XML 기반의 지능 에이전트 시스템의 설계 및 구현

†조영상[1] 정의현[2] 홍도원[2] 박용진[1]
한양대학교 전자공학과 네트워크 컴퓨팅 연구실[1]
대우통신 통신 소프트웨어 연구실[2]
{ysch0, park}@nclab.hanyang.ac.kr[1] {ehjung, dwhong}@moon.daewoo.co.kr[2]

A Design and Implementation Intelligent Agent System based on the XML

Young-Sang Cho[1] Eui-Hyun Jung[2] Do-Won Hong[2] Yong-Jin Park[1]
Dept. of Electronic Engineering, Hanyang University[1]
Telecommunication S/W Dept. Daewoo Telecomm. [2]

요 약

본 논문에서는 XML(eXtensible Markup Language)을 이용하여 지능형 에이전트 시스템의 기능을 동적으로 확장할 수 있는 지능 에이전트 시스템을 설계하고 이를 구현한다. 본 논문에서 제안한 지능 에이전트 구조는 지능 에이전트가 XML로 제시된 다수의 기능(behavior)을 원격지의 에이전트 기능 서버에서 다운로드 받아, 실행 시점에서 조립하여 실행할 수 있는 구조를 갖고 있다. 이 구조는 에이전트에게 다양한 기능을 실행시점에서 제공할 수 있으므로, 기능 중심의 적응성과 확장성을 제공할 수 있을 것으로 기대된다.

1. 서론

인터넷의 활성화로 인한 폭발적인 정보량의 증가는 단순한 정보의 수집이나 검색을 제공하는 기존 시스템의 사용자에게 적절한 시간 내에 정보를 소비하지 못하는 정보과다(Information Overload) 문제를 야기하고 있다. 이 때문에 다량의 정보를 사용자의 사전 요구에 맞추어 여과하고, 사용자의 개입 없이 처리하며, 이를 적절한 시간에 제공하는 지능 에이전트(Intelligent Agent)[1][2]의 필요성이 증대되고 있다.

일반적으로 지능 에이전트가 업무를 자율적으로 처리하기 위해서는 주도적 판단을 위한 지능(intelligence)과 업무처리를 할 수 있는 실제적인 기능(function or behavior)을 가져야 한다. 하지만 기존의 소프트웨어 공학적인 측면에서 설계된 지능 에이전트는 소유한 지능과 기능이 설계 시에 고정이 되기 때문에, 상황이 변동되는 경우에 적응해서 동적으로 지능과 기능을 확장하는 것이 쉽지 않은 문제점을 가진다. 특히, 에이전트의 동적인 기능 확장은 매우 어려운 문제로 생각되고 있는데, 이것은 지금까지의 에이전트 설계 방안이 에이전트 자체를 하나의 단일한 실행 모듈로 간주하여 에이전트를 설계하였기 때문에, 에이전트에 기능을 추가하기 위해서는 새로운 에이전트를 재작성 하거나, 기존 에이전트로부터 상속(inheritance)을 받아 구축하는 접근 방법밖에는 없기 때문이다. 그러나, 이런 접근 방안에서는 운용시(on-the-fly)에 동적인 확장이 불가능할 뿐더러, 기존 시스템에 추가될 때 필연적으로 상호운용성(interoperability) 문제를 야기하게 되기 때문에, 시스템을 재구성해야 하는 문제점을 야기한다.[3]

본 논문에서는 XML[4]과 Java의 원격 클래스로딩(Remote Class Loading)[5] 개념을 이용하여 운용되는 시스템에 영향을 미치지 않으면서 기능을 운용시에 동적으로 확장할 수 있는 지능 에이전트 구조를 설계 및 구현한다.

2. 기존 지능 에이전트 시스템의 고찰

현재의 에이전트의 시스템 구조에 대한 연구는 에이전트 시스템을 구성하고 있는 엔진이나 통신 모듈을 효과적으로 구축하기 위한 소프트웨어 공학적인 구축방안에 대한 연구가 주로 행해지고 있다. 그러나, 지능 에이전트를 구축하기 위해서는 단순한 소프트웨어 구성과 달리 에이전트의 특성인 자율성과 적응성, 지능을 갖도록 구성해야 한다. 하지만 기존의 시스템은 이러한 특성을 구현하는데 있어 다음과 같은 문제점을 가지고 있다.

(1) 지능 에이전트의 동적인 기능확장

에이전트의 기능이 설계 시에 고정되는 기존의 소프트웨어 개발 방식으로는 에이전트의 기능을 동적으로 확장하기 곤란하다. 이는 기존 소프트웨어 구조가 에이전트를 하나의 결합된 실행 모듈로 간주하는 정적결합(static-binding) 모델을 이용하기 때문이다. 이런 구조에서는 에이전트의 기능이 동적으로 추가되거나 변경되기 위해서는, 해당 기능을 가진 에이전트를 새롭게 설계·작성하여 이를 기존 시스템에 연동해주어야 한다.

(2) 기능 확장시의 상호 운용성 문제

에이전트 시스템이 구축된 후의 에이전트의 기능 확장은 기존 시스템과의 상호운용에 문제를 야기한다. 이는 에이전트를 기반으로 구축된 시스템은 에이전트에 종속적으로 구현되기 때문에, 에이전트의 구조가 바뀔 경우 시스템 전체를 수정해야 하는 문제점이 생기기 때문이다. 또한, 서로 같은 종류의 에이전트가 설치된 시스템간에도 확장된 에이전트가 설치된 시스템과 기존 시스템은 상호 운용할 수 없다.

3. XML 을 이용한 에이전트 설계 기법

XML은 W3C(World Wide Web Consortium)에서 제안한 새

로운 태그(Tag) 언어이며 문서의 내용(content)을 나타낼 수 있는 언어이기 때문에 기계가 분석하기 용이하고 기존 정보 시스템에 유연하게 결합이 가능한 특징을 갖고 있다[4]. 자바를 이용해서 에이전트 시스템을 구축하는 경우, 에이전트의 기능은 기본적으로 자바 가상머신 내의 클래스 인스턴스를 의미한다. 따라서, 기존 에이전트에 기능을 추가하기 위해서는 기능에 해당하는 클래스의 메서드를 호출할 수 있어야 한다. 에이전트의 기능을 확장하기 위해서는 두 가지 문제가 해결되어야 한다. 첫째로, 해당 에이전트에 주어진 기능을 에이전트가 해석할 수 있는 데이터 형이 정의되어야 한다. 두 번째로, 에이전트의 기능에 해당하는 클래스 인스턴스를 동적으로 활성화하고 에이전트가 해당 클래스 인스턴스를 참조할 수 있어야 한다. 이러한 사전 요구를 해결하기 위하여, 본 논문에서는 XML과 원격 클래스로딩 개념을 결합하였다.

3.1. 에이전트의 기능 명세서

에이전트의 기능을 명시하기 위한 방안으로 본 논문에서는 XML을 이용하였다. XML을 이용함으로써, 동작의 확장 부분에 대한 기능 명세와 실제 기능의 구현을 분리할 수 있게 된다. 사용자는 에이전트의 기능을 명시하기 위해서 XML을 이용하여 기능 명세서를 작성한다. 이 명세서는 지능 에이전트에 의해 동작 노드를 요소로 갖는 DOM(Document Object Model)[6] 문서 구조체로 변환된다. 에이전트의 기능이 소스 차원에서 코딩된 경우, 새로운 기능을 추가하기 위해서는 에이전트의 코드를 모두 바꿔주어야 한다. 그러나 XML로 기능 명세서를 주는 경우에는 에이전트의 기능이 DOM 구조체로 나타나기 때문에, 에이전트는 단지 XML을 파싱하고 해당 기능을 로딩하는 기능을 가지면 된다. 또 한가지 장점은, 사용자 입장에서도 사용자가 이해할 수 있는 언어로 에이전트의 동작을 지정할 수 있다는 장점을 갖는다. 그림 1은 각 단계에서의 태그와 시스템간의 관계를 보여준다.

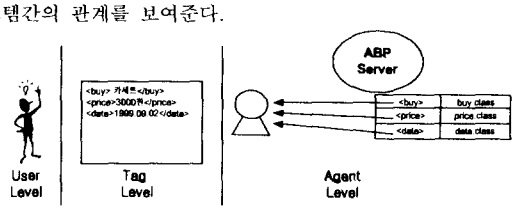

그림 1

3.2 기능의 원격 클래스로딩

에이전트의 DOM 문서 구조체에서 정의된 동작은 ABP (Agent Behavior Pool) 서버에서 다운로드되어, 에이전트가 위치한 에이전트 엔진으로 클래스로딩된다. 다운로드되어 클래스 인스턴스로 바뀐 동작은 에이전트의 동작으로 등록되며, 에이전트는 이중에서 주 동작(main behavior)에 해당하는 동작 클래스 인스턴스를 호출한다. 그리고, 호출된 주 동작은 에이전트에 등록된 다른 동작들을 제어하게 된다. 본 논문에서는 이러한 에이전트의 동적 기능확장을 위하여, 자바의 원격 클래스로딩을 이용하였다. 원격 클래스로딩은 원격지의 클래스 파일을 로컬(local) 자바 가상머신의 클래스 인스턴스로 바꿀 수 있는 메커니즘이다.

이러한 XML과 자바의 특징을 이용하여 에이전트의 기능에 해당하는 모듈을 XML과 자바 클래스를 이용하여 ABP 서버로 구축한다. 이 ABP 서버를 사용하여 동적으로 에이전트의 기능을 확장하고, 같은 ABP를 사용하는 에이전트간에 통신을 가능하게 한다. 이런 구조를 이용하는 경우, 사용자가 원하는 기능을 사전에 에이전트에 코딩 할 필요가 없으며, 새로운 기능을 필요로 하는 경우, ABP 서버에만 태그와 해당 클래스를 추가해주면 된다.

4. 시스템의 세부구현

본 논문에서 제안하는 지능에이전트 시스템을 적용한 프로토타입으로 에이전트 뉴스 시스템을 구현하였다.

4.1. ActiveNews 시스템 개요

ActiveNews 시스템은 기존 Usenet이나 게시판 시스템과 달리 사용자 에이전트가 뉴스 디렉토리를 감시하다가 사용자가 원하는 뉴스가 나타났을 때, 이를 사용자에게 알려주는 구조를 갖는다. ActiveNews 시스템의 서버구성은 그림 2과 같이 각 동작을 처리하는 동작 클래스를 저장할 수 있는 ABP 서버와 실제의 지능 에이전트가 생성되고 실행되는 여러 개의 에이전트 엔진으로 구성되어 있으며, 하나의 ABP 서버에 여러 ActiveNews 폴더 시스템이 결합된 구조를 갖는다. 본 시스템의 사용자는 XML로 명시된 에이전트 동작 명세서를 서버로 보내고 서버 시스템은 ABP로부터 동적으로 기능 추가가 가능하다.

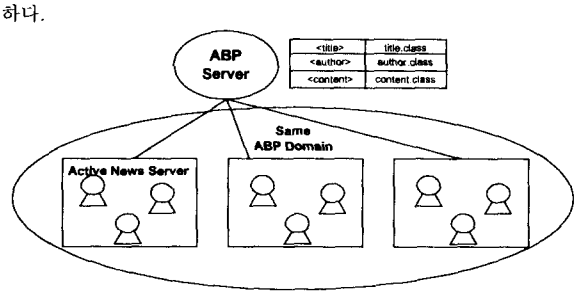

그림 2

4.2. 에이전트 엔진

본 논문에서 제안한 DTD는 크게 지능 에이전트 자체의 정보를 나타내는 부분, 지능 에이전트가 필요로 하는 부가 정보를 갖는 부분, 그리고 지능 에이전트의 동작을 지정하는 부분으로 나뉘어진다. 이렇게 구성된 DTD 각 부분의 세부 태그는 XML의 특성상 쉽게 확장이 가능하다. 최상위 태그를 제외한 태그들은 구성하려는 에이전트의 기능 요구를 만족하기 위해 설계시에 수정될 수 있으며, 심지어는 에이전트 시스템의 실행 도중에도 동적인 변경이 가능하다.

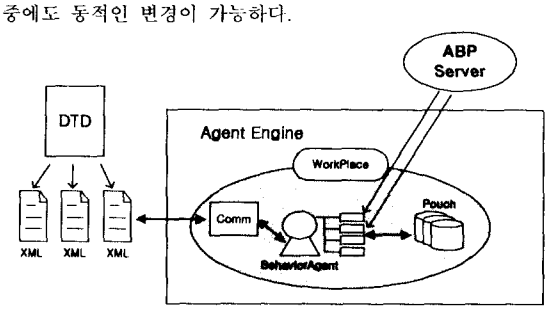

그림 3

에이전트 엔진은 위의 그림 3에서 볼 수 있는 것처럼 지능 에이전트의 프로필(profile)과 동작을 정의한 DTD와 에이전트의 명세를 지시하는 XML 문서, 엔진의 구성 관리를 담당하는 WorkPlace 클래스, 지능 에이전트가 요구하는 데이터를 저장

하는 역할을 해주는 Pouch 클래스, 통신 기능을 제공하는 Comm 클래스로 구성되어 있다. 그리고, 지능 에이전트 동작을 해주는 BehaviorAgent는 내부적으로 XML 형식의 기능 명세서를 파싱한 Node 클래스와 이를 처리해주는 여러 개의 Behavior 클래스를 ABP 서버로부터 로딩하여 등록한다.

4.3 실행 구조

사용자가 지능 에이전트를 이용하기 위해서는 에이전트 명세서를 XML로 서버로 전송해 주어야 한다. 이때, 엔진은 에이전트 명세서에 지시된 지능 에이전트를 생성해 주어야 한다. 지능 에이전트를 생성하기 위해 에이전트 엔진은 에이전트 명세서를 XML 파싱한 후, ABP 서버로부터 해당 클래스를 다운로드 받아 원격 클래스로딩을 이용해 조립한다. 이렇게 함으로써 원하는 에이전트의 기능을 동적으로 조립할 수 있다.

이 시스템의 사용자는 issuer와 reviewer의 두 종류로 나뉘며 아래와 같은 과정으로 시스템을 사용한다. (그림 4 참조)

① issuer가 서버에 접속하면 사용자 시스템은 ABP 서버로부터 뉴스를 작성하는 데 사용할 인터페이스와 모듈에 대한 정보가 기술된 XML을 다운 받아 각 모듈을 로딩한다. 따라서 시스템에 새로운 기능이 추가되거나 XML 형식이 변경되어도 즉시 반영이 가능하다.

② issuer가 자신의 뉴스를 작성하면 사용자 시스템이 XML 형식으로 뉴스 폴더 시스템에 추가한다.

③ reviewer는 issuer들이 보낸 뉴스를 에이전트를 통해서 검색할 수 있다. 이때, 에이전트에게 자신이 원하는 뉴스의 타입과 조건이 기술된 기능 명세서를 사용자 시스템을 이용해서 XML형태로 넘겨준다.

④ 받은 기능 명세서를 기반으로 에이전트는 ABP 서버에서 해당 자바 클래스를 다운로드 받아서 에이전트의 기능을 동적으로 추가하여 필요한 작업을 수행한다.

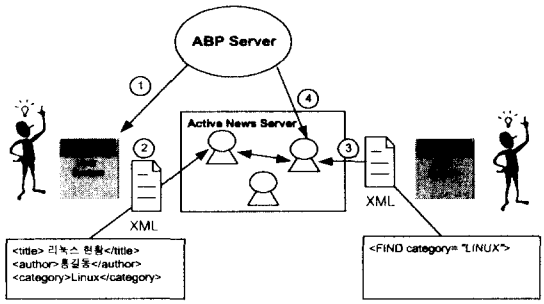

그림 4

4.4 동적 확장 검증

본 논문에서 제시한 기능의 동적 확장을 검증하기 위하여, 본 논문에서는 기본 DTD 모델에 <NOTIFY>/<WATCH>라는 새로운 태그와 처리 클래스를 ABP 서버에 추가하고, 이에 의한 새로운 기능이 기존 시스템과의 충돌 없이 가능함을 보인다.

설계의 목적에 따라 에이전트를 새로 구현하는 것이 아니고 ABP에 해당 XML 태그와 클래스를 추가함으로써 에이전트의 기능을 동적으로 확대시킬 수 있음을 보이는 것을 목표로 한다. 기본 DTD 정의에서는 검색 기능을 가진 <FIND> 태그 외에는 정의하지 않았으나, <NOTIFY>/<WATCH>라는 새로운 태그를 정의하는 확장 DTD를 적용하는 경우에는 <NOTIFY>/<WATCH> 태그를 처리하기 위한 BehaviorNotify 클래스와 확장 DTD를 작성하고, 또한 태그의 추가로 인해 추가되는 사용자 시스템의 인터페이스와 동작모듈도 작성하여 이것을 사용자 시스템에서 로딩할 수 있게 사용자 시스템으로 보내는 XML 문서도 수정하였다.

수정한 시스템은 다음과 같이 동작한다. (그림 5 참조)

① 수정한 DTD와 추가된 태그를 처리하기 위한 클래스, 사용자 시스템을 위한 추가 인터페이스와 동작모듈을 ABP 서버에 추가한다.

② 사용자 시스템은 추가된 인터페이스와 동작 모듈을 다운로드 받아 로딩한다.

③ 사용자는 추가된 기능을 사용하기 위해 XML 형식의 기능 명세서를 보낸다.

④ 사용자가 보낸 기능 명세서에 따라 에이전트는 추가된 기능을 ABP 서버로부터 다운로드 받아서 추가하고 필요한 작업을 수행한다.

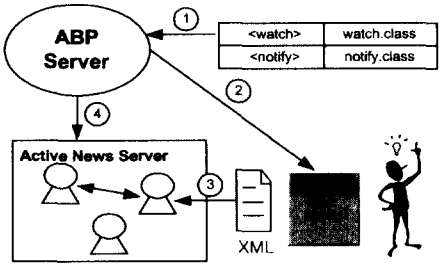

그림 5

실험 결과, 사용자 시스템에서는 새로운 태그에 해당하는 정보를 지원하도록 사용자 인터페이스가 자동으로 갱신되었으며, 서버에서 실제로 해당 동작이 지원되는 것이 확인되었다.

5. 결론 및 향후과제

기존의 에이전트 시스템은 설계 시에 기능이 고정되기 때문에 동작 시에 동적으로 기능을 확장하는 것이 불가능하였다. 본 논문에서는 이 문제를 해결하고자 XML과 원격클래스로딩을 사용하여, 동적인 기능 확장이 가능한 에이전트 시스템을 설계 및 구현하였다.

앞으로는 에이전트의 적응형 구조와 동작간의 충돌회피 알고리즘에 관하여 연구가 심화되어야 할 것이다.

6. 참고 문헌

1. P. Mates, "Agent that Reduce Work and Information Overload", CACM, vol.37, no.7, pp.31-40, Jul. 1994

2. D. Riecken, "Intelligent Agents", CACM, vol.37, no.7, pp.18-21, Jul. 1994

3. M. J. Wooldridge and et al., "Software Engineering with Agents: Pitfalls and Pratfalls", IEEE Internet Computing, May·June 1999

4. R. Khare and et al., "XML: A Door to Automated Web Applications", IEEE Internet Computing, July·August 1997

5. R. Macgregor and et al., "Java Network Security", pp.77-93, Prentice Hall PTR, 1998

6. "Document Object Model (DOM) Level 1 Specification", W3C, "http://www.w3.org/TR/REC-DOM-Level-1/", 1998