프록시 서버간 협업을 통한 분산적응시스템

이승화[1], 이은석[2]
성균관대학교 컴퓨터공학과[1 2]
{shlee[1],eslee[2]}@ece.skku.ac.kr

Distributed Adaptation System through Cooperations among Proxy Servers

Seunghwa Lee[1], Eunseok Lee[2]
Dept. of Computer Engineering, Sungkyunkwan University[1 2]

요 약

최근 무선디바이스의 확산과 함께 무선 네트워크 환경의 다양한 제약사항을 극복하고 항상 적절한 서비스 레벨을 유지하기 위한 'Adaptation'관련연구가 중요한 이슈가 되고 있다. 그러나 대부분의 기존 연구들은 무선디바이스의 문제에 주요 초점이 맞추어져 있으며, 실제 적응에 필요한 변환작업이 이루어지는 프록시 서버의 부하는 크게 고려되지 않고 있다. 하지만 실제로 컨텐츠의 포맷변환과 같은 작업은 많은 리소스가 소요되는 작업이며, 하나의 서버에서 이를 수행하는 경우 많은 작업부하가 집중된다. 이는 사용자가 증가함에 따라 더욱 심각해지며, 사용자의 컨텐츠 요청에 대한 응답시간을 증가시키는 결과를 초래한다. 따라서 본 논문에서는 계층적으로 구성된 주변 프록시들간에 협동작업을 통해, 적응에 필요한 작업의 부하를 분산시키고, 이를 통해 보다 빠르고 효율적인 서비스를 제공하는 새로운 프레임워크를 제안한다. 우리는 제안시스템을 평가하기 위해 프로토타입을 개발하여 Healthcare 시나리오에 적용하였으며, 작업량에 따라서 참여 프록시들에게 작업부하가 균등하게 분배되는 결과와, 이를 통해 적응 컨텐츠가 보다 빠르게 사용자에게 제공되는 결과로 시스템의 효율성을 증명하였다.

Keyword : Agent, Distributed Computing, Content Adaptation

1. 서 론

최근 인터넷과 무선네트워크 기술의 급격한 발전을 통해, 무선인터넷의 이용이 점차 증가하고 있다. 다양한 형태의 무선 디바이스는 이미 우리 생활의 일부가 되었으며, 우리는 이를 통해 전화뿐만 아니라, TV 를 보고, 게임을 즐기며, 인터넷에 접속을 한다. 이처럼 기존 유선인터넷과 데스크 탑 PC 를 통해 이루어졌던 서비스들이 무선 환경으로 점차 확대되면서 우리는 언제 어디서나 원하는 정보를 손쉽게 얻을 수 있게 되었다. 이러한 변화와 함께 무선 디바이스의 성능은 갈수록 강해지고 있지만, 기기의 휴대성이 강조되는 모바일 환경의 특성상, 아직까지는 데스크 탑 PC 에 비해 상대적으로 열악한 컴퓨팅 파워를 가질 수 밖에 없다. 또한 무선인터넷은 유선인터넷과는 달리, 사용자의 이동에 따라 bandwidth 가 크게 변화하며, 그에 따라 서비스의 품질도 크게 변화하는 특성을 가지고 있다.

따라서 이러한 무선네트워크 환경의 다양한 제약사항을 극복하고, 항상 적절한 서비스 레벨을 유지하기 위한 적응형 서비스의 제공이 최근 중요한 이슈로 대두되고 있다. 기존의 적응관련연구는 크게 '무엇을 적응시켜 제공할 것인가'와 '어디서 적응을 수행할 것인가'로 나눌 수 있다. 먼저 전자의 경우는 제공되는 컨텐츠의 품질을 조정하는 적응과 클라이언트의 모듈을 재구성(re-configuration)하는 연구로 나눌 수 있다. 그리고 후자의 경우는 적응이 이루어지는 위치에 따라 크게

Receiver Adaptation, Source Adaptation 과 Proxy Adaptation 으로 나눌 수 있다. 그리고 최근에는 한쪽으로 작업부하가 치우치는 것을 막기 위해, 적응 모듈을 적절히 분산시켜 효율적인 적응을 수행하려는 연구[1]도 진행되고 있다. 그러나 실제 적응 작업이 이루어지는 프록시 서버의 부하는 여전히 큰 문제로 남아있으며, 클라이언트의 수가 늘어날수록 그 문제는 커지게 된다.

본 연구에서는 이러한 문제를 해결하기 위해 적응을 수행하는 모듈을 프록시에 위치시키고, 작업부하에 따라 주변 프록시와의 협력작업을 통해 작업을 적절히 분산시켜 처리하는 시스템을 제안한다. 이를 통해 사용자의 요청에 대한 응답시간을 단축시켰으며, 전체적인 시스템의 운용효율을 높일 수 있다. 또한 상태 모니터와 적응에 대한 결정은 클라이언트에서 수행하도록 적응모듈을 분산시켜, 사용자의 Privacy 와 관련된 정보의 유출을 막을 수 있다.

우리는 제안시스템의 평가를 위해 프로토타입을 개발하여, Healthcare 시나리오에 적용하였으며, 작업량에 따라 주변 프록시 서버에 작업부하가 균등하게 분배되는 결과와, 이를 통해 적응 컨텐츠가 보다 빠르게 사용자에게 제공되는 결과로 시스템의 효율성을 증명하였다.

2. 관련연구

최근 Adaptation 관련연구는 많은 연구기관에서 다양한 방법으로 진행되고 있다. 이러한 기존 연구는 적응이 이루어지는 모듈의 위치에 따라 크게 Receiver Adaptation, Proxy Adaptation, 그리고 Source Adaptation 과 같이 세 가지로 분류가 가능하다[2].

먼저, Receiver Adaptation 은 대부분의 적응모듈이 클라이언트 디바이스에 내장되어, 클라이언트 스스로 주변상황을 모니터하고, 이에 따라 자신의 모듈을 재구성하거나 애플리케이션의 파라매터를 조정하는 방식[3]을 의미한다. 이 방식은 클라이언트의 모듈 재구성을 통한 적응에 적합하며, 보다 개인화된 적응서비스를 위해, Context Information 에 포함될 수 있는 다양한 개인정보의 유출을 막을 수 있다. 그러나 미디어 컨텐츠의 포맷변경과 같은 적응작업은, 휴대용 기기가 가지고 있지 않은 충분한 리소스를 필요로 하기 때문에 이루어지기 어렵다는 단점을 가지고 있다.

둘째로, Source Adaptation 방식은 적응모듈이 서버에 존재하여, 사용자가 정보를 요청할 때 Context 정보를 받아, 그 특성에 맞게 적절한 미디어 컨텐츠를 연결해주는 방식[4]이다. 서버는 일반적으로 컨텐츠를 저작하는 정보제공자이기 때문에, 보다 적절한 컨텐츠 버전을 제공할 수 있다는 장점을 갖는다. 그러나 적응모듈이 서버에 추가됨으로써, 서버는 컨텐츠 제공 외에 컨텐츠를 변환하는 작업 부하도 갖게 되며, 이는 응답시간의 증가를 초래할 수 있다.

마지막으로 Proxy Adaptation 은 Source Adaptation 방식의 문제를 보완하기 위해, 대부분의 적응모듈을 서버에서 독립시켜, 클라이언트와 서버 사이에 존재하는 프록시 서버에 내장시킨 방식[5][6][7]을 의미한다. 이 방식은 클라이언트와 서버의 모듈을 변경하지 않고, 단지 프록시에 적응 모듈을 추가하는 것만으로 적응이 가능하다는 장점을 가지고 있어 많은 연구에서 이 방식을 이용하고 있다. 그러나 프록시 서버는 일반적으로 적응작업을 효과적으로 처리할 수 있는 충분한 리소스를 가지고 있다고 가정할 수 있지만, 실제로 컨텐츠의 포맷변경과 같은 적응작업은 많은 리소스의 소비를 필요로 하며, 클라이언트의 요청이 늘어날수록 프록시 서버의 작업부하는 크게 증가하게 된다.

또한 Source adaptation 방식과 마찬가지로 Context Information 이 개인화를 위해 확장될 경우, 개인정보의 유출과 같은 단점도 가지고 있다.

다음 섹션에서는 이러한 기존 관련연구의 다양한 단점을 보완하고, 효율적인 적응작업을 위해 설계된 제안시스템을 자세히 설명한다.

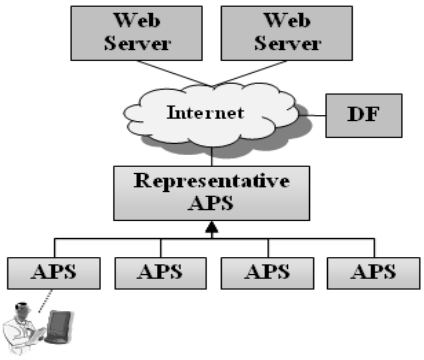
3. 제안시스템

제안시스템은 모바일 환경에서 발생할 수 있는 다양한 Context 를 인식하여 그에 적절한 적응 서비스를 수행하며, 적응작업이 이루어지는 프록시 서버의 작업부하를 분산시켜, 보다 빠르고 효율적인 서비스를 제공하기 위해 설계되었다.

3-1. 제안시스템 프레임워크

제안시스템의 전체적인 구조는 [그림 1]과 같다. Adaptation Proxy Server(APS)들은 Representative APS 를 중심으로 계층적인 구조로 구성된다.

Representative APS 는 네트워크 설계자에 의해 위치적으로 가장 적절한 프록시 서버가 선택되며, 같은 Zone 으로 연결된 APS 들에게 현재 작업로드와 가능한 서비스 정보를 받아 관리한다.



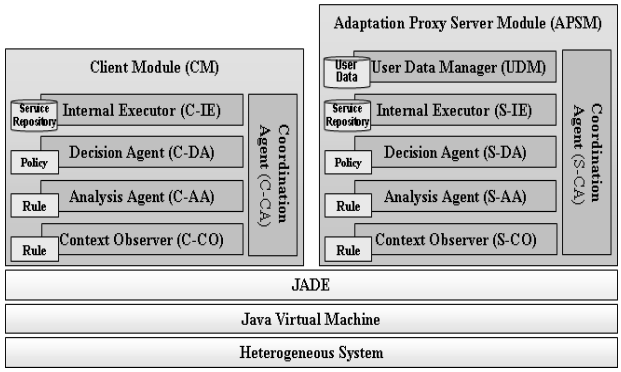
[그림 1] 제안시스템의 전체적인 프레임워크

클라이언트가 컨텐츠를 요청할 때, Client Module(CM)은 컨텐츠의 요청과 함께, 자신의 현재 주변상황과 사용자의 취향정보를 반영한 'Order'를 작성하여, 가까운 APS 에 전송한다. 이 'Order'는 적응서비스의 종류나 강도에 대한 요구사항을 나타낸다. APS 는 이 요청정보를 Representative APS 에게 전달함과 동시에, 자신의 작업로드와 가능한 서비스를 알린다. 그리고 Representative APS 는 DF 를 검색하여 요청 컨텐츠를 제공하는 웹 서버나 다른 Zone 의 APS 의 위치를 파악하고, 직접 그 서버와 상호작용을 통해, 요청 컨텐츠에 대한 메타데이터를 전송 받는다. 요청 메시지에 목적지가 명시되어 있는 경우에는 DF 를 경유하지 않고 직접 연결하여 데이터를 전송받는다.

이후, Representative APS 는 사용자의 'Order'와 웹 서버로부터 받은 메타데이터를 기반으로 최종 적응서비스를 결정한다. 그리고 자신이 속한 Zone 의 각 APS 의 여유리소스를 반영하여 적응작업을 APS 들에게 적절히 분배한다. 각 APS 는 메타데이터를 해석하고, 서버로 접속하여 컨텐츠를 전송받아 적응작업을 수행한다. 그리고 적응이 수행된 컨텐츠는 사용자가 처음 접속한 HomeAPS 나 CM 으로 전송되어 최종적으로 사용자에게 전송된다.

3-2. 시스템 구성요소

제안시스템의 구성요소는 크게 사용자의 기기에 내장되는 Client Module(CM)과 Adaptation Proxy Server (APS)에 내장되는 모듈로 나눌 수 있다. CM 과 APSM 간에 통신은 FIPA 표준을 따르는 JADE 플랫폼을 이용하여 이루어지며, 그 구조가 [그림 2]에 나와있다.



[그림 2] 제안시스템의 구성모듈

각 세부모듈의 역할은 다음과 같다.

- Context Observer (CO): Rule 을 기반으로 Context Information 을 수집하여, RDF 를 이용한 규격화된 문서를 생성한다.

- Analysis Agent (AA): 생성된 Context Information 과 진단 Rule 을 기반으로 현재의 상황을 해석하고 추론한다.

- Decision Agent (DA): AA 로부터 받은 정보를 기반으로 현재 가장 적절한 행동을 결정하여 'Order'를 생성한다.

- Coordination Agent (CA): 모듈간 메시지의 송수신을 수행하며, 외부 에이전트와 협상을 통한 상호작용을 수행한다. 외부 메시지의 경우, 암호와 복호를 수행한다.

- Internal Executor (IE): 선택된 Adaptation Service 를 실행한다. C-IE 의 경우, 애플리케이션의 파라매터 조정이나 모듈의 재구성(re-configuration)과 같은 내부적인 조정도 가능하다.

- Service Repository Adaptation 을 위한 서비스코드(i.e., translator, page reformatting)가 저장된다.

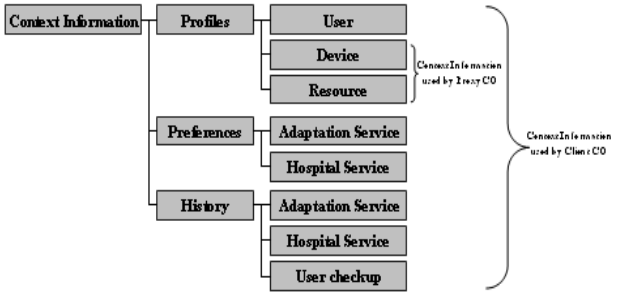
- User Data DB: 사용자의 기본정보가 저장된다. 4 장에서 소개할 Healthcare 시나리오에서, 병원에 위치하게 되는 APS 에는 진료 이후의 데이터와 이에 대한 메타데이터가 저장된다.

- Directory Facilitator (DF): 각 APS 나 웹 서버의 에이전트는 자신이 제공하는 컨텐츠와 기타 제시 조건, 위치정보를 DF 에 등록한다. 각 APS 는 이를 이용하여 필요한 에이전트의 위치를 찾는다.

이외에도 사용자와 상호작용을 위한 User Interface 가 필요하다. 이는 추천결과를 사용자에게 보여주고 선택하게 하거나, Preference 를 추출하기 위해 조건을 제시하는 역할을 한다.

3-3. 상황인식과 모듈간 주고받는 메시지

CO 는 사용자의 요청이 있을 때, 또는 주기적으로 자신의 상태를 수집하여 Context Information 을 생성하며, 고려되는 Context 는 개발자에 의해 시스템의 적용분야에 맞게 확장될 수 있다. 본 제안 시스템의 프로토타입으로 구현된 Healthcare System 의 Context Information 종류는 [그림 3]과 같다.



[그림 3] Context Information 의 분류

User Profile 은 진료기록과 병원예약에 필요한 정보를 포함하는 사용자의 기본정보를 담고 있다. Device Profile 은 기기의 정적인 Capability 정보를 나타내며, Resource Profile 은 동적으로 변화하는 주변상황(i.e., Bandwidth, CPU load)을 나타낸다. Adaptation Service Preference 는 적응서비스의 강도를 조절하기 위한 사용자의 취향정보를 나타내며, 이는 충돌을 최소화하기 위한 우선순위를 지정하는데 사용된다. Hospital Service Preference 는 사용자가 선호하는 병원에 대한 조건정보를 나타낸다. 이들은 각각 CM 이 설치될 때, User Interface 를 통해 사용자로부터 입력 받아 생성되며, 이후에는 사용자의 행동에 의해 지속적으로 변경된다.

Adaptation Service History 는 각 상황에 따른 적응작업과 그에 대한 사용자의 반응이 저장되며, Hospital Service History 는 병원 예약에 대한 결과와 사용자의 반응이 저장된다. User checkup History 는 사용자의 건강체크 기록이 저장된다. 이후 이 History 정보들은 데이터마이닝을 통한 패턴분석에 사용되며, 이는 'Order'를 생성하는데 반영된다.

이 중 Proxy 에 내장되는 S-CO 에 의해 생성되는 정보는 Device Capability 정보와 동적으로 변화하는 현재 Resource 사용양만 해당된다.

이처럼 CO 에 의해 생성된 Context Information 은 RDF 를 이용하여 규격화된 문서로 생성되며, AA 는 이 정보를 이용하여 Resource Profile 을 분석하고, 현재의 상황을 숫자를 통해 표현한다. 이러한 동작 알고리즘은 본 연구팀에서 개발한 Self-Growing Engine[8]을 이용하였다. 이후, DA 는 해석된 내용을 기반으로 'Order'를 생성한다. 이때, C-DA 와 APSM 의 S-DA 의 동작은 다르다. 먼저 C-DA 는 Profile 과 Preference, History 를 반영하여, 수행될 수 있는 후보 서비스를 선택하거나 병원예약에 대한 요청을 'Order'로 작성한다. 이와는 다르게 S-DA 는 자신의 리소스 상황에 따라 자신이 수행하기 어려운 서비스를 요청하는 내용으로 'Order'를 작성한다. [그림 4]에 CM 과 APSM 에서 생성된 'Order' 메시지의 예가 나타나 있다.

```
(INFORM
   :sender ( agent-identifier :name CM1@JB:1099/JADE )
   :receiver (set ( agent-identifier :name APS1@selab:1099/JADE )
   :content "(action text translator(null, kr to en)),
   (action image ImageCompression(30, Color to gray) AND
                 ImageConverter(null, vt to jpg)"
   …
)
```

```
(INFORM
   :sender ( agent-identifier :name CM1@JB:1099/JADE )
   :receiver (set ( agent-identifier :name APS1@selab:1099/JADE )
   :content "(entrust text translator(null, kr to en)),
   …
)
```

[그림 4] CM 과 APS 간에 주고받는 Order 메시지(위), APS 가 대표 APS 와 주고받는 Order 메시지(아래)의 예

3-4. 협력적 프록시 서버를 통한 분산 적응

클라이언트로부터 'Order'와 함께 정보요청을

받은 HomeAPS 는 자신의 상태에 따라, 웹으로부터 가져온 데이터를 주변 프록시와 협동작업을 통해 적응작업을 수행한다.

먼저 HomeAPS 는 클라이언트의 요청이 있을 때, 자신의 현재 리소스 상황에 따라 위임하고자 하는 작업들의 리스트(i.e., translator, image converter, movie clip converter)를 'Order'로 작성하여 Representative APS 에게 전송한다. 그러면 Representative APS 는 요청된 목적지 서버나 요청 컨텐츠를 가지고 있는 서버를 DF 를 통해 찾아 원본서버에 접속한 후, 컨텐츠의 메타데이터를 전송받는다. 메타데이터는 그 컨텐츠에 대한 간략한 정보를 담고 있는 문서파일로 파일의 형식, 크기, 지원언어와 그 구성내용 등이 규격화되어 기술된다. 이후, Representative APS 는 요청작업리스트와 메타데이터의 공통부분을 찾아 필요한 작업을 결정한다. 그리고 적응서비스 개발자가 미리 기술한 정보파일을 검색하여 적응서비스를 수행할 때 소요되는 리소스 소모량을 계산한다. 그리고 자신의 Policy 와 각 APS 의 상태에 따라 적절한 APS 를 선택하여 작업을 분배한다. 이 작업분배에 사용되는 'Order' 메시지는 원본 컨텐츠의 위치, 필요한 작업과 강도, 수행 이후 전송할 목적지 주소(HomeAPS or CM)로 구성된다. 이후 각 APS 는 서버로 접속하여 컨텐츠를 전송받아 적응작업을 수행하고, HomeAPS 나 CM 으로 결과를 전송한다.

4. 시스템 구현 및 평가

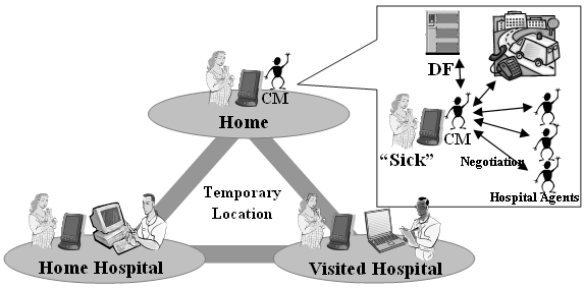
우리는 제안시스템의 평가를 위해 프로토타입을 구현하여 차세대 Healthcare 를 위한 시나리오에 일부 적용하였다.

4-1. Healthcare 시나리오

본 시나리오는 사용자가 이동할 수 있는 세 가지 Location 을 고려한다. APS 는 Home 과 각각의 병원에 존재하며, Temporary Location 에서 서비스를 위해, 무선네트워크의 AP 나 그와 가까운 곳에 위치하는 것으로 가정하였다.

먼저 Home Hospital 은 환자가 주로 이용하는 병원을 의미하며, 진료 이후의 데이터가 저장된다. 이 데이터는 CT 촬영결과, X-ray 필름, 의사소견서 등을 의미하며, 멀티미디어 컨텐츠로 존재하는 것을 가정한다. 그리고 Home 은 환자를 관리하는 HomeAPS 가 존재하는 환자의 집을 의미한다. Home 에서 시스템은 블루투스 기능을 내장한 센서를 통해 환자의 현재상태(i.e., 당뇨환자의 경우 혈당, 혈압 체크 또는 임산부와 태아를 위한 건강체크)를 수집하여, 사용자의 기기에 전송한다[9].

사용자의 휴대용 기기에 내장된 CM 은 이 정보를 통해 환자의 상태를 분석한다. 그리고 만약 환자의 건강에 이상이 발생하면, CM 은 이를 사용자에게 알리고, Policy 에 기반하여 작성된 'Order'를 HomeAPS 에 전송한다. HomeAPS 는 'Order'를 기반으로 다음과 같은 행동들을 수행한다.



[그림 5] Healthcare 시나리오

첫 번째 상황으로, 만약 환자에게 낮은 위험도의 이상이 발생한 경우에 'Order'는 사용자에게 '자가치료'에 관련된 정보를 수집하여 제공하라는 내용과 적절한 강도의 적응 서비스 리스트로 이루어진다. 이에 따라 HomeAPS 는 DF 에 관련 컨텐츠를 제공하는 서버의 위치를 질의하고, 웹을 통해 이에 관련된 컨텐츠를 수집한다. 그리고 이 정보를 사용자의 기기에 적합한 형태로 변환하여 제공한다. 이때 HomeAPS 는 보다 빠르고 효율적인 작업처리를 위해, 자신의 작업부하에 따라 이를 가까운 주변의 APS 와 함께 분산작업을 수행한다.

두 번째 상황으로, 만약 환자에게 병원진료가 필요한 위험도의 이상이 발생한 경우에 'Order'는 적절한 병원을 검색하여 추천하라는 내용으로 이루어진다. 이에 따라 HomeAPS 는 DF 에 환자의 증상과 관련된 서비스를 제공하는 주변 병원을 질의한다. 그리고 'Order'에 함께 들어있는 사용자의

Preference 와 스케쥴 정보를 기반으로 각 병원의 APS 와의 상호작용을 수행하고, 환자에게 필요한 의사의 존재여부, 스케쥴 등을 확인하여 병원 리스트를 필터링 한다. 이후, CM 은 필터링된 병원 리스트를 수신하여 사용자로부터 최종결정을 수행하도록 하고, 병원 APS 와 직접 상호작용을 통해 예약에 필요한 개인 정보를 전송하고 자동예약을 수행한다. 그리고 이때 병원 APS 로부터 지도나 안내서의 리스트를 제공받아 사용자에게 보여주고, 사용자가 선택하면 이를 다운 받아서 사용자의 기기에 적합한 형태로 변환하여 보여준다. 이때 제공되는 컨텐츠가 만약 Translator 나 동영상 품질변환과 같은 여러 적응서비스를 필요로 한다면 변환에 많은 시간이 소요될 것이다. 제안시스템은 이러한 상황에서 효과적으로 사용된다.

세 번째 상황으로, 만약 환자에게 위급한 상황의 이상이 발생한 경우에는 응급환자에 맞는 정책에 의해 'Order'가 생성된다. 이에 따라 HomeAPS 는 검색과 협상절차를 간소화하여 자율적으로 가까운 병원을 예약하며, 예약된 병원의 의사나 주치의에게 화상진료를 요청한다. 그리고 동시에 구급대에 연락을 취한다. 이후 도착한 구급대는 구급차안에 자체적으로 가지고 있는 이동 APS 를 이용하여, HomeAPS 로부터 요청 작업처리결과를 받아, 만약 화상진료가 가능한 경우, 병원 APS 와 연결을 수행하고, 환자가 병원으로 이송되는 동안 응급처지가 가능하도록 한다. 이때 역시 APS 는 구급대원의 PDA 나 구급차 안에 비치된 디스플레이 장치에 적합한 형태로 컨텐츠를 변환하여 제공된다. 그리고 화상진료가 가능하지 않은 경우에는 DF 를 통해 응급처치에 관련된 컨텐츠를 수집하여 제공한다.

환자가 Visited Hospital 에 도착하면, Visited Hospital APS 는 사용자 디바이스의 CM 과의 상호작용을 통해 신원파악과 인증을 수행하고, 의료보험조합과 같은 제 3 의 기관에 진료 History 를 요청한다. 이후, 진료기록이 있는 병원에 진료에 필요한 데이터를 요청하고, CM 으로부터 건강체크 History 를 제공받는다. 이렇게 전송 받은 데이터들은 의사의 기기에 적합한 형태로 재구성된다.

4-2. 프로토타입 구현을 통한 평가

프로토타입 구현에 사용된 Proxy 서버는 유선랜으로 연결되어 있고, CPU 는 Intel Pentium-4 3GHz, RAM 1Gbyte 의 사양을 가지고 있는 데스크탑 PC 8 대를 이용하였다. 각 PC 는 Windows 2000 을 운영체제로 사용하기 때문에, 프록시 서버의 Context Observer 는 C++을 이용하여 구현하였다. CM 은 Content 요청 시에 이 CO 를 호출하고, 실행된 CO 는 현재 Context Information 을 수집하여 RDF 문서를 생성한다. 그러나 이번 실험에서는, 클라이언트도 데스크탑에서 시뮬레이터를 이용했으며 Context Information 은 PDA 상황을 가상으로 직접 작성하여 이용하였다. 나머지 모듈은 모두 Java 를 이용한 JADE 를 기반으로 개발되었다.

우리는 먼저 HomeAPS 에서 몇 개의 적응 서비스(translator, Image Converter, Movie clip converter)를 동시에 실행하였을 경우의 소요시간을 측정하였다. 두 번째로, HomeAPS 에서 다수의 클라이언트 요청에 따른 작업부하를 시뮬레이트 하기 위해, dummy process 를 발생시키는 프로그램을 실행한 다음, 적응 서비스의 소요시간을 측정하였다. 그 결과, 부하에 따라 작업속도가 현저하게 저하되는 결과를 알 수 있었다. 세 번째는 제안시스템의 전체 모듈을 가동시켜, 작업들이 주변 두 개의 Proxy 로 분산되어 병렬처리되는 작동과정을 확인하였으며, 이를 통해 작업속도가 단축되는 결과를 확인하였다. 이때, 프록시간에 작업결과를 전송하는 전송시간을 추가하여 계산하였지만, 여전히 최종 적응에 걸리는 시간은 단축되었다는 것을 알 수 있다. 이에 대한 결과는 다음과 같다.

<표 1> 응답시간에 대한 비교표

Service Type	Source Type	Substance of adaptation	RT for test1	RT for test2	RT for test3
Translator	File size: 5Kbyte File type: xml Language: En	Translate >>xml >>Kr	Average 85sec.	Average 120sec.	Average 55sec.
Image Converter	File Size: 54kbyte File type: bmp 1024*768*24b	Convert >>jpg >>file format: width 320	Average 8sec.	Average 20sec.	Average 4sec.
Movie clip Converter	File Size: 47Mbyte File type: avi Frame rate = 23 bit rate = 448 display size = 720 * 304	Convert >>avi >>frame rate = 15 >>bit rate = 150 >> display size = 320*240	Average 85sec.	Average 120sec.	Average 55sec.
		Total Required Time	85sec.	120sec.	60sec.

우리는 본 실험을 통해, 제안시스템을 이용하여 보다 빠른 적응 서비스가 가능해짐을 알 수 있었다. 비록 네트웍 상에 트래픽이 일부 증가하지만, 프록시에 많은 작업부하가 있을 때 사용자에게 보다 빠른 서비스를 제공하고, 각 프록시간에 리소스를 공유함으로써 작업부하가 한쪽으로 치우치는 것을 방지할 수 있다. 또한 자신의 상태를 분석하여 결정한 'Order'만을 전송함으로써, 개인정보의 유출을 어느 정도 막을 수 있으며, 모듈간 주고받는 메시지의 크기를 줄이게 된다.

5. 결 론

우리는 실제 적응 작업이 이루어지는 프록시의 작업부하를 줄이기 위해, 프록시들을 계층적으로 구성하고, 주변 Proxy 와의 분산작업을 수행하여 보다 빠르고 효율적인 서비스를 제공하는 시스템을 제안하였다. 향후과제는 다음과 같다.

- 보다 효율적인 메시지 교환을 통한 네트워크 트래픽의 감소방안 연구
- 적응 컨텐츠의 재사용을 통해 사용자의 컨텐츠 요청에 대한 응답속도 향상
- 서버, 프록시, 클라이언트간 적응작업의 분산처리방안 연구

참고문헌

[1] Alvin T.S. Chan, Siu-Nam Chuang, "MobiPADS: A Reflective Middleware for Context-Aware Mobile Computing", IEEE Transaction on Software Engineering. Vol. 29, No.12 pp.1072-1085, Dec.2003

[2] Margaritis Margaritidis and George C. Polyzos, "Adaptation techniques for Ubiquitous Internet multimedia", Wireless Comm. and Mobile Comp., vol.1, No.2, pp.141-163, Jan.2001

[3] Brian Noble, "System Support for Mobile, Adaptive Applications", IEEE Vol.7 No.1, Feb.2000

[4] Paolo Bellavista, Antonio Corradi, Rebecca Montanari, Cesare Stefanelli, "Context-Aware Middleware for Resource Management in the Wireless Internet", IEEE Transactions on Software Engineering, Vol.29, No.12, Dec.2003

[5] Anel Pashtan, Shriam Kollipara, and Michael Pearce, "Adapting content for wireless web services", IEEE Internet computing, Sep.2003

[6] Timo Laakko and Tapio Hiltunen, "Adapting web content to mobile user agents", IEEE Internet computing, Mar.2005

[7] IBM WebSphere® Transcoding Publisher, http://www-306.ibm.com/software/pervasive/transcoding_publisher

[8] Seunghwa Lee, Jehwan Oh, and Eunseok Lee, "An Architecture for Multi-agent Based Self-adaptive System in Mobile Environment", LNCS 3578, pp.494-500, Jul.2005

[9] Byong-In Lim, Kee-Hyun Choi, and Dong Ryeol Shin, "A JXTA-based Architecture for Efficient and Adaptive Healthcare Services", LNCS.3391, pp.776-785, Jan.2005

[10] Wai Yip Lum and Francis C.M. Lau, "A context-aware decision engine for content adaptation", IEEE Pervasive computing, vol.1, pp.41-49, Jul.2002

[11] W3C - RDF Primer, http://www.w3.org/TR/rdf-primer, Feb.2004.

[12] W3C - Composite Capability/Preference Profiles (CC/PP), http://www.w3.org/Mobile, 2004

[13] http://www.fipa.org/specs/fipa00023

[14] M. Roman, C. K. Hess, R. Cerqueira, A. Ranganathan, R. H. Campbell, and K. Nahrstedt, "Gaia: A middleware infrastructure to enable active spaces", IEEE Pervasive Computing, pp.74-83, Oct-Dec 2002