WIPI(*Wireless Internet Platform for Interoperability*) Overview

충남대학교 컴퓨터공학과
분산시스템연구실
유 용덕

Contents

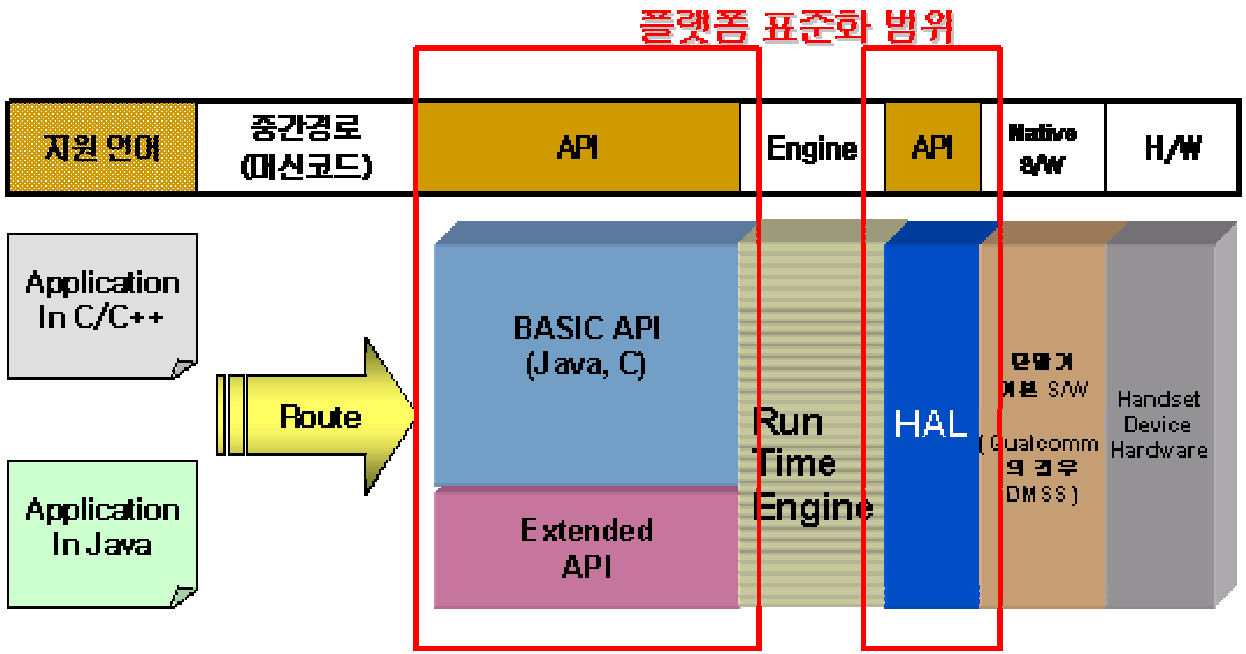
- Introduction
- 주요 기능 규격
- WIPI 플랫폼 구조
  - HAL 규격
  - API 규격
  - 단말기 최소 권장 사항
- 자바 성능 향상 방법
- 바이러리 자바
- 참고 문헌

Introduction(1/2)

- WIPI
  - 이동 통신 단말기에 탑재되어 응용 프로그램을 수행 할 수 있는 환경을 제공하는 모바일 표준 플랫폼 규격
  - 목적
    - 응용 프로그램 개발자 : 플랫폼간 컨텐츠 호환성 제공
    - 단말기 개발자 : 플랫폼 이식의 용이성 제공
    - 일반 이용자 : 다양하고 풍부한 컨텐츠 서비스 제공

Introduction(2/2)

- 플랫폼 표준화 범위
  - 개발자 의견 반영
  - 컨텐츠의 호환을 유지하기 위한 최소한의 API set 정의



주요 기능 규격(1/9)

- 응용 프로그램 머신 코드 규격
  - 플랫폼은 머신 코드 형태의 응용 프로그램을 다운로드 받아 수행
  - 자바 응용프로그램의 경우 자바 중간코드(바이트코드)를 AOTC(Ahead Of Time Compile)를 통해 머신 코드를 생성 후 다운로드 받아 수행
- 다중 응용 프로그램 수행
  - 동시에 여러 개의 응용 프로그램이 메모리에 적재되어 수행될 수 있는 환경 지원
    - 응용 프로그램간 우선 순위
  - 여러 응용 프로그램이 수행될 경우 각 응용 프로그램은 독립적으로 수행
    - 독립적인 응용 프로그램간 통신을 위한 지원 필요
    - 공유 메모리, 이벤트

주요 기능 규격 (2/9)

- 지원 프로그래밍 언어
  - 자바, C언어를 통한 기본 API 구현
  - **CLDC/MIDP** 채택 시, Extended API를 사용하여 응용 프로그램 작성
- API 추가/ 갱신 지원
  - API를 무선망을 통해 추가/갱신
    - 플랫폼 : API 추가/갱신에 따른 버전 관리 및 설치/삭제 기능
    - 추가/갱신된 API에 대해서 동일한 보안 수준 적용

주요 기능 규격 (3/9)

- 플랫폼 보안
  - 플랫폼은 보안수준에 따라 API와 디렉토리에 대한 접근 제한
  - 보안 수준
    - Public 수준
      - ✓ 가장 낮은 보안 수준
      - ✓ 신뢰할 수 없는 일반 개발자가 제공하는 응용 프로그램
      - ✓ 단말기에 영향을 미치거나, 개인 정보 등에 대한 접근 방지
    - CP(Contents Provider) 수준
      - ✓ 인지도가 있는 CP
      - ✓ 어느 정도 신뢰 할 수 있음
      - ✓ 단말기에 심각한 영향을 주지않는 범위 내에서 접근 허용
    - 시스템(System) 수준
      - ✓ 완전 신뢰
      - ✓ 모든 접근 허용

주요 기능 규격 (4/9)

- 보안 항목
  - API 보안
    - ✓ 특정 API 그룹을 보안 대상 그룹으로 구분
    - ✓ API 그룹의 접근 수준과 보안 수준 설정에 대한 정책
      - 플랫폼 이식 시점에 결정
    - ✓ 해당 그룹별 보안 수준 지정
      - NO ACCESS : 허용하지 않음
      - READ ONLY : 읽기만 허용
      - WRITE ONLY : 쓰기만 허용
      - READ/WRITE : 읽기, 쓰기 모두 허용
  - 디렉토리 보안
    - ✓ 개인 디렉토리 : 프로그램 관리자를 제외하고는 해당 응용 프로그램 개발자만 접근
    - ✓ 응용 프로그램 공유 디렉토리 : 서로 합의된 응용 프로그램간 공유
    - ✓ 시스템 공유 디렉토리 : 응용 프로그램에 관계없이 공유

주요 기능 규격 (5/9)

- 메모리 관리
  - 자동 메모리 해제
    - 응용 프로그램이 종료되면, 해당 프로그램과 관련된 모든 메모리 반환
      - ✓ 이벤트 등 각종 동적으로 사용된 메모리는 자동으로 모두 반환
  - 메모리 컴팩션
    - 동적으로 사용하는 메모리를 할당/해제할 때 메모리의 단편화 감소
  - 자바 가비지 컬렉션
    - 자바 언어 문맥에 따른 가비지 컬렉션 수행
  - 자바 스택
    - 자바 응용 프로그램별 스택 할당 및 해제
    - 메모리 한계를 넘을 경우, 예외 상황 전달
    - 예외 상황 발생 후 플랫폼은 정상 동작
  - 공유 메모리 지원
    - 응용 프로그램간 사용되는 메모리 영역은 서로 독립적
    - 응용 프로그램간 공유할 수 있는 메모리 지원

주요 기능 규격 (6/9)

- 응용 프로그램 관리
  - 관리 기능
    - 응용 프로그램 수행 시,
      - ✓ 날짜, 회수 제한 설정에 따른 기동 여부 판단
      - ✓ 설치, 삭제 기능 제공
      - ✓ 응용 프로그램 정지 기능 제공
        - 실행 프로그램만 삭제, 데이터 파일 보존
      - ✓ API 추가/ 갱신 기능 지원
      - ✓ 응용 프로그램 강제 종료 기능 제공
  - 응용 프로그램 다운로드 기능
    - 응용 프로그램 다운로드 받는 기능 지원
    - 다운로드 중 오류 발생 시, 초기 상태로 복구
    - 무선망 뿐만 아니라 시리얼 인터페이스를 통한 다운로드 기능 제공

주요 기능 규격 (7/9)

- 다국어 지원
  - 유니코드 지원
    - 지역 특성에 맞게 해당되는 문자 코드로 변환
  - 로케일 지원
    - 지역정보(locale)에 따라 참조하여 지원하는 문자셋으로 인식
      - ✓ 한국 : EUC_KR
  - 확장 유니코드
    - EUC_KR 문자셋에서 유니코드에 대응되지 않는 그래픽 문자
    - Private Use 영역을 사용하는 확장된 유니코드

주요 기능 규격 (8/9)

- CLDC 및 MIDP 지원
  - 1.2 규격 : 지원 할 수 있음(2003.12.31 이후 : 지원 필수)
  - **CLDC**(Connected, Limited Device Configuration)
    - CLDC 규격은 Sun Microsystems사의 CLDC 규격 1.1을 기준
    - 플랫폼은 바이너리 코드의 실행을 기반
    - CLDC 규격에서 정의하는 버추얼머신의 기능은 플랫폼 엔진에서 수용
      - ✓ 바이트코드와 코드 verification 처리는 플랫폼의 의미를 AOTC를 포함한 것으로 해석
  - **MIDP**(Mobile Information Device Profile)
    - MIDP 규격은 Sun Microsystems사의 규격 2.0을 기준

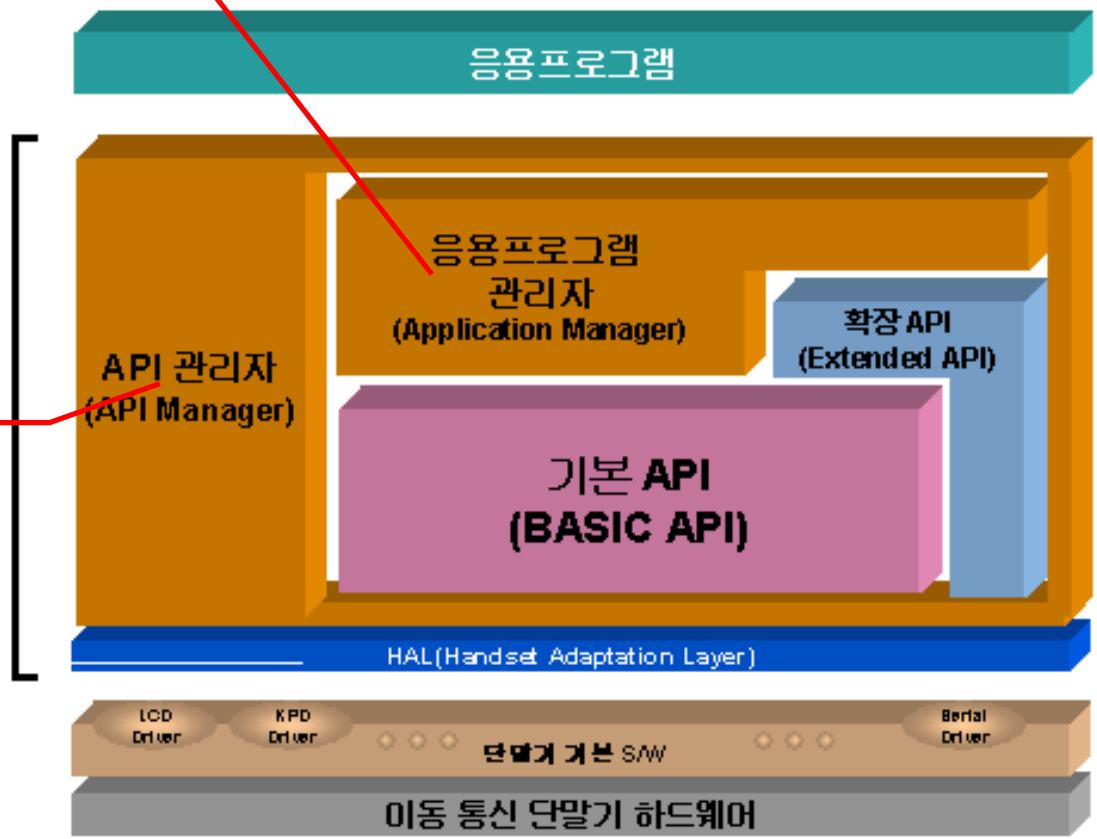
주요 기능 규격 (9/9)

- Extended API
  - CLDC/MIDP의 API를 Extended API로 정의
  - 필수 규격화 될 경우, Basic API로 편입
- Basic API와의 상호운용성
  - CLDC와 MIDP가 추후 필수 규격화 되는 시점에서 만들어질 규격에서 구체적인 방안 제시

WIPI 플랫폼 구조

정보 보기,
다운로드,
설치, 실행 등
전반적인
관리기능

*기본 API 및
확장 API 갱신/추가
*응용프로그램
관리자 다운로드
*DLL 방식 사용
*API 관리자의
업그레이드
가능



HAL(Hardware Adaptation Layer) 규격

- 플랫폼 이식에 있어 하드웨어 독립성을 지원하기 위한 계층

플랫폼이 제공할 API	플랫폼이 구현하여 제공하여야 되는 API 주로 HAL 하단에서 플랫폼으로 이벤트를 전달하거나, 플랫폼을 시작하기 위해서 필요 HAL 이식시에 구현해야 할 API가 아니며 플랫폼이 구현함을 가정하는 함수
시스템	단말기 정보 또는 이벤트 입수 관련 API 디버깅 정보 출력을 지원하는 함수와 크리티컬 섹션(Critical Section) 을 보호하기 위한 API 플랫폼이 관리할 메모리영역을 주는 규격 등을 지원하여 플랫폼의 핵심 기능을 구현 할 수 있도록 지원하는 API
Call	전화를 걸거나 받는 API로 구성 ->플랫폼 수행 중에 전화가 걸려 올 때 수신 여부를 결정할 수 있고, 플랫폼에서 전화를 걸 수 있도록 지원
Handset Device	단말기에서 지원하는 LED,백라이트, 진동 장치 등에 대한 제어를 지원하는 API
네트워크	PPP 연결 관련 기능 지원 API UDP, TCP를 사용하기 위한 API

HAL(Hardware Adaptation Layer) 규격

시리얼	시리얼 연결 및 제어 관련 기능 지원 API
SMS	SMS 메시지를 가져오고, 제어하는 기능을 지원하는 API
사운드	지원하는 Sound 포맷을 이용하여, 멜로디 및 벨소리 ,경고음, 키음 등을 지원하는 API
시간	현재 시간을 가져오는 기능과 타이머를 지원하는 API
Utility	유니코드와 지역코드로의 변환 관련 기능을 지원하는 API
파일 시스템	계층적 디렉터리 구조의 파일 시스템을 추상화하여 지원하는 API

HAL(Hardware Adaptation Layer) 규격

보코더	플랫폼에서 지원하는 Vocoder 장치의 녹음에 관련된 기능을 지원하는 API
입력기	다국어 입력기를 지원 하는 API
폰트	다양한 폰트를 화면에 출력 하거나, 화면에 출력 할 때 관련 정보들을 얻어오는 API
가상키	LCD화면에 프레임 버퍼의 내용을 출력하거나, 화면의 정보를 얻어오는 API 응용프로그램에서 단말기에 존재 하지 않는 키를 가상적인 기능키로의 사용을 지원하는 API

C API 규격

커널	동적 메모리 할당/해제 관련 API 다중 응용프로그램과 동적 라이브러리 지원을 위해 로딩과 수행 및 다중 응용프로그램간 공유 메모리를 지원하는 API 복수 타이머를 지원하며, 시스템 정보를 갱신하거나 가져오는 API
그래픽	화면이나 오프 스크린 프레임 버퍼(Off Screen Frame Buffer)에 다양한 그리기를 할 수 있는 API 다양한 이미지 포맷(BMP, PNG, GIF, AGIF)의 인코딩/디코딩을 지원하는 API 그래픽 이벤트 처리 및 문자 입력 처리 관련 API
데이터베이스	데이터를 레코드 단위로 저장하고, 검색하며 관리하기 위한 API
파일시스템	계층적 디렉토리 구조의 파일 시스템에서 파일과 디렉토리를 사용하기 위한 API
네트워크	PPP 연결, TCP/UDP 소켓 연결과 관련된 API HTTP 연결에 지원하기 위한 API

C API 규격

매체처리기	사운드나 동영상 등의 Media 처리에 관련된 API와 톤 재생 및 음성녹음 및 볼륨 조절에 관련한 API
시리얼	시리얼 포트 관련 제어 및 사용에 관련된 API
Phone	CALL과 SMS 송수신 관련 API
Misc	LED, 백라이트 등의 제어와 관련된 API
UI 컴포넌트	사용자 인터페이스 컴포넌트로 텍스트 박스, 날짜/시간 컴포넌트, 메뉴 컴포넌트, 라벨 컴포넌트, 리스트 컴포넌트와 관련된 API

Java API 규격

Core System (java.lang)	J2SE의 java.lang 패키지와 동일하거나 부분적으로 지원
High Level IO (java.io)	J2SE의 java.io패키지와 동일하거나 부분적으로 지원
Utilities (java.util)	J2SE의 java.util 패키지와 동일하거나 부분적으로 지원
Low Level IO (org.kwis.msf.io)	J2SE의 java.net 패키지와 유사하거나 부분적으로 지원
System (org.kwis.msf.core)	다중 응용프로그램과 동적 라이브러리 지원을 위해 로딩과 수행 및 다중 응용프로그램간 공유 메모리 API를 제공
Additional IO (org.kwis.msp.io)	J2SE의 file 관련 API 동일하거나 부분적으로 지원 PPP 연결 관련 API를 지원

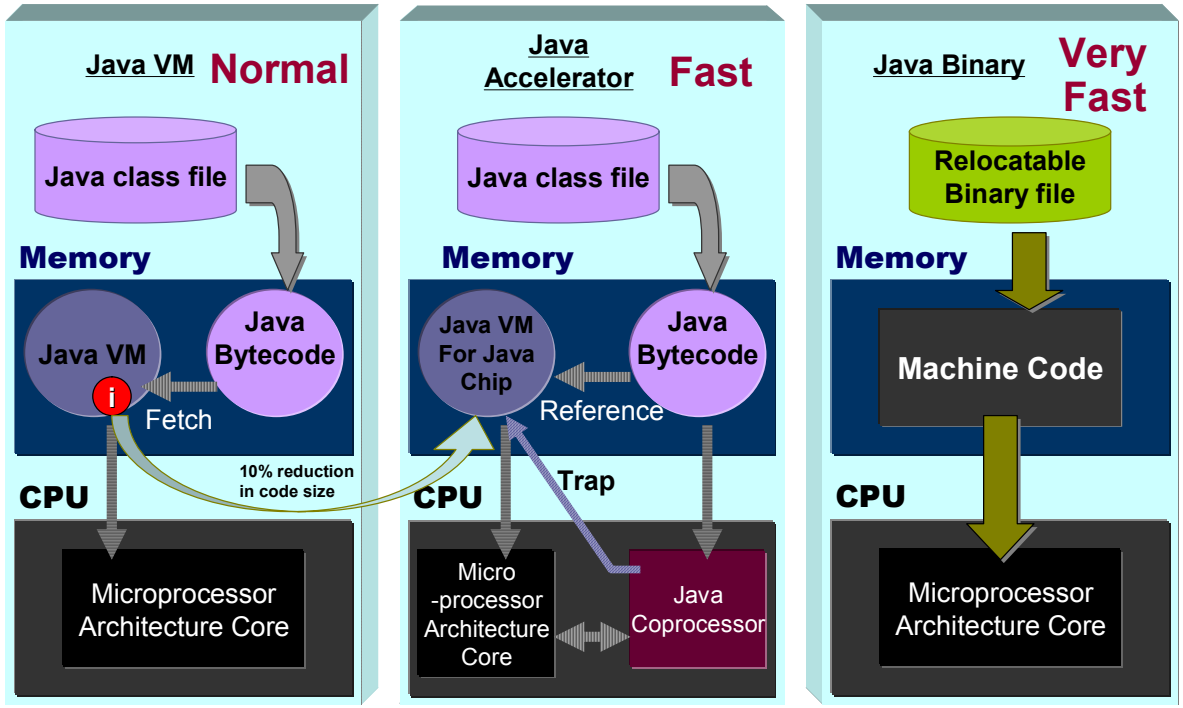
Java API 규격

Graphics (org.kwis.msp.lcdui)	화면이나 오프 스크린 프레임 버퍼(Off Screen Frame Buffer)에 다양한 그리기를 할 수 있는 API들로 구성 다양한 이미지 포맷(BMP, PNG, GIF, AGIF)의 인코딩/디코딩을 지원하는 API로 구성 그래픽 이벤트 처리 및 문자 입력 처리 관련 API로 구성
Database (org.kwis.msp.db)	데이터를 레코드 단위로 저장하고, 검색하며 관리하기 위한 API로 구성
UI Component (org.kwis.msp.lwc)	Graphics 관련 패키지 상에서 다양한 고수준 그래픽 인터페이스를 위한 API로 구성
Handset (org.kwis.msp.handset)	단말기 특수 정보, 백라이트, Call 에 관련된 API로 구성
Media (org.kwis.msp.media)	사운드나 동영상 등의 Media 처리에 관련된 함수와 톤 재생 및 음성녹음 및 볼륨 조절에 관련한 API로 구성

단말기 최소 권장 사양

종류	설명
디스플레이	* 스크린 크기: 96*54 * 색상 : 회색조 4색 이상 또는 천연색 256색 이상
입출력장치	* 입력 장치: 키패드 * 사운드 장치 : 진동 및 비프음 * 네트워크 : 무선 및 시리얼 전송
비휘발성 메 모 리	* 플랫폼 라이브러리가 사용할수 있는 비휘발성 메모리 600KB 이상 * 응용 프로그램 관리자 및 기본 응용 프로그램에서 사용할 수 있는 비활성 메모리 400KB 이상 * 응용 프로그램이 사용 가능한 파일 시스템 공간으로 500KB 이상
휘 발 성 메 모 리	* 응용 프로그램에서 사용 가능한 힙 영역으로 300KB 이상 * 플랫폼 라이브러리에서 사용 가능한 영역으로 20KB 이상

자바 성능 향상 방법(1/3)



인터프리터 하드웨어 가속기 AOTC

자바 성능 향상 방법(2/3)

- 인터프리터 방식
  - 자바 바이트코드를 소프트웨어 인터프리터로 수행하는 방식
  - 초기의 JVM(Java Virtual Machine)들에서 사용되었으며, 속도가 현저히 느림
    - 보완 방법 : 인터프리터 부분을 CPU 칩에 최적화
- 하드웨어 가속기 사용 방식
  - 소프트웨어 인터프리터를 하드웨어 가속기로 대체하는 방식
  - 자주 사용되는 일부 인스트럭션 -> 하드웨어 가속기
  - 복잡한 인스트럭션 -> trap이 발생되어 소프트웨어가 수행
  - 상당한 속도 개선이 있으나, trap으로 인한 수행상의 손실이 발생하고 ,부가적인 하드웨어로 인한 비용과 전력 소모 고려

자바 성능 향상 방법(3/3)

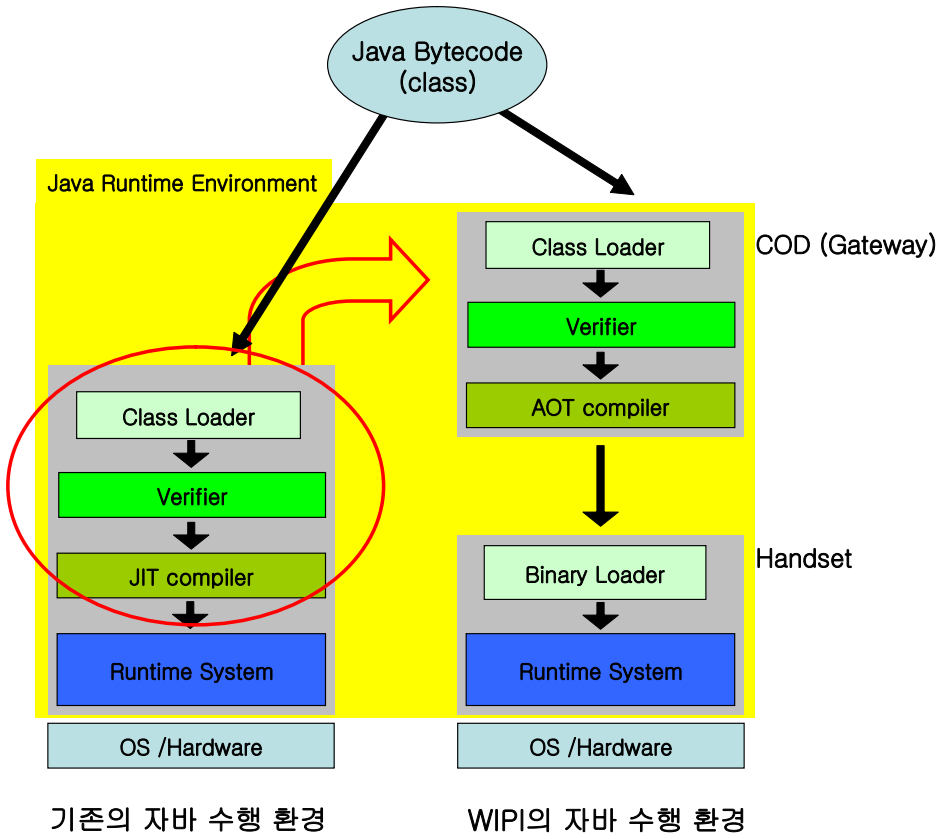
- Just In Time Compile(JITC) 방식
  - 수행 중에 바이트코드를 컴파일해서 CPU에 맞는 바이너리 코드를 생성하는 방법
  - 컴파일 속도가 중요하기 때문에 최소한의 최적화만 수행
  - 메모리 요구량이 커서 메모리 제약이 많은 단말기에는 부적절
- Ahead Of Time Compile(AOTC) 방식
  - 바이트코드로 되어 있는 자바 어플리케이션을 수행되기 전에 미리 컴파일 해서 단말기의 CPU에 최적화된 바이너리 코드를 생성하는 방법
  - JITC에 비해 충분한 최적화 시간
  - WIPI 규격 : AOTC 방식을 선택하여 바이너리 형태의 자바 어플리케이션을 수행

바이너리 자바(1/8)

- WIPI 규격에 따른 플랫폼에는 바이트코드 검증 부분 없음
  - 자바 바이트코드 검증(Verification)
    - 정상적으로 수행 될 수 있는 어플리케이션인지 수행 전에 확인 할 수 있음
- COD(Compile On Demand) 서버에서 구현
  - 기존의 JIT 컴파일러는 AOT 컴파일러의 형태로 COD 서버에 존재
  - COD :프로그램 개발자 또는 시스템 관리자의 요청에 의한 컴파일
    - 자바로 작성되는 어플리케이션에만 적용
    - 단말기가 여러 종류의 CPU 칩으로 서비스 될 경우에 각각의 CPU에 맞도록 서버단에서 컴파일

    -> C언어로 작성된 어플리게이션의 경우, 컴파일러가 개발자에게 제공

바이너리 자바 (2/8)



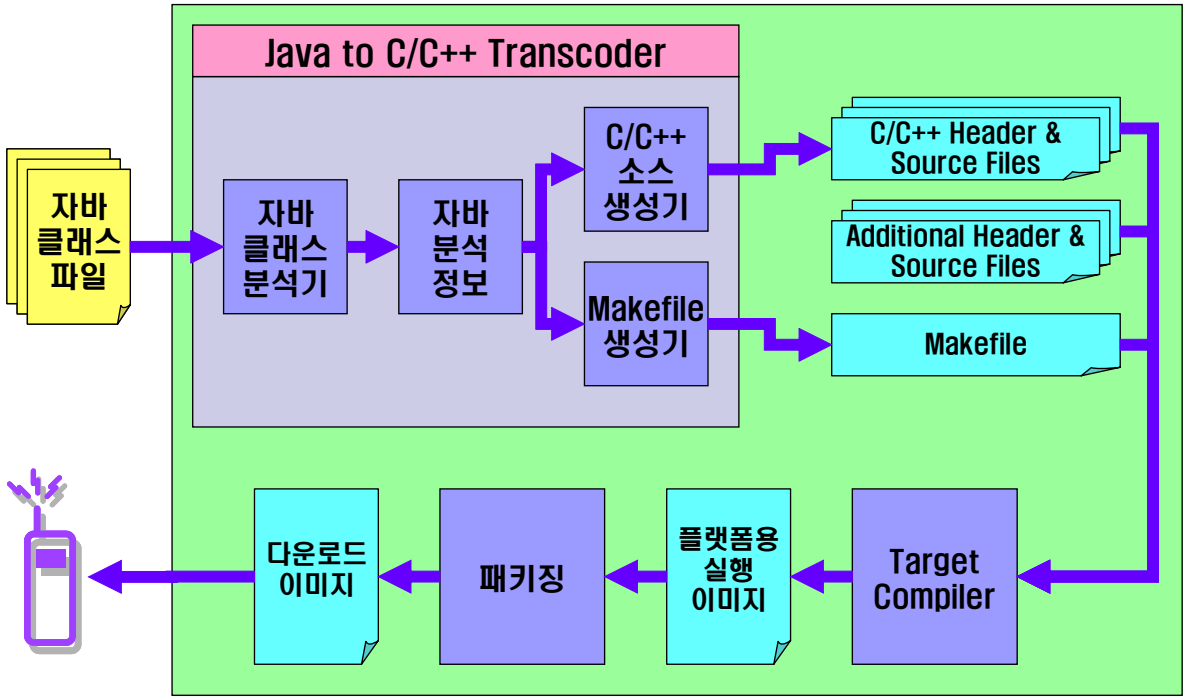
*COD를 거쳐 플랫폼으로 전달될 때 바이러리 이미지로 변경

바이너리 자바 (3/8)

- COD/AOTC는 왜 필요한가?
  - 기존의 무선인터넷 플랫폼
    - VM 기술과 Native 바이너리 기술
    - VM 기술
      - VM을 사용하여 프로그램 동작
      - 작성된 프로그램을 VM에서 동작할 수 있는 중간 코드로 변환
      - VM이 해당 코드를 번역하여 실제 CPU용 기계어로 번역, 실행
      - 중간 코드를 실행하는 과정에서 번역(interpreting) 과정 필요
      - CPU와 자원이 제한된 이동단말에서는 상대적으로 매우 느린속도
    - Native 바이너리 기술
      - CPU의 성능을 100% 활용
      - VM에 비해 매우 빠른 속도 제공
      - 프로그램 개발자에 의해 악의적인 메모리 접근을 통한 시스템 안정성을 위해 할 수 있는 요소가 있음
  - Java 언어의 장점 + Native 바이러리의 실행 성능
    - C/C++ 와 Java를 기본언어로 채택
    - 단말기에 탑재되는 어플리케이션은 Native 바이러리

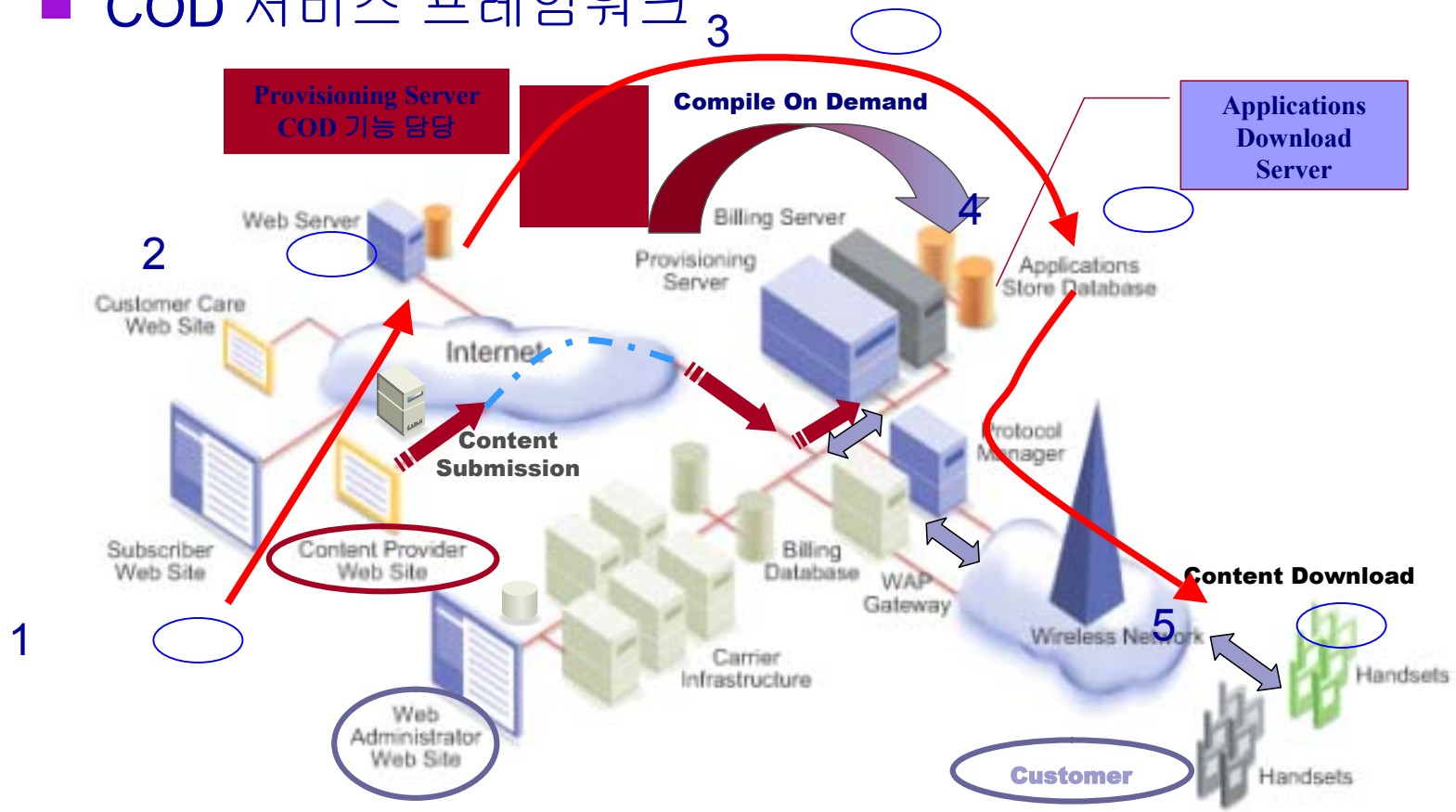
바이러리 자바 (4/8)

- COD 서비스 구조



바이러리 자바 (5/8)

- COD 서비스 프레임워크



바이러리 자바 (6/8)

- COD 서비스 프레임워크
  - 개발자의 프로그램 제공(단계1 + 단계2)
    - 개발자는 WIPI SDK 등에서 제공되는 기능을 이용하여 프로그램 개발→(1)
    - Java 컴파일러를 이용하여 클래스 파일 생성→(1)
    - Jar(Java Archive)형태로 프로그램 정보와 함께 Contents Provider Web Site에 제출→(1)
    - 검증과정을 거쳐 Provisioning Server 에 등록 → (2)
  - Command On Demand
    - 개발자에 의해 제출된 프로그램은 시스템관리자 또는 개발자 자신에 의해 컴파일 요청
    - 해당 프로그램은 단말기에서 작동 가능한 바이너리로 생성
    - 생성된 바이너리는 다운로드 형태로 패키징
    - Application Store Database에 등록

바이러리 자바 (7/8)

- Application Download
  - 사용자는 단말기를 이용하여 서버에 접속하여, 다운로드가 가능한 프로그램을 검색, 다운로드
  - 과정
    - ✓ 접속한 단말기의 종류 확인
    - ✓ 사용자가 정상적인 사용자인지 검사
      - 연령제한을 받는 프로그램이라면 이에대한 확인작업 선행

바이러리 자바 (8/8)

- COD의 장점
  - Java 언어를 무선인터넷 플랫폼을 위한 개발언어로 사용하여 얻는 이익과 동일
    - Java 언어 수준의 신뢰할 수 있는 안정성 제공
      - ✓ 개발자는 개발과정에서 발생할 수 있는 모든 종류의 오류로부터 자유로워 질 수 있음
      - ✓ Java의 예외처리 기능을 통하여 안정적인 프로그램 작성
      - ✓ 개발자의 실수에 의해 시스템에 오류를 발생시키는 문제를 근본적으로 회피
      - ✓ 쉬운 사용자 인터페이스 제공
  - AOTC 기술 사용을 통한 추가적인 장점
    - 단말기에서의 빠른 수행속도 제공

참고 문헌

- 모바일 표준 플랫폼 규격서(KWISF.K-05-001R3: 1.2.1)
- http://www.mobilejava.co.kr
- http://www.aromasoft.co.kr
- http://www.kwisforum.org (한국무선인터넷표준화포럼)