韓國 컴퓨터情報學會 論文誌
第11卷 第1號, 2006. 3.

2006-11-1-4-2

모바일 컨텐츠 개발에서 기존의 컨텐츠를 최대한 이용하기 위한 아바타 변환기 개발 연구

박 대 혁*, 강 의 선**, 홍마리아***, 이 근 수****, 임 영 환*****

Developing Avatar Converter by using existing Contents in Mobile Contents Development

Dae-Hyuck Park*, Eui-Sun Kang**, Maria Hong***, Keun-Soo Lee****, Young-Hwan Lim*****

요 약

모바일 단말기 장치가 발전하면서 컴퓨터로만 할 수 있었던 게임, 인터넷, 금융 서비스 등을 이동 단말기를 이용하여 처리할 수 있다. 본 논문에서는 펜티엄급 PC에서 유행하던 게임을 이동 단말기에서 재생 가능성 하도록 재구성하고, 자동화 가능한 재구성 요소를 정리하여 아바타 변환기로 개발하고자 한다. 즉, 기존의 자원을 최대한 재활용하여 이동 단말기용 컨텐츠 개발 속도를 향상시킬 수 있는 방법을 제안하고자 한다. 플랫폼의 제약사항을 고려하여 기존의 컴퓨터의 아키텍처를 모바일 플랫폼에서 동작 가능한 로직과 리소스로 구분하여 변환하는 동작이 아바타 변환기의 주 기능이다. 이동 통신 사업자의 플랫폼 종류에 따른 다수의 컨텐츠 개발에 공용의 로직과 리소스를 변환함으로써 개발 속도를 향상시킬 수 있다.

Abstract

Mobile device has the same service as PC with the development of mobile technology, such as game, internet, economic service. This paper we play the games in Mobile Device which were popular in Pentium 1 PC, and then develop Avatar Converter by summarizing Automating Elements. In other words, we propose a method that can accelerate developing speed of Mobile Contents by recycling existing resource. The main function of Avatar Converter is that transform existing architecture with dividing logic and resource which can play in mobile by considering restriction items. It can accelerate developing speed by transforming common logic and resource in various contents development for kinds of mobile company.

▸ Keyword : Avatar Convert, Mobile Platform, Resource Trans-coder

---

• 제1저자 : 박대혁
• 접수일 : 2005.11.28, 심사완료일 : 2006.02.10
*,** 숭실대학교 미디어학과 박사과정, *** 안양대학교 디지털 미디어공학과 교수,
**** 한경대학교 컴퓨터공학과 교수, ***** 숭실대학교 미디어학과 교수
※ 본 연구는 숭실대학교 교내연구비 지원으로 이루어졌음.

Ⅰ. 서 론

음성 통화를 중심으로 하던 이동 단말기 시장이 점차 데이터 통신을 기반으로 하는 벨소리, 문자 서비스, 게임, 금융 서비스 등으로 급성장 하고 있다. MMS (Multimedia Messaging Service), 카메라 모듈을 이용한 멀티미디어 기술이 단말기에 적용되어 2.5세대-3세대 멀티미디어 중심의 이동 단말기로 발전하고 있다. 특히, 이동 단말기 플랫폼인 SUN Java, Qualcomm BREW, GVM/GNEX, SK-VM, KVM, WIPI 등의 플랫폼 위에 다운로드 되어 실행되는 애플리케이션이 다양한 서비스 영역에서 사용가능하도록 개발되고 있다.[1] 특히 항상 휴대하고 다니는 작은 컴퓨터인 이동 단말기를 이용한 모바일 게임은 폭발적인 증가세를 보이고 있다. 따라서 본 논문에서는 모바일 게임 개발 프로세스에서 절약할 수 있는 요소를 자동화 할 수 있는 방법을 아바타 변환기를 이용하여 얻고자 한다.[2][3]

컨텐츠의 호응도가 중요한 비중을 갖는 게임 컨텐츠 개발의 성공을 위해서 기존에 도스 모드에서 개발되어 매우 높은 호응을 얻은 게임을 선택하여 이동 단말기에서 동작시키기 위한 프로토타입을 개발하면서 기존의 게임에 사용한 로직과 리소스를 최대한 활용하여 개발 자원을 절약하고 최상의 품질의 게임을 개발하고자 한다. 제 2절에서는 모바일 플랫폼과 기존에 흥행했던 마도전기 게임에 대한 소개를 하고, 제 3절에서는 이동 단말기에서 재생하도록 하면서 발생한 문제점과 해결 방안을 열거한다. 게임 컨텐츠를 포팅을 하면서 얻은 중요 정보를 기반으로 제 4절에서는 모바일 마도전기를 기반으로 아바타 변환기를 설계하고, 마지막으로 제 5절에서 결론을 맺는다.

Ⅱ. 관련 연구

2.1 Mobile Platform Service

모바일 플랫폼이란 모바일 기기에 탑재되어 기기의 기능을 동작하기 위한 중간 레벨의 잘 정의된 API 및 개발 환경을 제공하는 소프트웨어 시스템을 말한다. 일반적으로 알려진 플랫폼은 Sun의 Java와 퀄컴 Brew등이 있다. 플랫폼은 주로 사업자의 수익에 큰 영향과 관련 있으므로 사업자가 선정하는 GVM/GNEX, SK-VM, KVM등의 플랫폼으로 재구성되어 상용화 중이다. 망 개방을 앞두고 우리나라를 중심으로 모바일 플랫폼 WIPI를 제안하고 있다.[4]

표 1. 모바일 게임 플랫폼
Table 1. Mobile Platform

플랫폼	개발언어	수행방식	서비스사
GVM	Mobile C	VM	SK
SK-VM	Java	VM	SK
KVM	Java	VM	LG
BREW	C/C++	Native Bin	KTF
WIPI	Java/C/C++	통 합	KWISF, TTA, ETRI

〈표 1〉의 내용을 근간으로 모바일 플랫폼에 대해서 알아보면 다음과 같다. 모바일 게임 방식은 크게 Virtual machine(VM) 방식과 Native Binary 방식으로 나뉜다. VM 방식은 Sun의 Java, 신지소프트의 GVM/GNEX[5], XCE의 SK-VM[6]을 이야기 하며, 각각은 CLDC, MIDP와 결합하는 Java기반과 Mobile C에 의해서 만들어진 어플리케이션이 실행 환경을 제공한다. 개발된 프로그램을 프로그램이 실행될 CPU에서 동작되는 기계어 컴파일 하여 실행시키는 방식인 Native Binary 방식은 BREW, Symbian의 Ericsson, Nokia, 삼성전자 등 모바일 산업을 주도하는 기업들이 공동으로 설립한 컨소시엄에서 발표된 Symbian OS에서 동작된다[7].

VM 방식의 느린 속도와 Native Binary 방식의 이식성 문제를 해결하면서 표준화된 플랫폼이 요구되고 있으면 이

를 WIPI를 이용하여 모바일 플랫폼 표준하고 있다. WIPI는 다양한 플랫폼으로 구성되어 있던 모바일 플랫폼의 장점을 수용하여 한 차원 높은 서비스를 위한 기능을 추가하여 기존 플랫폼의 단점을 보완하기 위한 것으로 개발자는 호환성 및 개발의 용이성과 이기 종간에 많이 발생하는 이식의 문제점을 보완하게 될 것이다. 모바일 하드웨어의 발전과 플랫폼의 성능이 발전됨에 따라서 이동통신 사업자를 기준으로 뉴스, 은행, 게임, 멜로디, 캐릭터, 동영상 서비스와 같은 다양한 서비스가 개발되고 있으며, 기존에 PC를 이용하여 서비스를 하고 있던 인터넷을 이용한 많은 서비스들이 모바일 서비스에 적용되고 있다[8].

2.2 컴퓨터용 마도전기

마도전기 게임은 던젼 형태의 RPG 게임으로 97년도 Disc Station 시리즈물로 배포하여 많은 PC 사용자에게 인기가 있었던 PC용 게임으로 RPG 게임의 요소인 오프닝, 통상 모드, 전투모드를 갖으며 3D 시점을 갖는 던젼 형식과 많은 애니메이션, 많은 이벤트와 난의 도가 높은 퍼즐을 갖는다[9]. RPG 게임의 흥미를 높이기 위해서 스토리를 기반으로 화려한 오프닝에 많은 이미지를 사용하며 시나리오를 추가하여 조작자가 주인공이 되어 문제를 풀어나가면서 게임을 즐긴다.

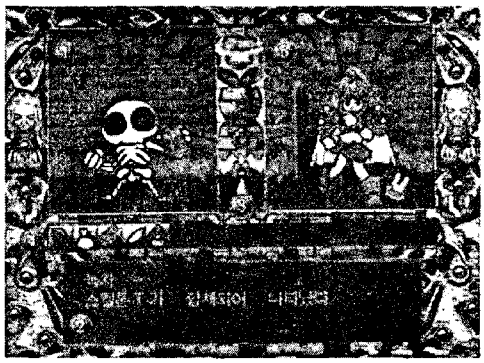

그림 1. 컴퓨터용 마도전기 게임 화면
Fig 1. Mado Game on PC

(그림 1)은 컴퓨터용 마도전기의 모습이며, 마도전기는 아르르가 주인공이 되어 8*8로 구성되어 있는 층에 제시된 퍼즐을 풀어가면서 게임을 진행한다. 던젼에서는 이동모드와 전투모드로 크게 구성되며 이동모드는 던젼의 퍼즐을 풀기 위해서 미로를 찾아다니게 된다. 이동 모드 중 이벤트성으로 전투 모드가 발생되며, 마도력을 지키면서 마법을 이용하여 적과 전투를 하게 된다. 마도력을 유지하면서 모든 층의 퍼즐을 해결하는 것이 마도전기(Mado)의 게임의 묘미이다.

III. 모바일 단말기에서의 재생

게임 머들에게 널리 알려있던 COMPILE의 아르르가 주인공이 되어 진행하는 마도전기는 그 당시 화려하고 움직이는 애니메이션으로 새로운 RPG 시장을 개척한 게임이다. 이러한 게임을 모방에서 동작할 수 있도록 하는 것은 기존의 시나리오와 이미지에서 예전 그 느낌을 다시 느낄 수 있도록 하는 것이 가장 중요하다. Disk Station의 모든 게임들은 CPU 펜티엄 100MHz이상, 메모리 16MB 이상, 사운드 블래스터 16이상의 하드웨어적인 성능을 요구하는 게임으로 이를 모바일에 동작 하는 것은 이전까지는 매우 큰 무리수이다. 본 논문에서는 이를 모바일 API를 이용하여 재구성한다. 즉, 기존의 시스템 요구 성능에는 매우 부족한 이동 단말기에서 동작가능하게 구현하는 것이 문제이다.

Windows95 OS, DirectX를 기반으로 구현되어 있는 소스를 SKT에서 서비스 가능한 신지소프트웨어의 GVM 코드를 이용하여 구현하였다. 구현을 하면서 기존의 화면해상도와 컴퓨터의 파워풀 한 기능에서 수반하는 애니메이션, 게임의 시나리오를 구성하는 맵과 이벤트에서 발생하는 문제점을 해결하는 것이 모바일 게임으로 구현하는데 매우 큰 문제점으로 제시되었다. 기존의 맵 정보를 그대로 인용하면서 저성능의 단말기에서 동작 가능하도록 하기 위해서 3개의 층을 하나의 맵으로 구성하고, 맵의 정보를 단순화 시켜서 자료의 효율성을 높였다. 또한 모든 동작은 모바일 플랫폼에서 제시한 이벤트 방식으로 호출되도록 구성하였으며, 게임의 시나리오의 이동모드와 전투모드를 기본으로 층간의 이동, 오프닝, 엔딩모드로 나누어서 블록단위로 독립적으로 동작 한다.

표 2. 게임 동작 시스템 요구사항
Table 2. Comparison of Operating Background

	컴퓨터 시스템	모바일 시스템
CPU	100M 이상	ARM7 or 9
메모리	16M 이상	0-512KByte
그래픽	16컬러 이상	256 이하 컬러
OS	Windosw	SK-VM, GVM

게임을 개발하면서 가장 큰 문제점은 기존의 pc의 경우 확장성 높은 메모리 자원을 이용하여 많은 데이터를 보관할 수 있었으나 모바일의 경우 최소 128Kbyte에서 최대 512Kbyte에 모든 게임의 로직과 데이터를 포함하여야 하는 문제점이 있다. 이를 해결하기 위해서 아르르의 특징을 나타내는 길고 복잡한 말투를 최소한의 캐릭터 특징을 손상하면서 단순화시키는 시나리오 작업과 이동 단말기의 디스플레이 장치에 적합한 이미지로 변환하여 최적화 하는 것이 가장 큰 문제이다. 다음 〈표 2〉는 이러한 게임이 동작되는 시스템의 요구사항을 명시한 것이다.

그림 2. 모바일 마도전기
Fig 2. Mado Play in Mobile Device

모바일에서는 다음과 같은 제약 사항을 갖으며 이를 해결하는 것이 기존의 게임인 마도전기(Mado)를 모바일 마도전기(Mado)로 개발하는데 문제점의 원인이다. 프로그램의 크기, LCD 사이즈의 제한 및 다양함, 인터페이스의 제한, Symbol 개수와 Array의 제한 255개, 자료형의 제한 int와 string뿐, 함수 Call 의 Depth 제한, 기존의 C에서 보였던 모듈 별 소스 개발 방식이 불가능, 처리 능력 및 반응 속도의 프로세스 별 다름, 이를 해결하기 위해서 기존의 마도전기(Mado)의 2개의 창을 (그림 2)에서 보인 것과 같이 컨텐츠를 재구성하였고, 기존의 640*480의 해상도의 게임을 120*146의 해상도의 게임으로 재구성, 단말기의 입력 방식인 키패드를 이용하여 동작 가능하도록 변경, 또한 타이머를 이용하여 애니메이션을 재구성하여야 하였다. 기존의 마도전기의 3D 시점 표시하는 방법인 9개의 셀을 이용하는 방법을 그대로 착용 변경하여 사용하였다.

IV. Avatar Trans-coder

모바일 플랫폼에서 동작되는 프로그램은 이벤트방식으로 동작되며, 플랫폼 별로 서로 다른 특징 점을 갖지만 거의 비슷한 구성으로 동작되는 것에서 착안하여 이러한 코드 간에 재 생성할 수 있는 방법인 아바타 변환기를 설계하고자 한다. 즉 플랫폼에서 동작되는 어플리케이션을 개발하는 언어가 Java와 C, C++을 기반으로 제작되어 있다. 또한 모든 동작이 VM 위에서 동작되므로 유사한 형태의 구조를 갖는 것이 특징이다.

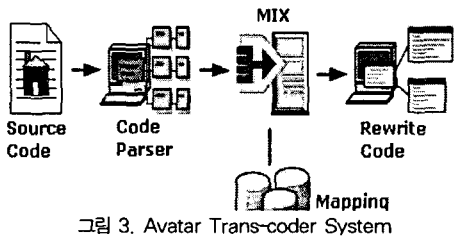

그림 3. Avatar Trans-coder System
Fig 3. Avatar Trans-coder System

아바타 변환기란 기존에 특정 플랫폼으로 개발되어 있는 게임과 같은 소스 코드를 다른 플랫폼에서 동작 가능하도록 소스 코드를 Java, C, C++의 형태의 다른 언어로 변경하는 것이 주요 기능이다. 또한 플랫폼 별로 동일한 기능에 대해서 특징적인 API를 사용하는 것을 플랫폼에 적합한 API로 변경하여 변경한 플랫폼에서 동작할 수 있도록 하는 것이 아바타 변환기의 주요 동작이다.[10] (그림 3)은 아바타 변환기의 시스템 구성도이며, Code Part와 재구성된 코드를 생성하기 위한 MIX가 주요 모듈이 된다. MIX에 의해서 생성된 코드를 플랫폼에서 동작 가능한 소스 레벨의 코

드로 생성하는 것이 필요하다[11][12]. 이러한 분석된 내용을 재구성하기 위해서 소스간의 관련성 및 각각의 특징점을 파악하는 것이 아바타 변환기의 성패를 결정할 것이다.

4.1 모바일 마도전기의 동작

사용자가 입체적인 미로 화면으로 인식하기 위해서 현재 아르르의 위치를 중심으로 앞으로 3칸 정도의 벽의 상태를 확인하고 디스플레이를 하여야 한다. (그림 4)에 나타난 것처럼 복잡한 미로를 구성하기 위해서 현재 아르르의 위치를 기준으로 현재 칸을 1단계, 앞 칸의 좌우중앙을 2단계, 앞의 앞칸을 3단계의 depth 값을 갖는다. 이러한 1, 2, 3 단계의 depth 값을 증가하며 가운데, 왼쪽, 오른쪽의 값을 조사하여 총 15개의 벽의 형태로 디스플레이 한다.

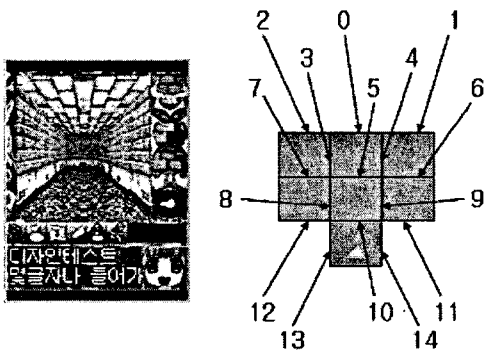

그림 4. 입체화면 구성
Fig 4.Composition of Three Dimension Screen

벽의 상태는 기존의 게임의 벽의 상태의 맵 데이터를 10진법을 이용하여 모바일에서 사용 가능한 데이터로 등록하여 사용하였다. 많은 층을 일관되게 관리하기 위해서 현재 작업 중인 층만을 임시 변수에 기록하여 빠르게 접근할 수 있도록 하였다.

이벤트로 구성된 PC 기반의 게임을 모바일 단말기에서 동작시키기 위해서 모바일 C의 이벤트 기반 동작 성을 이용하여 버튼 입력과 타이머의 이벤트의 상태에 따라서 서로 다른 동작을 할 수 있다. 이를 이용하여 하나의 층에 해당하는 8*8 블록에 해당하는 이벤트 리스트를 만들어 칸에 이벤트가 등록 되어 있으면 적적한 이벤트에 응답을 하도록 구현하였다.

CheckEvent()에서 현재 블록에 이벤트가 등록 되어 있는지를 검사하고, 등록 되어 있으면 적당한 이벤트 동작이 이루어지도록 구현하여 이벤트가 등록 되어 있지 않으면 위치가 빠르게 변경되면서 실시간 3D 입체화면이 구성되고, 이벤트가 등록되어 있으면 해당 이벤트에 적절하게 응답을 하도록 되어 있다. CheckEvent() 함수는 주기적으로 반복해서 검사 하도록 되어있다. 이벤트 속성은 현판(E_NOTICE), 문잠김(E_LOCK), 전투(E_BATTLE), 보물(E_PRESURE), 스위치(E_SWITCH), 계단(E_DOWNFLOOR, E_UPFLOOR), 회복샘(E_CURESPRING), 퍼즐에 맞는 이벤트들이 등록되어 있다. 등록되어 있는 이벤트들의 블록에 들어가면 일괄적인 내용은 ActionEvent()에서 처리가 되고 층마다 독특한 이벤트의 경우에는 각각의 해당 층에 해당하는 함수가 호출되어 연결 처리 된다. 다음의 리스트는 본 코드의 이벤트값을 기록한 것이다.

```
int EventList01[12][EVENT_INFO_SIZE] = {
        {1,4,6,D_NORTH,E_NOTICE,0,9,0},
        {1,4,6,D_NORTH,E_NOTICE,0,9,0},
        {1,1,7,D_WEST,E_NOTICE,9,2,0},
        {1,7,7,D_EAST,E_NOTICE,21,6,0},
        {1,5,8,D_WEST,E_NOTICE,27,8,0},
        {1,1,2,D_WEST,E_NOTICE,37,6,0},
        {1,8,5,D_NORTH,E_NOTICE,43,7,0},
        {1,7,1,D_EAST,E_NOTICE,43,7,0},
        {1,3,4,D_EAST,E_LOCK,0,0,3},
        {1,7,1,D_EAST,E_LOCK,0,0,4},
        {1,5,6,D_SOUTH,E_SWITCH,35,2,3},
        {1,3,1,D_EAST,E_PRESURE,100,1,4},
        {1,8,8,D_SOUTH,E_DOWNFLOOR,22,4,0}
};
int EventList02[26][EVENT_INFO_SIZE] = { ... ... ... ... ... ... }
int EventList03[26][EVENT_INFO_SIZE] = { ... ... ... ... ... ... }
int EventList04[20][EVENT_INFO_SIZE] = { ... ... ... ... ... ... }
int EventList05[25][EVENT_INFO_SIZE] = { ... ... ... ... ... ... }
int EventList052[27][EVENT_INFO_SIZE] = { ... ... ... ... ... ... }
int EventList06[29][EVENT_INFO_SIZE] = { ... ... ... ... ... ... }
int EventList07[18][EVENT_INFO_SIZE] = { ... ... ... ... ... ... }
```

4.2 Source Analysis

소스 분석기는 플랫폼에 적합한 API를 기반으로 구현 되어 있는 코드를 다른 플랫폼에 맞는 API로 매핑하기 위해서 현재 사용되고 있는 소스의 특징을 이해하여 순서적으로 논리적으로 분석하여 분석된 내용의 소스 코드를 선택된 플

랫폼의 특징적인 API로 변환하여 코드를 재구성하여 생성하고자 하는 플랫폼의 API로 변환하는 것을 기본으로 소스를 최적화 하는 기능을 갖도록 설계하고자 한다. 소스 코드의 특성상 라인단위, 함수 단위로 맵핑되는 점을 고려하여 처리 단위를 라인으로 하는 LR 파서를 제작하여 구성하였다. GVM의 코드의 경우에는 다수의 정의문과 하나의 소스로 구성되므로 이를 분석하는 것은 라인단위로 순차적으로 분석하는 것이 적합하다. 하지만 이벤트를 베이스로 동작되므로 중요 이벤트 별로 다수의 소스 분석이 필요하며 동작 모델을 만드는 것은 플랫폼에서 이벤트를 사용자가 사용한다는 상태도를 기반으로 구성하는 것이 적합하다.

Java혹은 C++을 기반으로 하는 코드의 경우에도 유사한 호출 시점을 갖는 것이 특징이다. 또한 코드의 구현은 이벤트를 중심으로 구현하는 경우가 많으므로 기존에 GVM으로 구현되어 있는 코드를 이벤트를 중심으로 로직을 재구성하면 문제점 없이 동작 가능하다. WIPI 경우에는 모든 동작이 Life Cycle이라는 상태 전이(흐름)도를 중심으로 동작이 되므로 GVM의 이벤트를 상태 전이도로 재구성함으로 플랫폼에 적합한 소스로 재생성 가능하다.

4.3 Source Mix

분석된 내용을 기반으로 플랫폼에 적합한 소스코드를 재생성할 수 있다. 이는 기본적으로 플랫폼에서 요구하는 소스 형태를 갖는 것이고, 분석한 소스의 내부의 주요 로직을 변형된 형태의 코드로 변환하는 것이 주요 기능이 된다. 프로그램의 소스는 동작되는 플랫폼의 특성에 따라 다소 다른 동작 모델과 API를 갖으며, 이는 적응 테이블을 이용하여 해결할 수 있다. 소스 코드는 크게 변수와 함수로 구성되며, Java 나 C++에서는 OOP 개념이 적용되어 구조가 상대적으로 객체 기반으로 하는 코드가 생성된다. 객체의 경우에 하나 이상의 경우에는 조금 복잡한 함수 형태의 호출로 해결할 수 있다는 연구를 이용함으로서 해결할 수 있다.

플랫폼에 따라서 사용하는 소스 언어에 따라서 변수의 형태와 단말기에서 사용되는 API 관련 함수, 시스템 관련 함수, 호출하는 방법, 디바이스를 제어하는 방법에 큰 차이점을 나타내었다. 특히 화면에 디스플레이 하거나 사운드를 재생하는 동작에 큰 차이점을 갖는다. 이러한 차이점을 해결하기 위해서 차이점을 갖는 함수 단위로 구성된 파싱 된 내용을 기반으로 코드를 생성해낸다. 구성도 (그림 5)의 내부 모듈이 코드를 순차적으로 생성하며 코드의 생산성을 높이기 위해서 함수 내의 라인을 처리할 수 있는 코드 생성기가 동작된다. MIX의 내부에는 코드 생성기가 세 단계로 구성되어 있는 것과 같으며 서로 코드를 생성할 수 있도록 되어 있다. 이때 사용되는 테이블을 맵핑 테이블이라고 한다. 맵핑 테이블은 인터프리터 언어에서 사용하는 1:1 매칭 맵핑 테이블과 같이 호출 모바일 플랫폼의 특정 API를 선택한 플랫폼의 API에 대응하는 맵핑 테이블이다.

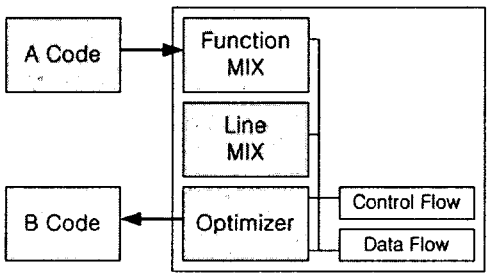

그림 5. 아바타 변환 시스템의 구성도
Fig 5. Structure of AVATAR Converter

변환기를 이용하여 정확도가 높은 코드를 생성하기 위해서 사용되는 소스의 구성 단계에 대한 이해가 필요하며 게임의 경우에는 MAIN, EVENT, DISPLAY, SOUND, MESSAGE로 구성되어 있으며, 호출되는 함수들이 나머지에 해당되며 키, 타이머, 소프트웨어 이벤트에 의해서 호출된다. API이 테이블은 플랫폼 별로 독특하게 구성되어 있으며, 특정한 파라미터를 이용하여 동작되도록 되어 있다. API는 Display 함수, String 함수, Mathematics 함수, Handset 제어 함수로 구성되며 이렇게 크게 4단계로 매핑 테이블을 구성하므로 합성 시에 정확도를 높일 수 있으며, 처리 속도 또한 향상될 수 있다. 또한 소스의 복잡도가 증가하므로 최적화기를 통해서 복잡 도를 조절하여 코드를 생성할 수 있다.

4.4 이미지 변환기

게임의 높은 비중과 제작 시 많은 자원을 요구하는 캐릭터 디자인 요소를 이미지 변환기를 이용하여 변환하고자 하는 것은 단말기마다 갖는 LCD 규격의 차이점과 모바일 게임 컨텐츠에서 다수의 파일 접근 불가능성에 의해서 매우 중요한 요구사항이 된다. 이동 단말기의 VM에서 지원하는 이미지로 변환하기 위해서 기존의 리소스를 축소하고 연출하여야 한다. 고해상도의 수많은 파일로 구성된 이미지 데이터를 규격의 해상도의 하나의 축소된 이미지로 만들어야 하는 것이 중요 동작이 된다. 축소 작업을 하면서 발생 가능한 흐려짐, 번짐, 인식률 저하 등의 문제점을 줄이기 위해

서 이미지의 데이터 영역과 캐릭터 영역을 구분하여 배경을 투명화하고 라플라시안 필터를 이용한 윤곽선의 곡부를 구분하고, 이를 이진 데이터 처리하여 강조하는 처리 후 축소하는 방법을 사용한다. 결과에 의해서 만들어진 이미지는 이동 단말기의 화면구성에서 배경과 에지가 구분되어 뚜렷하게 재구성됨을 확인 할 수 있다.

V. 결론

게임과 같은 이동 단말기에서의 엔터테인먼트 분야의 활용도가 급증하고 있다. 그에 따라서 PC에서 유행했던 RPG와 같은 시나리오를 갖는 종류의 컨텐츠의 요구가 증가하고 있다. 즉, PC에서 흥행했던 게임 컨텐츠를 이동 단말기에 포팅을 하고자 하는 요구가 증가하고 있다. 이를 위해서 본 논문에서는 이동 단말기에서 PC 컨텐츠로 흥행했던 게임인 마도전기를 단말기에서 재생 가능하도록 포팅 하였다. 제 3장, 제 4장과 같이 포팅을 하면서 PC용 컨텐츠에서 최대한 재활용 가능한 로직과 리소스를 선택할 수 있도록, 아바타 변환기의 중요 모듈을 설계한다. 이를 바탕으로 구현된 아바타 변환기에 의해서 중요 로직과 리소스를 최대한 자동으로 변환할 수 있도록 구현하였다. 구현에 의해서 게임 컨텐츠의 많은 부분의 반복적인 요소를 규칙성을 기반으로 변환하여 사용함으로써 컨텐츠 개발의 소요 자원을 줄일 수 있었다.

또한 기존 PC용 게임 컨텐츠의 모바일 서비스 가능성을 검토할 수 있었으며, 유사 다른 플랫폼에서의 동작 가능한 데이터 생성 또한 자동화 가능함을 예상할 수 있었다. 즉 플랫폼에 맞는 프로그램 개발을 완성하고 소스레벨에서 변환 테이블을 기반으로 하는 API 변환에 의해서 중요 로직 재활용을 확인 할 수 있었다. 앞으로 변환 로직과 리소스에 XML 데이터 형식의 설명을 추가하여 복잡도가 높은 로직, 리소스 변환이 가능하게 하는 연구와 이기종 플랫폼 간의 변환 신뢰도를 높일 수 있는 방법의 연구를 하고자 한다.

참고문헌

[1] Hyuck Yoo, Mobile Device S/W Platform Trend, KISS, January 2004

[2] 이현창, 최광돈, "온라인.모바일 환경에서 멀티미디어 컨텐츠 생성을 위한 학습 시스템의 설계 및 구현에 관한 연구", 한국컴퓨터학회논문집, 제10권 제1호, 2005.

[3] 이석기, 김성회, "기술수용모델을 활용한 모바일 소액결제 시스템의 인지적 특성 분석", 한국컴퓨터학회논문집, 제9권 제1호, 2004.

[4] Joon Sung Hong, Present standard status of WIPI and prospect of development, KISS, January 2004

[5] SinjiSoft GNEX CLUB.COM, http://www.gnexclub.com

[6] XCE developer zone, http://developer.xce.co.kr

[7] JungHyun Han, Mobile game : the state of the art, KISS, January 2004

[8] Mobile Java, http://mobilejava.co.kr/

[9] Disc Station Vol. 1-Vol.5 (COMPILE)

[10] Vlasimir Getov, "Multi-Language Programming Environments for High Performance Java Computing", In Scientific Programming, 1999

[11] Per Bothner and Tom Tromey, "Java/C++ integration : Writing native Java methods in natural C++"

[12] Alfred V.Aho, Ravi Sethi, Jeffrey D.Ullman, "Compilers: Principles, Techniques, and Tool"

저 자 소 개

박 대 혁

2004년 ~ 현재 : 숭실대학교 미디어학과 박사과정

〈관심분야〉 멀티미디어 기술, 무선 인터넷, 시스템 소프트웨어

강 의 선

2002년 2월 숭실대학교 석사 졸업

2005년 ~ 현재 : 숭실대학교 박사 과정

〈관심분야〉 멀티미디어, 무선 인터넷

홍마리아

2001년 숭실대학교 정보미디어학과 공학석사

2004년 숭실대학교 컴퓨터학과 공학박사

2004년 ~ 현재 : 안양대학교 디지털 미디어공학과 교수

〈관심분야〉 멀티미디어 스트리밍, MPEG21, 유비쿼터스, DMB

이 근 수

1983년 숭실대학교 전자계산학과 공학사

1988년 숭실대학교 전자계산학과 공학석사

1993년 숭실대학교 전자계산학과 공학박사

2003년 3월 ~ 2004년 2월 미국 George Mason University, 전자계산학과 객원교수

1989년 ~ 현재 : 한경대학교 컴퓨터공학과 교수

〈관심분야〉 패턴인식, 퍼지이론, 컴퓨터비전, 지식기반 시스템, 동작이해, 비디오검색 등임.

임 영 환

1985년 Northwestern University 전산학과(박사)

1979년 ~ 1996년 한국전자통신연구소 책임연구원

1996년 ~ 현재 : 숭실대학교 미디어학부 교수

〈관심분야〉 멀티미디어 기술, 유비쿼터스, 모바일 분야