무선 환경에서 자원 관리를 위한 모바일 에이전트의 설계 및 구현

나 승원°, 오 세만
동국대학교 컴퓨터공학과
(nasw, smoh)@dgu.ac.kr

Design and Implementation of Mobile Agent for Resources Management in Wireless Environment

Seungwon Na°, Seman Oh
Dept. of Computer Engineering, Dongguk University

요 약

이동통신과 인터넷 기술의 결합으로 탄생한 무선 인터넷은 휴대의 편리성을 제공하고 있으나 모바일 환경의 제약 사항 때문에 대중적인 인터넷 서비스로는 발전하지 못하고 있다. 무선 환경의 제한 요소 중 협소한 메모리 공간으로 모바일 디바이스에서는 효율적인 자원 관리를 수행하지 못하는 단점을 가지고 있다. 휴대성이 고려되어야 하는 하드웨어 특성상 충분한 메모리 용량을 확보하는 데에는 한계가 있으므로 디바이스 내부의 저장 장치에서 외부의 영역까지 저장 공간을 확대하여 데이터 처리가 수행될 수 있는 모바일 플랫폼 구조로 발전되어야 할 것이다.

본 논문에서는 모바일 디바이스의 내부에서 외부의 서버까지 메모리 공간을 확장하여 데이터 활용을 가능하게 하고 디바이스 내부의 파일을 효율적으로 관리할 수 있는 모바일 에이전트를 제안하며 이를 실현하기 위한 자원 관리시스템(RMS: Resources Management System)을 설계하고 구현하였다. 제안된 RMS를 적용한 디바이스는 '모바일 공간확장'으로 확대된 프로세스 적용이 가능하며 내부 파일을 효과적으로 관리하여 최적의 메모리 공간을 유지하는 효과가 있다.

1. 서 론

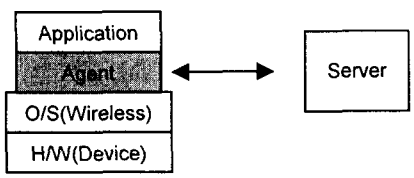
인터넷이 결합된 이동통신 기술은 무선 인터넷 서비스를 제공하게 되었고 무선 인터넷은 휴대의 편리성으로 차세대 인터넷 서비스로서 주목을 받고 있다. 그러나 모바일 환경은 작은 디바이스의 처리 능력이라는 한계를 가지고 있어서 무선 인터넷이 대중화 되는 데에는 많은 시간과 노력이 요구된다. 특히 협소한 메모리 공간 때문에 디바이스 내부에서 자원 관리를 효율적으로 수행하지 못하여 사용자 요구에 충족하는 서비스를 제공하는 데에는 한계가 있다. 따라서 향후 모바일 디바이스의 메모리 활용 형태는 내부의 저장 장치에서 외부의 서버 영역까지 확대하여 프로세스가 수행되는 모바일 플랫폼 구조로 발전되어야 할 것이다.

본 논문에서는 모바일 디바이스 내부의 저장 공간에서 외부의 서버까지 메모리 영역을 확장하여 데이터 활용을 가능하게 하고 내부의 자원관리 기능을 수행 할 수 있는 모바일 에이전트를 제시하고자 하며 이를 실현하기 위한 자원 관리시스템(RMS: Resources Management System)을 설계하고 구현하였다. RMS를 적용한 디바이스는 '모바일 공간확장'을 통해서 확대된 프로세스 적용이 가능하며 내부 파일을 효과적으로 관리하여 최적의 메모리 공간을 유지하는 효과를 가진다.

2. 에이전트 기술

에이전트 기술은 사용자가 수행 할 작업이나 반복적인 일 들을 자동적으로 처리해 주는 시스템이다[7]. 에이전트는 지시된 작업을 수행하고 조정하기 위한 기능을 제공하는 제어지식 에이전트와 수행할 역할의 기능을 규정하고 있는 영역지식 에이전트, 그리고 프로시저 콜과 메시지 교환을 처리하는 통신모듈 에이전트로 구성된다[4].

무선 환경에서의 에이전트 구성은 사용자가 필요로 하는 정보를 찾아내어 보여주는 형태로서 다운로드 시스템 등이 여기에 포함된다. 무선 환경에서의 에이전트는 오퍼레이션 시스템 위에서 다운로드 받은 파일을 내부의 애플리케이션과 인터페이스 관계를 형성하며 수행을 유도한다[3].

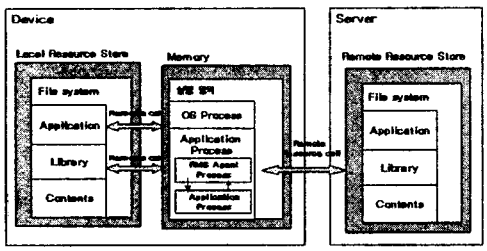


[그림 1] 무선 환경에서의 에이전트

3. RMS의 설계

3.1 RMS의 개요

RMS는 데이터 저장 공간을 확대하여 자원 관리를 효율적으로 운영하기 위한 시스템이다. 실행 파일이 필요한 리소스 파일을 호출 할 경우 RMS 에이전트는 서버에 요청하여 리소스 파일을 다운로드 받아 동적 로딩한다. 기존의 다운로드 시스템은 컨텐츠 파일을 다운로드 받아 수행하는 반면에 RMS는 확장된 프로세스를 수행하는 것에 차이가 있다. 리소스 파일은 DLL(Dynamic Link Library) 컴포넌트와 OCX(OLE Control Extension)파일, 기타 컨텐츠 파일 등을 대상으로 한다.

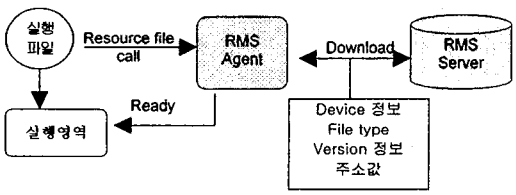


[그림 2] RMS 프로세스 수행구조

3.2 RMS 에이전트의 설계

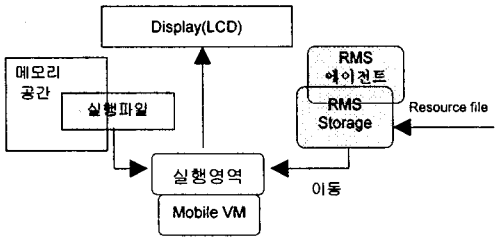
RMS 에이전트는 디바이스 내부의 애플리케이션 프로세스의 리소스 호출 범위를 로컬에서 리모트로 확장하기 위한 시스템으로 메모리에 상주하게 되며 RMS 에이전트 전용 API set을 정의한 코드 형태로서 디바이스와 서버의 자동조절 기능을 수행한다. 이 역할을 위한 세부 기능으로 다운로드 영역, 디스플레이 영역, 저장장치 관리 영역으로 나누어 에이전트를 설계하였다.

첫째, 다운로드 영역은 실행 파일이 리소스 파일을 호출할 경우 RMS 에이전트는 정보를 입력 받아 서버로부터 리소스 파일을 다운로드 받은 후 실행을 준비하는 영역이다. 파일 저장은 RMS가 관리하는 별도의 스토리지가 있으며 이곳으로 다운로드 된다.



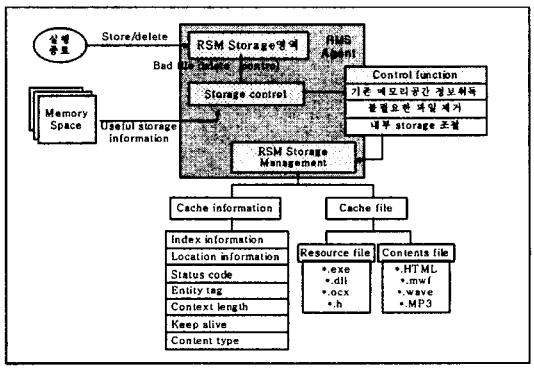
[그림 3] 다운로드 수행영역

둘째, 디스플레이 영역은 RMS Server로부터 다운로드 받아 적재된 리소스 파일이 실행 파일과 동적 링킹되어 수행된다. 이때 DLL은 실행 파일과 컨텐츠 파일은 디바이스에 내장된 애플리케이션과 연동된다. 리소스 파일이 RMS 스토리지에 이미 존재하는 파일 인 경우에는 캐쉬 기능으로 off-line 링킹되어 수행되어 진다.



[그림 4] 디스플레이 수행영역

셋째, 저장장치 관리 영역으로 실행 종료 후 리소스 파일을 RMS 스토리지에 저장하거나 삭제 할 수 있으며 디바이스 내부의 메모리 정보를 운영체제로부터 입력 받아 RMS 저장 공간을 조절하여 자원관리 기능을 제공한다. RMS 스토리지 공간은 리소스와 컨텐츠 부분으로 디렉토리를 구성하여 해당 파일이 적재된다.



[그림 5] 저장장치 관리 영역

3.3 RMS 에이전트의 API

RMS 에이전트 API set은 기존 디바이스의 운영체제에 종속적인 native API의 특정 부분에 대해서 일부 API를 보완하여 로컬 영역에서 리소스 파일 호출에 대한 리모트 호출 부분을 지원하는 기능을 제공한다.

<표 1> RMS 에이전트 API set

RMS API Set	기능
RMS_LoadResource()	실행 파일이 수행하기 위한 리소스 파일 호출 함수
RMS_SearchLocation()	리소스의 위치를 검색하여 위치 정보 제공
RMS_Link_App()	스토리지에 있는 리소스 파일이 기존의 어플리케이션과 링크됨
RMS_Get_Resource()	RMS_Search_Localtion의 위치 정보에 따른 리소스 핸들 반환
RMS_Process_Type()	리소스의 반환 및 관련 어플리케이션 정보 제공
RMS_VM_Interface()	리소스 호출 후 디바이스에 내장되어 있는 VM과의 호완성 유지

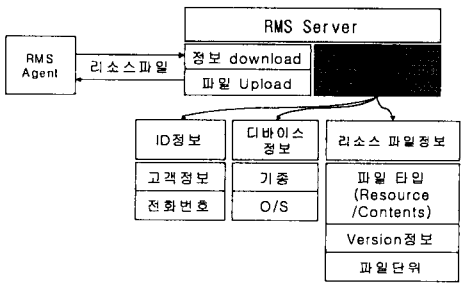
다음은 RMS API Set을 적용한 코드의 일부이다

```
#include "stdafx.h"
#include "RMS_Library.h"

RMS_Create()//RMS 실행을 위한 메인 인터페이스 함수
int RMS_LoadResource(szfileName,
                szfileName,
                                        //생략
                INT rc;
                switch (RMS_Search_Location(szfileName)) {
case IS_Remote_TCP : // Remote 위치일 경우
RMS_Get_Resource(szfileName);
if (Process_Type(szfileName) == IS_PROCESS)
    // 메모리에 Load된 Resource 타입이 Program일 경우
    rc = CreateProcess (szfileName, szCmdLine,
        NULL, NULL, FALSE,
        dwCreationFlags, NULL,
        NULL,NULL, &pi); // 신규 out-process 생성
                                        //생략
   CreateProcess(L_MOUNT_APP, szParameter, NULL,
    NULL, NULL, 0, NULL, NULL, NULL, &pi);
                                        //생략
   break;
case IS_LOCAL_CACHE : // Local Cache 위치일 경우
  break;
case IS_LOCAL_IN : // Local 위치일 경우
   break;
case IS_NON : // 위치 정보 값이 없을 경우
   break;      }
                                        //생략
{
    /* 메인 프로세스에서 test.dll 호출
    lstrcpy (szfileName, TEXT ("test.bmp")); //Image 호출
    lstrcpy (szfileName, TEXT ("test.mp3")); //Sound 호출
    lstrcpy (szfileName, TEXT ("test.wmv")); //movie 호출
   */
   lstrcpy (szCmdLine, TEXT (""));
    dwCreationFlags = 0;
rc = RMS_LoadResource (szfileName, szCmdLine,NULL,
          NULL, FALSE,
      dwCreationFlags, NULL,NULL,NULL, &pi);
          // 메인 프로세스에서 test.dll을 로드하기 위해
          // RMS Agent API의 Resource Load 함수를 호출
            if(rc) {
              CloseHandle (pi.hThread);
              CloseHandle (pi.hProcess);
 }
    return 0;
}
```

3.4 RMS Server의 설계

RMS Server에서는 파일 관리 테이블을 통해서 리소스 파일을 관리하며 RMS 에이전트에서 요청 시 해당 리소스 파일을 업로드 하는 기능을 제공한다. 파일 관리 테이블은 접속 모듈의 ID정보, 모바일 디바이스 정보, 리소스 파일 정보로 영역을 나누어 관리한다.



[그림 6] RMS Server 설계

4. RMS의 구현

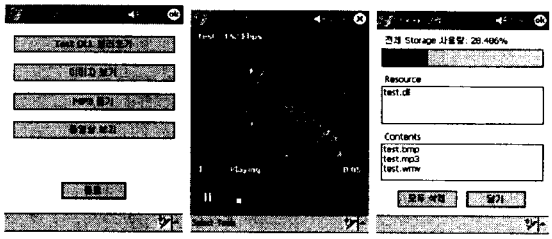
4.1 개발도구 및 구현환경

RMS의 테스트를 위한 PDA(Personal Digital Assistants)의 개발 도구 및 구현 환경은 다음과 같다.

- Language : Embedded Visual C++ 3.0
- PDA : POZ, iPAQ(O/S → PPC2002)

4.2 구현결과

[그림 7]에서 첫번째 화면은 호출을 위한 테스트 실행 화면이고 두번째 화면은 동영상 파일을 구동시킨 결과이다. 마지막 화면은 다운로드 받은 파일들이 스토리지 영역에 존재하고 있는 결과이다. RMS 스토리지 영역의 메모리 공간은 현재 5Mbyte로 책정하였으나 자유롭게 확장할 수 있다.



[그림 7] 구현결과

5. 결론

무선 인터넷은 이동의 편리성을 가지고 있는 대신에 메모리 용량이 협소하여 효과적인 자원 관리를 수행하지 못하는 단점이 있다. 본 논문에서는 이러한 문제점을 해결하고 모바일 저장 공간을 확장시켜 수행되는 플랫폼 구조를 제시하기 위해서 자원 관리시스템을 제안하였다. RMS를 적용한 '모바일 공간확장'으로 데이터 영역을 확장하고 불필요한 파일을 제거하여 내부의 메모리 공간을 최적의 상태로 유지할 수 있다

향후의 연구 계획은 모바일 에이전트의 기능을 다각화하고 서버에서 호출시 로드 밸런싱을 효율적으로 관리할 수 있는 방안에 대해서 연구 할 계획이다.

참고 문헌

[1]James Y. Wilson. Building Powerful Platforms with Windows CE, Aspi Havewala, 2002.
[2]Microsoft, PocketPC Official site, http://www.microsoft.com/mobile/pocketpc/
[3]T. Magedanz, "Intelligent Agent", CACM, Vol.37, No.7. pp.18-21, 1994.
[4]김종완, "PDA용 소프트웨어를 위한 무선 에이전트 클래스의 설계 및 구현", 정보과학회 춘계학술대회 VOL 28. NO 01. PP 0663~0639, 2001
[5]노영선역 Programming Microsoft Windows CE 개정판, 정보문화사, 2002.
[6]안태균외 3인, 포켓 PC와 함께하는 모바일 프로그래밍, 정보게이트, 2002.
[7]이재호, "p에이전트 시스템의 연구 및 개발 동향", 정보과학회지, 제 18권 5호, PP.4-6, 2000.