무선인터넷 플랫폼에서의 API 관리자 설계 및 구현

임형택[0], 장준, 최훈
충남대학교 컴퓨터공학과
e-mail : {htlim, jjang, hchoi}@ce.cnu.ac.kr

Design and Implementation of the API Manager on Wireless Internet Platform

Hyoung-Taek Lim[0], June Jang, Hoon Choi
Dept. of Computer Engineering, ChungNam National University

요 약

한국형 무선인터넷 표준 플랫폼인 WIPI 2.0 은 현재 동적 API 추가/갱신 기능을 필수 기능으로 채택하고 있다. 이를 위해 WIPI 플랫폼은 자신이 제공하는 API 들을 관리하고 API 정보를 응용프로그램에게 제공하는 API 관리자를 가지고 있어야 한다. 본 논문에서는 무선망을 통한 API 동적 추가/갱신/삭제 및 API 메타 정보를 통해 API 의 관리를 지원하는 API 관리자의 설계 및 구현에 대해 기술한다.

1. 서론

최근 무선인터넷에 대한 일반 소비자의 욕구가 증대됨에 따라 무선인터넷은 우리 실생활에 없어서는 안 되는 필수 통신수단으로 자리잡고 있다. 그러나 이동통신사 별로 각기 다른 무선인터넷 플랫폼을 사용함으로써 컨텐츠개발자(CP: Contents Provider)와 단말기 제조업체는 이동통신사의 각기 다른 서비스를 이용하기 위해 연구 및 생산에 중복 투자가 불가피했고 컨텐츠와 플랫폼의 호환성 측면에서도 많은 문제점들이 발생하였다. 이에 2001 년 10 월 초 SK 텔레콤, KTF, LG 텔레콤 등 국내 이동통신 3 사는 표준화를 위한 구체적인 요구사항을 제시하였고, 한국전자통신연구원(ETRI: Electronics and Telecommunications Research Institute)과 한국무선인터넷표준화포럼(KWISF: Korea Wireless Internet Standardization Forum)의 노력으로 WIPI (Wireless Internet Platform for Interoperability)가 제정되었다[1]. 현재 2004 년 2 월에 버전 2.0 이 표준화된 상태이다[2]. WIPI 2.0 에서의 큰 특징은 DLL(Dynamic Linking Library)을 통한 API(Application Programming Interface)의 동적 추가 및 갱신을 지원하는 것이다. 이렇게 함으로써 WIPI 플랫폼은 API 뿐만 아니라 다양한 플랫폼 구성 요소를 추가 또는 삭제하는 응용 레벨의 동적 재구성이 가능하다. 단말기 플랫폼을 재구성하는데 있어 사용자의 불편을 최소화하며, 응용프로그램 개발자는 플랫폼을 위한 다양한 컨텐츠를 개발할 수 있다. 이를 위해 WIPI 플랫폼은 자신이 제공하는 API set 을 관리하는 방법과 API 정보를 응용프로그램에게 제공하는 방법을 정의하는 API 관리자를 가지고 있어야 한다.

본 논문에서 제안하는 API 관리자는 WIPI 플랫폼의 모듈로서 동작하며, 동적 API 추가/갱신 및 삭제를 수행하고 플랫폼에서 제공하는 API 대한 정보를 관리한다. 또한, API 의 버전관리를 통해 API 의 삭제 또는 오류 발생 시 이전 버전의 API 로 복구를 제공한다.

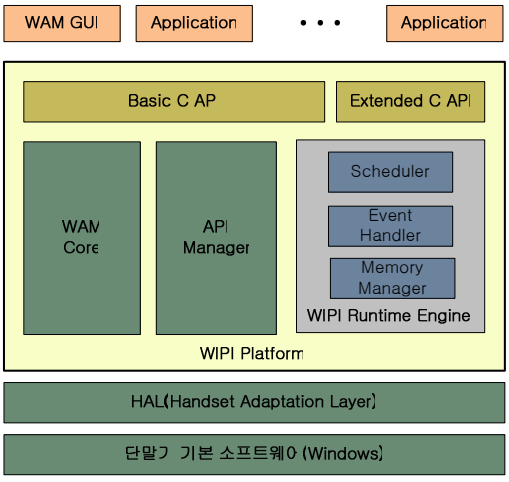
본 논문은 다음과 같이 구성되어 있다. 2 장에서는 본 연구팀에서 현재 개발 중인 윈도우용 WIPI Clet 에뮬레이터 2.0 의 내부 기능구조 및 특성을 기술하며, 3 장에서는 본 연구를 통해 진행 중인 API 의 동적 추가/갱신 위한 API 관리자의 설계 및 구현 사항을 기술하였다. 4 장에서는 결론 및 향후 연구방향에 대해 제시한다.

2. WIPI

2.1 WIPI Clet Emulator

* 본 논문은 한국과학재단이 지정한 지역협력연구센터(RRC)인 충남대학교 소프트웨어연구센터의 지원으로 수행되었음

WIPI 응용프로그램을 개발하고 검증하는데 있어서 좀더 쉽고 효율적인 방법은 일반 컴퓨터 환경에서 응용프로그램을 실행해 보는 것이다. 이를 위해서는 실제 단말기에서 WIPI 응용프로그램이 실행되는 것과 동일한 효과를 낼 수 있는 에뮬레이터 프로그램이 필요하다. 본 연구팀은 WIPI 규격 1.2 의 C API 를 지원하는 윈도우 실행환경 기반의 WIPI Clet 에뮬레이터를 개발하였고[4] WIPI 2.0 Clet 에뮬레이터를 개발 중이다[3]. WIPI Clet 에뮬레이터는 [그림 1]과 같이 단말기의 하드웨어 역할을 하는 GUI(Graphic User Interface)와 윈도우 OS(Operating System)용 HAL(Handset Adaptation Layer), 그리고 WIPI 플랫폼으로 구성되어 있다. WIPI 플랫폼은 메모리 관리자, 스케줄러, 이벤트 핸들러, 링커/로더 등 WIPI 응용프로그램을 실행하는데 꼭 필요한 핵심 부분인 C Engine 과 C API 관리자, WIPI 응용프로그램 관리자 및 응용프로그램을 위한 C API 로 구성된다.



[그림 1] WIPI Emulator 구조도

에뮬레이터를 실행시키면 단말기 운영체제가 먼저 사용자 인터페이스와 플랫폼을 구동시키고 플랫폼이 HAL 과 각 API 를 초기화 한다. WIPI 응용프로그램은 윈도우 DLL 형태로 제공된다.

2.2 API 모듈

WIPI 플랫폼에서 제공하는 API 모듈은 크게 Basic API 와 Extended API 로 나뉜다. Basic API 는 다양한 응용프로그램 개발을 촉진하기 위해 WIPI 에서 정의한 기본 API 이며 플랫폼에 기본적으로 탑재되어 있다. Extended API 는 응용프로그램에게 차별화된 기능을 제공하기 위해 응용프로그램 개발자가 작성한 API 이다. Basic/Extend API 모두 무선망을 통해 다운로드 및 동적 추가/갱신/삭제가 가능하다.

2.3 동적 공유 라이브러리 지원

기존의 단말기에서는 서비스센터를 방문하지 않고서 플랫폼을 수정할 수 있는 방법이 없었고 서비스 업그레이드에 신속하게 대처할 수 없었다. 따라서 WIPI 2.0 에서는 동적 API 추가 및 관리기능을 필수 규격화 하였고 플랫폼은 API 를 무선망을 통해서 추가 및 갱신할 수 있다. 또한, 추가 및 갱신된 API 는 지속적으로 유지되어야 한다. 동적인 API 추가/갱신을 지원함으로써 차별화된 API 확장과 멀티미디어 코덱이나 보안 모듈을 동적 라이브러리 형태로 개발할 수 있으며, 플랫폼이 설치된 후의 버그 패치 혹은 기능추가가 가능하다. 이를 위해 API 는 DLL 형태로 제공된다. Basic API 와 Extended API 는 하나의 DLL 로 작성되거나 API 의 기능상 여러 그룹으로 나뉘어 각각의 DLL 로 작성될 수도 있다.

3. WIPI API 관리자 설계

3.1 WIPI API 관리자

API 관리자는 WIPI 플랫폼에 포함되는 하나의 모듈로서 WIPI 플랫폼이 제공하는 Basic API set, Extended API set 에 대한 동적 추가/갱신 및 그에 따른 버전 관리, 설치/삭제 기능을 수행한다. 또한 사용자의 요구에 따라 이전 버전의 API 를 유지함으로써 역방향 호환성(backward compatibility)을 지원한다. 이를 위해 API 관리자는 DLL 메타(meta) 정보와 동적 API 추가/갱신을 위해 WIPI 규격서에 명시된 API 규격을 따르는 DLL 인터페이스 정보를 유지한다.

3.2 DLL 메타 정보

DLL 메타 정보는 플랫폼에 설치될 DLL 의 특성을 포함하는 데이터 파일로서 API 관리자에게 DLL 의 정보를 제공한다. 본 논문에서는 API 관리자가 동적으로 API 를 추가/갱신하기 위해 필요한 DDF(DLL Description File)를 [그림 2]와 같이 정의하였고, DDF 는 DLL 개발자에 의해 작성되고 공인된 인증서버를 통해 API 모듈을 포함하는 DLL 과 함께 다운로드 된다.

```
Authentication_Key = a1bb434
Name = 3d_library
FileName = 3D dll
Versior = 2 1
Vendor = WIPI mobile
Access_Leve = CP
Description = This contains API set related to 3D
```

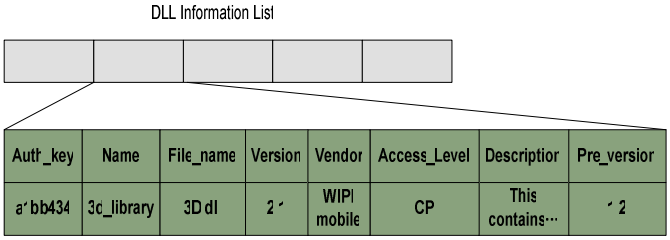
[그림 2] DLL Description File

DDF 에서 정의하고 있는 정보는 다음과 같다. Authentication_Key 필드는 인증서버에서 DLL 에 부여한 ID 이고, Name 은 어플리케이션이 참조하는 DLL 의 symbol 이름, Filename 은 MC_knlLoad()의 파라미터로 사용된다. Version 은 DLL 의 버전정보, Vendor 는 DLL 개발자를 나타낸다. Access Level 은 WIPI 플랫폼에서 정의한 세가지 보안 수준을 나타내며 API 별 보안 수준에 따라 일반(Public)수준, 컨텐츠 개발자(CP)수준,

시스템(System)수준을 갖게 된다. Description 필드는 사용자에게 DLL 에 대한 설명을 포함한다.

3.3 DLL Information List

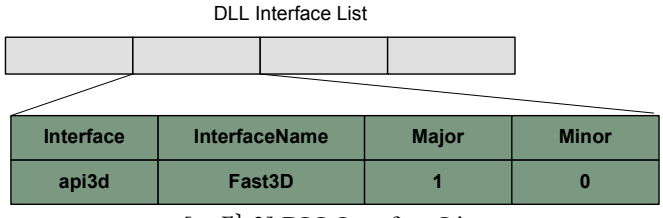
API 관리자는 DDF 을 통해 얻은 DLL 메타 정보를 관리하기 위해 [그림 3]과 같이 DLL Information List 를 생성하고 현재 플랫폼에서 제공하는 모든 API 모듈에 대한 정보를 유지한다. DLL Information List 는 DDF 로부터의 정보 이외에 추가적으로 Pre_version 필드를 갖게 되는데, 이는 해당 DLL 의 이전 버전에 대한 정보와 이전 버전의 존재 유무를 나타낸다. 이 필드를 통해 API 관리자는 사용자의 요구에 따라 구 버전 API 모듈로의 복구를 가능하게 한다.



[그림 3] DLL Information List

3.4 DLL Interface List

WIPI 규격서는 무선망을 통한 API 추가/갱신이 가능하도록 API 규격을 정의하였고, DLL 개발자는 규격에 맞게 DLL 을 구현해야 한다. DLL 은 구현과 인터페이스로 분리된다. 인터페이스는 응용프로그램이 API 를 참조하기 위한 수단이고, DLL 개발자는 응용프로그램 개발자에게 정의한 인터페이스에 대한 header file 을 제공해야 한다. API 관리자는 DLL 로딩 시 DLL 초기화 루틴을 통해 export 할 인터페이스 리스트를 DLL Interface List 에 등록한다.



[그림 3] DLL Interface List

Interface: API 응용프로그램 개발자 정의 인터페이스
InterfaceName: 인터페이스에 부여된 이름
Major: 인터페이스의 major 버전 넘버
Minor: 인터페이스의 minor 버전 넘버

3.4 API 관리

API 모듈은 WIPI 파일 시스템의 sys/dll/ 또는 sys/dll_backup/ 에 저장되고, DLL Information List 는 효율적인 추가/갱신/삭제를 위해 HAL 파일 시스템을 이용한 간단한 데이터베이스 파일로 구현되었다.

가. 디렉토리 별 관리
sys/dll/ : 현재 서비스되는 API 모듈 저장
sys/dll_backup/ : 삭제된 구 버전의 API 모듈 저장

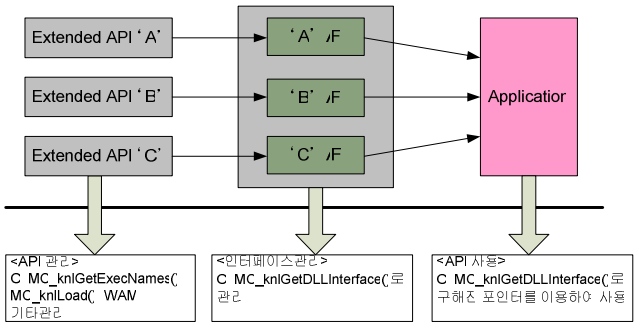
나. API 관리자 기능
① API 추가
다운로드 받은 DLL 을 sys/dll 에 저장한 후 DLL 메타 정보를 DLL Information List 에 등록한다.

② API 갱신
DLL Information List 를 검색하여 등록된 DLL 이 존재한다면 사용자의 요구에 따라 현재 제공되는 DLL 을 sys/dll_backup 으로 이동시키고 또는 삭제 후 다운로드 받은 DLL 을 sys/dll 에 저장하고 새로운 DLL 메타 정보를 DLL Information List 에 등록한다.

③ API 삭제
DLL Information List 로부터 해당 DLL 엔트리를 삭제하고 sys/dll 의 DLL 을 삭제한다. 만약 사용자가 이전 버전으로의 복구를 요구하면 sys/dll_backup 에 저장된 구 버전의 DLL 을 sys/dll 로 이동시키고 DLL Information List 에 구 버전의 DLL 메타 정보를 등록한다.

3.5 API 사용

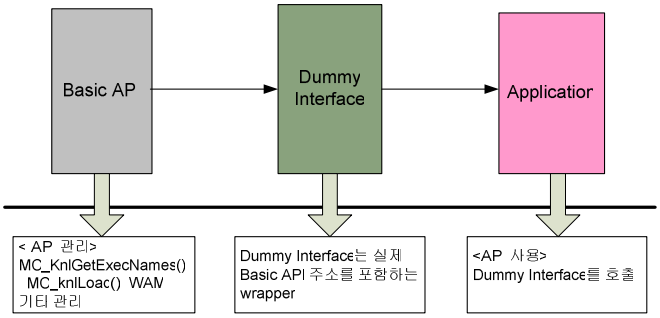
응용프로그램과 별도로 작성된 API 모듈을 링크하기 위해서 응용프로그램은 Basic API set, Extended API set 에 구현된 함수의 주소를 참조할 수 있어야 한다. 이때 Basic API 의 경우 이미 WIPI 플랫폼에 빌트인(built-in) 되었다고 가정한다면 WIPI 규격서에 따라 인터페이스가 정의된 Extended API([그림 4])와 달리 Basic API 의 함수 주소를 외부로 알리기 위한 방법([그림 5])이 필요하다.



[그림 4] Extended API 사용 시나리오

이를 위해 Basic API set 에 대한 dummy 인터페이스를 정의하였다. dummy 인터페이스는 Basic API 가 로딩 될 때 함수의 주소를 포함하게 된다. Dummy 인터페이스에 대한 주소는 응용프로그램 개발자에게 제공

된다. Dummy 인터페이스의 주소를 가지고 개발된 응용프로그램은 함수 포인터로 선언된 Basic API 의 주소 값을 dummy 인터페이스의 주소로 맵핑(mapping)하게 된다. 따라서 응용프로그램은 dummy 인터페이스를 통해 실제 Basic API 를 호출하게 된다. 즉, dummy 인터페이스는 Basic API 의 wrapper 가 된다.



[그림 5] Basic API 사용 시나리오

4. 결론 및 연구방향

본 논문에서는 API 의 동적 추가 및 삭제를 수행하는 WIPI API 관리자의 설계 및 구현에 대해 기술하였다. 제안된 API 관리자는 WIPI 플랫폼이 제공하는 API 에 대한 체계적인 관리 방법과 응용프로그램을 위한 인터페이스를 제공하며 사용자에게 동적 API 추가/갱신에 따른 플랫폼의 재구성을 위한 손쉬운 접근을 제공한다. 향후 다양한 API 모듈 또는 플랫폼 구성 요소가 플랫폼에 추가됨으로써 사용자가 임의의 DLL 을 삭제할 경우 발생할 수 있는 플랫폼 오류를 방지하기 위해 DLL 간의 의존성을 인지하여 적절한 동작을 취할 수 있는 API 관리자에 대한 연구가 필요하다.

참고문헌

[1] 모바일 표준 플랫폼 WIPI 1.2.1, KWISFS.K-05-001R3
[2] 모바일 표준 플랫폼 WIPI 2.0, KWISFS.K-05-002
[3] “윈도우용 Clet 에뮬레이터 개발”, ETRI 연구보고서, 충남대학교, 2004.8
[4] “휴대정보단말기용 웹 서비스 클라이언트 및 WIPI SDK”, ETRI 연구보고서, 충남대학교, 2003.12
[5] 유용덕, 김연수, 최훈, “동적 Upgrade 기능을 구비한 WIPI 개발환경 설계”, 한국통신학회 추계 종합 학술발표회 논문초록집, vol. 28, p.343, 2003.12
[6] “한국무선인터넷표준화포럼” www.kwisforum.org, 2004.9.9