

# 다양한 장치를 위한 적응력 있는 유저 인터페이스의 연구

김영태\*, 김치수\*

\*공주대학교 컴퓨터공학과

e-mail : zerot@kongju.ac.kr

## Study of Adaptable User Interfaces for Diverse Devices

Young-Tae Kim\*, Chi-Su Kim\*

\*Dept of Computer Engineering, Kong-Ju National University

### 요 약

많은 웹 기반 정보시스템은 다양한 클라이언트 장치와 사용자, 사용자 작업에 대한 시스템 유저 인터페이스의 적합성을 요구한다.

본 논문에서는 개발자가 high-level mark-up 언어를 사용하여 다양한 장치와 사용자, 사용자 작업에 적합한 웹 기반 인터페이스를 명세할 수 있도록 하는 접근방법을 제시하고자 한다. 본 시스템을 통한 단일 인터페이스 명세는 실행 시에 다양한 웹 장치에 대한 인터페이스를 자동으로 제공함으로써 시스템 개발에 유연성과 일관성을 부여할 수 있게 된다.

### 1. 서론

많은 웹 기반 정보시스템은 다양한 클라이언트 장치와 사용자, 사용자 작업에 대한 시스템 유저 인터페이스의 높은 적합성을 요구한다[1].

21세기 초의 매우 중요한 기술적 발전중의 하나는 모바일 폰이나 PDA와 같은 무선 장치를 통해서 인터넷에서 이용될 수 있는 엄청난 양의 정보를 신속하게 제공한다는 것이며, 점점 더 많은 사람들이 무선 장치들을 통해서 언제 어디서나 온라인에 접속하기를 원한다.

그러나 현재 PC를 위해 설계된 웹 페이지를 모바일 폰과 같은 무선 장치에서 사용하기 위해서는 완전한 인터페이스를 위해 재설계, 재사용을 필요로 한다. 그것은 모바일 폰을 지원하는 프로그래밍 언어인 WML이 어려울 뿐만 아니라 컴퓨팅 환경, 네트워크, 장치 화면 크기 등의 제약조건 또한 바뀌어야 하기 때문이다.

또한 상이한 사용자와 사용자 작업에 적용할 수 있는 유저 인터페이스를 필요로 한다. 그러나 콘텐츠의 서비스가 자주 변경되고, 모든 서비스들을 최신으로 유지시키고자 할 때 지속성이 요구되어지고 시간을 많이 소비하게 된다. 결과적으로 많은 수의

인터페이스가 개발되고 유지되어야 한다. 게다가, 각각의 장치마다 특별한 페이지 코드를 쓰는 것은 일관성의 부족과 함께 웹 사이트의 유지보수를 어렵게 만든다.

유저 인터페이스는 다양한 접근 장치의 특성에 적합하고 일관성과 유용성을 보존하면서 상이한 사용 환경에 적합하도록 편의를 도모하는 것이 필요하다.

본 논문에서는, 소프트웨어 개발자가 다양한 사용자와 장치에 대한 인터페이스를 좀더 쉽게 설계하고, 구축하며 전개하도록 하기위한 적응력 있는 유저 인터페이스 개발방법을 제시하고자 한다.

적응력 있는 유저 인터페이스를 위한 본 접근방법은 특정한 장치 특징들을 기반으로 하며, 콘텐츠의 표현 형식뿐만 아니라 사용자와 특정한 작업에 대한 적합성을 포함하고 있다.

### 2. 관련연구

#### 2.1 컴포넌트 기반 개발방법

최근, 컴포넌트 기반의 소프트웨어 구조의 사용에 대한 관심이 증가하고 있다. 이러한 구조는 다른 컴포넌트들에 의해 사용가능하도록 메소드와 속성, 이벤트를 공개하고 소프트웨어 응용프로그램을 구축하

기 위해 대형 컴포넌트 조합을 사용하는 소프트웨어 컴포넌트 객체 개념을 사용한다[2].

다양한 소프트웨어 구조들이 JavaBeans와 COM/DCOM, OpenDoc와 같은 컴포넌트를 사용하여 개발되었고, JBuilder, Visual Javascript, Visual Age 등과 같은 도구들은 컴포넌트와 컴포넌트 기반 응용프로그램을 명세하기 위해 사용된다.

지능적 컴포넌트 기반 유저 인터페이스는 대개 다양한 사용자와 사용자 작업의 적합성을 지원한다[3]. 그러나 클라이언트 장치에서 실행되고, 서버에서는 시행되지 않는 thick-client 인터페이스를 제공하고 장치 적응 능력은 거의 제공하지 않는다.

## 2.2 자동 변환기

다양한 장치들이 지원하는 여러 가지 Markup 언어 사이의 콘텐츠 변환을 위해 특화된 게이트웨이는 HTML을 WAP장치를 위한 WML로 자동 변환해주기 위해 사용된다[4].

자동 변환기 사용의 주요 이점은 다음과 같다.

- 개발자들의 생산성 향상에 따른 개발 속도 증가
- 구입가격이 저렴함
- 본래 웹 사이트에 독립적인 방법

자동 변환 기술들이 WML 브라우저를 위해 HTML을 WML로 변환을 적당히 성공시키지만, 사용자와 작업에 대한 적응을 지원하지는 않는다.

## 2.3 Palm의 웹 클리핑

Palm의 웹 클리핑 접근 방법은 HTML 문서의 많은 내용을 간소화된 WML 버전으로 만들기 위해 자르는 것이다. 전형적으로 웹 클리핑은 작고 동적으로 만들어진 웹 페이지이다. 또한 사용자 질의에 응답하는 CGI 스크립트에 의해 만들어졌다.

클리핑의 목적은 디스플레이 요구사항과 대역폭 사용을 최소화하는 것이다.

그러나 이러한 해결책은 여러 가지 제약을 갖는다. 사용자는 그래픽의 사용을 최소화해야 하며, 게다가 WML 장치를 지원하지 않는다.

## 2.4 사용자 태그와 태그 라이브러리

개발자들은 웹 기반 애플리케이션을 개발하기 위해 JSP를 사용한다. JSP의 가장 큰 장점은 개발자들로 하여금 사용자 태그라 불리는 태그 집합을 생성할 수 있게 해준다는 것이다. 사용자 태그는 하나의 태그 라이브러리 정의로 코드를 재사용할 수 있

다. 본 논문에서는 JSP의 사용자 태그를 다양한 장치를 위한 인터페이스를 생성할 수 있는 태그 집합을 작성하는데 사용한다.

## 3. 적응력 있는 유저 인터페이스의 요구사항

본 논문에서의 적응력 있는 접근방법은 개발자가 어느 장치에나 만족하는 유저 인터페이스를 생성할 수 있는 단일 시스템만을 생성하도록 하여 개발시간 단축과 작업부하 감소를 제공하고자 한다.

다음은 본 논문에서 제시하는 적응력 있는 유저 인터페이스를 위한 요구사항이다.

### 3.1 동적 마크업 언어 생성

다양한 장치들이 다른 마크업 언어를 지원하기 때문에 시스템은 사용자가 애플리케이션을 실행하기 위해 사용할 것으로 예상되는 장치에 적절한 마크업 언어를 동적으로 생성할 수 있는 능력을 제공해야 한다.

적용은 클라이언트 장치가 서비스를 요청할 때 이루어지며, 적용 시스템은 장치를 인지하고 마크업 언어를 결정하며 레이아웃을 표현하게 된다.

### 3.2 다양한 장치에 대한 적응

오늘날의 데스크 탑 장치에서 잠재적으로 무한한 용량의 RAM과 하드디스크 영역이 사용가능함에 따라, 개발자들은 소프트웨어 설계에 보다 적은 제약을 받는다. 그러나 모바일 폰이 낮은 해상도의 작은 화면, 한정된 메모리와 처리능력, 데스크 탑 시스템 보다 적은 대역폭을 갖는다는 것을 기억해야 한다.

또한 많은 장치들은 정보를 디스플레이 하는 스크린 크기가 다르다.

적용력 있는 인터페이스는 주어진 장치의 화면 크기나 메모리 능력에 따라 요소들이 디스플레이 될 수 있어야 한다.

본 논문에서는 적응력 있는 접근방법을 통하여 실행 시에 자동으로 각각의 특별한 장치들을 위해 알맞은 콘텐츠를 생성할 수 있도록 한다.

### 3.3 사용자와 사용자 환경에 대한 적응

많은 유저 인터페이스나 요소들은 어떤 사용자에게는 적합하지만 특별한 사용자 규칙이나 부속 작업이 이행되기에는 적합하지 않은 경우도 있다. 대부분의 적응력 있는 시스템들은 사용자와 사용자 선택에 대한 적응 능력에 초점을 맞춘다. 개별 사용자의

요구사항과 선택은 소프트웨어 시스템을 사용하면서 바뀔 것이다. 적응력 있는 시스템은 사용자의 활동 패턴을 감시하고, 시스템에 의해 제공되는 인터페이스나 콘텐츠를 사용자 선택에 따라 적응되도록 자동적으로 조절한다.

### 3.4 존재하는 클라이언트-서버 구조에 대한 적응

웹 개발자는 웹 애플리케이션의 개발기간 동안 존재하는 클라이언트-서버 구조에 대해 잘 알고 있다. 본 시스템은 개발자들이 잘 알고 있는 기존의 구조를 보다 많이 사용할 수 있게 만들 것이다.

개발자들은 새로운 웹 사이트를 개발하기 위해 새로운 지식을 익힐 필요 없이 그들의 기존 웹 개발 지식을 사용할 수 있다.

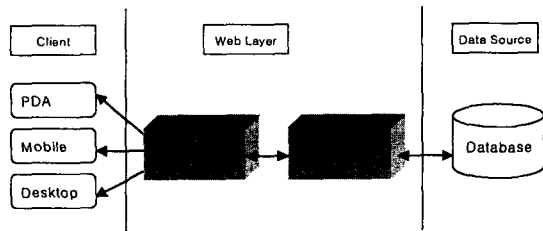
## 4. AUIDD의 설계 및 구현

적응력 있는 유저 인터페이스는 Adaptable User Interface for Diverse Devices(AUIDD) 페이지라 불리는 AUIDD 요소들의 집합을 사용하여 생성하게 된다.

### 4.1 AUIDD를 사용하는 전체 시스템 구조

AUIDD 시스템은 4-tier 구조를 갖는다. 클라이언트는 데스크 탑 PC, WML 기반 브라우저를 사용하는 무선장치나 PDA가 될 수 있다.

다음 [그림 1]은 AUIDD를 사용하는 시스템의 전체 구성도이다.



[그림 1] AUIDD 시스템의 전체 구성도

AUIDD 페이지는 클라이언트와 비즈니스 로직 사이에 위치한다. 웹 레이어는 디스플레이 되는 데이터와 포맷을 검색한다. AUIDD 표현 페이지에서 비즈니스 로직을 분리하는 것은 애플리케이션의 설계에 유연성을 부여하기 위함이다. 다중 사용자 인터페이스는 비즈니스 로직에서 명확하게 정의된 인터페이스의 어떤 변화 없이 표현 계층으로 구성되고

조립된다.

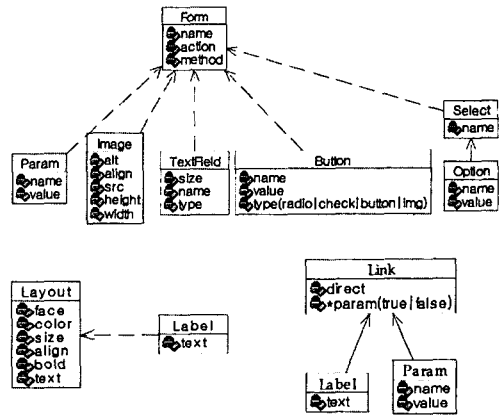
### 4.2 AUIDD 요소의 설계 및 구현

AUIDD 페이지는 AUIDD 요소로 이루어진다. 따라서 상이한 언어로부터 공통 요소들을 추출하여 고수준 요소들의 목록을 생성해야 한다.

AUIDD 페이지의 기본 구조를 명세하기 위해 "Template"이라 불리는 요소를 사용하게 된다. 문서 구조 요소는 HTML 문서와 WML 문서에서 모두 필요한 주된 구조를 제공해야 한다. AUIDD 페이지는 <template> 요소로 시작하고 </template> 요소로 끝나야 하며, 이 사이에 다른 형태의 요소들의 집합이 위치하게 된다.

#### 1) 일반적인 UI 요소들

각각의 인터페이스는 UI 요소들의 집합으로 이루어진다. 다음 [그림 2]는 해당되는 장치를 위한 페이지에 콘텐츠를 표현할 수 있는 적절한 마크업 언어를 생성하는데 사용하게 될 UI 요소들의 집합을 보여주는 다이어그램이다.



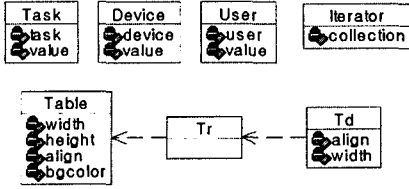
[그림 2] 일반적인 UI 요소들

#### 2) UI 관리 요소들

설계자는 각 장치에 해당되는 각각의 페이지에 대해서 같은 양의 작업을 하기 원할 것이다. UI의 상이한 유형을 관리하기 위해서 사용자, 작업, 장치의 3요소가 사용된다. 이 요소들이 동작하는 방법은 사용자의 유형이나 장치의 유형 등에 따라 AUIDD 페이지 레이아웃의 일부분을 숨기는 것이다.

UI 관리 요소가 UI 요소들이 디스플레이 될지 여부와 어떻게 디스플레이 될지를 결정한다.

다음 [그림 3]은 UI 관리 요소들을 보여준다.



[그림 3]은 UI 관리 요소들

3) 화면 분할 요소들

어떤 환경에서 다른 환경으로 유저 인터페이스를 적절하게 변경할 수 있어야 한다. 따라서 필요한 경우 다중 유저 인터페이스가 동시에 생성될 수 있어야 한다.

모든 아이템이 하나의 디스플레이 장치에 분명하게 디스플레이 되지 못할 때 화면을 다중 화면으로 분할하는 내부적인 변환을 갖게 하기 위한 Group 요소들 group/grouptr/grouptd를 설계했다. Group 요소들은 화면의 UI 요소들의 디스플레이를 관리하고, 컨텐츠가 다른 화면이나 card로 이동되어야 하는지를 결정한다.

4) 템플릿 태그

템플릿 태그는 AUIDD 페이지 상단에 나타나며 장치의 식별, AUIDD 페이지 구조의 제공, 닫힌 태그 </AUIDD:template>로 레이아웃의 출력을 수행하는 매우 중요한 태그이다.

[그림 4]는 템플릿 태그 클래스의 일부로서 요구하는 장치에 대해 각각 적합하게 미리 보호된 WML이나 HTML 문서 구조를 생성하는 코드이다.

```

if(device = "html")
    output.append("<html><head><title>* name= *</title></head><body>")
else{
    output.append("<?xml version= '1.0?'");
    pageContent.getResponse().setContent("text/vnd.wap.wml");
    output.append("<!DOCTYPE wml PUBLIC '//WAPFORUM/DTD WML 1.1//EN'
        http://www.wapforum.org/DTD/wml_1.1.xml><wml>");
}
    
```

[그림 4] 문서 구조를 생성하는 코드의 일부

레이아웃 출력을 위해 "templateTag" 클래스의 "doStartTag" 메소드에서 "output"을 호출하는 "StringBuffer" 개체가 생성되어야 하고, "output" 속성 값으로 할당되어야 한다.

[그림 5]는 출력을 처리하는 "templateTag" 클래스의 코드이다.

```

public int doStartTag() throws JspTagException{
    StringBuffer output=new StringBuffer();
    pageContext.getRequest().setAttribute("output", output);
    ...
    return EVAL_BODY_TAG;
}
public int doEndTag() throws JspTagException{
    output=(StringBuffer)pageContext.getRequest().getAttribute("output");
    ...
    bodyContent.getEnclosingWriter().write(output.toString());
}
    
```

[그림 5] 출력을 처리하는 코드의 일부

인터페이스의 사용 이전에 시스템에서 요구하는 데이터를 위한 데이터베이스 테이블의 생성이 필요하다. 애플리케이션을 디스플레이 하기 위해서 사용될 장치 정보, 사용자 정보, 사용자 선택 사항 등을 입력하게 되며, 입력된 정보를 기준으로 하여 적절한 유저 인터페이스를 생성하게 된다.

5. 결론 및 향후 연구과제

본 논문에서는 다양한 장치, 사용자, 사용자 작업에 따라 적합한 유저 인터페이스를 실행 시에 동적으로 생성하여 제공할 수 있는 방법을 제시하였다.

향후 연구과제로는 본 논문에서 적용한 HTML과 WML 이외의 다른 마크업 언어들에 지원할 수 있도록 시스템을 확장할 필요가 있으며, 제시한 도구에서 사용된 사용자 태그 또한 확장하여 적용할 수 있도록 부가적인 기능들에 대한 연구가 이어져야 할 것이다.

참고문헌

- [1] Van der Donckt,J., Limbourg,Q., Florins,M., Oger,F., Macq,B., "Synchronized, model based design of multiple user interfaces", Workshop on Multiple User Interfaces over the Internet, 2001
- [2] J.C.Grady, J.G.Hosking, "Developing Adaptable User Interfaces for Component based Systems"
- [3] Stephanidis, C. Concept of Unified User Interfaces, In Interfaces for All Concepts, Method and Tools, LAurence Erlbaum Associates, 2001
- [4] Fox,A., Gribble,S., Chawathe,Y., Brewer,E., "Adapting to Network and Client Variation Using Infrastructural Proxies:lessons and perspectives, "IEEE Personal Communications 5(4), 1998