 rdf:type을 하나의 문자열로 저장하는 ?types로 구성되어 있다.

개체 타입 같은 경우 보통 한 개체가 여러 가지의 속성 타입을 가지고 있다. 이를 별도로 처리하지 않을 경우 질의 결과가 ?res, ?label, ?lat, ?lng와 함께 하나의 속성이 하나의 결과 값으로 출력되게 된다. 결과적으로 하나의 개체가 가진 rdf:type 유형들이 개별로 출력되게 되어 속성의 개수만큼 ?res, ?label, ?lat, ?lng 값들이 중복되어 출력되게 된다. 이렇게 출력될 경우 결과의 개수도 불필요하게 많아지고 type 속성을 제외한 다른 변수들은 같은 값을 가지고 있기 때문에 많은 중복이 발생한다. 이는 추후 처리에도 비효율적이다. 이를 SELECT 절에서 (group_concat(distinct ?type;separator = ",") as ?types) 구문을 이용하여 해결하였다. 여러 개의 type 속성 값들을 "," 문자로 구분하여 하나의 문자열 형식으로 변환하여 ?types 변수로 지정하여 결과 값을 출력하도록 지정하였다.

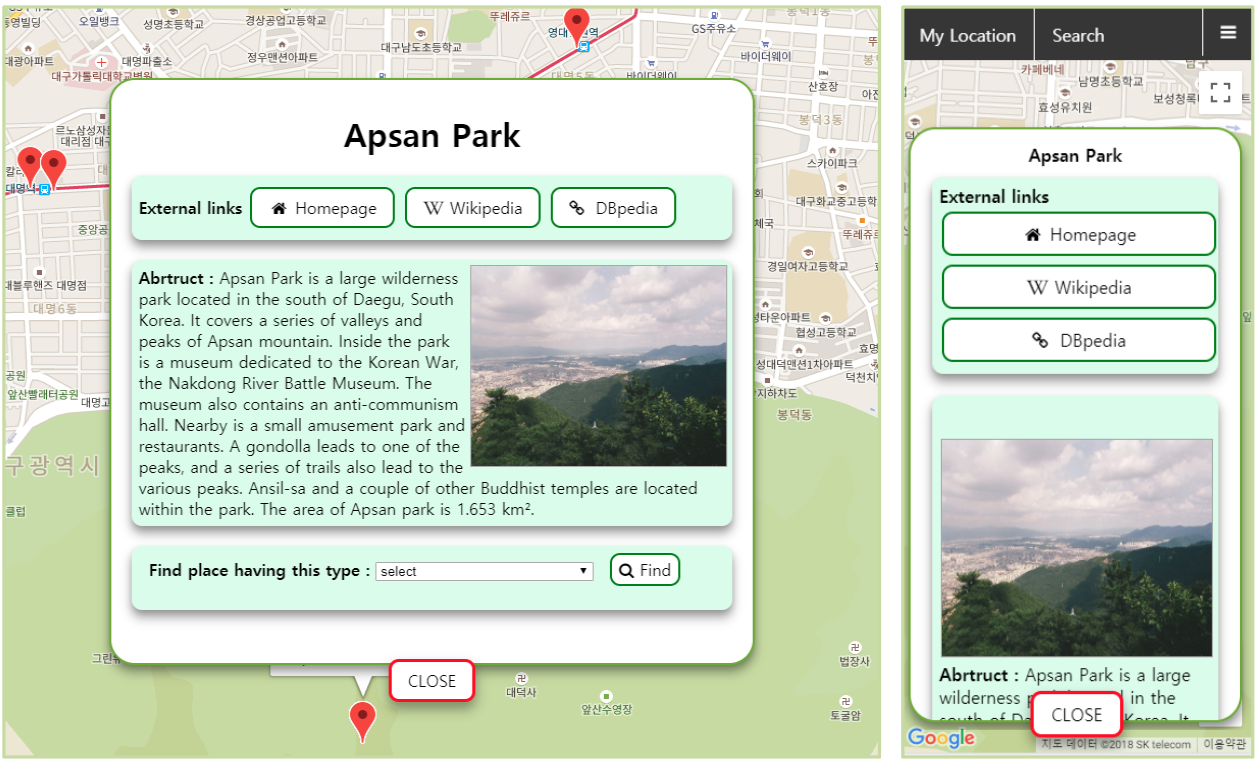
이후 where의 내용에는 ?res의 라벨과 위도, 경도, 타입들이 검색되도록 기술되어 있으며, FILTER 구문에서는 사용자의 위도, 경도가 입력되어 있는 lat, lng 변수와 함께 SearchRange 변수를 정의하였다. lat, lng 값을 기준으로 SearchRange 변수가 가진 값만큼 더하거나 빼서 특정 범위 안에 있는 개체들만 필터링 되도록 기술되어 있다. SearchRange 값은 0.1로 설정했으며, 이는 위도를 기준으로는 약 1.8km, 경도를 기준으로는 약 1.4km에 해당한다.

그림 9의 SPARQL 질의문을 DBpedia의 SPARQL Endpoint에 송신하면 그 내용을 수신한 후 그 결과를 처리하여 지도에 각 개체의 위도·경도 값을 이용하여 해당 위치에 마커 형태로 표현한다. 지도에 표시된 마커에 마우스 커서를 위에 올리거나, 혹은 모바일 환경에서는 해당 동작이 어렵기 때문에 원활한 사용을 위해 마커를 터치했을 때 마커의 라벨이 표시되도록 구성하였다. 표시된 라벨을 다시 한 번 클릭하게 되면 세부 정보창이 화면에 표시되고 세부정보 표시 프로세스가 시작된다.

3.5 세부정보 표시 프로세스

위치 기반 검색 프로세스를 통하여 표시된 마커를 클릭하면 해당 개체의 URI를 사용하여 서버에 전송하여 관련된 세부정보 페이지를 요청한다. 세부정보 표시 화면은 그림 10과 같이 구성되어 있다. 개체의 이름, 홈페이지, Wikipedia 링크, DBpedia의 전체 데이터에 접근할 수 있는 링크를 가진 버튼, 개체에 관한 요약정보, 개체와 관련된 사진, 그리고 앞선 시작 화면에 있는 Type Filtering 기능과 유사하게 탐색된 개체와 같은 속성을 가지는 개체들을 지도에 표시하는 기능인 속성 필터링 기능을 구현하였다.

서버에서 세부정보 요청을 받으면 함께 전달된 개체 URI를 이용하여 DBpedia SPARQL 질의를 실행한다. SPARQL 질의 내용은 세부정보 내용을 표시하기 위한 요소들로 이름, 요약정보, 개체의 속성들, 홈페이지 링크, 그리고 사진과 같은 데이터들을 읽어 들인다.



(a) PC 환경
(a) PC environment

(b) 모바일 환경
(b) Mobile environment

그림 10. 세부 정보 표시 화면
Fig. 10. Screen of showing detail information

```
PREFIX target: <url>
SELECT ?label ?comment ?depiction ?homepage
(group_concat(distinct ?type;separator= "," ) as ?types)
WHERE {
      target: rdfs:label ?label .
      target: rdfs:comment ?comment .
      target: rdfs:type ?type .
OPTIONAL {target: foaf:depiction ?depiction .}
OPTIONAL {target: foaf:homepage ?homepage .}
FILTER (lang(?label) = 'en' )
FILTER (lang(?comment) = 'en' )
}
```

그림 11. 세부정보 SPARQL 질의문
Fig. 11. SPARQL query statement for detail information

그림 11은 세부정보를 읽어 들이기 위해 사용한 SPARQL 질의문이다. 검색할 대상은 세부정보 요청과 함께 전달된 URI를 간략하게 표시하기 위하여 PREFIX에서 target으로 정의하였다. 이 후 질의 내용을 살펴보면 개체의 라벨을 나타내는 rdfs:label, 개체에 관한 대략적인 정보를 나타내는 rdfs:comment, 개체의 속성을 나타내는 rdf:type, 개체의 사진 주소를 가지고 있는 foaf:depiction, 그리고 홈페이지 주소를 가지고 있는 foaf:homepage가 기술되어 있다. rdf:type의 경우 앞선 위치 기반 검색에서와 동일하게 검색 결과의 최적화를 위해 ","로 구분된 문자열 형태로 출력되도록 지정하였다.

foaf:depiction과 foaf:homepage는 개체에 따라 해당 값을 가지지 않는 개체들이 존재한다. 이를 따로 처리하지 않고 그대로 검색할 경우 foaf:depiction 혹은 foaf:homepage 값을 가지지 않은 개체들은 검색되지 않기 때문에 OPTIONAL 구문을 이용하여 해당 값이 존재하지 않는 개체들도 검색되도록 구성하였다. 또한 rdfs:label과 rdfs:comment의 경우 한 가지가 아닌 여러 종류의 언어로 데이터가 존재하는 경우가 있는데, 이 경우 따로 처리하지 않으면 존재하는 모든 언어에 대한 결과가 출력되게 된다. 본 애플리케이션에서는 영어 한 가지 언어만 사용하여 정보를 제공하기 때문에 영어를 제외한 다른 언어는 필요하지 않다. 그래서 FILTER(lang(?label) = 'en')과 같이 FILTER구문을 이용하여 여러 데이터 중 영어로 구성된 데이터만 선택하여 읽어 들이도록 한다. 이 SPARQL 질의 결과를 바탕으로 세부 정보 페이지의 내용을 구성하여 이를 사용자에게 다시 전송하게 된다.

Ⅳ. 비교 분석

위치 기반 시맨틱 검색 시스템은 현재 연구가 활발히 진행되고 있는 분야이므로 아직까지 구체적인 구현이 완료된 시스템은 아직 찾아보기 힘든 상황이다. 따라서 규모나 개발 범위가 다른 시스템 간 성능 분석은 큰 의미가 없으므로 표 2와 같이 본 논문에서 구현된 웹 애플리케이션(SemanticWeb)을 기능이 유사한 GeoNames와 네이버 지도(NaverMap)와의 기능 스펙을 비교하였다.

GeoNames는 지형지물에 관한 시맨틱 데이터를 제공해 주는 서비스이다. GeoNames는 기본적인 RDF 형식의 데이터 외에도 이를 지도와 결합한 형태로 정보를 제공해준다. GeoNames의 경우 특정 지명이나 건물명을 입력하면 GoogleMap과 연동하여 지도 형태로 정보를 제공해 주지만, 별도의 위치 기반 검색은 지원하지 않는다. 제공되는 정보의 유형도 지형지물에 한정되어 있다. 또한 개체의 정보 역시 지형지물에 맞게 주소나 개체의 타입 등은 제공하지만 개체의 상세한 정보는 제공해주지 않는다. 서비스 제공 플랫폼은 PC 형태로만 제공하기 때문에 모바일 등 다른 환경에서는 이용하기 어렵다.

NaverMap은 국내에서 가장 높은 점유율을 가지고 있는 지도 서비스이다. 서비스는 PC 및 안드로이드와 iOS를 포함한 모바일 웹을 지원한다. NaverMap은 위치 기반 검색은 가능하지만 시맨틱 데이터를 사용하지 않는다. 또한 GeoNames와 같이 제공하는 정보의 유형이나, 개체가 가지고 있는 데이터에 관한 내용 역시 주소, 전화번호 등을 제공하지만 자세한 정보는 제공하지 않기 때문에, 추가적인 정보를 얻기 위해서는 별도의 검색 과정이 필요하다.

본 연구에서 제안한 SemanticWeb인 경우 온라인 백과사전을 기반으로 한 DBpedia를 이용한 정보를 제공하기 때문에 지형지물과 더불어 해당 지역에서 일어난 사건 등도 확인할 수 있어 더욱 다양한 개체 정보를 제공한다. 이는 개체의 유형에 상관없이

위치정보를 가진 개체에 관한 정보라면 모두 표시되기 때문이다. 또한 애플리케이션 내에서 상세 정보 페이지를 이용하여 개체에 관한 요약정보나 사진 등을 바로 확인할 수 있도록 제공해준다. 제공 플랫폼 또한 반응형 웹 디자인을 적용하여 다양한 환경에서 사용할 수 있다. 앞서 기술한 기존의 서비스와 본 연구에서 제안한 웹 애플리케이션의 특징 비교를 요약하면 표 2와 같다. 여기에서 세부정보 제공, 링크드 기반 브라우저, 그리고 매쉬업 기능 등은 본 시스템만이 갖고 있는 독특한 기능들이다.

표 2. 시스템 간 기능 비교
Table 2. Function comparison with other systems

		SemanticWeb	NaverMap	GeoNames
Semantic Function		Yes	No	Yes
Location-based Search		Yes	Yes	No
Information Type	geographic feature	Yes	Yes	Yes
	accidents incidents	Yes	No	No
Detail Information		Yes	No	No
SPARQL Support		Yes	No	Yes
Responsive Web		Yes	Yes	No
Link Traversal		Yes	No	Yes
Linked-based Browser		Yes	No	No
Mashup Function		Yes	No	No

V. 결 론

웹과 관련된 기술과 통신 기술의 발달로 인해 인터넷은 일상 생활과는 떼려야 뗄 수 없는 관계가 되었다. 이는 곧 방대한 양의 정보가 생성되고 유통되며 소비되는 것을 의미한다. 하지만 이러한 방대한 양의 데이터는 데이터가 너무 많아 오히려 효율적으로 정보를 관리, 검색하는 것이 어려운 상황에 이르렀다. 이런 상황을 극복하기 위해 시맨틱 웹이 제안되었고 시맨틱 웹을 활용한 다양한 애플리케이션들이 등장하고 있다.

본 논문은 시맨틱 웹을 응용하는 한 사례 연구로, 시맨틱 웹과 온톨로지 기술을 이용하여 시맨틱 데이터와 함께 사용자의 위치정보를 활용하여 사용자의 주변에 존재하는 개체를 검색하는 애플리케이션을 구현하였다. 시맨틱 데이터인 DBpedia 정보와 Google에서 제공하는 GoogleMap API를 매쉬업하고, 반응형 웹 애플리케이션 형식으로 구현되어 사용자에게 시맨틱 데이터를 보다 지능적으로 활용할 수 있도록 지원한다.

또한 검색된 개체들에 대하여 세부적인 시맨틱 데이터를 확인할 수 있는 링크드된 브라우징 기능을 제공하고 있으며, 검색된 개체와 유사한 개체를 찾을 수 있도록 도와주는 필터링 기능을 갖추었다. 구현된 웹 애플리케이션은 반응형 웹 디자인을 적용하여 특정 환경에서만 사용 가능한 것이 아닌, PC 및 모바일 등 다양한 접속 환경이나 플랫폼에서 사용할 수 있는 높은 접근성을 제공한다. 이를 통하여 누구나 다양한 환경에서 손쉽게 시맨틱 데이터에 접근하고 이를 활용할 수 있도록 해준다.

현재는 단순히 DBpedia 데이터를 활용하여 제공하는 기능만 가지고 있지만, 추후 여러 데이터세트들을 활용하여 더 다양한 정보를 제공하면 더욱 정확한 정보와 풍부한 데이터를 제공할 수 있을 것으로 기대된다. 이와 더불어 참고문헌 [16][17]과 같은 최근의 유사한 시스템과의 성능 비교 분석을 통해 본 논문의 차별성을 보다 부각시킬 필요가 있다.

References

[1] E. Kim, "A Study on the Progress and Development of E-Government", Master Thesis, Dong-A University, Feb. 2017.

[2] Doosan Corp., "Mashup", http://www.doopedia.co.kr [accessed: Jan. 21, 2019]

[3] Y. Lee, "Semantic-Based Data Mashups Using Hierarchical Clustering and Pattern Analysis Methods", Journal of Information Science and Engineering, Vol. 30, No. 5, pp. 1601-1618, Sept. 2014.

[4] Y. Lee, "Web 3.0 is Changing the World", Bomoongak, Jun. 2010.

[5] E. Jung and Y. Lee, "Linked Data based Storage/Application Platform and Implementation of Analysis System Visualization", Journal of

KIIT, Vol. 16, No. 9, pp. 95-102, Sep. 2018.
[6] M. Lee and S. Choi, "Research on Minimizing Access to RDF Triple Store for Efficiency in Constructing Massive Bibliographic Linked Data", Journal&Article Management System, Vol. 48, No. 3, pp. 233-257, Sep. 2017.
[7] S. Ha, J. Yim, and H. Rieh, "A study on the Procedure for Constructing Linked Open Data of Records Information by Using Open Source Tool", Korea Society for Information Management, Vol. 34, No. 1, pp. 341-371, Mar. 2017.
[8] D. Jo and M. Kim, "Linked Legal Data Construction and Connection of LOD Cloud", Korea Society of Computer and Information, Vol. 21, No. 5, pp. 11-18, May 2016.
[9] M. Cho, W. Oh, and J. Park, "Linked Data R&D Report: Mainly of Subject, Name and Author Name Authority Data", National Library of Korea, Dec. 2011.
[10] https://ko.wikipedia.org/wiki/ [accessed: Jan. 21. 2019]
[11] C. Bizer, T. Heath, and T. Berners-Lee, "Linked Data - The Story So Far", International Journal on Semantic Web and Information Systems, Vol. 5, No. 3, pp. 1-22, Jul. 2009.
[12] https://lod-cloud.net. [accessed: Jan. 21, 2019]
[13] https://wiki.dbpedia.org. [accessed: Jan. 21, 2019]
[14] Node.js, "Introduction to Node.js", https://nodejs.dev. [accessed: Jan. 21, 2019]
[15] Wikipedia, "Responsive Web Design", https://ko.wikipedia.org/wiki/ [accessed: Jan. 21, 2019]
[16] Y. Lee, "Semantic Mobile Mashup Using Linked Open Data", Journal of KIIT, Vol. 14, No. 11, pp. 93-100, Nov. 2016.
[17] D. Lukovnikov, C. Stadler, D. Kontokostas, S. Hellmann, and J. Lehmann, "DBpedia Viewer - An Integrative Interface for DBpedia Leveraging the DBpedia Service Eco System", World Wide Web Workshop: Linked Data on the Web, Apr. 2014.

저자소개

이 수 형 (Suhyoung Lee)

2017년 : 경북대학교 컴퓨터시스템공학전공(학사)
2019년 : 경북대학교 컴퓨터학부(공학석사)
관심분야 : 시맨틱 웹, 빅데이터, 웹 데이터베이스

이 용 주 (YongJu Lee)

1985년: 한국과학기술원 정보검색전공(공학석사)
1997년 : 한국과학기술원 컴퓨터공학전공(공학박사)
1998년 8월 ~ 현재 : 경북대학교 IT대학 컴퓨터학부 교수
관심분야 : 링크드 데이터, 시맨틱 웹, 빅데이터, 웹 사이언스