

모바일 게임개발을 위한 자바 프로그램 최적화 설계

A Study on the Game System Development Design of Mobile JAVA

최 성*

남서울대학교 컴퓨터학과

Sung Choi*

Dept. of Computer Science, Namseoul University

요 약

무선용 자바용 게임의 개발에 대한 게임 개발 업체들의 관심이 높아지면서, 표준화 된 게임 어플리케이션의 상용화에 대한 필요성이 대두되고 있다. 휴대용 게임 같은 실시간 어플리케이션을 작성하는 것은 디바이스의 제한된 속도와 메모리 등의 한계를 극복해야만 가능하다. 먼저 무선 인터넷에 대하여 연구하고, 자바를 게임개발에 어떻게 응용을 하여, 효율적인 게임개발을 함으로서 우수한 게임 제품을 개발 할 것인가에 대한 연구하였다.

1. 서론

무선 게임 분야는 최근에 활성화된 산업분야로서 폭넓고 미래에 영향력 있는 분야로 인정받고 있다. 모바일 게임 분야에 있어서 기존의 단순한 Text방식의 WAP서비스만으로 지속적인 사업 성장을 할 수 없으며 다양한 잇점을 제공하는 새로운 게임 소프트웨어 개발방법을 필요로 하고 있다. Java는 정적인 웹화면에 동적이고 자유로운 응용 게임이 가능하도록 만들었고(applet) 결과적으로 웹 게임서비스에 있어서 동적인 서비스에 시초가 되었던 것과 같이, 모바일 분야에서도 java의 역할이 유사한 개념을 제공하는 시초가 되고 있다. 본 연구는 소형기에 있는 제한 응용 프로그램을 작성하는 방법을 바꿔야만 하는 이유를 제시하고, Java를 사용하여 무선 단말기를 위한 프로그래밍에 대하여 연구하였다.

2. 무선인터넷

2.1 무선인터넷이란?

무선 인터넷은 단말기를 통해서 언제 어디서나 인터넷에 접속하여 다양한 정보검색과 전자상거래까지 하는 서비스와 환경이며, 무선 인터넷 서비스를 이용하면 사이버 쇼핑, 금융거래, 생활정보, 게임 등 다양한 응용 서비스를 이용한다. 현재 이동통신 단말기를 매체로 이용하여 Notebook, 핸드폰, PDA등을 통해 데이터에 접근하는 무선데이터 서비스와의 구별이 필요하다.

2.2 무선인터넷의 특징

첫째, 시간에 관계없고 어느 장소에서나 어떤 사람

이라도 이용할 수 있다는 것이고 둘째, 데이터 전송속도의 제한이 있으며 셋째, 입력도구의 제한으로 인한 텍스트 입력이 어렵다. 넷째 사용 중에 서비스 요금이 부과되고, 제한된 화면을 가지고 있다는 특징이 있다.

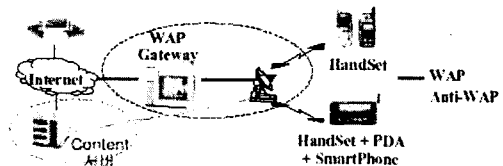
2.3 무선인터넷 프로토콜

2.3.1 WAP(Wireless Application Protocol)

WAP은 셀룰러폰이나 무선호출기 등과 같은 무선장치들이 전자우편, 웹, 뉴스그룹 및 IRC 등의 인터넷 액세스에 사용될 수 있는 방법을 표준화하기 위한 통신 프로토콜들의 규격이다. 무선 환경에 최적화 프로토콜이며 WAP 포럼-Open Standard로 사용되고 있다. 유럽을 중심으로 전세계 통신업체에서는 무선용 언어(WML, WMLScript 채택)로 사용되고 있다.

2.3.2 Anti-WAP

TCP/IP, HTTP를 사용함으로써 인터넷과의 완전한 통합을 추진하였고 HTML보다 좀더 확장된Subset (m-HTML, c-HTML, s-HTML)을 사용하며 콘텐츠 서버를 통해서 웹과 연결된다. MicroSoft, NTT Docomo, 삼성전자에서는 표준으로 채택하고 있다.



(그림1 : WAP-AntiWap의 프로토콜 관계도)

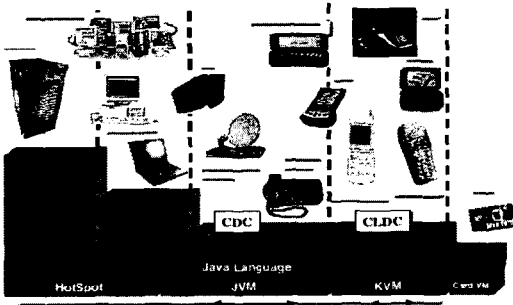
3. 무선 인터넷에 사용 되는 자바

3.1 자바플랫폼

자바 플랫폼은 두 가지 요소로 구성된다. 자바 언어로 작성되고 컴파일된 자바 실행 코드를 수행해 주는 자바 가상머신과 자바 프로그램을 작성하기 위해 기본적으로 사용할 수 있는 클래스 라이브러리(표준 API 집합)으로 구성된다. 그리고 이러한 자바 플랫폼은 자바2 플랫폼이라는 개념이 가미된 새로운 플랫폼으로 진화하면서 J2EE(Java2 Enterprise Edition), J2SE(Java2 Standard Edition), J2ME(Java2 Micro Edition)라는 세 가지의 영역으로 분할되었다. 이것은 각각 엔터프라이즈 서버, 데스크톱, 임베디드 디바이스를 목표로 한 역할 분담이다.

3.2 KJAVA

1) CLDC : 제한된 리소스를 가지고 네트워크를 이용하는 작업을 주로 담당하는 (connected) Embedded system들에 대한 공통된 API를 제공하기 위하여 JCP에서 정의한 클래스 규격이다.



(그림2 : 자바 플랫폼)

2) MIDP : 무선 네트워크 기능을 가지는 이동정보기기(휴대폰 등)에 대하여 표준적인 API를 제공하도록 하기 위하여 JCP에서 정의한 클래스 규격으로 PC급 이상의 시스템보다 제한적인 사용자 인터페이스를 제공한다.

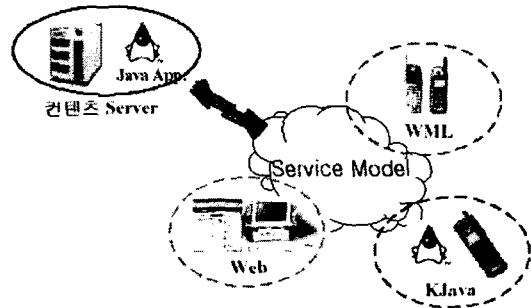
3) KVM 소형기기의 적은 용량에 맞게 설계된 JVM으로 작은 메모리를 갖는 가상머신을 재설계 하였다.

3.3 JAM(Java Application Manager)

CLDC/MIDP 플랫폼의 응용모델을 지원하기 위한 애플리케이션 관리 소프트웨어다. JAM의 역할은 MIDP 애플리케이션, 즉 MIDlet을 다운로드로 설치, 업그레이드, 실행, 삭제한다. 실행과 업그레이드, 삭제 과정은 일반적인 애플리케이션이 된다. 즉, 사용자 인터페이스를 통해 설치한 애플리케이션을 찾아 수행, 업그레이드, 삭제 작업을 하는 프로그램이다.

3.4 무선인터넷에서의 자바의 필요성

- 1) 동적 애플리케이션의 다운로드 : 자바는 실행 코드의 네트워크 이동성을 보장하는 우수한 솔루션이다.
- 2) 크로스 플랫폼 호환성: 플랫폼 간 호환성은 차세대 모바일 솔루션의 기본이다. 사용자들은 동일한 애플리케이션을 원하고 있으며, 자바는 플랫폼 호환성에 관해서는 경쟁력을 갖고 있다.
- 3) 사용자 경험과 역동성 : 동적으로 다운로드된 자바 애플리케이션은 보다 향상된 그래픽과 더욱 빠른 응답 속도를 통해 사용자들의 만족도를 높여 주고 있다.
- 4) 보안 문제 : 자바 자체에 보안 모델을 갖고 있으며, 무선상에서 보안 문제를 해결할 수 있다.
- 5) 비연결성 : 자바 애플리케이션을 사용함으로써, 네트워크에 연결되지 않았을 때 작업한 내용을 네트워크에 연결할 수 있는 환경에서 동기화하는 것이 가능하다.



(그림3 : 자바의 게임 서비스 모델)

3.4 자바 사용 게임 개발

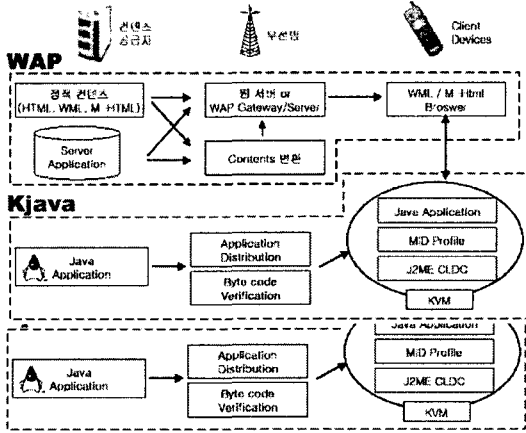
위에서 설명 된 무선인터넷 서비스들을 모두 제공하고 있으며 특히 게임에서는 자기가 원하는 게임을 다운로드 하여 실행, 삭제할 수 있으며 멋진 그래픽 인터페이스를 제공하며 매신저에서는 WAP방식과 달리 실시간으로 서비스 가능하고 파일교환도 가능하다. 또 멀티미디어 콘텐츠 등을 지원하는 브라우저도 개발되어 있다.

4. 최적화 연구

4.1 자바와 성능

프로그램의 성능의 나쁜 이유는 첫째가 성능 조정을 제대로 수행하지 않았을 때이고 두 번째는 비효율적 알고리즘이나 데이터의 구조를 가지고 있을 때 성능이 나빠진다. 이를 비추어 자바 프로그램의 성능이

나쁜 이유로 들수 있는 것은 플랫폼 독립성을 위한 소프트웨어 레이어를 가지고 있는 것이고, 재사용/유연성/안전성을 위한 동적 특성과 힙 할당, 가비지 수집, 참조 의미론, 동적 바인딩, 형 변환과 첨자에 대한 철저한 검사를 하는데 있다. 이런 특징들은 프로그래밍이 쉬울지 몰라도 성능은 많이 떨어지게 하는 원인이 된다.



(그림4 : 자바 활용 무선인터넷 서비스)

4.2 모빌리티와 성능

모바일 자바는 J2SE의 고급 API가 대부분 지원되지 않아 필요한 기능을 직접 구현하여야 할 경우가 자주 발생한다. 모바일 기기는 저성능 CPU, 저용량 기억장소, 불편한 입출력수단을 가지고 있으며 배포본 프로그램 규모도 최소화할 필요가 있다. 이러한 모바일 환경은 통신 품질은 나빠면서 요금 부담은 부담이 된다. 따라서 성능 최적화의 중요성은 강조될 수밖에 없다.

4.2 성능조정 의 일반 원칙

- 가. 사용자가 받아들일 수 있는 수준의 성능
- 나. 사용자 관점에서의 성능을 추구할 것 (UI와 반응성)
- 다. 성능을 좌우하는 부분을 집중 조정할 것
- 라. 임의적 작업이 아니라 과학적 과정 적용
- 마. 주된 병목(bottleneck)을 판정한다
- 바. 손쉽게 해결할 수 있는 병목을 선정, 집중 개선한다
- 사. 이러한 과정을 사용자가 만족할 때까지 반복한다

4.3 응용 게임개발 프로그램 최적화 방안

1) 의심이 가면 Java를 사용하지 않는다.

가장 간단한 방법으로, 이단이 되는 것이다. 프로그램을 작성할 때, 기능적인 측면 아니라, 성능면에서 합당하지 않다면, Java를 피하는 것이 좋다. 이런 간단한 명제는, Java를 애호하는 예찬론자들이 종종 간파하기 쉽다. Desktop이나 Server에서는 별 문제 없는

중복의 문제가 소형기기에서는 해로운 문제가 된다. 소형기기의 제한으로는 CPU의 속도가 비교적 빠르지 않은 것에 기인하는 연산 능력 부족, 메모리 양의 부족이 있다. 예를 들면, Palm에서는 KVM상에서 64K가 넘어가는 프로그램은 동작하지 않았다.

2) Server에서 계산을 하도록 한다.

만약 Java를 사용하기로 결정되었다면, 다음 전략은 간단하다. 계산이 집중된 작업을 소형 기기에서 하도록 하지 않고, Server에서 하도록 한다. 이것은 마치 대부분의 작업이 Web Server에서 일어나고, 브라우저는 그것의 결과를 보여주도록 하는 것과 마찬가지로이다. 그것도 여의치 않다면, 그 집중된 계산동안 User Interface등의 여타 다른 작업을 잠시 멈추고 계산에 CPU를 할당하도록 하여, 짧은 시간에 작업을 끝내도록 하는 것이 좋다. 물론, 그 멈추는 시간은 사용자가 받아들일 만한 시간이어야 하지만 한다.

Server가 할 일과 기기가 할 일의 균형을 맞추는 것은 다소 애매한 작업으로, 그 기기의 Connectivity에 의존적이다. 빠르고 무료인 Cradle 통신과, 느리고 고가의 무선 통신 두 가지가 모두 가능하다면 시간과 비용을 고려한 균형을 잡아야 한다.

3) 작은 프로그램을 만들도록 한다.

다음 단계는 크기가 작은 프로그램을 만들도록 한다. 이것은 기기에 응용 프로그램이 차지하는 메모리를 작게 하고, 인스톨 시에 시간을 절약하도록 해준다. 압축된 JAR를 사용하는 것은 작은 응용 프로그램을 만들어주는 방법이다. 그렇지만 반대로, 이것은 load 할 때 압축을 푸는 시간이 소요되는 단점이 있다. 그러므로, 구동 시간이 문제가 된다면, 압축되지 않은 JAR를 사용하도록 하고, 그렇지 않은 보통의 경우라면 압축된 JAR를 사용하도록 한다.

4) Run-Time시에 적은 메모리를 사용한다.

보통의 소형 기기의 Run-Time시의 가용한 메모리는 상당히 제한되어 있다. 예를 들어, Palm OS의 경우, 실행시에 응용프로그램이 stack이나 heap으로 동적으로 사용할 수 있는 양은 32K에서 256K이다. 그 이상의 메모리를 필요로 할 경우, 영속적인 느린 메모리를 접근하게되어 성능이 떨어진다.

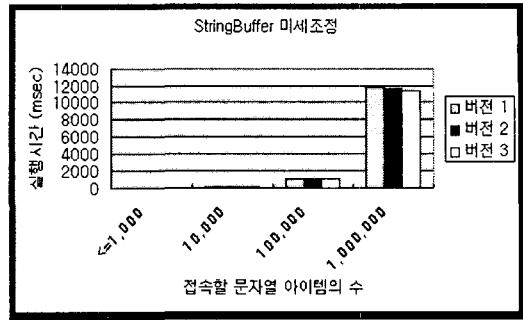
5) 성능을 염두에 둔 Code 작성을 염두에 둔다.

지역 변수를 사용하면 보통 지역 변수에 접근하는 것보다, class member 변수에 접근하는 것이 느리다. member 변수를 계속 재사용 할 필요가 있을 경우에는, 지역 변수를 하나 선언하는 것이 좋을 수 있다. 물론, 그 데이터가 여러 thread 공유하는 경우가 아닐

때의 경우다. 이것은 array의 한 변수를 계속적으로 접근할 때도 마찬가지이다. array의 변수에 접근을 할 때는 bound check를 하기 때문에 역시 지역 변수로 할당 받는 것이 좋다.

6) MIDP 프로그래밍시 고려해야 할 사항

CPU 속도가 느리며, 과도한 그래픽 이미지의 사용을 최대한 자제하여야 하며, 메모리 사용공간이 한정적이다. 코드의 최적화 기술이 중요한 요소와 네트워킹 vs 영속 저장공간의 RMS를 통한 영속 저장공간의 효율적인 활용이 필요하다. UI(유저 인터페이스)의 최적화를 하고, 하위 레벨 API를 활용한 직관적인 인터페이스 제공한다.



(그림 5 : 미세조정 결과 그림)

5. 최적화 알고리즘(스트링버퍼 미세조정)

```
static String makeString(int count)
{
    StringBuffer sbuf = new StringBuffer("");
    for (int i = 0; i < count; i++)
        sbuf.append(i).append(",");
    return sbuf.toString();
}

static String makeString(int count)
{
    //Pre-sizing StringBuffer
    StringBuffer sbuf = new StringBuffer(count*2);
    for (int i = 0; i < count; i++)
        sbuf.append(i).append(",");
}
```

(알고리즘1. 스트링버퍼의 조정1, 2)

1)최적화프로그래밍

```
static String makeString(int count)
{
    StringBuffer sbuf = new StringBuffer(count*2);
    int limit = count, i = 0;
    count /= 10;
    for (; i < count; i += 10) //Loop unrolling
        of grain 10
        sbuf.append(i).append(",").append(i+1).append(",")
            .append(i+2).append(",").append(i+3).append(",")
            .append(i+4).append(",").append(i+5).append(",")
            .append(i+6).append(",").append(i+7).append(",")
            .append(i+8).append(",").append(i+9).append(",");
    while (i < limit)
}
```

(알고리즘2 스트링버퍼의 미세조정 3)

6. 결론

자바는 생산성, 이식성, 안전성을 제공하는 매우 우수한 언어이다. 그러나 그 대가로 효율성을 희생하는 경우가 있다. 모바일 기기/환경에서는 효율성이 보다 중요하므로, 이에 맞게 성능을 조정하면 사용자가 만족하는 성능을 보일 가능성도 매우 높다. 성능 조정은 임의적 작업이 아닌 체계적 과정이고, 병목을 집중 개선하면 큰 효과를 얻을 수 있다. 게임 응용 소프트웨어 개발 시 제한된 메모리와 속도를 고려해야 한다. 또한, 게임 소프트웨어의 종류에 따라 속도를 가속시킬 수 있는 방법은 여러 가지가 있으며, 최대한 프로그램을 가속시킬 수 있는 알고리즘을 사용해야 한다.

[참고문헌]

- [1] 온라인 게임의 원소스 멀티유즈 활용 방안에 관한 연구-김보수
- [2] 실시간 멀티플레이어 온라인 게임을 위한 게임 트래픽 소스 모델 분석- 이문규(연세대간: 2001)
- [3] 무선 인터넷에플리케이션 프로그래밍, 삼양출판사, LG-CNS시스템,2001
- [4] about WAP, 영진.com, 2001간
- [5] J2ME Specification : <http://java.sun.com/j2me/> 참조
- [6] CLDC 1.0 : <http://java.sun.com/products/cldc> 참조
- [7] MIDP 1.0 : <http://java.sun.com/products/midp> 참조
- [8] TIA/EIA/IS-707 : Data Service Option for Wideband Spread Spectrum System
- [9] TIA/EIA/IS-99 : Data Service Option Standard for Wideband Spread Spectrum Digital Cellular System
- [10] <http://my.dreamwiz.com/june8th/> 김준동
- [11] 조인혜, 자바기술과 정보, 소프트웨어진흥원, 2000.
- [12] LDAP Programming with Java