인터넷 정보 검색을 위한 XML 기반 동적 탐색 에이전트 개발연구

이양원
호남대학교

XML based Dynamic Search Agent for the Internet Efficient Multicast

Yang Weon Lee
*Honam University
E-mail : ywlee@honam.ac.kr

요 약

WWW의 정보량의 증가에 있어서 그 속도는 어마어마하게 빠르게 늘어나고 있다. 탐색 엔진의 역할은 좀 더 필요하고 중요한 정보를 찾는 것에 있다. 현재 대부분의 검색엔진은 더 많은 자료를 찾는 것을 제 1차 목표로 하고 있기 때문이다. 그러므로 찾은 데이터에는 많은 쓰레기 정보가 포함되어 있다.

본 연구에서는 "정보에 관한 정보"를 사전에 인덱싱하여 놓음으로서 XML 기반으로 구조적으로 구성하여 데이터베이스에 저장하여 놓았다. 그리고 검색 에이전트를 사용하여 효과적으로 데이터를 찾을 수 있는 방법을 구현하였다.

ABSTRACT

As the amount of information on the WWW is increasing at a very high speed, the role of search engines has become more necessary and important. Yet, current popular search engines have a limitation of scalability because the more information those engines gather, the more garbage hit search results are produced, thus, XML based metadata has recently been the issue of much focus. Metadata is information about information, which using a small potion of data can express the fundamental information about other information. XML enables data to be expressed better organized and well structured. Moreover, the information in those current efficiency and make them disappointed by dead links and obsolete pages. this is because the information was collected and indexed in advance before users do searching. therefore, a mechanism is required that cam search from current and up-to-date information resources dynamically in real time to avoid retrieving out of data information. In this study, a new concept search agent system for the WWW by using XML based technology is proposed. the implementation of the prototype of this proposed system, the comparison with traditional search engines, and the evaluation of the prototype system are also discussed.

키워드
WWW, Internet, XML , search agent

Ⅰ. 서 론

본 연구의 목적은 탐색정확도와 탐색엔진들이 안고 있는 공통적인 문제들을 개선한 탐색시스템을 개발하는 것이다. 일반적으로 대부분의 탐색엔진은 두 가지 형태로 분류할 수 있다 : 분류된 목록형식과 자유 텍스트 기반 키워드 검색. 그러나 오늘날에는 목록으로 분류되는 디렉터리 전용 검색엔진은 드물고 대부분이 자유 텍스트 기반 키워드 검색을 지원하고 있다. 이들 검색엔진들의 공통점은 분류된 정보 목록과 키워드를 저장하기 위해서 방대한 데이터베이스를 가진 중앙 시스템을 구축하고 있다는 것이다. 역사적으로 볼 때 목록 기

반 검색엔진은 YAHOO가 시초라고 볼 수 있다. 야후는 목록 정리는 처음에는 사람이 직접하였으나 현재는 자동으로 컴퓨터가 분류하고 있다. 그러나 전체를 자동으로 분류하는 것은 아직 하지 못하고 있다. 반면에 키워드 기반 검색엔진은 사용자가 토픽에 따라서 손쉽게 정보를 얻을 수 있으며 또한 데이터베이스화도 자동으로 할 수 있어서 매우 빠르게 발전하고 있다. 그러나 이 것 역시 몇 가지 문제점을 안고 있다.

키워드 기반 검색엔진은 크게 세 가지 문제점을 가지고 있다. 첫째로, 이들 엔진은 너무나 많은 쓰레기 정보를 제공한다는 점이다. 왜냐면 정보를 분류할 수 있는 능력이 없기 때문이다. 둘째로 이미 한물간 정보나, 링크가 죽은 사이트 정보, 이미 사라진 내용들을 데이터베이스에 가지고 있으므로 보여준다는 것이다. 셋째로는 이들 엔진은 시스템 자체가 scalability 문제를 가지고 있다는 것이다. 정보의 양이 증가함에 따라서 데이터베이스의 크기를 무한정 증가시킬수 없다는 문제가 대두된다. 따라서 데이터를 수집하는 에이전트는 적절한 주기로 데이터를 수집할 수 없는 문제가 발생한다.

이러한 문제를 해결하기 위해서는 XML 기반 동적인 데이터 검색이 이루어지는 시스템의 개발이 필요하다. 최근에 점점 더 XML기술을 이용한 데이터 저장 방법의 사용이 증가되고 있다. XML 문서는 기존의 문서형식에 비해서 텍스트 기반 포맷을 가지고 있기 때문에 시스템에서 읽기 능력이 뛰어나다. 즉 XML 문서는 바이너리 형식의 파일로 저장된 문서들과는 완전히 다른 형태를 가지고 있다. 그러므로 XML 문서는 한 응용에서 다른 응용으로 쉽게 전이가 가능하다.

II. XML 기반 탐색

기존의 탐색 엔진의 문제점을 극복하기 위해서는 두 가지 개념이 필요하다. 첫째는 XML 기반이고 둘째는 동적인 내용 획득이다. 이 같은 사항은 XML 기반으로한 메터데이터를 사용하면 좀 더 높은 정확도를 가진 검색엔진을 개발 할 수 있다.

2.1 XML 기반 정보

XML 형식은 다음 예와 같이 잘 조직화된 구조체 형식이다. 탐색 결과의 정확도를 향상시키기 위해서 표적 데이터는 조직화되고 구조화되어야한다.

```
[예1]
<H3>DVD player</H3>
가격 <B>4000원</B>
[예2]
<H3>소개 </H3>
이 연구는 <B>XML</B>에 관한 것이다.
```

2.2 동적인 탐색 에이전트 모델

본 연구에서는 4개의 잠정적인 대안을 제시한다. 그리고 각각의 에이전트에 대한 강점과 약점을 기술한다.

2.2.1 클라이언트측 에이전트 모델

클라이언트측 모델에서 에이전트 프로그램은 각 클라이언트 머신에서 동작한다. 처음에 시작페이지에서 시작하여 하이퍼링크에 따라서 작동하며 사용자가 지시한 탐색 조건에 맞게 XML 문서를 탐색한다. 이 모델의 강점은 각 에이전트 프로그램이 각 사용자에 대해서 쉽게 조절할 수 있다는 점이다. 그러나 네트워크 밴드폭이 항상 충분하지 않다는 단점이 있다. 사용자들이 사용하는 클라이언트 단말의 속도가 빠르지 않다는 것이다.

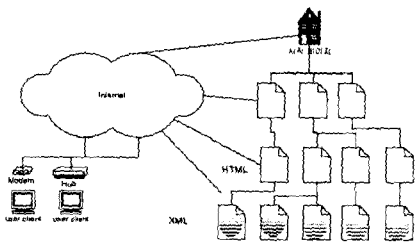

그림 2.1 클라이언트측 에이전트 모델

2.2.2 서버측 에이전트 모델

서버측 모델에서 에이전트 프로그램은 서버 머신위에서 작동한다. 에이전트는 사용자 클라이언트로부터 탐색 조건을 얻는다. 그리고서 http 연결과 하이퍼링크를 통하여 XML문서를 검색한다. 사용자들은 웹브라우져를 이용하여 에이전트 프로그램과 통신할 수 있다.

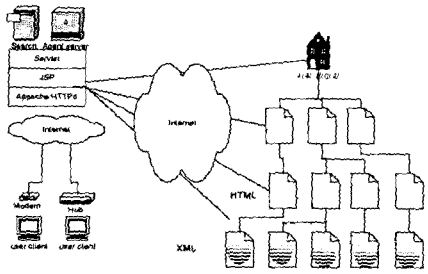

그림 2.2 서버측 에이전트 모델

클라이언트측 모델과 다른점은 대부분의 http 접속은 에이전트 서버와 표적 웹사이트간에 설정된다는 점이다. 오직 질문과 탐색결과만이 사용자 클라이언트에게 전송된다는 점이다. 그러므로 사용자는 웹브라우져 외에 다른 프로그램을 설치할 필요가 없으며, 이 것은 가격을 절감시키는 효과를 가져온다. 그러나 이 모델의 단점은 많은 클라이언트의 접속으로 인한 집중 현상이 발생한다는 것이

다. 에이전트 서버는 매우 높은 성능과 scalability를 가지고 있어야만 혼잡을 피할 수 있는 단점이 있다.

2.2.3 분산 에이전트 모델

그림 2.3에 분산에이전트 모델을 보였다. 에이전트 서버는 모든 표적 웹사이트에 배치되어 설치되어있다. 하나의 중심 서버는 각각에게 질문을 보내고 검새결과를 수집하는 역할을 수행한다. 분산 에이전트 서버는 자신의 내부 사이트에서만 임무를 수행한다.

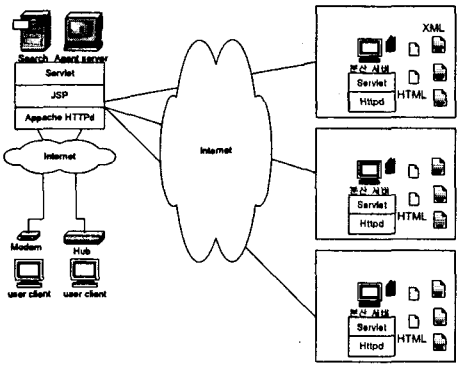

그림 2.3 분산 에이전트 모델

서버측 에이전트 모델과 유사하게 탐색 질문과 탐색결과는 오직 사용자 클라이언트와 에이전트 서버사이에 낮은 전송 링크를 통하여 전송된다. 추가로 인터넷을 통하여 주고받는 정보는 질문과 결과이므로 네트워크의 혼잡을 피할수 있다. 또한 탐색 시간도 절약할 수 있다. 이 모델의 단점은 각각의 서버를 설치하기 위해서는 많은 비용이 든다는 점이다. 그러므로 에이전트 서버를 많은 웹사이트에 배치하는 것은 매우 어려운 현실이다. 더군다나 각 웹사이트는 각기 운영 정책이 다를수도 있기 때문에 모든 웹마스터가 에이전트 서버의 설치를 동의하지 않을 수도 있다.

2.2.4 피어투피어 에이전트 모델

이 모델은 현재 냅스터(Napster)나 그누텔라(GNutella)에서 구현하고 있는 전형적인 방법이다. 냅스터가 중앙서버를 가지고 있는 반면에 그누텔라는 가지고 있지 않다. 냅스터 모델에서 중앙서버는 각기 다른 분산 서버에서 위치하고 있는 파일의 정보를 제공하거나 관리한다. 그렇지만 두개의 모델 공히 데이터 전송은 피어투피어 방식으로 수행된다. 그림 2.4에 이 모델의 구조를 보였다. 최근에 텍스트 기반 탐색엔진은 scalability 문제, 효율성, 데이터 신선도문제 등의 문제를 개선하기 위해서 피어투피어 탐색 엔진모델을 관심있게 연구하고 있다. 그렇지만 그누텔라 모델에서 만일 요구하는 서버가 요구된 데이터를 가지고 있지 않으면 탐색 요구는 다른 서버에 이전될 것이다. 결국 이 같은 행위가 반복해서 이루어지면 심각한 네트워크 혼잡을 가져오게 될 것이다. 이리하여 냅스터와 그누텔라는 오직 파일 이름 서비스만 제공하고 있는 것이다.

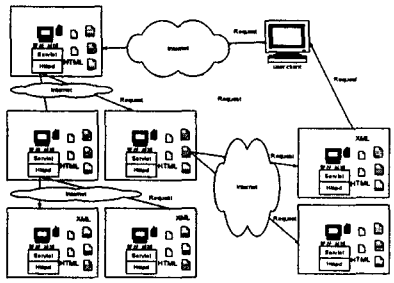

그림 2.4 피어투피어 에이전트 모델

2.5 추천 모델

4개의 제안된 모델 중에서 추천 모델은 서버측 에이전트 모델이다. 왜냐면 이 모델은 성능과 구현면에서 강점을 가지고 있다. 다음 표는 이 것들에 대한 비교표를 보인 것이다.

표 2.1 모델간 특징 비교

	Client side	Server side	Distributed	P-to-P
성능	미흡	중	최우수	우수
가격	최우수	우수	미흡	보통
네트워크 사용량	미흡	보통	최우수	미흡
유지보수	미흡	보통	미흡	미흡

III. 에이전트 구현

3.1 시스템 구조

그림 3.1에 보인것과 같이 제안한 시스템의 구조는 MVC 모델과 같다. XMLRetriver와 ShowMatch는 자바 servlet 프로그램으로서 클라이언트로부터 요구를 받아서 수행할 책임이 있다. 에이전트 프로그램 내에 있는 모든 클래스와 에이전트 프로그램 그 자체는 모두 자바프로그램이다. 그리고 모든 탐색 결과는 JSP를 통하여 보인다.

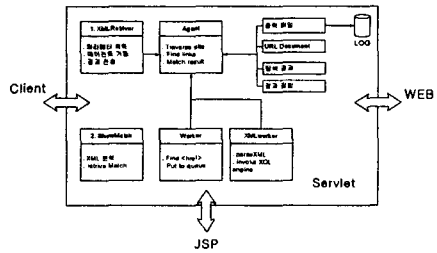

그림 3.1 시스템 구조

3.2 구현 결과

시스템을 구현한 결과는 다음 그림3.2와 3.3과 같다.

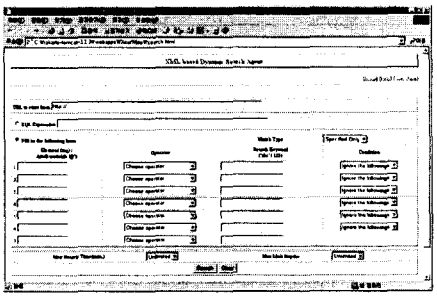

그림 3.2서치 에이전트 시작페이지

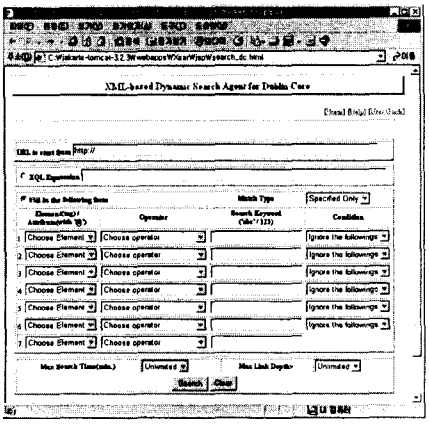

그림 3.3 서치에이전트 실행 예

V. 결 론

제안한 탐색엔진 에이전트는 현재 사용되고 있는 키워드 기반 검색엔진에서 보이고 있는 심각한 문제들을 해결할 수 있는 잠재적인 능력이 있음을 확인하였다. 그러나 이러한 잠재력은 좀더 정밀한 실험 계획을 세워서 평가를 수행할 계획이다. 네트워크 환경과 컴퓨터 성능이 향상되므로서 제안한 시스템의 접근기법은 인터넷 정보 검색에 의한 자료 획득 과정에서 획기적인 변혁을 가져올 수 있을 것이다.

참고 문헌

[1] Tim Bray, Jean Paolo, C.M. Sperberg-McQueen, Eve Maler, XML 1.0
[2] Lycos, Inc., Lycos, See http://www.lycos.com
[3] Yahoo!Inc., Yahoo!
[4] Sun microsystem, Inc., Java API
[5] Sun microsystem, Inc., Java Mail
[6] Sun microsystem, Inc., JDBC
[7] Sun microsystem, Inc., Java CORBA