u-PC: Personal Workspace on a Portable Storage

Injung Kim*, Min Kyung Hwang, Woojoong Lee, Chanik Park

Department of CSE/GSIT*
Pohang University of Science and Technology,
San 31, Hyoja-dong, Pohang, Kyungbuk, Republic of Korea
{pypupipo, hmk0119, wjlee, cipark} @ postech.ac.kr

ABSTRACT

In these days, people easily meet a public computer at any place. For instance, when visiting someone's office to take part in a meeting, a user usually faces an unfamiliar workspace on the computers. Even if the user has a Microsoft PowerPoint file on the USB flash device, it is common that the user can not open the file because of different versions or absence of appropriate applications. In order to avoid this situation, people carry on their laptop or try to access e-mail clients.

In this paper, we propose a framework called ubiquitous personal computing environment (u-PC) that supports mobility of personal workspace within any portable storage. This framework is specially designed for fast and light-weight switching between carried workspace on the portable storage and native one on the public computers. For supporting the application mobility, we present a mechanism performed by filtering file and registry I/Os in order to extract platform-dependent resources such as system libraries, application settings and registry information, during application installation process. Then, the filtered resources are stored to the portable storage so that the framework can provide the resources for applications by forwarding I/O requests which enable to execute these applications independently when being invoked and requesting the filtered resources. Additionally, the framework includes the module supporting the wireless portable storages that is designed by using iSCSI and UPnP standards for automatic detection and initialization.

Finally, we implement a prototype of the framework to prove our concept and to show adaptability in real-world conditions. Some limitations and future works are also presented.

Categories and Subject Descriptors

D.4.3 [**Operating Systems**]: File Systems Management – *Access methods.*

D.4.4 [**Operating Systems**]: Communications Management – *Input/output.*

General Terms

Application, User Profile, User Data, Portable Storage, iSCSI, UPnP, File I/O, Registry, Windows Kernel.

Keywords

Personal Workspace, Application Mobility, File and Registry Hooking, File Forwarding.

1. INTRODUCTION

In the emerging ubiquitous computing, people want to use their customized personal computing environment at any time and any place. Let us consider a Windows operating system based desktop which is used every day by a user. Once the Windows OS boots up, it loads initial processes and brings up configuration related to the user who has logged in [3]. For the user, Windows OS constitutes the personal computing environment including desktop settings, application settings, internet settings, and personal data. We define it as personal workspace in this paper.

The user may require certain files or own personal workspace from public computers while being away from own computer. Although it is possible to carry the data files by using USB flash devices (UFD), e-mail clients, or FTP programs, it is difficult to open such files on other computers due to different versions or absence of appropriate applications.

Therefore, there is a strong necessity for a framework which provides mobility of personal workspace. However, it is not simple to move the applications or user profile because they are closely correlated to the Windows OS. Unlike Linux, application installation requires not only uncompressing of application binary files but also configuring and registering of the application in the Windows OS. Windows system has a special mechanism to maintain application configuration and system configuration, that is, registry. It is difficult to control registry since the user can not directly monitor it in the user mode. Also, some host dependant files created during application installation need to be set up on the special location like system root. Therefore, our technique presents mobility of applications and user profile because the user may not know where they are placed or how they are related with the system.

We also present a framework for detecting a wireless portable device and loading u-PC software automatically. By using iSCSI and UPnP standards, the wireless portable storage is detected by

Table 1. Comparison with similar systems to u-PC

	U3	MojoPac	Ceedo	u-PC
Pros	- Open standard based (U3 middleware platform, a specialized H/W specification) - Security policy	- Application mobility supports for Windows - Supports any type of UFD - Virtual workspace like a native Windows OS - Security policy	- Application mobility supports for Windows - Supports any type of UFD	- Personal workspace mobility supports for Windows - Supports any type of portable storage
Cons	- Applications must be modified - Requires specially designed H/W	- Based on UFD - Relatively long initialization and setup time	- Based on UFD - Mainly supports non-commercial applications	- A few applications supported currently - Requires host PC set up UPnP & iSCSI client

any public computer and starts to load u-PC software package. Then, the user can work in the personal workspace without extra application installation or setting up the connection.

There are similar systems such as Thin-clients, SoulPad [6], DeskPod [7], U3 [9], MojoPac [4], and Ceedo [1]. Each of them supports relatively more applications, security policy, or clear division of personal workspace from the host PC than the u-PC framework. However, they have some disadvantages of requiring high quality of network latency, having to reboot the system, supporting only Linux environment, needing a special hardware, or having long setup time. Therefore, we propose a system, called u-PC, which supports mobility of personal workspace simply and easily.

The user can easily access many types of flash memory devices to carry personal workspace, for example, USB hard drives, iPods, and MP3 players. It needs enough space to store applications and user data, but minimum capacity for using u-PC is only about 30MB. Since the portable storage devices tend to be cheaper and have faster access, it is reasonable to use them for moving applications and user profile.

u-PC can be represented by two strengths. First, it supports mobility of various types of applications, including editor, image viewer, media player, FTP server, and compression applications. We have tested some applications such as Hangul, Acrobat Reader, Edit Plus, GhostScript, GVim, Winamp, Adrenaline, and so on. Also, we plan to extend the number of applications. As for executing u-PC applications on some host PC, the user does not need to worry about conflicts of the same applications which are already installed in the host PC, since u-PC registers the registry information of applications into the u-PC registry hierarchy not the original one.

Second, mobility of user profile makes a user feel comfortable when starting the u-PC software package on any host PC since visible environment is almost the same as the user's desktop. This user profile can be desktop surroundings, internet surroundings and application data. Mobility of user profile aims at application attributes as well as familiar computing environment such as a wallpaper, a first page of web browser, bookmarks, and so on.

Generally, portable storage is used for carrying user data files. u-PC, however, serves mobility of applications as well as user profile and data files within any portable storage device. It affects the development of portable storage device industries by providing a new kind of services.

The remainder of this paper is organized as follows. In Section 2, we describe related work. Section 3 discusses the u-PC architecture, and Section 4 explains how the u-PC device is detected and u-PC software package loads into the host PC. Section 5 presents how u-PC supports the application mobility. In Section 6 we present the technique of how user profile could be mobile. Finally, Section 7 concludes this paper and highlights ideas for future work.

2. RELATED WORK

Due to decreasing cost of the hardware, increasing computer infrastructure and trade-off among performance, weight, and size, recent research tends to focus on the software-approach. It is more efficient to move data with portable storage such as USB drives instead of laptop or sub-laptop computers. To an extent, recent research in software-approach to support user's mobile computing environment is composed of two approaches, network-based approach and storage-based approach.

Thin-client systems [5, 8], which are the representative network-based methods, require network access for the central server infrastructure. Such systems put all applications and state information to the server, and then logical executions run on the server, that is, server-side computing. Also, remote desktop of Windows system and XTerminal of UNIX system are a type of network-based methods. To achieve a better performance of the network-based approach, reliable network environment such as high-bandwidth and low latency is essential.

Another approach, storage-based approach, is self-contained and runs on any local host. Some environments such as SoulPad [6] and DeskPod [7] achieve these features by providing a virtualization mechanism. However, SoulPad needs to reboot and configures for applying virtual machine monitors (VMMs) into the host, and DeskPod only works on the Linux OS even though many users are familiar with Windows OS.

Other environments such as U3 [9], MojoPac [4], and Ceedo [1] are very similar to u-PC as shown in Table 1. They all support application mobility for Windows and use a sort of portable storage. U3 platform provides security policies and an open standard based platform which is the U3 middleware platform, but requires every application to be modified or programmed by U3

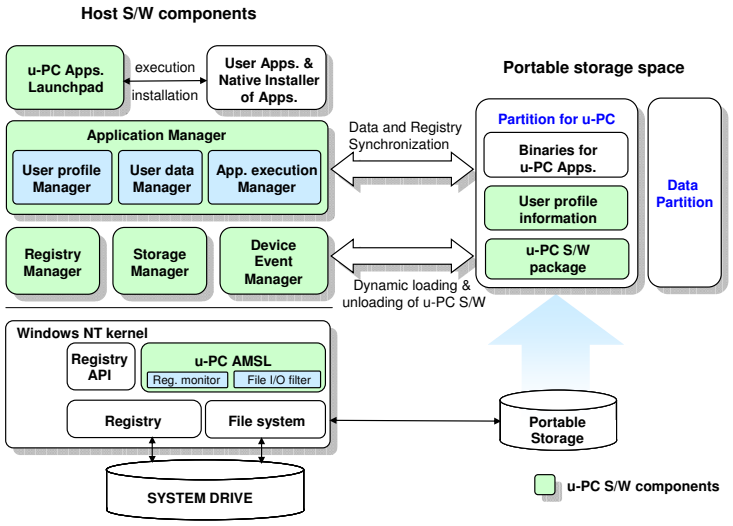

Figure 1. u-PC architecture

specification, and using a specially designed U3-compatible hardware is also essential. On the other hand, MojoPac proposes virtual workspace like a native Windows OS, and supports security policies. This system, however, has relatively long initialization and setup time due to the change from a native Windows OS to the virtual workspace. At last, Ceedo is in the process of development. Therefore, it mainly supports non-commercial applications which are opened sources. In comparison with u-PC, these systems offer a number of applications, and support the security policies to counter the loss of devices.

Many other systems have been proposed to support application mobility, but they need to change an application itself, require a specific hardware, take long startup time, and influence the system settings on the host system. u-PC has been built from previous work by [2], a system only for the application mobility in Windows 2000. In this paper, we show the developed and extended u-PC framework on several aspects which are provided by working in Windows XP as well as Windows 2000, creating mobility of ubiquitous personal computing environment that includes user profile, user data, and applications, and supporting more applications than before.

3. u-PC ARCHITECTURE

u-PC runs on either Windows 2000 or XP with any portable storage device such as a USB flash or a hard drive. The overall architecture of the system is depicted in Figure 1. The left side on the figure describes how the host PC is composed once the u-PC device is detected and the u-PC software components are automatically loaded. The other side shows which data is stored in the u-PC device.

Once the u-PC software package is installed onto the device, a user is able to save own user profile information and applications that the user wants to retrieve from the desktop to the hand. For example, the user profile can be a wallpaper, a first page of web browser, and bookmarks. Since most visible environment is similar to the user's desktop, user profile mobility can make the user feel comfortable whenever the u-PC software package is executed by UPnP and iSCSI standards which are described in Section 4. Along with the user profile, applications could be stored into the u-PC device by using the u-PC software package. Also, the user can save one's data in the data partition of portable storage space.

The u-PC device detected by UPnP automatically loads the u-PC software package into the host PC by the iSCSI server and client. Once the u-PC software package is dynamically loaded, the device event manager composes the u-PC software package that consists of u-PC application launchpad, application manager, registry manager, and storage manager.

When the device event manager shows the u-PC application launchpad to the user, the user could select any action of u-PC by using the u-PC application launchpad, a simple user interface. The actions include applying user profile, accessing user data, installing applications, executing applications, and removing applications. When the user selects an action, the u-PC application launchpad sends a request to the application manager.

The application manager has three different managers which are core components of u-PC, and they are user profile manager, user data manager, and application execution manager. Each manager synchronizes data files and registry whenever they are modified, and supports mobility of user profile, user data, and applications. Also, each manager sends respective actions to the lower managers, the registry manager and the storage manager. The application manager supports these managers by registering the u-PC application mobility support layer (u-PC AMSL) to the Windows NT kernel.

First of all, for mobility of user profile, the user profile manager backs up the user profile of the host PC by executing the registry manager that uses registry API to extract the host PC's user profile information from the system drive to the u-PC device. Then, it applies user profile of u-PC into the other host PC. At that time,

the user can see the changed wallpaper, user's bookmarks on the web browser, and so on. The user profile manager also goes back to the original user profile of the host PC when device event manager detects that u-PC device is not plugged anymore.

Second, for mobility of user data, the user data manager allows the user to use data easily from the shortcut of data partition in the u-PC device. The user can access it from the user interface of the u-PC, that is, the u-PC launchpad.

Lastly, for mobility of desktop applications like Acrobat reader, Photoshop, and Winamp, the application execution manager provides two functions, the application installation and the application execution. Both functions make applications portable and execute them in the host environment without any failure of file I/Os.

In the Windows kernel level, the u-PC AMSL works with registry API, registry, and file system. In fact, Windows system supports the layered kernel mode drivers so that I/O request packets (IRP) can pass several drivers when handling the I/O requests. It means that it is possible to add a driver under or above the function driver, called a filter driver. The filter driver filters the I/O requests in order to add or modify these behaviors. In this paper, we have produced the personal workspace mobility by implementing the u-PC AMSL, which includes two filter drivers, registry monitor and file I/O filter.

The registry monitor is for monitoring when registry keys and values are created, deleted, or modified. After the observation, it records the changes of registry, and returns the information to the registry manager by using registry API. The registry API is already defined as how to access the system registry.

On the other hand, file I/O filter allows the storage manager to access the file system, and if the user wants to execute the applications, access the user profile, or obtain user's data within the u-PC device, the file I/O filter makes them appear to exist in the host system.

Finally, when the u-PC device is plugged out, Windows system broadcasts a message called WM_DEVICECHANGE to the entire system whenever the status of portable storage is changed. At that time, the device event manager catches the massage carefully and judges whether the device is composed of u-PC or not. When the device event manager confirms that it is the u-PC device, it starts to clean the host PC just like the environment before using u-PC.

4. u-PC WITH UPnP/iSCSI STANDARDS

For the automatic operations of detecting a u-PC device and loading a u-PC software package, u-PC is designed with UPnP/iSCSI standards. We assume that a host PC has already set up the UPnP and iSCSI client and is running them. Also, the u-PC device is assumed to have set up the iSCSI server.

First, to detect the u-PC device, the host PC needs a special protocol, UPnP. The UPnP technique, which represents universal plug and play, is designed by Microsoft for the independent and unified network environment. It enables to work as peer-to-peer networking instead of central concentrated networking, and executes on the different operating systems or platforms due to using standard network techniques such as IP or HTTP. Devices using the UPnP technique could have several operations which are addressing, discovery, description, control, eventing, and presentation. These six operations exist for the universal plug and play. To detect the u-PC device, we especially use the discovery operations which enable to detect any portable device without central concentrated database.

Second, to load the u-PC software package, the host PC is required to be set up the iSCSI client, and the u-PC device has to be set up the iSCSI server. Once, the host PC detects the u-PC device, it starts to load the u-PC software package by using the iSCSI standards. The iSCSI technique, which means internet SCSI or SCSI over IP, is a block-oriented storage access protocol that enables a user to recognize a remote storage as their own local block device through general TCP/IP networks. Since the iSCSI uses the standard Ethernet switch and router for this kind of access, it can not only be applied to Ethernet technologies, but can also be used to create a storage networking system without any distance restrictions that can equally be applied to wireless network environment. Accordingly, focusing on this applicability, this paper presents automatically loading environment to consist of personal workspace from the u-PC to the host PC.

5. APPLICATION MOBILITY

In this section, we describe how u-PC supports the application mobility. For the application mobility, we have to consider how u-PC collects the information created during application installation and how it works on the host PC without any conflict or error. The former is described in Section 5.1, while the later is presented in Section 5.2. The following two sub-sections present the method of supporting the application installation and execution with detailed explanation of u-PC AMSL which we have previously shown in the architecture of u-PC in Figure 1.

5.1 Supports for the Application Installation

The Windows system uses a special type of database called registry for keeping application settings as well as the entire system. In this part, for carrying the entire application files and registry to the portable storage, we hook the file and registry requests during application installation processes. In Figure 1, the AMSL component is in charge of this hooking process and consists two parts of kernel mode driver, the file I/O filter and the registry monitor.

The former, the file I/O filter, filters any file requests created during the application installation and stores file information such as a file path, a file name, and so on. The latter is the registry monitor that monitors the changes of registry, and then records detailed information such as registry values and registry keys. The registry monitor handles registry actions with the Windows registry API which is defined of creating, changing, and deleting registry.

When the application installation process is over, the application execution manager sends all of the created files and registry from the system drive to the u-PC device by using the recorded information. Finally, u-PC automatically removes all of these files and registry from the system drive.

5.2 Supports for the Application Execution

u-PC makes the host PC to think that it can execute applications encapsulated within the portable storage device as shown in Section 5.1. It also makes an illusion and makes the user feels as if the applications were installed on the system drive. To support the

Table 2. Install time of three applications on the three different cases of devices, the USB device with u-PC, the USB device without u-PC, and the system derive.

Application (size)	The USB device with u-PC	The USB device	The system drive
Winamp 5.23 (19.2MB)	12 (min)	10 (min)	1 (min)
Acrobat reader 7.0 (42.9MB)	10 (min)	9 (min)	1 (min)
Hangul 2005 (288MB)	43 (min)	37 (min)	2 (min)

application execution, the u-PC AMSL works inside the Windows kernel in order to filter file and registry requests from u-PC applications.

The registry monitor enables to update the registry dynamically which has been already registered by the registry manager when the user starts to execute applications in the u-PC launchpad. To monitor the changes of the registry, the registry monitor loads a device driver which hooks the system calls on the Windows NT kernel. When a user-mode process issues a privileged system call, its control is transferred to the software interrupt handler, and it takes a system call number which indexes into a system service table to find the address of the NT function to handle the requests.

Before monitoring, the register monitor replaces those entries in the table with pointers to filtering functions. Therefore, it is possible to monitor the NT system services because the registry monitor modifies the value of pointer on the function which corresponds to the system call of modifying registry. Once the pointer of adequate function is changed and the registry monitor starts to monitor, the registry monitor updates the changed registry from the system drive to the u-PC device whenever the applications try to modify one's registry.

Next, the file I/O filter plays an important role in the changed file requests from the system disk to the portable device in order to run applications which are encapsulated on the u-PC device. It monitors every file requests inside the Windows NT kernel, catches the file requests for the u-PC applications, and finally changes the file requests toward the u-PC device where corresponding files are placed. This occurs since the application installation processes store all the files into the system drive. Therefore, although the applications point the system drive, the file I/O filter changes the requests to the portable storage and returns successful responses of the file requests.

In conclusion, these routines make the u-PC applications work as if they were installed on the system drive, and the applications encapsulated on the u-PC device can work well without any failure of the file I/O requests.

6. USER PROFILE MOBILITY

In this paper, we define the user profile for supporting carriable personal workspace, which is composed of the following three parts:

6.1 Application Configuration and Desktop Environment

The mobile user wishes to execute applications consistently on any host PC. For example, when a user manipulates some u-PC applications, one can change the application settings such as closing drawing tool bar and opening the table tool bar. Once the user terminates to use the applications, moves around with the u-PC device, and starts the applications again on the other host PC, one may want to see the same status of the applications with the table tool bar and no drawing tool bar. u-PC supports these attributes by dynamically updating whenever the application's registry is modified.

Also, to supply much more familiar desktop environment, u-PC user profile considers desktop environment such as a wallpaper, Windows theme, attributes of folder, and so on.

6.2 Internet Environment

The u-PC user profile regards the internet environment as a necessity to be mobile since most computing users use the web browser. Therefore, u-PC carries the user's favorite bookmarks, recently opened web pages, cookies and so on.

6.3 Application Data

When a user is working on the Microsoft Outlook, the application stores the received and sent e-mails as a data file. If the Microsoft Outlook has no data when the user moves with u-PC device, and runs it on the other host PC, then the user feels that it is useless. We call it, application data, and the u-PC user profile framework carries it on the u-PC device.

As a result, the u-PC user profile maintains the application configuration, the desktop environment, the internet environment, and the application data. It also supports dynamic updates by the registry monitor, as we have mentioned in Section 5.

Another remaining aspect to consider about the u-PC user profile is the entire sequence. As a matter of fact, the u-PC user profile does not register as soon as the u-PC device is detected. At first, the u-PC registry manager has to backup the current host's user profile, and then register that of the u-PC. For the backup of the host's user profile, the u-PC registry manager makes the registry hierarchy of u-PC information. Soon after, when the u-PC device is ejected physically, the device detect manager starts to clean up the host PC as same as before starting u-PC. To clean the host PC, the device detect manager accesses and uses the back-up information in the u-PC registry hierarchy.

7. CONCLUSIONS AND FUTURE WORK

In this paper, we presented a framework called u-PC which guarantees the portability of a personal workspace that includes applications, user profile, and data. The key features of the framework are not only filtering and logging the file I/Os but also monitoring registry changes in order to obtain some platform-dependent installation information during the installation process

made by application-specific installers. Then, it stores the information to a portable storage playing a role in a u-PC device.

When the u-PC device is connected to another machine and encapsulated applications are invoked, the information is used to forward the file requests from the system drive to the portable storage in case of requesting the filtered files.

u-PC will be improved in terms of installation time of applications into the u-PC device. In fact, compared to the general time of application installation into the system drives, the application installation into the u-PC device takes much more time due to write latency of the portable storage device. To verify this, we conducted some tests using LG IBM laptop with Mobile Intel Pentium 4-M CPU 1.90GHz, 1GB RAM, Windows XP SP2, and 1GB USB device with 2.0 bandwidth. Moreover in Table 2, we suggested that u-PC has a pre-installation progress that does not install into the portable storage device but copy application packages which are already installed by using the u-PC framework. As another test of applications installed to the system drive, the time to copy Winamp from the system drive to the USB device took about 2 or 3 minutes, and other applications, Acrobat Reader and Hangul, took only 3 or 4 minutes.

We hope that u-PC can support much more applications and user profile, which will be a part of our future work. For the applications, we need to research on not only monitoring file I/O requests but also process migration. Furthermore, the user profile needs to be considered on another approach with no latency of setup time such as a virtual workspace on MojoPac. In the future, we will extend the u-PC environment to become more flexible and feasible.

8. REFERENCES

[1] Ceedo, *http://www.ceedo.com*

[2] Kyonghoon, L. *A Technique for Application Programs*

Mobility Using Portable Storage in Windows Operating System, Master Thesis, Pohang University of Science and Technology, Pohang, Republic of Korea, 2005

[3] Mark, E. R. and David, A. S.*, Microsoft Windows Internals, Fourth Edition: Microsoft Windows Server 2003, Windows XP, and Windows 2000,* Microsoft Press, Redmond, Washington, 2005.

[4] MojoPac, *http://www.mojopac.com*

[5] Ricardo, A. B., Shaya, P., Gong, S., and Jason, N. MobiDesk: Mobile Virtual Desktop Computing. *In Proceedings of the Tenth Annual ACM International Conference on Mobile Computing and Networking (MobiCom 2004)*, Philadelphia, PA, 2004, 1-15.

[6] Ramón, C., Casey, C., Chandra, N., and Mandayam, R. Reincarnating PCs with Portable SoulPads. *In MobiSys 05: The Third International Conference on Mobile Systems, applications, and Services*, Seattle, WA, 2005.

[7] Shaya, P. and Jason, N. Highly Reliable Mobile desktop computing in Your Pocket. *In Proceedings of the 30th Annual International Computer Software and Applications Conference (COMPSAC 2006)*, Chicago, IL, 2006, 247-254.

[8] Tristan, R., Quentin, S., Kenneth, R. W., and Andy, H. Virtual Network Computing. *IEEE Internet Computing*, 1998, 33-38.

[9] U3 Platform, *http://www.u3.com*