碩士學位論文

# 유비쿼터스 환경을 위한 규칙 기반 CC/PP 프로파일 생성 방법

濟州大學校 大學院

컴퓨터工學科

宋 在 炅



2005年 12月

# 유비쿼터스 환경을 위한 규칙 기반 CC/PP 프로파일 생성 방법

指導敎授 李 尙 俊

宋 在 炅

이 論文을 工學 碩士學位 論文으로 提出함

2005 年 12 月



宋在炅의 工學 碩士學位 論文은 認准함

審査 委員長 _____ 印

委 員 _____ 印

委 員 _____ 印

濟州大學校 大學院
2005 年 12 月

# A Rule-based CC/PP profiling method in ubiquitous environment

Jae-Kyoung Song

(Supervised by professor Sang-Joon Lee)

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF MASTER OF ENGINEERING



DEPARTMENT OF COMPUTER ENGINEERING

GRADUATE SCHOOL

CHEJU NATIONAL UNIVERSITY

2005. 12.

# 목 차

# 그 림 목 차

# 표 목 차

# Summary

In the ubiquitous computing environment, service providers should able to deliver their services to various devices, such as PDA, Smart Phone, Telematics devices. For this purpose, W3C announced 'Composite Capabilities / Preference Profile (CC/PP) Standard' for delivery-context description. At present, only a small number of mobile phone supports the CC/PP standard, But most of contents servers do not. In this paper, by analyzing request headers which are received from various ubiquitous devices, a rule-based CC/PP profiling method is developed. Service providers can use these profiles for providing optimized services to the various devices which have different capabilities.

# Ⅰ. 서 론

인터넷 기술의 진보와 모바일 기기 및 임베디드 컴퓨팅 기술의 발전으로 인해, 일상생활에서의 컴퓨팅 환경은 언제 어디서든지 컴퓨터 기술을 활용할 수 있는 유비쿼터스 환경으로 변해가고 있다. 이러한 변화의 한가운데에는 PDA, 스마트폰, 텔레매틱스 기기와 같은 유연하고 이동성이 있는 최첨단 기기들이 자리 잡고 있으며[1], 성공적인 유비쿼터스 환경을 위해서는 현재의 다양한 단말뿐만이 아니라 앞으로 출현할 새로운 기능과 제약사항을 갖는 단말들에 대해 최적화된 서비스를 제공할 수 있는 컴퓨팅 환경의 구축이 요구되어지고 있다.

그러나, 이러한 다양한 이종 기기들의 증가는, 각각의 단말들이 갖는 서로 다른 입출력 기능과 하드웨어, 소프트웨어, 네트워크상의 특성으로 인하여, 서비스를 제공할 각각의 단말의 상황에 최적화된 어플리케이션 개발의 복잡성을 증가시키고 있다. 이상적으로 볼 때 서비스 제공자들은 한 가지 버전의 서비스를 개발함으로서, 해당 서비스를 모든 단말에 동일하게 제공할 수 있기를 바라지만, 단말들의 다양한 특성으로 인해 각각의 단말에 적합하도록 동일한 내용의 서비스를 서로 다른 형식으로 중복하여 개발하여야만 하는 것이 지금의 현실이다.

이러한 문제를 제거하기 위하여, 많은 비용을 들이지 않고도 다양한 단말기기들을 지원할 수 있는 장치 독립적(device-independent) 접근방식에 대한 연구가 진행 중이다[2-4]. 이러한 장치 독립적 접근방식을 사용한다면 서비스 제공자는 하나의 서비스만을 개발하고, 이를 서비스를 요청하는 클라이언트의 특성에 최적화된 형태로 변환하여 제공함으로서 이전의 불필요한 서비스 개발의 비용을 절감할 수 있게 된다.



단말의 특성을 기반으로 한 장치 독립적 서비스 제공을 위해서는 먼저 상황기반 정보가 필수적으로 요구된다. 이러한 상황기반 정보에는 단말의 특성뿐만 아니라 네트워크의 특성, 사용자의 선호도, 그리고 사용자의 위치나 사용하는 언어 등 추가적인 어플리케이션에 특화된 파라미터 값 또한 포함된다. 여기서 서버가 상황기반 정보를 바탕으로 사용자의 단말에 최적화된 서비스를 제공할 수 있도록, 이러한 정보를 어떻게 서버로 제공할 것인가 하는 콘텐츠 협의(Content Negotiation) 문제가 발생한다. 현재 표준 HTTP 헤더에는 이러한 콘텐츠 협의를 위한 몇 가지 정보를 포함하고 있다[5]. 그러나 HTTP는 브라우저 특성에 대한 기술만을 대상으로 설계되어 있어 상황기반의 콘텐츠 협의에 충분히 활용될 수 없는 한계를 가지고 있다.

CC/PP 표준[6-9]은 하드웨어, 소프트웨어, 네트워크, 지원하는 서비스 정보 등 일련의 단말 특성 및 제한조건과 사용자 취향에 관한 정보 기술 및 전송에 대한

규격으로 RDF(Resource Description Framework)[10] 로 작성되는 프로파일을 통해 상황기반 정보를 기술한다. 이러한 CC/PP 기술을 이용하여 현재의 다양한 단말뿐만이 아니라 앞으로 개발될 새로운 기능의 단말들이 자신의 특성에 대한 프로파일을 전송한다면, 장치 독립적 서비스 제공을 위한 콘텐츠 협의 문제는 쉽게 해결될 수 있을 것이다.

그러나 현재 CC/PP 를 지원하는 단말은 UAProf[11] 기반의 프로파일을 사용하는 소수의 휴대폰에 한정되어 있으며, 그 외의 많은 단말들은 이를 지원하지 않고 있는 현실이다. 이러한 CC/PP 기술의 적용이 더딘 이유는 클라이언트와 서버 양자 간에서 나타나는 문제로서, 이러한 문제는 클라이언트 단말들이 프로파일을 전송하기 전까지 서버는 프로파일에 대한 처리를 지원 하지 않으며, 반면에 서버가 프로파일 처리를 지원하지 않기 때문에 클라이언트 개발자들은 프로파일 지원기능을 단말에 추가하지 않음으로 인해 발생하고 있다. 현재 이러한 문제를 해결하기 위해 모든 단말에 대한 프로파일을 제작함으로서 CC/PP 기술의 구현 레퍼런스를 늘려야 한다는 요구들이 나오고 있다[12-13]. 이렇듯, 현재의 컴퓨팅 환경은 CC/PP를 지원하는 단말과 지원하지 않는 단말들이 혼재한 상황이며, 이 중 CC/PP를 지원하는 단말은 UAProf 을 지원하는 소수의 휴대폰에 국한되어 있음으로 인해 서비스 제공측이 장치 독립적 서비스 제공의 기반이 되는 단말 특성 프로파일의 처리를 지원하지 않는 현실이다.



이에 본 논문에서는 유비쿼터스 환경 하에서 CC/PP 프로파일을 지원하는 단말뿐만이 아니라 지원하지 않는 단말에 대해서도 그 특성을 추출, 분석할 수 있도록, 프로파일을 지원하지 않는 단말에 대해 서비스 요청정보를 분석, 규칙에 기반하여 프로파일을 생성하여 단말의 특성 정보를 얻을 수 있도록 하는 규칙기반 CC/PP 프로파일 생성 방법을 제안한다. 제안한 방법은 다양한 이기종 단말들의 특성을 추출, 분석하여 이를 기반으로 해당 단말에 적합한 서비스를 제공할 수 있는 인프라의 역할을 할 것이다.

본 논문의 구성은 다음과 같다. Ⅱ장에서는 CC/PP를 중심으로 한 관련연구를 살펴보고, Ⅲ 장에서는 본 논문에서 제시한 규칙 기반 CC/PP 프로파일 생성 방법에 대해 살펴보며, Ⅳ장에서는 본 논문에서 제안한 방법을 적용한 결과를 보이고, 마지막 Ⅴ장에서 결론 및 향후 연구 방향을 제시한다.

# Ⅱ. 관련 연구

## 1. HTTP 요청헤더

기존의 웹서버들은 클라이언트 단말의 특성을 파악하기 위해서 HTTP 요청헤더 상에서 User-Agent 필드와 Accept 요청헤더 필드를 분석하는 방법을 사용한다. 이 중 User-Agent 항목은 다음과 같은 포맷으로 구성된다.

Browser / version (platform ; security-level ; OS-or-CPU description)

첫 부분에 브라우저명과 그 버전을 표시하며 괄호 안에 단말의 플랫폼과 보안레벨, 운영체제 또는 CPU 에 대한 정보를 세미콜론으로 구분하여 표시한다. 실제 사용되는 예는 다음과 같다.

Mozilla/4.04 (X11; I; SunOS 5.4 sun4m)
Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
Mozilla/1.22 (compatible; MSMB13; LG-KP6100; CellPhone)



마이크로소프트사의 인터넷 익스플로러 브라우저가 Mozilla 라는 브라우저 명을 사용하는 이유는, 익스플로러가 개발된 당시의 서버가 프레임을 지원하는 브라우저를 모질라만 인식하였기 때문이다. 이러한 이유로 마이크로소프트사는 브라우저명은 Mozilla로 두고 compatible 항목에 MSIE를 지정하는 방식을 사용하는 에이전트 은폐(cloaking)를 사용하였고, 후에 개발된 브라우저들은 이러한 방식을 그대로 따르게 되었다.

이러한 User-Agent 필드를 통한 클라이언트 단말의 특성 파악 방법은 크게 두 가지 방법으로 구분되어 진다. 첫 번째 방법은 단말 특성과 적용 콘텐츠의 묵시적인 연결방식으로 프로그램이 User-Agent 항목을 검사하여 적용되는 콘텐츠를 선택하는 방식이다. 즉 User-Agent 상에 Nokia가 있는 경우와 MSIE가 있는 경우, 제공하는 콘텐츠를 달리하는 방식이다. 이러한 방식은 단말 특성에 관한 명시적인 내용 없이 단말의 아이디만으로 제공할 콘텐츠를 결정한다.

두 번째 방법은 단말 특성과 해당 단말에 적용할 콘텐츠를 명시적으로 연결하는 방식으로서 단말 특성 데이터베이스를 이용하는 방법이다. 콘텐츠 요청을 받았을 때, 서버는 단말의 특성을 결정하기 위해 단말의 아이디를 데이터베이스에

넘기고, 데이터베이스는 단말 아이디에 맞는 해당 단말의 특성을 반환한다. 물론 이 방식도 단말의 아이디를 통해 특성을 결정하고는 있지만, 제공할 콘텐츠 결정에 있어서는 단말의 아이디를 통해 직접 결정하는 것이 아니라, 단말의 특성을 통해 제공할 콘텐츠가 결정되는 방식이다. 사실 이러한 방식은 다음 절에서 설명할 CC/PP와 별 반 다를 것이 없어 보인다. 허나 한 가지 주요한 차이점이 존재하는데 그것은 CC/PP 지원 단말인 경우에는 자신의 특성 정보를 서버에게 스스로 전송하기 때문에, 새로운 단말이 나왔을 때 서버상의 단말 특성 데이터베이스를 주기적으로 업데이트 할 필요가 없다는 장점을 갖게 된다는 것이다.

User-Agent 필드 이외에도 HTTP 요청헤더는 다음과 같은 네 가지의 Accept 요청헤더 필드를 제공한다.

Accept: User-Agent 가 받아들일 수 있는 MIME 타입
Accept-Charset: 사용자가 선택한 캐릭터셋
Accept-Encoding: User-Agent 가 제공하는 인코딩
Accept-Language: 사용자가 선택한 언어

다음은 그 실제 사용 예이다.



```
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg,
        application/x-shockwave-flash, application/vnd.ms-excel,
        application/vnd.ms-powerpoint, application/msword, */*
Accept-Language: ko
Accept-Encoding: gzip, deflate
```

위의 Accept 요청헤더 필드를 분석함으로서 해당 단말은 한국어를 기본으로 사용하며, 인코딩으론 gzip 형식의 압축파일과 평문을 지원하며, gif, x-xbitmap, jpeg, pjpeg 형식의 이미지 파일을 표현할 수 있고, 플래시와 마이크로소프트사의 엑셀, 파워포인트, 워드 형식의 파일을 서비스하는 것이 가능함을 알 수 있다. 콘텐츠를 제공하는 서버는 이러한 내용들을 이용하여 단말의 특성을 어느 정도까지 가늠할 수 있다. 그러나 불행히도 많은 단말들이 Accept 문에 어떠한 MIME 타입과도 매치될 수 있는 확장자인 */* 를 기술함으로서 모호한 정보를 제공하고 있는 것도 사실이다.

이렇듯 HTTP는 근본적으로 사용자, 문맥, 단말의 특성을 기술하기 위한 것이 아니기 때문에 헤더 상에서 직접적으로 제공되는 정보가 부족하며, 헤더 상에 브라우저 특성을 표현하는 방식 또한 정확한 표준이 존재하지 않아, 요청을 받은

서버는 해당 요청헤더에 기술된 정보를 추출한다 하더라도 Accept 필드를 통한 단말이 표현 가능한 MIME 타입과 User-Agent 필드를 통한 요청 웹 브라우저의 식별 등 대략적인 클라이언트의 상황밖에 추정할 수가 없는 한계를 가지고 있다.

## 2. CC/PP

W3C 에서 표준으로 내세운 CC/PP (Composite Capabilities/Preference Profile) 기술은 하드웨어, 소프트웨어, 네트워크, 지원하는 서비스 정보 등 일련의 단말 특성 및 제한조건과 사용자 취향에 관한 정보 기술 및 전송에 대한 규격으로서, 서버나 콘텐츠 제공자에게 전달되어 클라이언트의 요구사항에 맞추어진 서비스 콘텐츠를 클라이언트에게 제공할 수 있도록 한다. 이번 절에서는 이러한 CC/PP 표준에 대해 살펴보고, 클라이언트가 서버로 프로파일을 전송하는 방법에 대해 알아본다.

### 1) CC/PP 와 UAProf



CC/PP 프로파일은 RDF(Resource Description Framework)로 기술되며, 크게 두단계의 계층 구조를 갖는다. 프로파일은 최소한 하나 이상의 컴포넌트(Component)를 가지며, 각 컴포넌트는 최소한 하나 이상의 속성(Attribute)을 갖는다. 프로파일의 첫번째 계층인 컴포넌트는 단말기의 하드웨어 플랫폼, 소프트웨어 플랫폼, 브라우저와 같은 개별적인 어플리케이션 등의 주요 구성요소를 나타내며, 이러한 컴포넌트의 하위 계층으로 오는 속성은 이름과 값이 쌍을 이루는 집합구조로 해당 컴포넌트에 속하는 세부 특성을 나타낸다.

CC/PP 프로파일을 구성하는 컴포넌트와 속성들의 이름 및 제약사항, 그리고 데이터 타입은 프로파일 내에 URI(Uniform Resource Identifier)로 기술되는 용어집(Vocabulary)에 의해 결정된다. CC/PP 표준에서는 특정 용어집을 지정하지 않고, 어떠한 어플리케이션도 자신만의 용어집을 정의하여 프로파일 기술에 사용할 수 있도록 하고 있는데, 현재 CC/PP를 지원하는 휴대폰 기기들은 자신의 프로파일 기술을 위해 WAP 포럼에서 개발한 UAProf 용어집을 사용하고 있다. UAProf 용어집은 Table 1 에서 보는 바와 같이 크게 6가지의 컴포넌트를 정의하고 있다.

Fig. 1 은 이러한 UAProf 용어집을 따르는 CC/PP 프로파일 구성의 예를 보인다. 프로파일 구성을 보면 HardwarePlatform, SoftwarePlatform, BrowserUA 등

Table 1. Component of UAProf Vocabulary

| Component | Description |
|---|---|
| HardwarePlatform | describe the hardware charateristics of the terminal |
| SoftwarePlatform | describe the operating environment of the device |
| BrowserUA | describe the browser application of the device |
| WapCharacteristics | WAP capabilities of the terminal |
| NetworkCharacteristics | network related infrastructure |
| PushCharacteristics | desctibe the push specification of the device |

의 컴포넌트들은 URI로 지정된 용어집 상에 정의된 컴포넌트들이며, 이러한 각각의 컴포넌트 밑에는 ImageCapable, ScreenSize, BrowserName과 같은 속성들이 그들의 데이터 타입에 맞게 기술되어 있다.



Fig. 1 Structure of CC/PP profile

UAProf 용어집은 개별적인 속성을 구별할 수 있는 이름과 속성유형, 집합(collection) 유형, 그리고 분석(resolution) 규칙의 집합으로 컴포넌트를 구성하는 각 속성들을 정의한다.

4가지로 구분되는 속성유형은 BrowserUA 컴포넌트 내의 BrowserName 속성과 같은 일반적인 문자열의 값을 갖는 String 타입, HardwarePlatform 내의 ImageCapable 과 같이 Yes/No 중 하나의 값만을 가질 수 있는 Boolean 타입, BrowserVersion 과 같이 숫자를 그 값으로 갖는 Number 타입, 마지막으로 ScreenSize 와 같이 양수의 쌍으로 나타내어지는 Dimension 타입 중 하나로 정의된다.

집합 유형은 하나의 값만을 갖는 Simple 타입, 순서가 없는 여러 개의 값을 가질 수 있는 Bag 타입, Bag 과 같이 여러 개의 값을 가질 수 있으나 해당 값들이 순서적으로 우선순위를 갖는 Seq 타입으로 구별된다.

속성과 관련된 값 중 마지막인 분석 규칙은, 프로파일 내에 같은 이름의 속성이 둘 이상 선언되어 있을 때의 분석방법 또는, 단편화된 여러 개의 프로파일을 하나로 합병할 때 같은 이름의 속성들을 어떻게 처리할지에 대한 규칙을 정의한다. 먼저 분석 규칙이 Locked으로 정의된 속성은 첫 번째로 정의된 속성 값만을 사용하고 그 후에 나오는 같은 이름의 속성 값들은 무시한다. 분석 규칙이 Override로 정의되면, Locked와 달리 제일 마지막에 나오는 속성 값만을 사용하며, Append 규칙으로 선언 되었다면 프로파일 내에 나타나는 모든 속성 값들을 Bag 집합유형과 같은 형태로 사용한다.

2) W-HTTP 프로토콜

앞에서 설명한 바와 같이 CC/PP 기술은 일련의 단말 특성 정보를 클라이언트가 서버에게 스스로 전송한다. 그러나 기존의 HTTP 프로토콜에서는 요청헤더상에 프로파일을 추가하기 위한 필드가 정의되어 있지 않아, W3C에서는 CC/PP 프로파일의 HTTP 프로토콜을 통한 전송을 위해 HTTP-ex 라고 알려져 있는 HTTP Extension Framework에 기반을 둔 CC/PP exchange protocol[8]을 표준화 하고 있다. 이와는 별도로 WAP 포럼에서는 CC/PP 프로파일 전송을 위해 W-HTTP(Wireless Profiled HTTP) 프로토콜을 제안하고 있는데 이는 HTTP-ex에 기반을 둔 CC/PP 프로토콜과 기능적으로 일치하며 현재의 HTTP 프로토콜에 적합한 형태로서 CC/PP 프로파일을 지원하는 휴대폰들은 현재 이 방식을 채택하여 사용하고 있다. Fig. 2 는 W-HTTP 프로토콜을 사용한 요청헤더의 예이다.

W-HTTP 프로토콜의 요청헤더는 일반적인 HTTP 요청헤더에 두 가지의 필드가 추가된 형태를 보인다. x-wap-profile 필드와 x-wap-profile-diff 필드가 그것인데, 먼저 x-wap-profile은 해당 클라이언트 단말 특성 프로파일의 위치를 URI를 통해 표현한다. Fig. 2 에서는 "http://localhost:8080/ccpp/profiles/test09defaults.rdf" 라는 위치에 있는 프로파일을 자신의 프로파일로 지정하고 있다. 이러한 URI 다음에 오는 1-Rb0sq/nuUFQU75vAjKyiHw== 라는 문자열은 profile-diff digest로, 처음에 나오는 1- 은 profile-diff 순서번호를 나타내고 나머지 부분은 해당 번호의 profile-diff를 MD5 알고리즘과 Base64 알고리즘을 이용하여 요약한 것이다. 다음으로 x-wap-profile-diff 필드를 보면, 첫 부분에 x-wap-profile에서 지정되었던 순서번호 1 이 나오는 것을 볼 수 있고, 뒤이어

```
GET /ccpp/html/ HTTP/1.1
Host: localhost
x-wap-profile:"http://localhost:8080/ccpp/profiles/test09defaults.rdf",
       "1-RbOsq/nuUFQU75vAjKyiHw=="
x-wap-profile-diff:1;<?xml version="1.0"?>
 <rdf:RDF xmlns="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:prf="http://www.wapforum.org/profiles/UAPROF/ccppschema-20010430#">
  <rdf:Description rdf:ID="MyDeviceProfile">
     <prf:component>
       <rdf:Description rdf:ID="HardwarePlatform">
          <rdf:type
  rdf:resource="http://www.wapforum.org/profiles/UAPROF/ccppschema-
    20010426#HardwarePlatform"/>
             <prf:BitsPerPixel>16</prf:BitsPerPixel>
       </rdf:Description>
     </prf:component>
  </rdf:Description>
 </rdf:RDF>
```

Fig. 2 W-HTTP request header

참조된 프로파일에 추가되어야 할 단말 특성이 XML 단편화로 기술된 형태를 보인다.

요청헤더에 기술된 프로파일은 미리 정의된 프로파일 분석규칙에 따라 분석된다. 먼저 x-wap-profile로 지정된 프로파일이 분석되며, 만일 해당 프로파일이 외부의 다른 프로파일을 참조하고 있다면 해당 프로파일 또한 같이 분석된다. 이후 profile-diff의 값은 용어집에 정의된 각 속성의 분석 규칙에 따라 프로파일과 합성되어 해당 단말기에 대한 최종 프로파일로 해석된다. 이때 같은 이름의 속성이 여러 개 존재한다면, 속성이 갖는 분석 규칙에 따라 해당 값을 선택 또는 합병하게 된다. Fig. 2 의 예에서는 먼저 test09defaults.rdf가 분석된 후 profile-diff에서의 BitsPerPixel 항목이 분석된 프로파일에 추가되는 형식이 된다. profile-diff에서 사용된 용어집인 ccppschema-20010426에 의하면 BitsPerPixel 속성의 분석 규칙은 Override로 지정되어 있기 때문에, test09defaults.rdf에 이미 BitsPerPixel 값이 지정되어 있더라도 이 값은 무시되고 profile-diff에서 지정된 BitsPerPixel 값인 2 가 사용된다.

이렇듯 CC/PP 기술은 단말 특성을 효과적으로 표현할 수 있는 기술임을 알 수 있다. 허나 CC/PP를 지원하는 단말, 즉 자신의 프로파일을 가지고 이를 요청헤더를 통해 서버에 전달하는 단말은 현재 그리 많지가 않다. CC/PP를 지원하는 휴대폰들이 출시되고는 있으나 기존에 생산된 휴대폰들 중 대다수가 이를 지원하지 않고 있으며, PDA와 PC의 경우 CC/PP를 지원하지 않고 있어, 서버가 프

로파일의 분석을 통해 단말의 특성에 최적화된 콘텐츠를 제공할 수 있다고 하더라도 현 상황에서는 그 의미가 크게 줄어드는 현실이다.

## 3. PandA

PandA[14,15]는 Portable and Adaptable CC/PP Proxy and Agent의 약어로서 다양한 설정 하에서의 CC/PP의 퍼포먼스를 측정하기 위한 프로토타입으로 개발되었다. 이를 위해 클라이언트의 요청에 지정된 프로파일을 분석하고, 이러한 프로파일에 적합하도록 웹 콘텐츠의 변환을 처리하는 CC/PP Proxy와 다양한 설정에 따라 프로파일을 구성하여 요청헤더를 통해 전송할 수 있도록 개발된 panda라는 모바일 브라우저로 전체 시스템을 구성하였다.

panda 브라우저는 화면크기, 사용 가능한 색상 수, 표현가능한 이미지 포맷, 기본 프로파일과 표현가능 마크업 언어를 설정함으로서 클라이언트의 특성을 변경할 수 있으며, 이러한 클라이언트의 특성을 CC/PP 프로파일로 표현하기 위해 CC/PP 표준을 따르는 자체 용어집을 정의하였다. Table 2 는 PandA가 사용하는 용어집에서 정의된 항목들이다.

Table 2. PandA Vocabulary



| Component | Attribute | Description |
|---|---|---|
| HardwarePlatform | Model | Model name or number |
| | ScreenSize | Display size |
| | ScreenDepth | Color depth |
| | Speaker | Sound support |
| SoftwarePlatform | HTMLVersion | HTML version |
| UserPreference | ImageResize | Image size |
| | Image | Image format |

이러한 프로파일을 지원하는 PandA의 동작은, 먼저 panda브라우저가 웹서버에 콘텐츠를 요청하면 CC/PP Proxy가 해당 요청을 전달받아 프로파일을 파싱하고 요청 자체는 웹서버로 포워딩 시킨다. 또한 웹서버에서의 응답 콘텐츠는 CC/PP Proxy에 의해 프로파일에 기술된 특성에 적합한 형태로 변환되어 클라이언트에게 전달된다.

이렇듯 프로토타입 형태의 프로젝트인 PandA는 CC/PP 프로파일을 지원하도록 자체 제작된 panda 브라우저의 요청만을 처리할 수 있는 등 제약적인 환경을

갖고는 있으나, 해당 클라어언트의 특성 프로파일에 기초한 콘텐츠 변환을 통해 클라이언트 특성에 최적화된 콘텐츠를 제공할 수 있는 방법을 보여주고 있다.

## 4. CC/PP 비지원 단말에 대한 처리

서론에서 언급한 바와 같이, 현재 사용되는 단말들 중 CC/PP를 지원하는 단말은 UAProf에 기반을 둔 프로파일을 사용하는 소수의 휴대폰에 한정되어 있으며, PDA나 일반 PC상의 브라우저, 그 외의 휴대폰등 많은 단말들은 이를 지원하지 않고 있는 현실이다. 이렇듯 대다수를 차지하는 CC/PP 비지원 단말들이 프로파일을 전송하지 않는 상황으로 인해, 서버가 프로파일 처리에 대한 지원을 하지 않고 있는 문제를 해결하기 위해, 비지원 단말에 대해 프로파일을 지원하도록 하는 방법을 거론하는 몇 가지 연구들을 살펴보도록 한다.

1) DELI

HP 연구소에서 개발된 DELI[16]는 CC/PP를 지원하는 단말의 요청헤더에서 프로파일을 추출하고 이에 대한 처리를 가능하게 해주는 자바 서블릿 기반의 오픈소스 라이브러리로서, W-HTTP 프로토콜을 따르는 헤더 상에서 x-wap-profile과 x-wap-profile-diff를 추출, x-wap-profile상의 URI에서 프로파일을 패치하여 diff와 합성, 최종 단말 프로파일을 생성하는 역할을 한다.

또한 DELI는 기존의 CC/PP 비지원 단말에 대한 프로파일도 지원하고 있는데, 이를 위해 User-Agent 필드 값에서 클라이언트 단말이 사용하고 있는 에이전트 아이디를 추출하고, 시스템 내에 작성된 매핑설정에 따라 에이전트 아이디에 매핑되는 미리 작성된 프로파일을 해당 단말의 프로파일로 사용한다. 다음은 DELI에서 이러한 매핑을 위해 사용되는 legacyDevice.xml 설정파일 형식이다.

```
<?xml version="1.0"?>
<device>
   <legacyDevice>
      <useragentstring>MSIE</useragentstring>
      <profileref>http://www.profiles.org/legacyProfiles/msie.rdf</profileref>
   </legacyDevice>
   <legacyDevice>
      <useragentstring>mozilla</useragentstring>
      <profileref>resources/deli/legacyProfiles/mozSample.rdf</profileref>
   </legacyDevice>
```

</device>

useragentstring에는 User-Agent 필드 상에 나타나는 단말을 유일하게 식별 가능한 문자열을 사용하며, profileref에는 프로파일 저장소 내의 관련 프로파일 위치를 지정한다. 이때, 마이크로소프트 인터넷 익스플로러가 프레임 지원을 위해 User-Agent 필드 상에 Mozilla를 표기하기 시작한 것과 같이 에이전트에 대한 은폐(cloaking)와 단말기 제조업자가 에이전트 명에 자신들의 회사이름을 넣기 위해 에이전트 이름을 특별 제작하는 문제를 고려해야만 한다.

DELI 가 사용하는 User-Agent와 프로파일의 매핑 방법은 CC/PP 비지원 단말에 대한 처리에는 손쉬운 방법이 될 수 있을 것이다. 허나, 현재 사용되는 클라이언트의 종류는 휴대폰만을 생각하더라도 그 수가 상당하며, 앞으로 개발될 다양한 유비쿼터스 단말들 까지 고려한다면 모든 단말에 대한 프로파일을 만들어두고 이에 대한 매핑정보를 유지한다는 자체가 매우 소모적인 작업이 될 것이라 예상할 수 있다. 또한 에이전트 아이디 만으로 프로파일을 1:1 매핑 시킴으로서 에이전트에 종속적이지 않은 특성까지도 해당 단말의 특성으로 사용될 수 있는 단점을 가지고 있다.

2) JSR 188 Reference Implementation



썬 마이크로시스템즈 사에서는 CC/PP 데이터를 처리하기 위한 API 의 개발을 위해 JSR(Java Specification Request)에서 JSR 188 CC/PP Processing을 구성하였다. 여기에서 개발된 J2EE CC/PP Processing Reference Implemantation에서는 CC/PP 비지원 단말의 처리를 위해 API를 사용함에 있어서 4가지 방법 중 한 가지를 구현하도록 제안하고 있다[17]. 이 중에서 비지원 단말인 경우 무시해버리는 방법과 모든 단말에 대해 미리 정해진 프로파일 하나만을 반환하는 방법은 본 논문의 논점과 어긋나기 때문에 배제하도록 한다. 또한 세 번째의 매핑방법은 앞서 DELI에서 보았기 때문에 생략한다. 비지원 단말을 위한 마지막 방법은 요청헤더 상의 Accept 필드와 User-Agent 필드에서 추출 가능한 값으로 프로파일을 생성하는 방법이다. 그러나 요청헤더상에서 직접 추출 가능한 정보만을 통해 프로파일을 생성하는 방법은 직접 추출 가능한 정보의 수가 적다는 한계에 의해 충분한 단말의 특성표현이 어렵다는 단점을 가지고 있다.

# Ⅲ. 규칙 기반 CC/PP 프로파일 생성

Fig. 3 은 프로파일을 생성하기 위해 본 연구에서 제안한 프로파일 생성 흐름도이다. 단말 특성정보 요청이 들어오게 되면, 먼저 클라이언트 단말의 요청헤더를 통해 해당 단말이 CC/PP 지원 단말인지 검사한다. 단말이 CC/PP 프로파일을 지원하는 단말로 확인되면 시스템은 헤더 상에 기술된 프로파일 정보를 얻게 되지만, 만일 클라이언트가 CC/PP 프로파일을 지원하지 않는 비지원 단말인 경우, 시스템은 요청헤더 자체를 파싱하여 유효정보들을 추출하고, 해당 정보를 기반으로 추가정보 생성규칙을 적용하고 사용자 에이전트 매핑 규칙을 적용하여 프로파일을 생성한다.

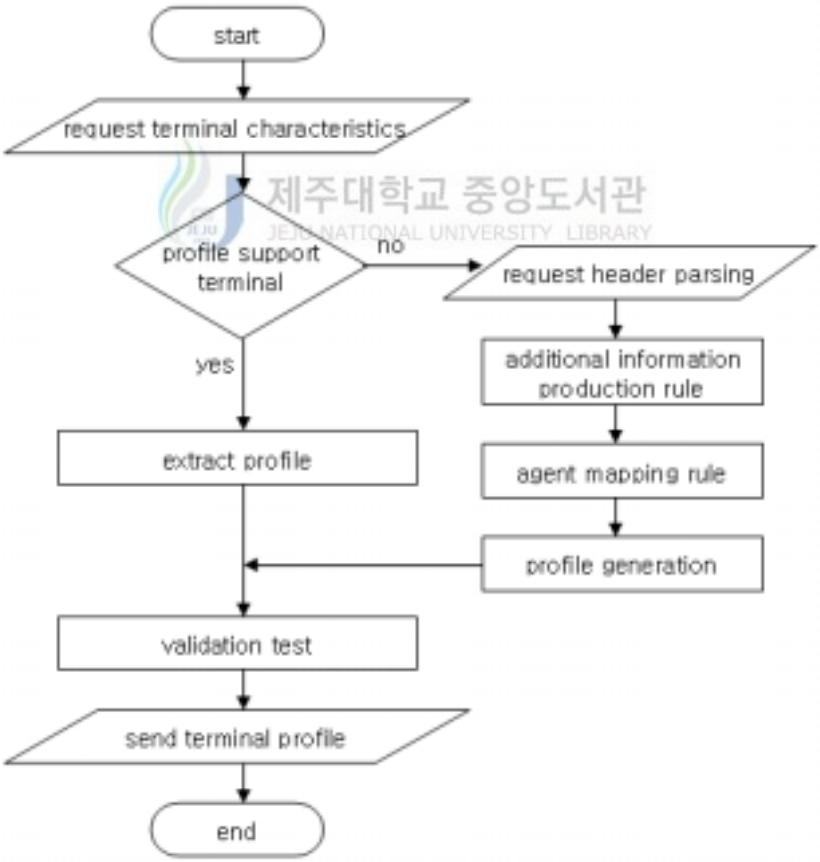



Fig. 3 Flowchart of rule-based CC/PP Profiling

## 1. 용어집

규칙 기반의 단말 프로파일 생성에 앞서, 해당 프로파일이 따라야 할 시스템 용어집을 결정하였다. 규칙 기반으로 생성되는 프로파일이 사용하는 용어집은 WAP 포럼에서 정의한 UAProf 용어집을 기반으로 하였으며, 해당 용어집 중 PandA에서 콘텐츠 변환을 위해 사용한 용어들을 위주로, 일반적인 이미지와 콘텐츠를 표현하기 위해 필요한 속성들을 선별하여 Table 3 과 같이 결정하였다.

Table 3. System Vocabulary

| Component | Attribute | Resolution Rule | Description |
|---|---|---|---|
| HardwarePlatform | ColorCapable | Locked | color support |
| | BitsPerPixel | Locked | color depth |
| | ImageCapable | Locked | image support |
| | SoundOutputCapable | Locked | sound support |
| | ScreenSize | Locked | display size |
| | Model | Locked | model name/number |
| BrowserUA | HtmlVersion | Locked | HTML version |
| | JavaScriptEnabled | Locked | javascript support |
| | JavaScriptVersion | Locked | javascript version |
| | TablesCapable | Locked | table support |
| | FramesCapable | Locked | frame support |
| | BrowserName | Locked | browser name |
| | BrowserVersion | Locked | browser version |
| SoftwarePlatform | CcppAccept | Append | MIME type |
| | CcppAccept-Encoding | Append | encoding |
| | CcppAccept-Language | Append | language |
| | CcppAccept-Charset | Append | character set |
| WapCharacteristics | WmlVersion | Locked | WML version |
| | WmlScriptVersion | Locked | WMLScript version |

시스템 용어집 내의 다양한 속성들 중 콘텐츠 변환을 위해 생성하여야 할 최소한의 속성들은 이미지 지원 여부를 나타내는 ImageCapable과 색상정보를 위한 ColorCapable, BitsPerPixel, 화면에 표시할 콘텐츠의 크기 결정을 위한 ScreenSize 및 표현형식을 위한 TablesCapable, FramesCapable과 마지막으로 단말이 지원하는 MIME 형식을 나타내는 CcppAccept로 결정하였다. 이는 PandA의 용어집을 기반으로 한 단말에 서비스되는 웹 콘텐츠의 변환을 위해 필요한 최소한의 속성들이다.

## 2. 요청헤더 상의 정보 추출

요청헤더는 크게 프로파일을 지원하는 W-HTTP 요청헤더와 프로파일을 지원하지 않은 요청헤더로 구분이 가능하다. 이 중 프로파일을 지원하지 않는 비지원 단말인 경우, 요청헤더 자체를 분석하여 단말기의 특성을 알아내어야 한다. 요청헤더를 구성하는 필드 중 단말의 특성을 추론하는데 사용 가능한 필드는 클라이언트 단말이 PC, PDA, 휴대폰인 경우가 각기 다르다. 이에 먼저 프로파일을 지원하는 요청헤더의 분석방법을 알아보고, 이어 HTTP 표준헤더 상에서의 추출 가능한 정보, 그리고 HTTP 표준헤더를 기반으로 하지만 각 단말 및 개발사 별로 확장하여 사용하는 비표준 요청헤더에서 프로파일 생성에 필요한 필드 값을 추출하는 방법을 기술한다.

1) W-HTTP 요청헤더

프로파일을 지원하는 W-HTTP 요청헤더인 경우, W-HTTP 프로토콜 규격에 따라 해당 요청헤더를 파싱하여 프로파일 추출을 위한 정보를 얻어낸다. W-HTTP 프로토콜은 CC/PP 프로파일을 위해 두 가지의 필드를 사용한다. x-wap-profile은 URI를 통해 해당 단말의 프로파일을 지정하며, x-wap-profile-diff는 RDF로 URI 로 표현된 해당 단말의 프로파일에 추가되는 추가 특성을 표현한다. 그러므로 요청헤더 상에서 x-wap-profile로 지정된 URI와 x-wap-profile-diff인 RDF 구문을 추출함으로서 요청헤더상의 정보 추출을 완료한다.

2) HTTP 표준 요청헤더

프로파일을 지원하지 않는 요청헤더의 경우 HTTP 요청헤더 상의

User-Agent 필드와 Accept 요청헤더 필드에서 정보를 추출한다. 이 중 User-Agent 항목에서는 에이전트 명과 버전, 플랫폼을 추출하며, Accept 필드에서는 지원하는 MIME 타입, 언어, 캐릭터셋, 인코딩을 추출한다. 다음은 마이크로소프트 인터넷 익스플로러의 요청헤더 예이다.

```
GET / HTTP/1.1
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg,
        application/x-shockwave-flash, application/vnd.ms-excel,
        application/vnd.ms-powerpoint, application/msword, */*
Accept-Language: ko
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
Host: localhost
Connection: Keep-Alive
```

먼저 Accept, Accept-Language, Accept-Encoding 필드는 프로파일의 속성들 중 SoftwarePlatform 컴포넌트 내의 속성인 CcppAccept, CcppAccept-Language, CcppAccept-Encoding으로 매핑 된다. 이때 Accept 필드의 값 중 모든 MIME 타입에 적용되는 */* 는 그 특성상 특성정보에서 배제된다.

User-Agent 필드에서는 MSIE 6.0과 Windows NT 5.1 값을 추출한다. MSIE 6.0은 BrowserUA의 BrowserName과 BrowserVersion으로 매핑되며, Windows

Table 4. Extraction value from HTTP request header

| Header Field | Description | Attribute |
|---|---|---|
| Accept | list of MIME types | CcppAccept |
| Accept-Language | list of preferred languages | CcppAccept-Language |
| Accept-Charset | list of charater sets | CcppAccept-Charset |
| Accept-Encoding | list of transfer encodings | CcppAccept-Encoding |
| User-Agent | name of user agent | BrowserName |
| | agent version | BrowserVersion |
| | platform | Model |

NT 5.1은 HardwarePlatform의 Model로 설정한다.

Table 4 는 표준헤더상에서 직접 추출 가능한 정보와 그에 대한 설명 및 프로파일 생성시 해당 정보가 어떤 속성 요소들로 변환 가능한지를 보여주고 있다.

표준은 지키고 있으나 User-Agent 필드 상에서 직접적으로 에이전트 명과 버전, 플랫폼을 추출할 수 없는 경우도 있다. 다음 예는 SKT 단말이 사용하는 SK-VM 1.0.1 규격에 따른 User-Agent 필드의 값이다.

User-Agent: STI45SS3000111281282108120693385 3;150;4;32904;494;2236

위의 예에서 보이는 User-Agent 구문은 SK-VM 1.0.1 규격에 따른 규칙을 적용, 단말의 특성을 추출하여 사용하게 된다.

3) 비표준 요청헤더

단말의 특성을 표현하기 위해 표준헤더 상에 몇 가지 비표준 확장 필드를 추가하여 사용하는 단말들이 존재한다. 표준은 아니지만 이러한 확장 필드 상에서 추출할 수 있는 값은 프로파일을 구성하기 위한 좋은 정보가 된다. 다음은 이러한 비표준 요청헤더를 사용하는 단말 중 PocketPC 헤더의 예이다.



User-Agent = Mozilla/4.0
(compatible; MSIE 4.0.1; Windows CE; PPC; 240x320)
Accept = */*
Accept-language = euc-kr
Accept-encoding = gzip, deflate
UA-OS = Windows CE (Pocket PC) -Version 4.21
UA-color = color16
UA-pixels = 240x320
UA-CPU = Samsung S3C2440
UA-Voice = FALSE
UA-Language = JavaScript

PocketPC인 경우는 UA 접두사를 갖는 비표준 확장 필드를 사용한다. Table 5. 는 PocketPC 헤더 상에서 직접적으로 추출 가능한 정보와 해당 정보가 프로파일 생성시 어떠한 속성들로 변환 가능한지를 보인다. 이러한 비표준 확장 필드를 요청헤더에 추가하는 방식은 PocketPC 이외에도 Openwave의 UP 브라우저, AvantGo 브라우저, KTF 단말 등이 사용하고 있다. 다음 Table 6과 7, 8은 Openwave의 UP 브라우저와 AvantGo 브라우저, KTF 단말의 헤더에서 직접적으로 추출 가능한 정보와 프로파일 생성시 어떠한 속성들로 변환 가능한지를 보인다.

Table 5. Extraction value from PocketPC request header

| Header Field | Description | Attribute |
|---|---|---|
| User-Agent | agent name | BrowserName |
| | agent version | BrowserVersion |
| | platform | Model |
| Accept-Language | list of languages | CcppAccept-Language |
| Accept-encoding | list of encodings | CcppAccept-Encoding |
| UA-color | color depth | BitsPerPixel |
| UA-pixels | display size | ScreenSize |
| UA-Language | javascript support | JavaScriptEnabled |
| UA-Voice | sound support | SoundOutputCapable |

Table 6. Extraction value from Openwave request header

| Header Field | Description | Attribute |
|---|---|---|
| User-Agent | agent name | BrowserName |
| | agent version | BrowserVersion |
| | model name | Model |
| Accept-Language | languages | CcppAccept-Language |
| Accept-encoding | encodings | CcppAccept-Encoding |
| X_UP_DEVCAP_SCREENDEPTH | color depth | BitsPerPixel |
| X_UP_DEVCAP_SCREENPIXELS | display size | ScreenSize |
| X_UP_DEVCAP_ISCOLOR | color support | ColorCapable |
| X_UP_DEVCAP_CHARSET | character sets | CcppAccept-Charset |



Table 7. Extraction value from AvantGo request header

| Header Field | Description | Attribute |
|---|---|---|
| User-Agent | agent name | BrowserName |
| | agent version | BrowserVersion |
| | model name | Model |
| Accept-Language | languages | CcppAccept-Language |

| | | |
|---|---|---|
| Accept-encoding | encodings | CcppAccept-Encoding |
| X-AvantGo-ColorDepth | color depth | BitsPerPixel |
| X-AvantGo-Screensize | display size | ScreenSize |

Table 8. Extraction value from KTF request header

| Header Field | Description | Attribute |
|---|---|---|
| User-Agent | agent name | BrowserName |
| | agent version | BrowserVersion |
| | model name | Model |
| Accept-Language | list of language | CcppAccept-Language |
| HTTP_DEVICE_INFO | display size, color depth | ScreenSize, BitsPerPixel |
| HTTP_DRIVER_INFO | image support | ImageCapable |
| | sound support | SoundOutputCapable |



## 3. 규칙 기반 프로파일의 생성

요청헤더의 분석이 완료된 후 분석된 값들을 기반으로 프로파일을 생성하게 된다. 생성과정은 크게 두단계를 거치게 되는데, 먼저 요청헤더에서 추출된 정보를 토대로 추가정보 생성규칙을 적용하여 추가적인 속성들을 생성하고, 이어 요청헤더에서 추출된 정보 중 브라우저 이름과 버전을 통해 매핑 되는 에이전트 프로파일을 추출하여 합성함으로서 해당 단말에 대한 프로파일 생성을 마친다.

1) 추가정보 생성 규칙

요청헤더상에서 추출된 정보는 그 정보 그대로도 단말의 특성을 표현하지만, 분석을 통해 추가적인 특성 정보의 추출이 가능하기도 한다. 추가정보 생성을 위해서는 요청헤더상에서 추출된 모든 정보를 대상으로 생성규칙이 적용되는지를 검사하여, 해당하는 정보에 따른 추가정보를 생성한다. 여기에서는 추가정보 생성을 위해 사용되는 규칙들에 대해 몇 가지 예를 보인다. 먼저 User-Agent 필드 값의 예를 본다.

SKT 단말 : STI45SS3000111281282108120693385 3;150;4;32904;494;2236
KTF 단말 : Mozilla/1.22 (compatible; KUN/1.2.3; SPH-V6900; CellPhone)
UP.Browser : UPG1 UP/4.0.10 UP.Browser/4.0.10-XXXX
UP.Link/4.1.HTTP-DIRECT

위의 예와 같이 User-Agent 필드를 구성하는 방법은 단말 및 개발사에 따라 형식이 틀림을 알 수 있다. 그러나 이러한 User-Agent 필드에는 부가적인 정보가 존재하는데 SKT의 경우 자체적인 SK-VM 규격에 따라 User-Agent를 기술함으로 이를 토대로 부가 정보를 추출하는 것이 가능하다. 먼저 Table 9 는 User-Agent의 분석을 통해 알아낼 수 있는 추가정보 생성 규칙이다.

Table 9. Creation of additional information from User-Agent

| User-Agent | Description | additional information |
|---|---|---|
| SKT, STI,<br>KTF, HSP, LGT | SK-VM spec. | SK-WML |
| 111, 121 | AUR browser | WML |
| itouch and UP.Browser/4.1 | 017, UP4.1 terminal | WML, bmp/wbmp<br>sound not support |
| itouch only | 017 terminal | HDML, bmp<br>sound not support |
| ezweb and UP.Browser/4.1 | 019, UP4.1 terminal | WML |
| ezweb only | 019 terminal | HDML |
| MSMB12C | ME browser | mHtml, 256 color<br>frame support |
| MSMB13 | ME browser | mHtml,<br>frame not support |
| UP.Browser/3.1,<br>UP.Browser/3.2 | UP 3.1/3.2 | HDML |
| UP.Browser/4.0 4.1 5.0 6.0 | UP 4.0/4.1/5.0/6.0 | UP-WML |
| KUN | 016, KUN browser | kHtml, jpg |

앞서 설명한 바와 같이 SKT는 SK-VM 규격에 따라 일련의 문자열 규격인 AAABCCDDEEEEFFGGGHHHIIIJJKKLLLMNNNNNNNN로 User-Agent를 구성한다. 다음 Table 10 은 SK-VM 규격에 따른 User-Agent 필드 상에서 추출 가능한 정보들이다.

Table 10. Creation of additional informations from SK-VM

| Format | Description | |
|---|---|---|
| HHHIII | display size | HHH X III |
| LLL | sound support | LX0 is not support sound |
| M | color depth | 1 : black and white<br>2 : 4 gray<br>3 : 16 Color<br>4 : 256 Color |

User-Agent 필드에 의한 정보 추출 외에 Accept 필드에서 제공되는 지원 MIME 타입 또한 추가정보의 추출에 이용된다. MIME 타입내의 주요 정보로는 먼저 이미지 포맷을 통한 해당 단말의 색상지원 여부와 지원 마크업 언어를 통한 프레임 및 테이블의 지원여부를 알수있다. 다음 Table 11 은 이미지 포맷별 색상 지원여부이며, Table 12 는 마크업 언어별 프레임과 테이블 지원여부이다.

Table 11. Creation of additional informations from image type

| Image format | color | ColorCapable |
|---|---|---|
| wbmp | black and white | No |
| bmp | b&w, gray, color | No |
| jpg | color | Yes |
| png | color | Yes |
| nbmp | black and white | No |
| gif | gray, color | No |

추가정보 생성 규칙을 통해 생성된 정보는 기존 헤더 정보에 추가되어 프로파일로 생성된다. 여기서 생성된 프로파일은 다음의 에이전트 매핑 규칙을 통한 프로파일과 합성되어 해당 단말의 최종 프로파일로 사용된다.

Table 12. Creation of additional informations from markup language

| Mark-Up language | FramesCapable | TablesCapable |
|---|---|---|
| HTML | Yes | Yes |
| SK-WML | No | Yes |
| HDML | No | No |
| mHTML | Yes (ME12) / No (ME13) | Yes |
| UP-WML | No | Yes |

2) 사용자 에이전트 매핑 규칙

사용자 에이전트 매핑 규칙은 요청헤더에 대한 추가정보 생성 규칙을 통해서 알아내기 힘든 단말이 갖는 특성을 기술하기 위해 사용한다. 이는 추가정보 생성 규칙의 적용 대상인 요청헤더의 정보 자체가 해당 단말에서 사용되는 브라우저 위주의 특성을 주로 기술하고 있기 때문에, 단말 자체가 갖는 하드웨어적인 특성을 생성하기 어렵다는 단점을 보완하기 위한 것이다.

매핑은 User-Agent 필드 상에서 나타나는 브라우저명과 모델명 각각에 대해 매핑 되는 프로파일이 있는지 검사하고, 매핑 프로파일이 있을 때 해당 프로파일을 추출하여 사용한다. 여기서 브라우저명과 모델명을 구분하여 매핑 시키는 이유는 해당 플랫폼이 갖는 특성과 그 플랫폼 상에서 동작하는 브라우저가 갖는 특성이 서로 독립적이기 때문이다. 이를 위해 브라우저 명에 매핑 되는 프로파일인 경우 추가정보 생성 규칙에서 생성 가능한 정보 이외의 BrowserUA 컴포넌트 위주의 특성을 기술하고, 모델명에 매핑 되는 프로파일은 해당 단말의 하드웨어적 특성인 HardwarePlatform 컴포넌트 위주로 그 특성을 기술한다.

브라우저 명에 의한 매핑에서 고려해야할 사항은 해당 브라우저가 Mozilla 계열인지 구분하여야 한다는 점이다. 마이크로소프트 인터넷 익스플로러와 같이 에이전트 은폐를 위해 Mozilla를 User-Agent에 기술하는 브라우저로는 KUN, ME와 같은 모바일 브라우저와 파이어폭스, 컹커러와 같은 데스크톱 브라우저가 존재한다. 이러한 모질라 계열은 모두 Mozilla라는 문자열을 User-Agent에 포함하고 있으므로, 먼저 Mozilla를 포함한 에이전트 명을 추출하고 그 안에서 실재 클라이언트가 사용하는 에이전트 명을 다시 검색하여 그 이름에 매핑 되는 프로파일을 찾게 된다. User-Agent에 Mozilla를 기술하지 않는 모바일 브라우저로서는 Openwave 사의 UP 브라우저, AUR 브라우저 등이 있다.

다음은 마이크로소프트 인터넷 익스플로러 버전 6 에 해당하는 프로파일

msie_6.rdf의 일부이다. 해당 버전의 인터넷 익스플로러는 Html 4.0과 JavaScript 1.3 버전을 지원한다.

```
<rdf:Description rdf:ID="BrowserUA">
    <rdf:type rdf:resource=
      "http://www.wapforum.org/profiles/UAPROF/ccppschema-20010430#BrowserUA"/>
    <prf:BrowserName>Microsoft</prf:BrowserName>
    <prf:BrowserVersion>6.0</prf:BrowserVersion>
    <prf:HtmlVersion>4.0</prf:HtmlVersion>
    <prf:JavaScriptEnabled>Yes</prf:JavaScriptEnabled>
    <prf:JavaScriptVersion>1.3</prf:JavaScriptVersion>
</rdf:Description>
```

다음은 모델명에 의한 매핑 프로파일의 예로 SPHX4200 휴대폰이 갖는 하드웨어적 특성을 기술하고 있다.

```
<rdf:Description rdf:ID="HardwarePlatform">
    <rdf:type rdf:resource=
"http://www.wapforum.org/profiles/UAPROF/ccppschema-20010430#HardwarePlatform"/>
    <prf:TextInputCapable>Yes</prf:TextInputCapable>
    <prf:Vendor>Samsung</prf:Vendor>
    <prf:SoundOutputCapable>Yes</prf:SoundOutputCapable>
    <prf:Model>SPHX4200</prf:Model>
    <prf:keyboard>PhoneKeypad</prf:keyboard>
</rdf:Description>
```



3) 프로파일 합성

단말의 최종 프로파일을 생성하기 위해 요청헤더에서 추출된 정보와 추가 생성 규칙에 의해 생성된 정보, 사용자 에이전트 매핑 규칙에 의해 추출된 매핑 프로파일을 합성한다. 프로파일의 합성은 요청헤더상에서 추출된 정보와 추가정보 생성 규칙에 의한 추가 정보를 토대로 생성한 프로파일을 기본으로 하여, 여기에 매핑 규칙에 의해 추출된 프로파일을 합성하여 최종 프로파일을 생성한다. 프로파일의 합성시 같은 이름의 속성이 중복되어 나타날 경우는 용어집에 정의된 분석 규칙에 따라 해당 속성의 값을 결정한다.

# Ⅳ. 구현 결과 및 평가

이번 장에서는 규칙 기반 CC/PP 프로파일 생성 방법을 구현한 프로파일 생성 시스템의 전체 구조와 이를 구성하는 주요 구성 부분 및 동작방식에 대해서 서술하고, 각 단말 별로 프로파일 생성에 대해 테스트한다.

## 1. 구현 시스템

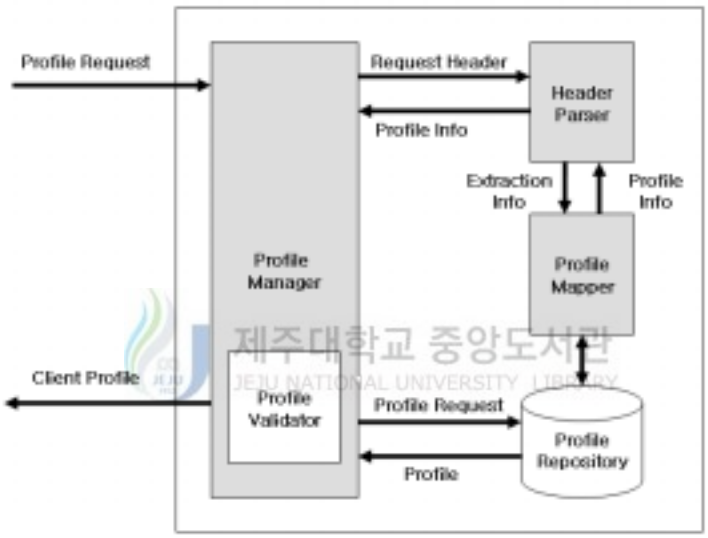



Fig. 4 Structure of CC/PP profiling system

프로파일 생성 시스템의 전체구조는 Fig. 4 와 같이 네 부분으로 구성된다. 프로파일 매니저(Profile Manager)는 프로파일에 관한 정보 요청을 받고 추출 및 생성된 프로파일을 반환하는 역할을 담당하며, 이때 프로파일 검증기(Profile Validator)를 이용 프로파일의 유효성 여부를 검사하게 된다. 헤더 분석기(Header Parser)는 요청 헤더의 분석을 통해 요청 단말의 프로파일 지원여부를 판단하여, 프로파일을 지원하는 단말의 경우 해당 프로파일 정보를 추출 반환하며, 프로파일을 지원하지 않는 단말인 경우 요청헤더의 분석을 통해 프로파일 생성에 필요한 정보를 추출한다. 프로파일 맾퍼(Profile Mapper)는 헤더 분석기에 의해 추출된 정보를 기반으로 추가정보 생성규칙 및 사용자 에이전트 매핑 규칙을 통해 적합한 단말기 프로파일을 생성하며, 프로파일 생성을 위한 기본 정보들의 저장 및 관리는 프로파일 저장소(Profile Repository)가 담당한다.

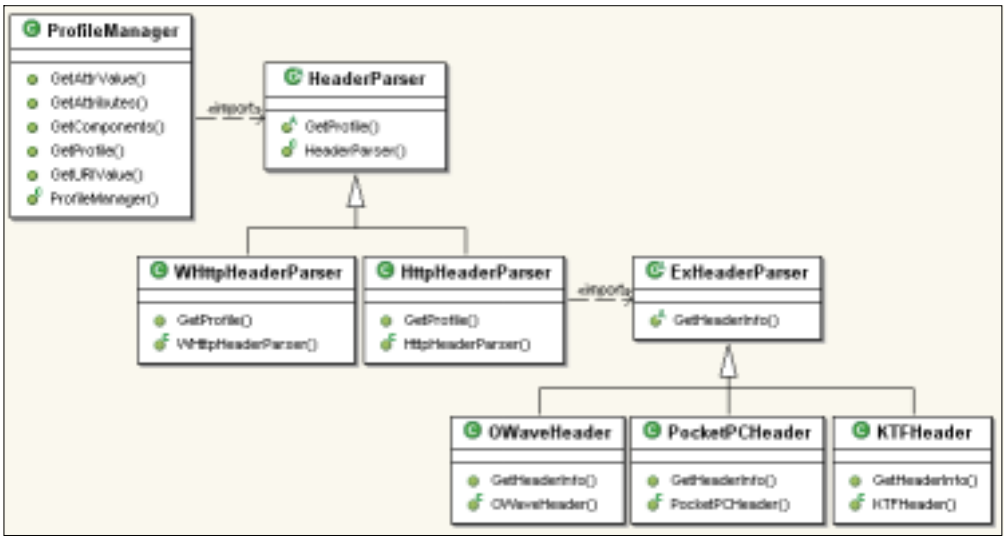


Fig. 5 ProfileManager/HeaderParser class diagram

Fig. 5 는 프로파일 매니저와 헤더 분석기의 클래스 다이어그램이다. 헤더분석기 클래스는 프로파일 매니저에게서 전달받은 요청헤더를 통해 CC/PP를 지원하는 W-HTTP 프로토콜인 경우 WHttpHeaderParser, CC/PP를 지원하지 않는 일반 HTTP 헤더인 경우 HttpHeaderParser를 이용해 헤더를 파싱한다.

WHttpHeaderParser는 W-HTTP 프로토콜을 따를 때는 해당 헤더상의 프로파일 URI를 추출하여 프로파일 맵퍼에게 전달하여 해당 프로파일을 추출할 수 있게 한다. HttpHeaderParser는 표준 HTTP 헤더를 파싱하며, 비표준 확장 HTTP 헤더의 파싱을 위해 ExHeaderParser를 이용한다. 비표준 확장 HTTP 헤더는 각 단말의 비표준 규격에 따라 해당 클래스를 정의하여 파싱하도록 하였으며, 각 클래스는 ExHeaderParser를 상속받아 구현되도록 하여 확장성을 갖추었다.

파싱된 요청헤더 정보는 헤더 분석기에서 프로파일 맵퍼에게 전달되어 프로파일 생성을 위해 사용된다. Fig. 6 은 프로파일 맵퍼 및 프로파일 생성을 위해 사용되는 클래스에 대한 다이어그램이다. 프로파일 맵퍼에게 전달된 헤더 정보는 MakeExtInfo에 의해 추가정보 생성규칙을 거치고 MakeAcceptRequest에 의해 필요정보가 추출되어 MakeHeaderProfile에 의해 프로파일로 변환된다. 이어 MakeMappingProfile을 통해 추출된 사용자 에이전트 매핑 프로파일과 합성되어 최종 프로파일로 생성되며, 프로파일 매니저에서 유효성 검사를 마친 후 최종 프로파일로 사용된다.

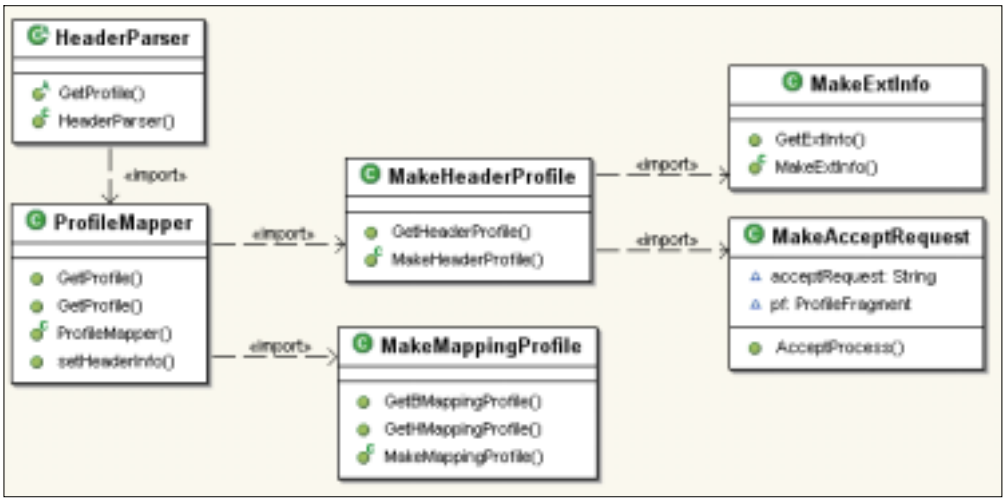


Fig. 6 ProfileMapper class diagram

## 2. 프로파일 생성 결과

프로파일 생성결과를 보기 쉽게 하기 위하여 프로파일 생성 시스템에서 추출 및 생성된 프로파일은 파싱 과정을 거쳐 컴포넌트별로 정렬되어 출력 하였다.

먼저 비지원 단말에 대한 프로파일 생성 테스트에 앞서 CC/PP 지원 단말에 대해 요청헤더상에 기술된 프로파일을 추출하여 처리하는지에 대해 테스트하였다. 지원 단말에 대한 테스트는 Openwave SDK 6.2.2 에뮬레이터를 사용하였다. Fig. 7 은 에뮬레이터의 요청헤더로서 W-HTTP 헤더 규격을 따르며 프로파일은 http://developer.openwave.com/uaprof/OPWVSDK62.xml 을 사용하고 있음을 알 수 있다. Fig. 8 은 해당 단말의 요청헤더 상에서 프로파일을 추출하여 이를 파싱한 결과 중 하드웨어 플랫폼 컴포넌트 부분으로서 CC/PP를 지원하는 단말인 경우 헤더분석을 통해 해당 단말이 사용하는 프로파일을 읽어옴을 알 수 있다.

```
host = localhost:8080
x-up-tpd-session-headers = User-Agent, Accept-Charset, Accept-Language,
user-agent = OPWV-SDK/62 UP.Browser/6.2.2.1.208 (GUI) MMP/2.0
accept-charset = ks_c_5601
accept-language = ko
x-wap-profile = "http://developer.openwave.com/uaprof/OPWVSDK62.xml"
accept-encoding = deflate,gzip
x-up-proxy-enable-trust = 1
x-up-devcap-cc = 1
accept = application/smil, application/vnd.phonecom.mmc-xml, applicatio
connection = close
```

Fig. 7 Request header of Openwave SDK 6.2.2

```
Component [ HardwarePlatform ]
        BitsPerPixel = 24
        ColorCapable = true
        OutputCharSet = [johab, euc-kr, ibm862, windows-1256, iso-8859-3,
        ScreenSizeChar = 12x8
        InputCharSet = [johab, euc-kr, ibm862, windows-1256, iso-8859-3,
        CPU = Pentium
        SoundOutputCapable = true
        ImageCapable = true
        PointingResolution = character
        ScreenSize = 120x160
        NumberOfSoftKeys = 2
        Keyboard = PhoneKeypad
        TextInputCapable = true
        VoiceInputCapable = false
        Model = OPWV-SDK/62
        Vendor = Openwave
        PixelAspectRatio = 1x1
        StandardFontProportional = true
```



Fig. 8 Profile of Openwave SDK 6.2.2

CC/PP 비지원 단말인 경우 프로파일을 생성하는지 여부를 테스트하기 위하여 두개의 휴대폰 에뮬레이터를 사용하였다. Fig. 9 는 CC/PP를 지원하지 않는 UP.Simulator 4.0의 요청헤더로서 비표준 확장 필드를 사용하고 있으며, 이를 기반으로 생성된 프로파일은 Fig. 10 에서 보이고 있다. 또한 Fig. 11 은 또 다른 비표준 확장 필드를 사용하는 KTF의 ME 1.3 브라우저의 요청헤더이며, 해당 요청헤더를 통해 생성된 프로파일은 Fig. 12 와 같다.

```
accept-language = en
content-type = application/x-www-form-urlencoded
accept-charset = KS_C_5601-1987, UTF-8, *
x-up-subno = ssaky_zhayoung
x-upfax-accepts = none
x-up-devcap-charset = KS_C_5601-1987
x-up-devcap-smartdialing = 1
x-up-devcap-screendepth = 1
x-up-devcap-iscolor = 0
x-up-devcap-immed-alert = 1
x-up-devcap-max-pdu = 2000
x-up-devcap-numsoftkeys = 2
x-up-devcap-screenpixels = 171,108
x-up-devcap-msize = 8,18
accept = application/x-hdmlc, application/x-up-alert, application/x-up-cacheop,
user-agent = UPGi UP/4.0.10 UP.Browser/4.0.10-XXXX UP.Link/4.1.HTTP-DIRECT
```

Fig. 9 Request header of UP.Simulator 4.0

```
Component [ SoftwarePlatform ]
        CcppAccept = [text/x-wap.wml, text/html, application/vnd.wap.wml,
        CcppAccept-Language = [en]
        CcppAccept-Charset = [UTF-8, KS_C_5601-1987]

Component [ HardwarePlatform ]
        BitsPerPixel = 1
        ImageCapable = true
        ColorCapable = false
        ScreenSize = 171x108

Component [ WapCharacteristics ]
        WmlVersion = [UP-WML1.1]

Component [ BrouserUA ]
        TablesCapable = true
        FrameCapable = false
```



Fig. 10 Profile of UP.Simulator 4.0

```
user-agent = Mozilla/1.22 (compatible; MSNB13; SPHX4200; CellPhone)
counter = 1
http_phone_number =
http_phone_system_parameter = BASE_ID:326, NID:36, SID:2189, BASE_LAT:0, BASE_LONG:0
http_device_info = LX:120,LY:128,CL:16
http_driver_info = IMG:MSIS|NBMP|GIF,SND:MA2|SMAF
accept = */*
host = 127.0.0.1
accept-language = en
```

Fig. 11 Request header of ME 1.3

```
Component [ HardwarePlatform ]
        ColorCapable = true
        BitsPerPixel = 16
        keyboard = PhoneKeypad
        TextInputCapable = true
        Vendor = Samsung
        Model = SPHX4200
        SoundOutputCapable = true
        ImageCapable = true
        ScreenSize = 120x128

Component [ BrowserUA ]
        TablesCapable = true
        FramesCapable = false
        BrowserVersion = 1.22
        BrowserName = Mozilla

Component [ SoftwarePlatform ]
        CcppAccept = [NBMP, text/mhtml, SMAF, MSIS, GIF, MA2]
        CcppAccept-Language = [en]
```

Fig. 12 Profile of ME 1.3

테스트 결과 CC/PP 지원 단말의 경우 해당 단말이 사용하는 프로파일을 제대로 추출하였음을 알 수 있었고, 비지원 단말의 경우도 규칙에 의하여 프로파일을 생성하였음을 알 수 있었다.

## 3. 평가

생성된 프로파일의 검증을 위해 PandA에서 콘텐츠 변환을 위해 사용한 속성들을 기반으로 마크업언어, 이미지, 색상정보 위주의 몇 가지 속성 요소를 정하여, 각 단말별로 해당 속성들을 생성할 수 있는지를 검사 하였다. 테스트에는 CC/PP 지원 단말인 Openwave와 비지원 단말 중 휴대폰 브라우저로는 UP.Browser 4.0과 KUN 1.2, ME 1.3의 세 가지, PDA 용으로는 마이크로소프트 모바일 익스플로러 3.0, 그리고 일반 데스크톱용 브라우저인 마이크로소프트 인터넷 익스플로러 6.0 버전을 사용하였다.

Table 13 은 각 단말별 생성된 프로파일 검사 결과이다. 결과에 의하면 CC/PP 지원 단말인 경우 해당 단말의 프로파일을 파싱하여, 모든 속성 테스트 요소에 대한 값을 추출 할 수 있었고, 비지원 단말인 경우에서도, 비표준 요청헤더를 사용하는 경우 테스트 요소 속성 값을 모두 생성할 수 있음을 볼 수 있었다. 그러나 마이크로소프트 인터넷 익스플로러 6.0과 같이 표준 요청헤더만을 사용하는

Table 13. Result of attribute generation

| Attribute/device | Openwave | UP 4.0 | KUN 1.2 | ME 1.3 | MME 3.0 | MSIE 6.0 |
|---|---|---|---|---|---|---|
| ColorCapable | O | O | O | O | O | O |
| ImageCapable | O | O | O | O | O | O |
| BitsPerPixel | O | O | O | O | O | X |
| ScreenSize | O | O | O | O | O | X |
| TablesCapable | O | O | O | O | O | O |
| FramesCapable | O | O | O | O | O | O |
| CcppAccept | O | O | O | O | O | O |

경우에는, BitsPerPixel, ScreenSize와 같은 단말기기에 종속적인 정보를 생성할 수 없었는데, 이는 해당 속성이 인터넷 익스플로러 6.0 브라우저가 동작하는 윈도우즈 플랫폼의 성격 상 하드웨어 종속적인 속성이면서 또한 사용자의 선호도에 속하는 속성이여서 매핑 프로파일에 고정된 값을 적용하기가 어렵기 때문이다. 이러한 경우에는 매핑 프로파일 상에 해당 속성이 가질 수 있는 기본 값을 기술함으로서 문제를 해결할 수 있을것이다.

다음으로는 예상하지 않은 단말의 요청 상황에서도 프로파일이 생성되는지를 테스트 하였다. 테스트에는 유닉스 워크스테이션에서 사용되도록 개발된 Lynx 브라우저를 사용하였는데, Lynx는 마우스 없이 키보드만으로 조작할 수 있고 텍스트만을 지원하는 브라우저로서 쉘 인터페이스 상에서 동작하며 표준헤더만을 사용한다. Fig. 13 은 Lynx의 요청헤더로서, 표준헤더를 따르며 추가적인 비표준 확장 필드가 없음을 알 수 있다. 또한 user-agent 필드의 값도 에이전트 매핑 규칙상에 정의되지 않은 에이전트 명으로서, 매핑 프로파일이 미리 정해지지 않은 상황이다.

```
host = 203.253.207.59:8080
accept = text/html, text/plain, audio/basic, audio/x-wav, audio/x-aiff,
accept-encoding = gzip, compress
accept-language = en
user-agent = Lynx/2.8.5rel.1 libwww-FM/2.14 SSL-MM/1.4.1 GNUTLS/1.0.16
```

Fig. 13 Request header of Lynx

Fig. 14 는 Lynx의 요청헤더에 대해 생성된 프로파일을 보인다. 생성된 프로파일이 많은 정보를 보여주고 있지는 않으나, 요청헤더상에서 나타나는 정보 이외에 규칙에 기반을 두어 추가 정보들이 생성되었음을 알 수 있다. 결과적으로 예상하지 않은 단말에 대해서도 해당 단말의 요청헤더를 통해 최소한의 프로파일이 생성 될 수 있음을 알 수 있다.

```
Component [ BrowserUA ]
        FramesCapable = true
        TablesCapable = true

Component [ HardwarePlatform ]
        ImageCapable = false
        SoundOutputCapable = true

Component [ SoftwarePlatform ]
        CcppAccept-Encoding = [compress, gzip]
        CcppAccept-Language = [en]
        CcppAccept = [application/ogg, text/html, audio/basic, audio/x-wav,
```

Fig. 14 Profile of Lynx

# Ⅴ. 결론 및 향후 연구

본 논문에서는 유비쿼터스 환경 하에서 사용되는 다양한 이기종 단말에 대하여, 자신의 특성 프로파일을 제공하는 CC/PP 지원 단말뿐만이 아니라, 이를 지원하지 않는 단말에 대해서도 프로파일에 기반을 둔 특성 추출 및 분석이 가능하도록 하는 규칙기반 CC/PP 프로파일 생성 방법을 제안하였다.

이를 위하여 CC/PP 지원 단말인 경우 요청헤더 상의 단말 프로파일 정보를 이용, 해당 단말의 프로파일을 추출하였고, 비지원 단말인 경우 여러 단말의 서비스 요청헤더 정보를 분석하고, 분석된 요청헤더 정보를 바탕으로 추가적인 특성 정보를 생성할 수 있는 추가정보 생성규칙을 정의하였다.

또한 해당 단말의 브라우저명과 모델명을 기반으로 프로파일을 매핑할 수 있는 사용자 에이전트 매핑규칙을 정의하여, CC/PP 비지원 단말에 대해서도 해당 단말의 특성을 나타내는 프로파일을 생성하였다. 그 결과 CC/PP 지원 단말뿐만이 아니라 비지원 단말에 대해서도 클라이언트 단말의 특성 프로파일을 얻을 수 있었으며, 예상하지 않은 단말에 대해서도 최소한의 프로파일을 생성할 수 있음을 보였다.

본 논문에서 제안한 규칙기반 CC/PP 프로파일 생성 방법은 단말의 CC/PP 지원 여부 관계를 떠나 서버 상에서 클라이언트 단말의 특성을 파악할 수 있도록 함으로써, 앞으로 다양한 이기종 단말들의 특성을 추출, 분석하여 이를 기반으로 해당 단말에 적합한 서비스를 제공할 수 있는 인프라의 역할을 할 수 있을 것이다.

향후 연구 과제로는 CC/PP 프로파일을 통한 장치 독립적 서비스 제공을 위해, 단말 특성에 맞도록 콘텐츠를 제공하는 콘텐츠 협의 기술과, 기존의 콘텐츠를 해당 단말의 특성에 최적화된 콘텐츠로 변환하는 콘텐츠 변환 기술에 대한 추가 연구가 진행되어야 한다.

[참고문헌]

[1] Uwe Hansmann, Lothar Merk, Martin S.Nicklous, Thomas Stober, “유비쿼터스 컴퓨팅 핸드북”

[2] Mark Butler, Fabio Giannetti, Roger Gimson, and Tony Wiley, “Device Independence and the Web”, HP Labs, Bristol, IEEE Internet Computing, September/October 2002, pp 81-86.

[3] “W3C Device Independence Work Group”, http://www.w3.org/2001/di/.

[4] “Delivery Context Overview for Device Independence”, W3C Working Draft,13 December 2002, http://www.w3.org/TR/di-dco/.

[5] “RFC 2616: Hypertext Transfer Protocol -- HTTP/1.1”, 71, 100-104, 145 http://www.faqs.org/rfcs/rfc2616.html.

[6] “Composite Capability/Preference Profiles (CC/PP): Struture and Vocabularies 1.0”, http://www.w3.org/TR/CCPP-struct-vocab/.

[7] Luu Tran, “CC/PP: Structure and Vocabularies Implementation Report”, Sun Microsystems, http://www.w3.org/2003/07/ccpp-SV-PR/test-suite-20030827/implementation-report.html.

[8] “CC/PP exchange protocol based on HTTP Extension Framework”, http://www.w3.org/TR/NOTE-CCPPexchange.

[9] “CC/PP Implementors Guide: Privacy and Protocols”, http://www.w3.org/TR/CCPP-trust/.

[10] “Resource Description Framework (RDF)”, http://www.w3.org/RDF/.

[11] Wireless Application Group, “WAG UAProf proposed version 10-Nov-1999”, www.wapforum.org/what/ technical/SPEC-UAProf-19991110.pdf.

[12] Stuart Lewis, “CC/PP from a developers perspective”, The W3C Workshop on Delivery Context, 4-5 March 2002.

[13] Amy Cowen, “DELI-Style Profile Resolution: Approaching Device Independence”, HP mpulse Magazine, Feb 2002. http://chinese-school.netfirms.com/computer-article-device-independence-resolution.html.

[14] Kinuko Yasuda, "Implementation and Evaluation of Keio CC/PP Implementation", Talk for CC/PP Implementor's Day, Karlstad, 2000 Nov.

[15] Kinuko Yasuda, Takuya Asada, Tatsuya Hagino, "Effects and performance of Content Negotiation Based on CC/PP", Second International Conference on Mobile Data Management, 2001.

[16] Mark H.Butler , “DELI: A DElivery context LIbrary for CC/PP and UAProf”, 2002.



[17] Sun Microsystems, “Composite Capability/Preference Profiles (CC/PP) Processing Specification”, 2003.

[18] Openwave, "UP.SDK Getting Started Guide version 3.1 for HDML", http://developer.openwave.com/docs/31h/start.pdf.

[19] Openwave, "UP.SDK Developer's Guide UP.SDK Release 4.1", http://developer.openwave.com/docs/41/devgd.pdf.

# 감사의 글

쌀쌀해진 바람을 맞으며 뒤돌아보니 어느새 지나온 2년간의 대학원 생활이 제 뒤에 자리잡고 있습니다. 제가 이 길을 제대로 걸어올 수 있었던 것은 아낌없는 가르침을 주셨던 이상준 교수님의 보살핌이 있었기 때문임에, 지도 교수님이신 이상준 교수님께 먼저 감사하다는 말씀을 드리며 지금의 저를 있게 해준 모든 분들께 보내는 감사의 글을 시작할까 합니다.

먼저 논문이 완성되기까지 세심한 지도를 아끼지 않아주셨던 김장형 교수님, 안기중 교수님, 곽호영 교수님, 변상용 교수님, 송왕철 교수님, 변영철 교수님, 김도현 교수님께 감사를 드리며, 특히 변영철 교수님의 지도에 고마움을 전하고 싶습니다.

연구실 내에서 같이 지냈던 선후배님들께도 감사를 드립니다. 갓 입학한 풋내기인 저에게 대학원생의 길이 무엇인지 실천으로 보여주신 김영민 선배님과 호탕한 웃음으로 반겨주시던 이종헌 선배님, 언제나 연구실 살림을 신경 써주신 정은경 선배님, 그리고 항상 밝은 모습으로 제 뒷자리를 지켜준 우종이에게 고맙다는 말을 전합니다. 또한 대학원 생활에 대해 많은 조언을 해주신 김정희 선배님과 한경복 선배님, 허지환 선배님, 오상현 선배님, 양동호 선배님, 그리고 친누나 같이 저를 아껴주신 이정하 선배님, 동생이지만 항상 든든한 친구같이 함께 있어준 경진, 정윤, 그리고 인석이와 훈에게도 이 면을 통해 감사하다는 말을 전합니다. 또한 학교밖에서 저를 지탱시켜준 93학번 동기들인 석민, 재만, 창훈, 철승, 동혁, 영수와 격려와 힘을 주었던 십인방 친구들에게도 고맙다는 말을 남깁니다.

마지막으로 언제나 사랑으로 저희 가족을 지켜주신 어머니와 누님들, 그리고 든든한 형님에게 사랑한다는 말을 전하고, 하늘에서 저희 가족을 하염없는 사랑으로 지켜보고 있을 아버지에게 사랑을 전하며 감사의 글을 마칩니다.

2005년 12월 새로운 발걸음을 내딛으면서...