Towards Simplicity: an introduction

Maomao Wu, Oliver Storz, Nigel Davies, and Adrian Friday

Computing Department, Infolab21, South Drive,

Lancaster University, Lancaster, UK, LA1 4WA

{maomao, oliver, nigel, adrian}@comp.lancs.ac.uk

ABSTRACT

With the rapid development of mobile devices, wired and wireless communication technologies, and network based services, people are able to access information ubiquitously using a variety of devices. At the same time, users are often overwhelmed by the complexity of interacting with different sorts of devices and services available to them. Moreover, the lack of agreements and standardisation complicates the design and deployment of adaptive services by service providers and network operators. This excessive complexity prevents novel information technologies from being quickly accepted by end users and widely adopted by service providers.

In this paper we argue that a flexible and extensible framework is required to extricate different parties from the burden of this complexity. The European Union funded Simplicity research project introduces the notion of a Simplicity Device to end users. Simplicity Devices allow them to store personal profiles and preferences in a secure way and enable them to customise devices and services with minimal effort. Simplicity offers a flexible and extensible Brokerage Architecture to service providers and network operators, facilitating unified treatment of device personalisation and service adaptation by employing policy based decision mechanism. This simple framework can be used by third party application components to acquire the necessary information for configuring and adapting themselves. Moreover, Simplicity is able to provide support for heterogeneous platforms on a wide range of devices, e.g., PCs, PDAs, mobile phones, MP3 players, TV sets and smart spaces.

I. INTRODUCTION

Recent years have witnessed the rapid proliferation of mobile devices, wired and wireless communication technologies, and network based services. Most of us already own a considerable number of personal computing devices, such as PCs, laptops, PDAs and mobile phones, and enjoy access to a growing number of services. The abundance of devices and services aim to simplify our lives by providing us with ubiquitous information access. However, inconsistencies in the ways we interact with different devices and services often tend to complicate things unnecessarily. More than often, users are forced to manually edit network settings on their laptop when trying to access different wireless networks. On each machine they are using, users have to manually change the settings of desktop and applications to acquire a familiar look-and-feel; more than often they are required to remember and type in a variety of usernames and passwords to get access to the different online services they use, such as online banking, shopping and auctions. On the other hand, the lack of agreements and standardisation also complicates the jobs for service providers and network operators. Heterogeneous networks, devices, platforms and applications make it very challenging to provide tailored information access to a wide range of users in a unified manner.

This excessive complexity prevents novel information technologies from being quickly accepted by end users and widely adopted by service providers. The Simplicity [1] project was proposed to extricate those different parties from the burden of this complexity. The project introduces the concept of Simplicity Devices (SDs). Simplicity Devices enable users to customize different devices and services (tailored to different environments) with minimal effort. For service providers and network operators, Simplicity offers a unified brokerage framework to support various types of adaptation by taking into account the network capabilities, service availabilities, device characteristics, and user preferences. By offering a simple way of acquiring information required for configuration and adaptation through this unified framework, Simplicity provides support for third party applications both on PCs and on other information and communication technology (ICT) devices such as phones and PDAs, and even on non-ICT-related platforms, such as MP3 players, TV sets and smart spaces. The expected extensive adoption of the Simplicity Framework will make the SD universally accepted over a wide range of information appliances.

The rest of the paper is organised as follows: In section two we provide a brief survey of related work in the field of device and service personalisation; we describe the key contributions of the Simplicity project from end users' and service providers' perspectives in section three. In section four, we analyse the benefits and drawbacks of the different approaches, and finish with concluding remarks in section five.

II. Related Work

Besides Simplicity, a small number of projects have identified the need for automatic configuration and adaptation of devices and services. Approaches range from remote machine access to the automatic personalisation of selected services and devices.

Virtual Network Computing (VNC) [2] provides users with access to personal data and applications on a remote machine by using a thin client application. The user just needs the account information (e.g., username and password) to get access to the remote machine. The underlying technology is a low-level remote display protocol that enables a local machine to render the graphical user interfaces of applications running on the remote machine. While user inputs are transferred from the thin client on the local machine to remote applications, all application states are maintained by the applications themselves on the remote server. Users are able to walk away from one machine and resume working on what has been previous left from another machine. VNC does not actually address the problem of device personalisation, since it does not physically change any settings on the device that the user is directly interacting. It simply redirects input and output to and from the applications running on the remote machine. Moreover, this approach requires good and continuous network connectivity between client and server in order to get the graphics rendered with a reasonable performance.

Analogous to suspending and resuming a session on a laptop, Internet Suspend/Resume (ISR) [4] allows users to suspend their desktop on one machine and resume the session on another without having to carry any physical device. ISR requires host machines to have network connectivity (including access to a central share on a distributed file system, such as Coda [6]). In addition, software providing a virtual machine (VM) abstraction (e.g. VMware [5]) is expected to be present on each host. The user always works using the operating system (OS) that is executed within the VM, called guest OS. When he suspends a session, the complete volatile state of the VM can be captured by a Virtual Machine Monitor (VMM) [7] and stored within a file (called VM image) on a remote file server. This VM image is then copied across the network to the local file system before resuming a session on another machine. Several techniques have been employed to optimise the performance of this process, especially the delay encountered while transferring images across the network. Proposed solutions to speed up the resume procedure by reducing the amount of data that needs to be transferred across the network include compression of the VM image and look-aside caching [8].

Migo [9], a commercial product, supports the personalisation of applications on PCs by keeping users' personal profiles and files related to these applications on portable USB storage devices, e.g. USB smart watches, USB flash memory, or even iPods. At the moment, Migo supports the personalisation of applications only on Windows-based platforms and is limited to work with a few popular Windows applications, including Windows Desktop, Microsoft Outlook and Internet Explorer. Migo approaches personalisation on a per-application basis, i.e. by directly personalising applications one-by-one. Since each application might have different ways of managing personal profiles and user data, the designers of Migo are required to understand the internals of every single application. These dependencies might prevent Migo from scaling to a larger number of applications across a variety of operating systems.

While the developers of ISR more or less assume ubiquitous availability of high-bandwidth networking facilities, different assumptions have lead to the development of another approach. The Personal Server [3] by Intel research is a portable device that offers storage, computing and communication capabilities. Envisioning the ubiquity of computing devices with rich user interfaces, the authors of [3] believe that the Personal Server requires neither display nor input devices. Instead, their concept builds on the ability to detect the presence of I/O devices nearby and use them. These I/O devices would then provide conventional user interfaces to access the information stored on the Personal Server. Information access is seen to happen via short-range wireless communication technologies, such as Bluetooth. Personal Servers still have processing power. They are powered by the Linux OS and are able to host software stacks for many different purposes, e.g., communication, discovery or security. Although Personal Servers could be used by users to store their profiles and data files, they do not natively provide any means for personalising devices or services that users are interacting with. Personal Servers are battery powered and are expected to be always on.

In contrast to ISR, which completely frees users from having to carry any device, IBM research proposes that users should be able to carry everything with them on devices called SoulPads [10] [11], in order to reduce infrastructure requirements to a minimum. A SoulPad is a regular USB2 portable disk with a complete software stack on it, including an auto-configuring host OS (Knoppix) and a suspended VM image. By plugging a SoulPad into the USB port of an arbitrary PC and booting from the SoulPad, users are able to resume their personalised computing environments on arbitrary computers (called EnviroPCs). Just before a user finishes using an EnviroPC, the state of the session will be suspended and stored on the user's SoulPad so that changes made by the user can be maintained. The SoulPad is a passive storage device that requires no battery. Therefore users do not have to worry about the battery life and recharging. The personalisation procedure requires no network connectivity and does not require EnviroPCs to have any specific software installed or preloaded. The relatively low level of infrastructure requirements makes this approach acceptable to a wide range of conditions, and the developers believe it to be especially useful for developing countries where Internet connection is not widely accessible.

III. SIMPLICITY

Simplicity ("Secure, Internet-able, Mobile Platforms LeadIng Citizens Towards simplicitY") is a European Union funded research project involving industrial and academic partners from Austria, Finland, Germany, Greece, Italy, and the United Kingdom. The aim of the project is to simplify the way that the end users interact with different devices and services by providing means for convenient personalisation, and to provide a simple and unified architecture for adapting services based on network and device capabilities as well as user preferences.

A. End User Experience

From an end user's perspective, the world with Simplicity is expected to look as follows (the Simplicity website provides detailed descriptions of user scenarios and business models [1]):

Bob is a business manager who enjoys using different types of personal devices, including a mobile phone, an iPod, a PDA, a laptop, and PCs at home and in his office. To simplify interactions with different services and devices, he adopted the solution offered by Simplicity. Bob now always takes his Simplicity Device (SD) with him, which acts as his key to the Simplicity-enabled world. The SD stores Bob's personal profiles and optionally data that he'd like to use to personalise the devices and services that he encounters. Each time Bob plugs (not necessarily physically, because "connection" could be wireless) the SD into one of his personal devices, the existence of the SD is automatically discovered by a software called Simplicity Personal Assistant (SPA) on the device. The SPA displays a user interface asking Bob to grant the device the right to access the SD. Bob has to authenticate himself, e.g. by entering username and password. After successful authentication, the device is authorised to access to profiles and data on the SD and automatically starts the personalisation process. For example, Bob's home PC can get the same appearance of the working environment of the office PC by automatically configuring itself using Bob's office PC profile and copying personal data from Bob's SD. The automatic personalisation takes into account the differences in device and network capabilities in different environments. For example, the monitor of Bob's home PC offers a much lower resolution than Bob's office PC. As a result, the display resolution finally configured on the home PC will not be exactly the same as that of the office PC. For networking, a DSL connection is used at home, whereas Bob's office PC is configured to use the office's Local Area Network (LAN).

Simplicity-enabled devices do not only exist in domestic environments that Bob is familiar with, but are also common in other places, such as brother companies and public spaces. While visiting a brother company, Bob will be able to use his SD to personalise the guest PC in the company. For example, the email client on a guest PC will be configured in exactly the same way as the one on Bob's office PC, with the same accounts being set up and same folder structures being presented. In public places such as train stations, Bob will be able to use his SD on the automatic ticket machine, which automatically issues Bob window seat tickets in a non-smoking coach. On the train, Bob can use his PDA to read the news that have been downloaded to his SD, and Bob's Simplicity-enabled web-browser automatically displays the news in plain text with hyperlinks to the pictures – according to Bob's preferences for the PDA. Even when visiting an unfamiliar city, Bob is able use a Simplicity-enabled electronic tour guide system to help him create a personalised tour, which makes sure that the attractions suggested are of interest to Bob. The tour guide system also displays detailed descriptions of the tourist attractions that Bob is visiting, and the contents of the description are tailored to Bob's interests, with a focus on the history of attractions rather than architectural details.

B. Simplicity System Architecture

The Simplicity Framework (illustrated in Figure 1) consists of two key components: the SD and the Simplicity Brokerage Architecture.

Conceptually, a SD is the entity that stores users' personal profiles and preferences that enable convenient customisation of devices and services. The SD could be a physical device (e.g., a USB memory stick, a CompactFlash card, a Java card or a Bluetooth-enabled mobile phone) or a virtual device, in which case user-related data would be stored within the network. The SD is the key for gaining access to the Simplicity Brokerage Architecture and for exploiting the benefits of it. It is therefore crucial to keep personal data on the SD stored in a secure manner so that it cannot be released without a user's permission. In order to minimise user involvement in the personalisation process, the SD should be automatically discovered by the Simplicity framework and Simplicity-enabled components. One specific goal of the Simplicity project is to make SDs universally recognisable, so that they can be used in conjunction with a wide range of computing devices that users encounter. The format of the user profile is important for it to be widely accepted, and possible methods for standardisation are currently investigated, including Generic User Profiles (GUP) in 3GPP [12]. Following the project's vision, a SD can be simply plugged (e.g., USB stick) or integrated (e.g., SIM card) into other devices. It should also be possible to have SDs that can be connected without physical contact, e.g., via short range wireless communication technologies, such as Bluetooth.

The Brokerage Architecture facilitates the design and deployment of adaptive services by managing different aspects of adaptation processes in a unified fashion. A broker consists of a number of subsystems, each of which is specialised to perform a specific task, e.g. the management of information such as user profiles, location date and the capabilities of devices or networks. Asynchronous communication between subsystems on the same device is enabled by a software component called Mediator, which dispatches messages (e.g., events) according to a set of pre-specified policies. In Simplicity, each broker undertakes management actions on local resources, while the overall administration of resources is achieved through inter-broker cooperation and coordination. A Simplicity Broker Communication (SBC) subsystem exists on every Simplicity broker. It is responsible for the communication between brokers on different entities.

Simplicity brokers also employ policy engines to facilitate the management of device and service adaptation. Rules for adaptation can be specified in a uniform way using a high-level policy language. By separating these rules from the components that perform the actual act of adaptation, developers and users are able to add new policies and modify existing ones conveniently without the need to modify these components. Policies can also be ported to different services and devices with minimal effort.

We classify Simplicity brokers into two types: brokers deployed on end terminals are called Terminal Brokers (TB). Amongst other things, Terminal Brokers host subsystems that handle user interaction with the Simplicity Framework. Brokers deployed to remote servers hosting network services are called Network Brokers (NB).

TBs are responsible for retrieving personal information stored on SDs and manage the interaction between the SD and the terminal device through a subsystem called Simplicity Device Access Management (SD-AM). SD-AMs offer interfaces to a variety of different types of SDs, including USB memory sticks and Bluetooth enabled mobile phones. TBs also allow legacy third party applications to exploit the benefits of Simplicity and customise themselves through Application Programming Interfaces (APIs) exposed by a special subsystem called Simplicity Application Interface Management (SAIM).

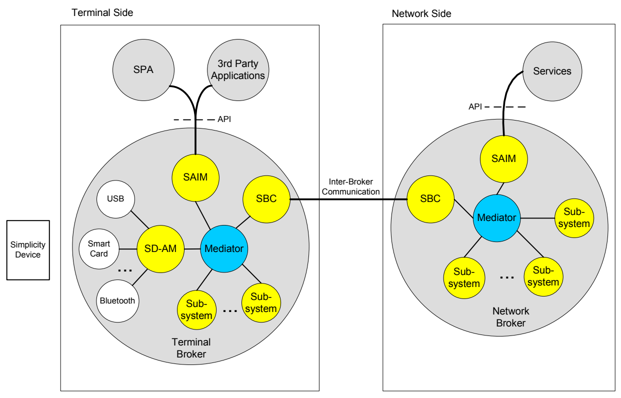

Figure 1: Architecture of the Simplicity Framework

Simplicity Personal Assistants (SPAs, mentioned in the "End User Experience" section) are standalone applications running on terminal devices to facilitates the process of personal profile transfer from SDs to TBs (e.g., by providing user interfaces for authentication). SPAs interact with the rest of the Simplicity framework through APIs exposed by SAIM subsystems.

To allow end users to select services and personalise them, Simplicity employs NBs on those servers that host network-based services. Service Management subsystems attached to Network Brokers collaborate with Service Management subsystems on TBs to provide support for service description, advertisement and discovery. By retrieving additional environmental information from context-aware modules, e.g. from a location management subsystem or a network capability subsystem, NBs are able to coordinate the adaptation of services based on various factors, such as location, network bandwidth or the availability of resources on servers . In order to treat those different factors in a unified fashion, Network Brokers make use of their built-in policy management mechanisms.

More detailed description of the architecture of the Simplicity system is beyond the scope of this paper. Please refer to [13][14][15][16] for more information.

IV. Analysis

As we have illustrated in section III, Simplicity provides a unified framework for configuring and adapting a wide range of devices and applications. The Simplicity framework itself is modular and extensible. Subsystems communicate using a light-weight, asynchronous, event-based interface, making it possible to easily extend the Simplicity platform for the needs of future applications.

By providing a framework that can be used by other application components to acquire the necessary information for configuring and adapting themselves, Simplicity is able to provide support not only for PCs and PC-based applications, but also for other ICT devices, such as phones and PDAs, and even non-ICT-related platforms, such as MP3 players, TV sets and even smart spaces. This ability to support heterogeneous platforms is one of the main features distinguishing Simplicity from other approaches that have been presented in section II. In contrast, most of these approaches are centred around either desktop PCs or workstations as targeted platforms.

While VNC, ISR and SoulPad merely support the migration of existing working environments that were already configured (e.g. complete images of the state of systems in the case of ISR and SoulPad, and input/output to a remote system in the case of VNC) to other machines, Simplicity enables arbitrary existing devices, services and applications to configure and adapt themselves according to users' preferences.

We note that there is a clear relationship between the amount of information a user is able to carry with him and the degree of dependency with respect to additional infrastructural services.

Approaches such as VNC or ISR do not require users to carry any devices but mandate permanent high-bandwidth network links (VNC) or at least intermittent network connectivity (ISR). By enabling users to have their personal Simplicity Devices with them at all times, Simplicity is able to operate without relying on network connectivity. Moreover, by allowing users to select from a wide variety of Simplicity

Devices, Simplicity enables users to select an option that is well suited for their own personal needs.

Tradeoffs have to be made between the versatility of devices and battery lifetime. Platforms, such as Intel's Personal Server technology, represent small embedded systems that are programmable and can possibly be used for a wide variety of tasks. However, their nature of being "always on" causes batteries to deplete more quickly. The Simplicity framework does not proscribe the use of any specific Simplicity Device, as long as the device exposes a standard, uniform SD interface. It is therefore possible to use both passive, unpowered devices as well as active, more versatile platforms, providing users with the flexibility they require.

Not unlike Simplicity, Migo aims at configuring existing applications and services according to users' needs. Migo pursues the philosophy of actively configuring applications and services. We personally believe that it is not feasible to expect a configuration and personalisation framework to provide modules for configuring every single existing application and service. Not only does the development of such components represent a great effort. It would moreover require developers of such components to track application development very closely to be able to support, for example, changes affecting the format of configuration information. This process is likely to require collaboration from the side of application developers, which is why we believe that it is more effective to leave the development of application-specific configuration and adaptation approaches to application developers themselves. Simplicity therefore provides a lightweight framework for retrieving and managing information necessary for the tasks of configuration and adaptation.

V. Conclusion

In this paper we have introduced the concepts of Simplicity, a novel framework for easing the burden of configuration and adaptation of devices and applications. By analysing Simplicity's approach and by comparing it with related work, we identified the following key contributions:

1) By providing a modular, extensible framework paired with well-defined interfaces through which third parties are able to use to retrieve information, we believe that Simplicity is well-equipped to lead citizens towards simplicity and to simplify interactions with a variety of devices and services.
2) The policy-driven decision mechanism employed in the framework facilitates uniform treatment of device personalisation and service adaptation, and will significantly increase the flexibility of the framework and ease the task of maintenance.
3) By providing a framework that can be used by other application components to acquire the necessary information for configuring and adapting themselves, Simplicity is able to provide support for heterogeneous platforms on a wide range of devices that extends well beyond the scope of other approaches.

We strongly believe that the benefits of the proposed framework will help Simplicity to quickly experience significant uptake by service providers, network operators as well as end users, and we are all looking forward to the arrival of Simplicity.

Acknowledgment

This work has been partially funded by the European Union in the framework of the project IST-2004-507558 Simplicity. The authors would like to acknowledge the contributions of all the Simplicity partners.

References

[1] SIMPLICITY project homepage: http://www.ist-simplicity.org/
[2] T. Richardson, Q. Stafford-Fraser, K. R. Wood, and A. Hopper, "Virtual Network Computing," *IEEE Internet Computing*, 2(1), Jan/Feb 1998.
[3] R. Want, T. Pering, G. Danneels, M. Kumar, M. Sundar and J. Light, "The Persona Server - Changing the Way We Think about Ubiquitous Computing," *Proceedings of Ubicomp 2002: 4th International Conference on Ubiquitous Computing*, Springer LNCS 2498, Goteborg, Sweden, Sept 30th-Oct 2nd, 2002, pp194-209.
[4] M. Kozuch, and M. Satyanarayanan, "Internet Suspend/Resume," *Proceedings of the Workshop on Mobile Computing Systems and Applications (WMCSA2002),* Callicoon, NY, June 20-21, 2002
[5] VMware homepage: http://www.vmware.com/
[6] M. Satyanarayanan, J. J. Kistler, P. Kumar, M. E. Okasaki, E. H. Siegel, and D. C. Steere, "Coda: A Highly Available File System for a Distributed Workstation Environment," *IEEE Transactions on Computers*, 39(4), April 1990.
[7] R.P. Goldberg, "Survey of Virtual Machine Research," *IEEE Computer*, 7(6), June 1974.
[8] N. Tolia, J. Harkes, M. Kozuch, and M. Satyanarayanan, "Integrating Portable and Distributed Storage," *Proceedings of the Third USENIX Conference on File and Storage Technologies (FAST '04)*, San Francisco, California, March 31-April 2, 2004.
[9] Migo homepage: http://www.4migo.com/
[10] M. T. Raghunath, C. Narayanaswami, C. Carter, and R. Cáceres, "Reincarnating PCs with Portable SoulPads," IBM Research Report RC23418, November 2004, Vol. 9, No. 1, February 2002. Available at http://www.kiskeya.net/ramon/work/pubs/ibm2004.pdf
[11] R. Cáceres, C. Carter, C. Narayanaswami, M. T. Raghunath , "SoulPad: Personalized Computing with Minimal Infrastructure," demo at the *Sixth IEEE Workshop on Mobile Computing Systems and Applications (WMCSA 2004)*, English Lake District, UK, 2-3 December 2004.
[12] Homepage of the 3rd Generation Partnership Project (3GPP): http://www.3gpp.org/
[13] N. Blefari-Melazzi, S. Salsano, R. Seidl: "The Simplicity Project: Initial System Architecture Specification," *E2R Workshop on Reconfigurable mobile systems and networks beyond 3G*, 5 September 2004, co-located with IEEE PIMRC 2004, in Barcelona (Spain).
[14] N. Blefari-Melazzi, S. Salsano, G. Bartolomeo, F. Martire, E. Fischer, C. Meyer, Ch. Niedermeier, R. Seidl, E. Rukzio, E. Koutsoloukas, J. Papanis, I. S. Venieris, "The Simplicity System Architecture," *14th IST Mobile & Communication Summit*, Dresden, Germany, 19-23 June, 2005.
[15] E. A. Fischer, C. Meyer, Ch. Niedermeier, R. Seidl, S. Kapellaki, G. Prezerakos, "Realizing Simplicity: Ambient Aware Service Adaptation," *14th IST Mobile & Communication Summit*, Dresden, Germany, 19-23 June, 2005.
[16] N. Blefari-Melazzi, D. Di. Sorte, M. Femminella, G. Reali, "Access Network Control within the Simplicity Brokerage Framework," *14th IST Mobile & Communication Summit*, Dresden, Germany, 19-23 June, 2005.