어플리케이션의 이동성을 기반으로 한 유비쿼터스 시스템

장재형° 이태동 정혜선 유승훈 임중호 최기영 정창성
고려대학교 전자공학과
{jjh3368°, lyadlove, sepia5706, friendyu, jhlim, twoxx195}@snoopy.korea.ac.kr,
sjeong@charlie.korea.ac.kr

MuSa : Mobility-based Ubiquitous System for Application

Jae-Hyung Jang° Tae-Dong Lee, Hae-Sun Jung. Seung-Hun Yoo, Joong-Ho Lime,
Gi-Young Choi, Chang-Sung Jeong
Dept. Electronics Engineering Graduate School, Korea University

요 약

Ubiquitous Computing은 소프트웨어 Architecture를 구현하는데 있어서 많은 도전들을 필요로 한다. 그 중에 하나는 User가 한 컴퓨터에서 다른 컴퓨터로 이동할 경우의 User Mobility를 고려한 것으로 User뿐 아니라 Application도 같이 이동하여 예전의 컴퓨팅 환경을 그대로 복원 하게 되며 User가 작업하던 일을 계속해서 할 수 있게 만드는 것이다. 본 논문은 이러한 Ubiquitous Computing 환경을 만들기 위해 MuSa Architecture를 제안하고, Application 이동에 필요한 Manager들을 설명 하도록 한다.

1. Introduction

User를 중심으로 여러 대의 컴퓨터가 존재하는 환경에서 현재 작업하던 컴퓨터에서의 Application들이 사용자가 다른 컴퓨터로 자리를 옮기면 자동적으로 옮긴 자리의 컴퓨터로 예전에 보여줬던 화면 그대로 보여 지게 만든다면 좀 더 사용자가 편하게 사용할 수 있는 환경이 될 것이다. 본 논문은 이러한 환경을 지원하는 MuSa Architecture를 제안하고, Manager들을 설명 한다.

2장에서는 본 논문과 유사한 내용을 진행 중인 프로젝트를 설명하고, 3장에서는 MuSa Architecure와 각 Manager들에 대해 설명하며, 4장에서는 User 이동시 MuSa의 시나리오를 설명하고, 끝으로 5장에서는 결론과 Future Work에 대해 언급하겠다.

2. Previous Work

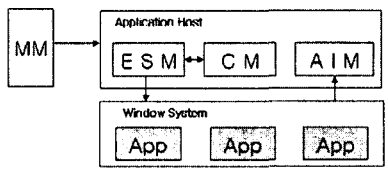
현재 Microsoft에서는 Easy Living[1] 프로젝트를 진행하고 있다. Easy Living 프로젝트는 여러 가지 시나리오를 바탕으로 진행하고 있는데 그 중에서 Automatic Behavior란 주제로 본 논문과 유사한 시나리오를 가지고 진행 중에 있다. 본 논문과의 차이점으로는 사용자가 이동시 전 컴퓨터가 자동적으로 log off되고 사용 할 컴퓨터는 자동으로 log on 되면서 전에 사용한 Application이 보여 지게 되지만, 본 논문에서는 Root로 로그인 한 상태이며, 일반 사용자가 아닌 시스템 관리자인 Root만을 고려하였다.

Illinois 대학의 Digital Computer Lab에서 진행 중인 Gaia[2] 프로젝트는 Ubiquitous Computing을 위한 미들웨어를 개발하고 있다. 사용자는 현재 작업하던 Active Space내에서 다른 Active Space로 이동하는 경우 사용자의 Application이나 데이터들도 사용자를 따라 이동한다는 측면에서 비슷한 점이 있지만, 데이터를 관리하는데 있어 CFS(Context File System)을 사용하였고, MuSa 시스템에서는 NFS을 이용 했다는 점에서 차이가 있다.

Carnegie Mello University의 Computer Science에서 진행 중인 Aura[10] 프로젝트는 각 Manager들 마다 고유한 통신 프로토콜을 만들어 데이터를 교환하게 되며, 실행을 위한 application은 통신이 가능한 Wrapper를 이용해 Application 데이터를 얻어 올 수 있게 하여, 본 논문과 유사한 Application Mobility를 제공한다.

3. MuSa Architecture

MuSa Architecture는 [그림 1]과 같다. Window System에서 실행된 Application들은 Application Host내의 각 Manager들에 의해 Application 정보를 얻고, 조작하게 된다.



[그림 1] MuSa Architecture

AIM(Application information Manager)은 User가 사용 중인 Application들에 대한 정보를 얻어 오고, MM(Mount Manager)는 Device를 소유한 User를 감지하여 원격의 디렉토리를 마운트 하고, Application Host를 호출하게 되며, ESM(Execution/Stop Manager)은

Application의 실행과 종료를 담당한다. 각 Manager에 대한 자세한 내용은 다음에 설명한다.

3.1. AIM (Application Information Manager)

AIM은 Application Host내에 존재하는 Manager 중 하나로써 User가 사용 중인 Application들에 대한 정보를 담은 IF(Information File)를 주기적으로 생성한다. IF파일의 정보는 각 Application 마다 할당되어진 Window ID를 이용하여 각종 속성 값들을 구하고, 그 데이터들은 XML형식을 이용해 저장 했다. Figure1은 IF 저장 데이터를 보여 준다.

```
<MuSa:AppInfo>
  <MuSa:Application>
    <MuSa:AppName>TextEditor</MuSa:AppName>
    <MuSa:upper_left_X>50</MuSa:upper_left_X)
    <MuSa:upper_left_Y>25</MuSa:upper_left_Y>
    <MuSa:Width>300</MuSa:Width>
    <MuSa:Height>100</MuSa:Height>
    <MuSa:Depth>8</MuSa:Depth>
  <MuSa:DataFile>
    <MuSa:Origin>TestFile<MuSa:origin>
    <MuSa:Format>txt</MuSa:Format>
  </MuSa:DataFile>
</MuSa:Application>
<MuSa:Application>
  ...
</MuSa:Application>
</MuSa:AppInfo>
```

Figure 1. IF (Information File) 저장 데이터

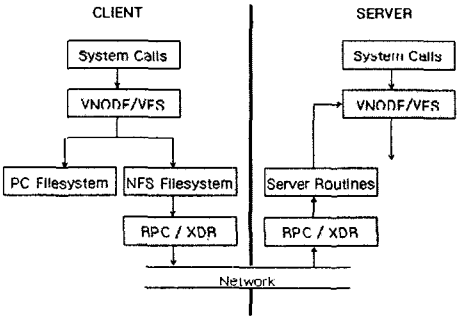
3.2. MM (Mount Manager)

MM은 Device를 인식할 수 있는 Device Reader와 직접 연결이 돼서 Device를 가진 User가 [그림4]와 같이 Computer B로 가까이 오면 인식이 되어져 Device내의 데이터를 가지고 와서 마운트를 하게 된다. 그리고 User가 Device가 인식되지 않는 범위로 벗어나면 MM은 Application Host내에 있는 ESM을 통해 Application을 종료 하고 마운트를 해제 시킨다.

3.2.1 MuSa에서 사용되는 File System

MuSa에서 사용하는 File System은 NFS(Network File System)을 이용한다. NFS은 원격지에 있는 데이터를 공유함으로써 로컬 파일 시스템에 있는 것처럼 사용할 수 있게 만든다. 그림[2]는 NFS의 Interface에 대한 구조적 다이어그램을 보여 주고 있다. 자세한 내용은 [2]를 참조하기 바란다.

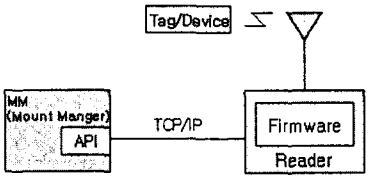
NFS 서버는 마운트를 허용할 디렉토리 정보와 접근 권한을 설정해 주는 파일인 /etc/exports 파일을 읽어 들인다. File Space는 컴퓨터상에 특정 디렉토리를 의미하는 것으로 exports파일에는 File Space 디렉토리가 설정되어 있어서 다른 컴퓨터에서 File Space를 언제든지 마운트 하여 공유 할 수 있도록 되어 있다.



[그림 2] schematic diagram of the filesystem interface and how NFS uses it

3.2.2 Device

MuSa 시스템에서 사용하는 Device는 RF시스템을 기반으로 하고 있어서 Device에 마운트를 위한 데이터를 저장 하고, 그 Device를 가진 사용자가 Reader기 근처에 가면 인식이 되면서 Device에 저장된 데이터를 읽을 수 있는 시스템을 이용하였다. 이러한 device에는 Active Badges, Smart Jewelry, Smart Watches 등이 있다. [그림 3]은 Device System을 나타내고, [Figure 2]는 Device내에 저장된 Data를 보여 준다.



[그림 3] Device System

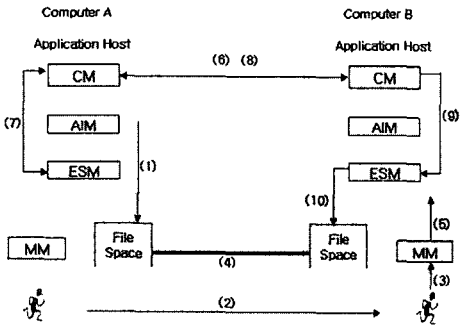
```
<MuSa:Storage>
  <MuSa:Owner>jang</MuSa:Owner>
  <MuSa:Host>rainy.korea.ac.kr</MuSa:Host>
  <MuSa:Path>/Test</MuSa:Path>
</MuSa:Storage>
```

Figure 2. Device에 저장된 정보

3.3. ESM (Execution/Stop Manger)

ESM은 CM의 Application 종료 완료 메시지를 받고 File Space에서 Application에 관한 정보를 담고 있는 IF(Information File)을 읽어서 속성에 맞게 Application을 띄우게 되고, CM의 Application 중지 메시지를 받고 Application을 종료 시킨다. ESM의 중요 역할은 양쪽의 컴퓨터에서 동시에 똑같은 Application을 실행 하지 못하게 하는데 있다. 만약 Application을 중지 시키지 않고 하나의 Data 파일을 두 개의 Application이 동시에 참조하고 있으면 데이터 동기화에 문제가 생기게 되므로 반드시 예전에 있던 Application은 중지 시켜야 한다.

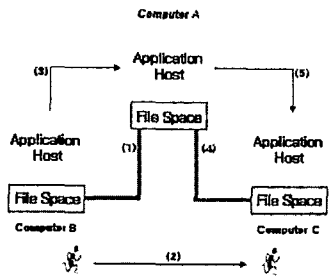
4. MuSa 시나리오



[그림 4] MuSa Architecture

[그림 4]은 MuSa의 시나리오를 나타낸다. User는 작업을 하기 위해 Application을 띄우고, (1)Application에 대한 정보는 AIM(Application Information Manager)에 의해 File Space에 저장 되게 된다. (2)User는 File Space를 마운트 하기 위한 정보를 가진 Device를 가지고 Computer A에서 Computer B로 이동한다. (3)Device에 저장된 정보는 MM(Mount Manager)에 의해 마운트 되어지는데 사용 된다. (4)마운트가 완료되면 Computer A에 있는 File Space가 Computer B에 있는 File Space로 마운트 되어 짐으로써 File Space가 공유가 되어 진다. (5)MM는 Application Host를 호출하게 된다. (6)호출 되어진 Application Host는 CM(Communication Manager)를 통해 Computer A로 현재 작업 중인 Application을 중지하라는 메시지를 보낸다. (7)Computer A의 CM은 ESM(Execution/Stop Manager)를 통해 현재 작업 중인 Application을 중지시키고 완료 메시지를 보낸다. (8) CM을 통해 완료 메시지를 받는다. Computer B의 CM은 ESM을 통해 File Space에 있는 Application과 데이터들을 읽어 Computer A에서 보여 주었던 Computing환경 그대로 Application을 보여주게 된다.

[그림 4]은 두 대의 Computer만을 고려하였다. [그림 5]는 User가 Computer B에서 Computer C로 이동할 경우의 시나리오를 그린 것이다.



[그림 5] Computer B에서 Computer C로의 이동 과정

(1)User는 Computer B를 사용하고 있다. (2)User가 Computer C로 이동할 경우 (3)Computer B는 Application을 중지시키고 마운트를 해제 하고, Application은 Computer A로 이동하게 된다. (4)Computer C에서는 User를 인식해서 [그림4]과 같이 마운트를 하게 되고, Application은 Computer C로 이동하게 된다.

5. Conclusion and Future work

미래의 컴퓨팅 환경은 더욱 더 사용자 중심의 환경으로 변할 것이고, 현재도 Ubiquitous 환경을 만들기 위해 많은 프로젝트들이 진행 중이다. MuSa 시스템은 그 중에 RF 시스템과 유선 네트워크 시스템을 통합해 좀 더 사용자 중심의 컴퓨팅 환경을 만들기 위한 것이다. MuSa 시스템은 현재 간단한 Application의 이동만을 고려하였다. 후에는 Configuration이 필요한 복잡한 Application까지 이동이 가능하게 만들 것이다.

7. Reference

[1] Manuel Roman, "Gaia: A Middleware Infrastructure to Enable Active Spaces", Revised Paper #20, 7/1/2002

[2] Russel Sandberg, "The Sun Network Filesystem: Design, Implementation and Experience", Sun Microsystems, 1986

[3] Christopher K.Hess, Roy H. Campbell, "A Context File System for Ubiquitous Computing Environments", University of Illinois, July, 2002

[4] David k, "Semantic File System", Programming Systems Research Group, 1988

[5] Ichiro Satoh, "Physical Mobility and Logical Mobility in Ubiquitous Computing Environments", Japan *Science and Technology Corporation*, LNCS 2535, pp.186-201, 2002

[6] Kari Kangas, "Using Code Mobility to Create Ubiquitous and Active Augmented Reality in Mobile Computing", University of Oulu, MOBICOM'99, August, 1999

[7] Brain Pawlowski, "The NFS Version 4 Protocol", 2000

[8] Jalal Al-Muhtadi, "A Flexible, Privacy-Perserving Authentication Framework for Ubiquitous Computing Environments", University of Illinois, ICDCSW'02, IEEE, 2002

[9] Steven A. N. Shafer, "Ubiquitous Computing and the EasyLiving Project", Microsoft Corporation, 2001

[10] Joao Pedro Sousa and David Garlan, "Aura: An Architectural Framework for User Mobility in Ubiquitous Computing Environments", Carnegie Mellon University, 2002