메시지 후킹 메커니즘을 이용한 적응형 하이퍼미디어 시스템과 외부 응용 프로그램의 결합

정효숙[†] · 박성빈[††]

요 약

사용자는 적응형 하이퍼미디어를 항해하면서 적응형 하이퍼미디어가 아닌 다른 응용 프로그램을 함께 사용할 수도 있다. 만일 사용자가 그 응용 프로그램을 통해 적응형 하이퍼미디어와 관련된 정보에 접근하였다면, 이러한 사용자의 행동은 적응형 하이퍼미디어가 제공하는 내용에 대한 사용자의 지식이나 흥미에 영향을 줄 수 있다. 그러나 적응형 하이퍼미디어 시스템은 페이지 접근과 같은 탐색 활동을 통해 사용자 행동을 이해하며, 다른 응용 프로그램에서 발생한 사용자의 행동을 인식하여 사용자 속성을 변화시키기 어렵다. 본 논문에서는 적응형 하이퍼미디어 시스템이 다른 응용 프로그램에서 발생한 사용자 행동을 인식하여 사용자 프로파일을 갱신시킬 수 있도록 함으로써 현재 사용자의 특성을 보다 정확하게 파악하여 적응형 내용 제시와 적응형 항해를 제공하고자 한다. 후킹 메커니즘을 이용하여 다른 응용 프로그램에서 발생한 사용자 이벤트를 분석하고, XML 번역기를 이용하여 시스템에 저장된 사용자의 프로파일을 갱신할 것이다.

키워드 : 적응형 하이퍼미디어, 메시지 후킹

Combination of an adaptive hypermedia system and an external application using a message hooking mechanism

Hyosook Jung[†] · Seongbin Park[††]

ABSTRACT

While a user is using an adaptive hypermedia system, the user can also use an external application. If the user accesses the information which is related to the contents provided by the adaptive hypermedia system, it can affect a user profile that contains the information about the knowledge or interests of the user. However, the adaptive hypermedia system understands user's behavior based on whether a page is accessed or not and it is difficult for the system to recognize user's behavior that can occur outside the adaptive hypermedia system. In this paper, we propose an approach that can detect user's behavior using a message hooking mechanism so that both user's behavior inside an adaptive hypermedia system and behaviors that occur outside the system can be reflected in a user profile. We analyze user events using a hooking mechanism and update a user profile using an XML parser.

Keywords : adaptive hypermedia, message hooking

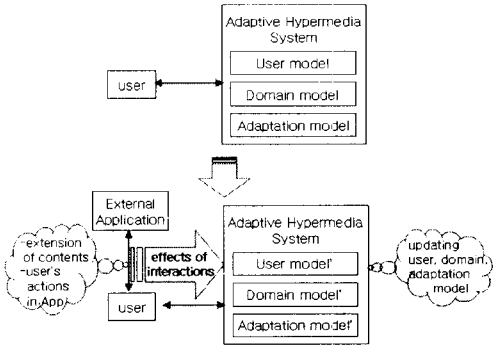
1. 서 론

하이퍼미디어(hypermedia)는 사용자들이 수많은 정보 노드(node)를 자유롭게 항해할 수 있도록 하기 때문에 정보를 관리하고 검색하는데 편

† 정 회 원: 고려대학교 컴퓨터교육학과 박사과정
†† 종신회원: 고려대학교 컴퓨터교육과 교수(교신저자)
논문접수: 2005년 6월 21일, 심사완료: 2005년 7월 14일

리하다. 그러나 자유로운 항해로 인해 사용자들은 방향 상실감(disorientation)이나 인지적 과부하(cognitive overload)를 경험하게 된다. 이러한 문제점을 극복하기 위해서 적응형 하이퍼미디어(adaptive hypermedia)는 사용자 모델(user model)을 기반으로 개별화된 컨텐츠와 링크를 제공한다[1]. 적응형 하이퍼미디어 시스템은 사용자 모델을 통해서 사용자의 특징을 반영하고 사용자에게 시스템의 다양한 시각적 측면을 적응적으로 제공하는데 사용자 모델을 적용하는 모든 하이퍼텍스트와 하이퍼미디어 시스템을 의미한다[2]. 적응형 하이퍼미디어 시스템은 세 가지 모델로 구성되어 있다. 사용자 모델은 사용자들의 다양한 특성을 저장하고 관리하며, 도메인 모델(domain model)은 시스템이 제공할 내용을 개념(concept)과 개념 간의 관계(concept relationship)로 표현하고 있으며, 적응성 모델(adaptation model)은 사용자 모델과 도메인 모델을 연결하여 어떻게 적응성을 수행할 것인가에 대한 규칙을 담고 있다[3]. 적응성을 제공한다는 의미는 저작자가 적응형 하이퍼미디어를 만들기 위해 어떤 방법을 사용할 것인가를 결정하고 위의 세 가지 모델을 설계하는 것을 의미한다. 따라서 적응성을 제대로 수행하기 위해서 저작자는 사용자의 행동과 시스템을 통해 제공할 내용에 대해 정확히 알고 세 가지 모델을 설계해야 한다.

그러나 적응형 하이퍼미디어 시스템이 예상하지 못한 사용자의 행동, 즉, 사용자가 적응형 하이퍼미디어를 사용하면서 다른 응용 프로그램을 사용할 때 발생하는 행동들이 적응형 하이퍼미디어 시스템에서 정의한 세 가지 모델과 관련성이 있는 유의미한 행동일 때, 적응형 하이퍼미디어 시스템이 현재 사용자의 특성을 정확하게 반영하여 적응성을 발휘한다고 볼 수 없다. 예를 들어, 사용자가 적응형 하이퍼미디어를 사용하면서 웹 브라우저를 실행시켜 적응형 하이퍼미디어의 내용과 관련된 정보를 찾아 볼 수도 있다. 이러한 사용자의 행동은 적응형 하이퍼미디어의 특정 내용과 관련된 지식이나 흥미를 증가시켜 내용 이해를 촉진시킬 수 있다. 또한 검색된 외부의 정보가 유용한 것이라면 도메인 모델에 새로운 개념으로써 추가시킬 수도 있으며, 다른 개념들과의 관계를 형성해야 할 것이다. 이러한 도메인 모델의 변화는 사용자 모델에 새로운 개념과 관련된 속성을 추가시켜야 하는 상황을 발생시킨다. 이로 인해 새로운 적응성 규칙들도 필요하게 되므로 적응성 모델도 변화시켜야 할 것이다.



[그림 1] 외부 응용 프로그램내의 사용자 행동이 적응형 하이퍼미디어 시스템에 미치는 영향

[그림 1]은 사용자가 적응형 하이퍼미디어 시스템을 사용하면서 외부 응용 프로그램을 사용할 때, 그 프로그램에서 발생한 이벤트가 적응형 하이퍼미디어 시스템에 미치는 영향을 나타내고 있으며, [표 1]은 시스템의 사용자 모델, 도메인 모델, 적응형 모델에 어떤 변화가 발생하는지를 이벤트가 발생하기 전과 발생한 후를 비교한 것이다.

[표 1] 외부 응용 프로그램내의 사용자 행동으로 인한 적응형 하이퍼미디어 시스템의 세 모델의 변화

이벤트 모델	외부 응용 프로그램 실행 전	외부 응용 프로그램 실행 후
사용자 모델	미리 설계된 사용자 속성	사용자 프로파일의 속성값 변경 및 새로운 속성 추가
도메인 모델	미리 설계된 개념들 및 개념들 간의 관계	새로운 개념 및 개념들 간의 관계 추가
적응성 모델	미리 설계된 적응성 규칙	새로운 적응성 규칙 추가

그러나 일반적으로 적응형 하이퍼미디어 시스템은 사용자가 하이퍼미디어 링크를 순서대로 따라가는 것과 같은 사용자의 행동을 관찰하여 사용자 모델링을 수행한다. 즉, 사용자가 적응형 하이퍼미디어를 사용하는 과정에서 발생하는 탐색 활동에 관심을 두고 있다. 또한 시스템이 제공할 내용은 동적으로 변하는 것이 아니라, 저작자에 의해 이미 결정된 정적인 것이다[4]. 본 논문에서는 외부 응용 프로그램에서 발생한 사용자의 행동 인식 및 사용자 모델의 변화에 대한 문제를

해결하기 위해서 메시지 후킹 메커니즘[5]을 사용하여 외부 응용 프로그램에서 사용자 이벤트로 인해 생성된 메시지들을 모니터하면서 특정 메시지가 발생하였을 때, 적응형 하이퍼미디어 시스템의 사용자 모델을 갱신시키도록 하였다. 또한 이렇게 갱신된 사용자 모델을 기반으로 한 적응형 하이퍼미디어 시스템은 그 사용자에게 적응형 내용과 적응형 항해를 제공할 수 있게 되었다.

2. 관련 연구

2.1. 적응성의 개념

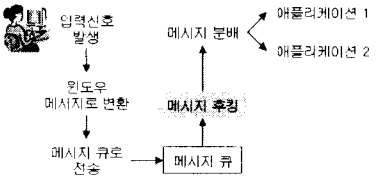
적응형 하이퍼미디어에서의 적응성은 지식수준, 목적, 흥미 등 사용자의 다양한 특성을 표현하고 있는 사용자 모델을 기반으로 사용자가 거대한 하이퍼 공간에서 자유롭게 항해하면서도 그들에게 개별화된 정보를 안내하는 것을 말한다[3]. 적응성은 내용 적응성(content adaptation)과 링크 적응성(link adaptation)으로 나눌 수 있다. 내용 적응성은 사용자의 지식, 목적, 개인적 특성 등에 따라 다른 방식으로 어떤 주제에 대한 정보를 제시하는 것이다. 링크 적응성은 링크의 구조를 변화시키거나 설명을 첨가하는 방식으로서 일반적인 하이퍼미디어의 특성인 자유로운 항해 기능을 유지하면서도 방향 상실감의 문제를 줄일 수 있도록 링크 구조를 단순화하려는 것이다[2].

2.2. NEO_VAM과 AHA!의 결합

NEO_VAM[6]은 학습자가 오토마타 이론의 개념을 이해하는데 도움을 주기 위한 상호작용 시뮬레이션 학습 프로그램이다. [7]에서는 비록 NEO_VAM 자체는 적응성 기능을 구현하고 있지 않지만, 학생들에게 적응적으로 학습 내용을 제공하기 위해서 NEO_VAM과 AHA! 시스템[8]을 결합하였다. NEO_VAM은 사용자의 행동 중에서 학습 진행에 영향을 미치는 학습자의 행동(예를 들어, 기계나 전이함수의 생성)을 인식하여 그 행동에 대한 메시지를 생성한 후, AHA! 시스템에 전달한다. AHA! 시스템은 전달 받은 메시지에 따라 관련된 개념의 지식을 변화시킨다.

2.3. 윈도우 메시지 후킹

윈도우 운영 체제에서 훅은 이벤트(메시지, 마우스 움직임, 키보드 입력)가 응용 프로그램에 도착하기 전에 이벤트를 가로채는 방법을 제공한다[9]. 이 방법을 통해 이벤트에 반응하여 동작하거나 이벤트를 수정 또는 삭제할 수 있다. 예를 들어 어떤 훅 함수는 모든 종류의 키보드와 마우스 이벤트를 받을 수 있다. 일반적으로 윈도우즈 시스템은 입력 디바이스로부터 신호가 들어오면 윈도우즈 메시지로 변환을 해주고 그 메시지를 메시지 큐로 받아서 분배해주는 역할을 한다. 메시지 후킹은 [그림 2]와 같이 메시지 큐에서 흘러나오는 모든 메시지를 중간에 가로채거나 임의의 메시지만 걸러내서 가로채는 것으로서 메시지를 가로챈 후 다른 메시지로 변형시켜 넣는 것도 가능하다.



[그림 2] 메시지 후킹 메커니즘

3. 적응형 하이퍼미디어 시스템과 외부 응용 프로그램의 결합 유형

적응형 하이퍼미디어 시스템과 외부 응용 프로그램의 결합은 외부 응용 프로그램의 소스코드에 대한 접근 가능성과 적응형 하이퍼미디어의 사용자 이벤트 처리 능력에 따라 달라질 수 있다. 외부 응용 프로그램의 소스코드 접근 가능성이란 개발자가 연결하고자 하는 외부 응용 프로그램의 소스코드를 변경할 수 있는지, 없는지를 말한다. 적응형 하이퍼미디어 시스템의 사용자 이벤트 처리능력이란 적응형 하이퍼미디어 시스템이 외부 응용 프로그램에서 발생한 사용자 이벤트 정보, 즉 외부 이벤트도 감지할 수 있는지, 또는 단지 사용자가 적응형 하이퍼미디

어를 탐색하는 것과 같은 내부 이벤트만 감지할 수 있는지를 말한다. 이러한 요인에 따라, 본 논문에서는 [표 2]와 같이 네 가지 유형으로 분류하였다.

[표 2] 응용 프로그램과 적응형 하이퍼미디어 시스템 결합 유형

시스템 / 기준 / 유형	적응형 하이퍼미디어		외부 응용 프로그램	
	내부 이벤트	외부 이벤트	코드변경 가능	코드변경 불가능
1	○		○	
2	○			○
3	○	○	○	
4	○	○		○

① 유형 1 : 적응형 하이퍼미디어 시스템의 로컬 페이지를 접근하는 것과 같은 시스템 내부 이벤트만 처리할 수 있고 저작자가 외부 응용 프로그램의 소스코드에 접근하여 변경할 수 있는 경우이다. [유형 1]의 경우, 적응형 하이퍼미디어는 외부 응용 프로그램에서 발생한 사용자의 이벤트를 처리할 수 없지만, 소스코드를 변경할 수 있으므로 이를 변경하여 적응형 하이퍼미디어의 내부 이벤트를 유발시키도록 한다. 앞서 언급한 NEO_VAM과 AHA! 시스템의 결합은 [유형 1]에 속한다. [7]은 NEO_VAM에서 학습하면서 AHA! 시스템의 적응형 기능을 활용하기 위해 NEO_VAM의 소스코드를 변경한다. 즉, 학습에 중요한 영향을 미치는 사용자의 특정 이벤트가 발생하였을 때, AHA! 시스템의 로컬 페이지를 고의로 호출하도록 하여 사용자가 페이지를 접근한 것과 같은 효과를 발생시킨다. AHA! 시스템이 사용자 모델을 갱신하고, 이를 기반으로 적응형 기능을 수행하도록 하였다.

② 유형 2 : 적응형 하이퍼미디어 시스템은 자신이 제어할 수 있는 내부 이벤트만 처리할 수 있고 저작자는 외부 응용 프로그램의 소스코드에 접근이 불가능하여 응용 프로그램을 변경할 수 없는 경우이다. 따라서 두 시스템을 연결시키기 위해서는 새로운 메커니즘이 필요하게 된다. 본 논문에서는 메시지 후킹을 이용하여 두 시스템을 결합하고자 한다.

③ 유형 3 : 적응형 하이퍼미디어 시스템이 페이지 접근과 같은 내부 이벤트뿐만 아니라, 외부의 이벤트도 처리할 수 있으며, 외부 응용 프로그램의 소스코드에도 접근할 수 있는 경우이다. 적응형 하이퍼미디어 시스템에서 외부 응용 프로그램의 사용자 이벤트에 따라 알맞은 적응성을 발휘하도록 하거나 외부 응용 프로그램의 소스코드를 직접 변경하여 적응형 하이퍼미디어의 내부 이벤트를 유발시키는 두 가지 모두 활용할 수 있으므로 저작자가 작업하기 편리하거나 개발 목적에 맞는 방법을 선택하면 된다.

④ 유형 4 : 적응형 하이퍼미디어 시스템이 내부 이벤트와 외부 이벤트를 모두 처리할 수 있지만, 외부 응용 프로그램의 소스코드에 접근할 수 없는 경우이다. [유형 4]의 경우, 적응형 하이퍼미디어 시스템에서 외부 응용 프로그램의 메시지를 처리할 수 있도록 적응성 규칙을 설계하면 된다.

4. 적응형 하이퍼미디어 시스템의 외부 이벤트 처리 기법 설계 및 구현

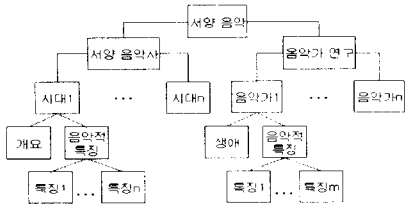
4.1. 적응형 하이퍼미디어의 생성

적응형 하이퍼미디어를 생성하기 위해서 AHA! 시스템을 활용하고자 한다. 본 논문에서는 [서양 음악사]와 [음악가 연구]에 대한 정보를 제공하는 적응형 하이퍼미디어를 구성하였다.

4.1.1. 도메인 모델

도메인 모델은 적응형 하이퍼미디어가 제공할 내용에 대한 개념적인 설명으로서 개념들과 개념들 간의 관계로 구성되어 있다. 사용자에게 보여지는 모든 페이지는 각각 상응하는 개념을 갖고 있어야 한다. 페이지들은 절이나 장과 같은 상위 수준의 그룹으로 묶일 수 있기 때문에 개념 구조는 계층적 구조를 형성하게 된다. 논문에서 생성한 적응형 하이퍼미디어의 도메인 모델의 개념들은 크게 서양 음악사와 음악가 연구로 나뉘며, 서양 음악사는 시대별로 나누어 각각의 개요와 특징들에 대한 내용을 제공하고 음악사 연구는 서양 음악사에 업적을 남긴 여러 음악가의 생애와 음악적 특징에 대한 내용을 제공한다. [그림

3]은 도메인 모델 개념들의 구조를 계층적으로 표현한 것이다.



[그림 3] 도메인 모델의 계층적 구조

[표 3]은 서양 음악사의 개요와 바로크 음악이 필요조건 관계를 형성하고 있음을 나타내고 있다. 즉, 바로크 음악은 서양 음악사의 개요에 대한 knowledge가 '0'보다 클 때, 즉 바로크 음악에 대한 내용을 읽기 전에 최소한 서양 음악사의 개요를 읽었을 때, 접근할 수 있음을 나타낸다.

[표 3] 서양 음악사의 개요와 바로크 음악이 필요조건 관계

```
<concept>
 <name>music.history.baroque</name>
 <description>baroque</description>
 <expr>((music.history.overview.knowledge>0 ))</expr>
</concept>
```

4.1.2. 사용자 모델

사용자 모델은 속성과 속성값을 지닌 개념들의 집합으로 구성되어 있다. [표 4]는 음악가 중에서 [바흐] 개념에 대한 초기화된 사용자 프로파일 중 일부분을 보여주고 있다.

[표 4] 초기화된 사용자 프로파일의 일부분

```
<record>
    <key>music.musician.bach.knowledge</key>
    <type>1</type>
    <persistent>true</persistent>
    <value>0</value>
</record>
<record>
    <key>music.musician.bach.interest</key>
    <type>1</type>
    <persistent>true</persistent>
    <value>0</value>
</record>
<record>
    <key>music.musician.bach.visited</key>
    <type>1</type>
    <persistent>true</persistent>
    <value>0</value>
</record>
```

사용자가 AHA! 시스템에 처음 등록하면 개인 프로파일이 생성되며 각각의 속성값은 '0'으로 초기화된다. 사용자 모델은 오버레이 모델(overlay model)로서 도메인 모델의 각 개념에 대응하는 개념이 사용자 모델에 존재한다. 또한 도메인 모델과는 상관없지만 시스템에 접속할 때 필요한 사용자의 ID나 비밀번호와 같은 사용자 개인의 특성을 표현하는 개념도 포함되어 있다. 각 개념은 미리 정의된 속성을 갖고 있는데, 예를 들어, 'knowledge'와 'interest'는 어떤 개념에 대한 사용자의 지식과 흥미를 말한다. 이러한 속성들의 값은 사용자가 학습하는 동안 점차 증가된다.

4.1.3. 적응성 모델

적응성 모델은 어떻게 사용자의 행동이 사용자 모델 갱신에 반영되고 적응형 내용 제시와 적응형 항해를 생성하는지를 정의하는 적응성 규칙들로 구성되어 있다. 사용자가 AHA! 시스템에서 제공하는 페이지에서 어떤 링크를 클릭하는 순간, AHA! 시스템은 링크의 개념이 어떤 페이지와 연결되어 있는지 또는 페이지가 어떤 개념과 연결되어 있는지 찾는다. 적응성 모델에 의해 작동하는 적응성 엔진(adaptation engine)은 이 순간 요청된 개념의 'access'속성과 관련된 규칙들을 실행시킨다. 각 규칙은 사용자 모델에서 속성값을 갱신하고, AHA! 시스템은 규칙이 실행될 때 무슨 페이지를 사용자에게 보여줄 것인지 결정한다. [표 5]는 음악가 [바흐] 개념과 관련된 적응성 규칙 중 사용자가 해당 페이지에 접근 하였을 때, 적용되는 규칙의 일부를 나타내고 있다. 사용자가 처음 [바흐] 페이지를 접근하였다면, [바흐] 개념에 대한 'knowledge'는 '10'으로 할당되고, [바흐] 페이지를 다시 접근할 때마다 원래 'knowledge'의 10 퍼센트씩 증가하게 된다.

[표 5] 적응성 규칙의 일부분

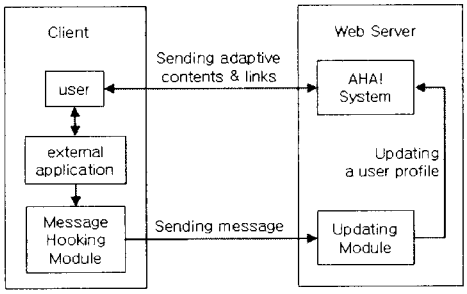
```
<assignment>
  <variable>music.musician.bach.knowledge</variable>
  <expr>10 </expr>
</assignment>
<assignment>
  <variable>music.musician.bach.knowledge</variable>
  <expr>music.musician.bach.knowledge+(0.1*
        _musicmusician.bach.knowledge) </expr>
</assignment>
```

4.2. 적응형 하이퍼미디어의 외부 이벤트 처리

외부 응용 프로그램에서 발생한 이벤트 메시지

를 적응형 하이퍼미디어 시스템에 전달하여 사용자의 프로파일을 변경시키는 작업은 크게 두 가지 과정으로 나누어 볼 수 있다.

첫째, 클라이언트에서 실행시킨 외부 응용 프로그램에서 발생한 메시지들을 후킹하면서 사용자 프로파일의 속성을 변경시키는 조건을 만족하는지 판단한 후, 조건을 만족시킨다면, 이를 서버에 알린다. 둘째, 서버는 클라이언트로부터 전달받은 정보를 통해 사용자 프로파일의 어떤 부분을 어떻게 변경할 것인지 판단한 후, AHA! 시스템에 저장된 사용자 프로파일을 찾아 변경시킨다. AHA! 시스템은 갱신된 프로파일에 따라 적응형 내용 제시와 적응형 항해를 제공하게 된다. [그림 4]는 사용자의 로컬 컴퓨터에서 발생한 메시지를 후킹 하여 사용자 모델을 갱신하는 과정을 나타내고 있다.



[그림 4] 메시지 후킹을 통한 사용자 모델 갱신

'Message Hooking Module'은 어떤 이벤트를 가로챌 것인지, 가로챈 이벤트를 어떻게 처리할 것인지를 DLL 파일에 정의한 후, DLL 파일에 정의된 함수들을 이용하여 사용자의 이벤트 메시지를 후킹한다. 예를 들어, 사용자가 적응형 하이퍼미디어 응용 프로그램의 내용을 탐색하면서 외부 사이트로 연결되는 링크를 클릭했을 때, 새로운 인터넷 익스플로러 창이 실행된다. 인터넷 익스플로러에서 발생한 사용자의 이벤트 메시지 중에서 현재 열려있는 적응형 하이퍼미디어의 페이지 정보, 실행시킨 파일들에 대한 정보, 키보드 이벤트 등을 후킹 한다. 후킹한 메시지들이 특정 조건을 만족시키는지 판단한 후, 조건을 만족한다면, HTTP 프로토콜 통해 후킹 정보(예를 들어, 현재 하이퍼미디어 페이지 이름, 실행시켰던 파일들의 이름 등)를 서버의 'Updating Module'에게 전송한다.

'Updating Module'은 우선 사용자 컴퓨터에 설정된 쿠키 정보를 통해 얻은 사용자 정보(예를 들어, 사용자 ID)를 이용하여 AHA! 시스템에 저장된 해당 사용자의 프로파일을 찾는다. AHA! 시스템은 사용자가 등록하였을 때, 사용자 ID에 대한 인덱스를 생성하고 각 사용자의 프로파일은 'index.xml'의 형식으로 저장되어 있다. 예를 들어, 사용자 ID의 index가 '100'이라면, 사용자 프로파일은 '100.xml'이 된다. 따라서 'Updating Module' 사용자 ID를 받아서 AHA! 시스템이 운영되고 있는 서버의 디렉토리에서 인덱스를 관리하는 파일을 찾고, 사용자 ID에 해당 인덱스를 검색한 후, 인덱스를 가지고 서버에 저장된 사용자 프로파일을 찾는다.

후킹 정보를 분석하여 사용자 프로파일의 어떤 속성을 변경시킬 것인지 판단하기 위해 현재 하이퍼미디어 페이지 이름을 이용하여 페이지가 어떤 개념과 연결되어 있는지 찾은 후, 그 개념과 관련된 속성의 값을 변경시킨다. 예를 들어, 후킹한 정보가 해당 개념의 knowledge 값을 증가시키는 조건을 만족시킨다면 사용자 프로파일에서 knowledge 속성값을 증가시킨다. 사용자 프로파일은 XML로 작성되어 있으므로 XML 번역기(예를 들어, Xerces)를 이용하여 XML 파일을 파싱한다. DOM API를 사용하여 XML 문서의 내용에 접근 한 후, 문서의 일부분(특정 개념의 속성값)을 원하는 내용으로 변경해야 한다. 즉, 사용자 프로파일의 XML 파일명과 변경해야 하는 속성 및 속성값을 가지고 DOM API를 이용하여 사용자 프로파일인 XML 문서를 파싱하면서 변경하고자 하는 속성이 정의되어 있는 노드를 찾아 값을 증가시킨다.

4.3. 외부 이벤트 처리 시나리오

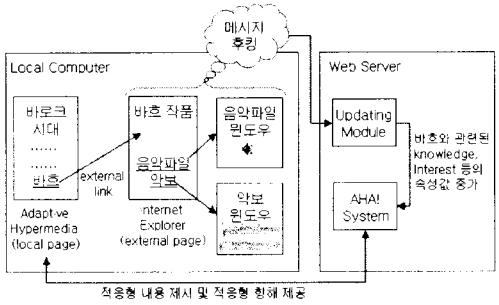
외부 응용 프로그램에서 특정 이벤트 메시지가 발생하였을 때, 이를 후킹 하여 적응형 하이퍼미디어 시스템의 사용자 모델을 변경시키는 시나리오는 [표 6]과 같다. 저작자는 [서양 음악사] 중에서 [바로크 시대]의 음악적 특징에 대한 정보를 제공하는 페이지를 만들 때, 유명한 음악가로 바흐를 언급하면서 바흐의 음악을 들을 수 있는 외

부 웹 사이트를 미리 링크시켜 놓았다.

[표 6] 시나리오의 상황적 정보

요소	상황 정보
적응형 하이퍼미디어에서 사용자의 현재 위치	서양 음악 / 서양 음악사 / 음악가 연구 / … 바로크 … / … 바흐 … / … 음악적 특징 … / … / … 대표 음악가 …
실행된 외부 응용 프로그램	인터넷 익스플로러
외부 응용 프로그램 실행 후, 사용자의 행동	새로운 윈도우 창에서 PDF 파일 및 미디 파일 실행
변경할 사용자 프로파일의 속성	[음악가연구]→[바흐]의 개념에 대한 knowledge, interest, visited의 값 변경

어떤 사용자가 바로크 시대를 읽는 도중에 그 링크를 클릭하여 해당 사이트에 접속하였을 때, 음악을 듣기만 한 것이 아니라 악보도 함께 보았다. 악보는 PDF 파일이며, 음악 파일은 미디 파일이었다. 두 파일 모두 새로운 윈도우창이 실행되면서 사용자에게 제공되었다. 단순히 음악을 듣는 것이 아니라 악보를 보며 들었다는 사용자의 행동은 음악에 대한 배경 지식이 높은 편이며, 바흐 음악에 대해 관심을 갖고 있거나 바흐 음악에 대해 배경 지식을 갖고 있다고 볼 수 있다. 따라서 이 사용자가 [음악가 연구]에서 [바흐]에 정보를 제공하는 페이지에 처음 방문하더라도 배경 지식이나 관심이 적은 사람보다는 전문적인 자료와 관련 링크를 제공하는 것이 필요하다. [그림 5] 는 메시지 후킹을 이용하여 특정 사용자 프로파일의 속성값을 변경시키는 과정을 설명하고 있다.



[그림 5] 메시지 후킹을 통한 사용자 모델 갱신의 예

후킹 프로그램은 사용자의 로컬 컴퓨터에서 사용자 PDF 파일을 보고 미디 파일을 실행시켰는지를 후킹하고, 이를 웹 서버에 전달한다. 웹 서버에서는 사용자가 바흐의 음악을 듣고 악보도 보았음을 인식할 수 있으며, 이러한 정보를 통해 [바흐]의 'knowledge' 속성과 'interest' 속성의 값을 증가시킨다. 또한, 비록 실제 [바흐]를 접근한 이벤트는 없었지만, 처음 방문한 사람과 달리 [바흐]에 대한 기본적인 내용은 알고 있는 것으로 판단하여, 'visited' 속성값도 변화시킨다.

[표 4]에서 제시된 초기화된 사용자 프로파일의 속성값은 [표 7]로 변경된다. 즉, 'knowledge'의 값은 '30'로, 'interest'의 값도 '15'로, 'visited' 속성값은 '100'으로 할당한다(AHA! 시스템은 사용자가 어떤 페이지를 접근하면, 'visited' 속성값을 '100'으로 할당함).

[표 7] 갱신된 사용자 프로파일의 일부분

```
<record>
    <key>music.musician.bach.knowledge</key>
    <type>1</type>
    <persistent>true</persistent>
    <value>30</value>
</record>
<record>
    <key>music.musician.bach.interest</key>
    <type>1</type>
    <persistent>true</persistent>
    <value>15</value>
</record>
<record>
    <key>music.musician.bach.visited</key>
    <type>1</type>
    <persistent>true</persistent>
    <value>100</value>
</record>
```

이렇게 갱신된 사용자 프로파일 정보를 바탕으로 앞에서 제시한 적응성 규칙이 수행되므로, [바흐] 페이지를 처음 접근하였을 때, 다른 사용자들은 'knowledge'가 '0'에서 '10'으로 할당되지만, 프로파일이 변경된 사용자는 페이지를 이미 접근한 것으로 판단하여, 기존의 'knowledge'를 '10' 퍼센트 증가시킨 '33'이 할당된다.

5. 결 론

본 논문에서는 적응형 하이퍼미디어 시스템을 사용하면서 동시에 다른 응용 프로그램을 사용할 때, 그 응용 프로그램에서 발생한 사용자 행동을 분석하여 사용자 모델을 갱신할 수 있도록 하였다. 이러한 방법을 통해 적응형 하이퍼미디어 시스템은 외부 응용 프로그램에서 발생한 사용자 행동까지 고려하여 개별화된 컨텐츠와 링크를 제

공할 수 있으므로 현재 사용자의 특성에 보다 적응적인 서비스를 제공할 수 있게 된다. 그러나 외부 응용 프로그램에서 발생한 사용자의 모든 행동을 메시지 후킹만으로 알아낼 수 없으며, 수많은 사용자 이벤트들을 모두 후킹 하여 처리하는 것이 비효율적일 수 있다. 본 논문에서는 외부 응용 프로그램에서 발생한 사용자의 행동 중 일부를 처리하였으나, 이를 적응형 하이퍼미디어 시스템에 전달하고 처리할 수 있는 다양한 방법에 대한 연구가 더 필요하다.

참 고 문 헌

[1] De Bra, P., Brusilovsky, P., Houben, G. J., Adaptive Hypermedia : From Systems to Framework, ACM Computing Surveys, Symposium Edition, ACM, 31(4es), 1999.

[2] De Bra, P., Houben, G. J., Wu, H., AHAM: a Dexter-based reference model for adaptive hypermedia, HYPERTEXT '99: Proceedings of the tenth ACM Conference on Hypertext and hypermedia, ACM, 1999, pp. 147-156.

[3] De Bra, P. Wu, H., De Kort, E., Design Issues in Adaptive Hypermedia Application Development, Proceedings of the Second Workshop on Adaptive Systems and User modeling on the World Wide Web, 1999, pp. 29-39.

[4] Aroyo, L., De Bra, P., Houben, G.J., Embedding Information Retrieval in Adaptive Hypermedia: IR meets AHA!. Proceedings of the AH2003 Workshop, TU/e CSN 03/04, May 2003, pp. 63-76.

[5] Ivanov, I. API hooking revealed, The Code Project, 2002.

[6] 이기우, 박성빈, 컴퓨터 과학 교육을 위한 CAI 프로그램 설계 및 구현, 2005년도 컴퓨터교육학회 동계학술대회, 2005, 제 9권 제 1호, pp. 331-336.

[7] Lee, K., Jung, H., Park, S., Applying adaptive hypermedia technologies to a learning tool, The 5th IEEE International Conference on Advanced Learning Technologies (ICALT 2005).

[8] De Bra, P. and Calvi, L., AHA! An open Adaptive Hypermedia Architecture, The New Review of Hypermedia and Multimedia, vol. 4, pp. 115-139, Taylor Graham Publishers, 1998.

[9] 김성우, 해킹/파괴의 광학, 정보게이트, 2001

정 효 숙

1998 서울교육대학교 교육학과 (교육학학사)

·2001 서울교육대학교교육대학원 컴퓨터교육과(교육학석사)

2003~현재 고려대학교 컴퓨터교육학과 박사과정

관심분야 : 적응형 하이퍼미디어, 컴퓨터교육

E-Mail : est0718@comedu.korea.ac.kr

박 성 빈

1990 고려대학교 전산과학과 (이학사)

1993 University of Southern California (전산학 석사)

1999 University of Southern California (전산학 박사)

2003~현재 고려대학교 컴퓨터교육과 조교수

관심분야 : 하이퍼텍스트, 컴퓨터교육, 알고리즘, 계산이론

E-Mail : psb@comedu.korea.ac.kr