Instant Personalization and Temporary Ownership of Handheld Devices

Jürgen Bohn

Institute for Pervasive Computing
ETH Zurich, Switzerland
bohn@inf.ethz.ch

Abstract

As we increasingly depend on inexpensive handheld devices at work and in daily living, ensuring the accessibility of those devices and the availability of the personalized services they provide becomes a major challenge. In this paper, we present a system for instant personalization and temporary ownership of mobile devices that addresses these issues. The system enables the user to make the transition from requiring a specific individual *device to utilizing* any *device at hand. This significantly raises the degree of redundancy of devices accessible to the user from one to a potentially unlimited number of devices of a certain type. The system that we have prototypically implemented further provides support for periodic data backup, data recovery, and data confidentiality when devices are lost or stolen.*

1. Introduction

Handheld devices have become inexpensive and popular companions that support activities of daily living. Today, an estimated 30 million of Personal Digital Assistants (PDAs) and about 1.3 billion mobile phone devices are in use worldwide, with sales being expected to increase considerably within the next years [14]. Thanks to their portability and ease of use, mobile user devices enable convenient ubiquitous access to personal user data in situations where bulkier devices such as laptop and desktop computers are inappropriate. Being small and lightweight everyday companions that fit into a pocket, they can be employed while being on the move, often even supporting hands-free operation.

As we gradually depend more and more on the assistance of mobile user devices at work and in daily living, the reliability and availability of those devices and of the particular services they provide becomes a crucial issue. Firstly, a handheld device might be at hand but may not function properly due to a technical defect or because of an empty battery, for example. This is aggravated by the fact that low-cost, mass-produced handheld devices are more prone to suffer from hardware failures than higher-priced quality products used for more demanding professional activities. Secondly, a personal device may be physically unavailable, either only temporarily when the user has forgotten to take his or her device along, or permanently in case the handheld has been lost or stolen.

To address these issues, we present a system for instant personalization and temporary ownership of arbitrary mobile devices. The main idea is to make the transition from using a specific *individual* device permanently owned and pre-configured by the user to utilizing *any* available device that is instantly turned into a fully personalized device containing both a person's user data and meta data. In doing so, we increase the accessibility of specialized functionality offered by personalized handheld devices and the availability of personal user data. The system we prototypically implemented also facilitates data recovery, enabling the user to retrieve private data from physically unavailable devices. It also assists the user in preventing illegitimate data access on behalf of third parties in case that a personalized device has been lost or was left behind.

The remainder of the paper is organized as follows: In Section 2 we give a survey of related work. In Section 3, we describe the instant personalization of mobile user devices. In Section 4, we present the conceptual framework and the architecture of the system we developed, followed by a discussion in Section 5. In Section 6, we give an account of the current implementation status of our prototype, before we draw some conclusions in Section 7.

2. Related Work

Our work is closely related to the research domain of *ubiquitous data access*, where the major goal is to achieve *anytime*, *anywhere* access to user data.

A prominent approach for ubiquitous data access is the *UbiData* system by Zhang et al. [22], an application-transparent middleware architecture which provides device-independent access to data from heterogeneous sources.

Here, device independence relates to the fact that the system allows the user to switch among his or her various personal devices (such as the personal office PC, laptop, and PDA). Typically these devices are permanently owned and personalized by the user. In contrast, we are suggesting a diversification of data access by enabling the user to pick any *impersonal* handheld device of a certain type, thus considerably increasing the choice of devices from a small number of personally owned to a potentially unlimited number of available devices. Further, different from our work, Zhang et al. focus on issues of automatic and device-independent selection, hoarding, and synchronization of data, but they are not supporting an instant and temporary personalization of arbitrary mobile devices. In our approach, instant personalization gives the user not only access to the specific user data he or she normally uses with a certain type of device (e.g., personal calendar for the PDA, personal address book of the mobile phone, etc.), but also temporarily installs the specific meta-data that is required for the proper and convenient functioning of the characteristic services and applications provided by the particular type of device (including customizations, application settings, passwords, etc.). As a result, the user no longer relies on an individual set of devices, but may temporarily and interchangeably utilize any device that happens to be available. However, the diversification of user data access by means of instant personalization would blend well with the UbiData system, as it could help to significantly reduce the dependence on individual personally owned and therefore permanently personalized devices of a kind. While a person may concurrently use n different types of devices as part of a "horizontally" diversified ubiquitous data access, each of these device types can in return be "vertically" diversified by enabling the user to instantly load his or her device-specific portion of user data and meta data onto any device out of a virtually unlimited number of devices of the same kind.

Want et al. proposed a different approach to achieve ubiquitous data access in the face of user mobility. They use a portable mobile storage device enhanced with wireless communication capabilities, the *Personal Server* [21], which enables nearby devices to get access to the user's personal files. Currently, the Personal Server does not support an instant personalization of other mobile devices, but it could in principle be used as a local personalization server. However, compared to our approach, the Personal Server constitutes a single point of failure, and in this respect it suffers from the same shortcomings as any individual mobile device a user owns and carries along: if it breaks down beyond repair, or in case it is lost or stolen, all the data stored on the Personal Server which has been modified since the last backup is definitely lost. And if the Personal Server is only temporarily unavailable (e.g., battery is depleted or the user unintentionally left the device behind), the user has no means to access his or her personal data on the spot, either.

There has already been a significant amount of research in the field of personalization of services in ubiquitous computing environments [2, 9, 12, 18]. However, here the main focus is on providing personalized services to mobile devices (such as personalized content delivery, content and service adaption, and personalized interfaces for interaction with nearby devices, for instance) rather than to instantly personalize the mobile devices themselves.

The Microsoft Active Directory service for Windows-based personal computers supports user mobility within a distributed computing environment inside of an organization. It provides a single-log-on capability and a central repository for information, simplifying user and computer management and providing access to networked resources within a Windows domain. Compared to our work, the Microsoft Active Directory is a heavyweight infrastructure focusing on centralized user and computer management, while our approach is lightweight, targeting resource-limited handheld devices, increasing the accessibility of device functionality and user data, and protecting user data on personalized devices that are lost or left behind by means of a server-triggered or timeout-triggered data recovery mechanism.

Further related is the field of network computing. Here the main idea is to utilize a thin client with basic input/output capabilities to control applications executed on a remote computer. One example hereof is the Remote Desktop technology by Microsoft for Windows-based computers. Another example is the Virtual Network Computing (VNC) system [17], which provides access to home computing environments from anywhere and any device via a network connection by using a simple platform-independent display protocol. In contrast to our work, personalization of mobile devices is generally not an issue in network computing, as it is rather aiming at providing an abstraction from the underlying device hardware to achieve device-independence. As a result, the network computing approach is not suited to exploit the particular device-specific functionality of mobile user devices. Moreover, a thin client usually requires a permanent network connection while controlling a service on a remote computer, whereas in our approach, a mobile device can operate autonomously and without requiring a network connection once it has been personalized, thus being unaffected by transient disconnections or network delay, for instance.

3. Instant Personalization of Mobile Devices

In this section, we first discuss the roles that user data and meta data play in personalization. Then we explain the terms of "instant personalization" and "temporary ownership", which are central to this paper. Finally, we show how the instant personalization is experienced from the user's perspective.

3.1. The Roles of User Data and Meta Data

Today, users of handheld devices typically personally own one device of a kind, each of which serves a particular purpose and therefore provides functionality for which it has been designed and optimized. A mobile phone is optimized for making phone calls, a PDA is convenient for keeping track of appointments or for taking quick notes, or a smart electronic book (e-book) is a compact means of carrying with you the content of multiple books while enabling the user to search for words and phrases, to append written annotations, or to add bookmarks, for example. A handheld device may offer several of such services if it meets the requirements with respect to technological capabilities and ease of use (e.g., a PDA with a high-quality display can be used as an e-book, or a smart phone combines the capabilities of PDAs and traditional mobile phones).

The operation of handheld devices involves personal *user data*, either because the primary purpose of the device is to edit and manage such user data (e.g., taking notes, entering a new contact or appointment), or because the data are needed for the provisioning of specific services (e.g., a reminder service needs access to the user's personal calendar, a smart phone utilizes the user's list of contacts to retrieve the phone number of the person that is to be called, an e-book requires the digital version of the book the user wants to read). The *individuality* of a handheld device, however, is mainly determined by the user's individual preferences and device settings which make up the so-called *meta data*. Such meta data not only improves the efficiency and the ease of use of certain services provided by the mobile device (e.g., bookmarked web addresses, application-specific defined shortcuts, customized views and program settings, etc.), but often constitutes an essential element of these services (e.g., mail account settings, remote file server settings, passwords in general, etc.).

Besides user data and meta data, the personalization of mobile devices could also include the temporary installation of *personal applications* which are not part of the standard software that comes with a certain type or brand of device. In principle, if an application is self-contained, it can be treated as ordinary user data: it simply has to be copied into the correct folder on the mobile device. The deletion of such personal software later on is straightforward, too, since it is sufficient to delete the previously copied files. Otherwise, it may not be advisable to install applications that are not self-contained, since they often require the presence of certain libraries or runtime environments, or because their installation procedure may not be fully reversible or cannot be performed in an unattended fashion, for example.

3.2. Instant Personalization and Temporary Ownership

The goal of an *instant* personalization of mobile devices is to transform an arbitrary device, devoid of any personal user information, very quickly into a fully personalized device and with minimal involvement of the user. Further, the instant personalization process should be started the very moment the user actively and deliberately initiates it, no matter when (*anytime*) or where (*anywhere*).

So with an infrastructure for instant personalization in place, it becomes possible for a user to take *temporary ownership* of arbitrary devices he or she does not own personally but which are only available to him or her for a limited period of time. Once the instant personalization of a device is completed, it is indistinguishable from a personally owned device of the same type with respect to the particular personalized functionality and user data.

To be in a position to instantly personalize an arbitrary device on demand, anywhere, and anytime, the device requires access to a background service (that is a service offered by the background infrastructure) which provides the user's personal data and meta data. The access should favorably be performed by means of a wireless connection in order to not impede device mobility and portability. Further, the mobile device itself needs to know how to retrieve and install the data needed for instant personalization. Once the device is no longer needed, it has to be able to write back those parts of the data which have been modified since the personalization was effected, before being "released" from service and available again to be temporarily claimed and personalized by other users. So the *release* of a device is the inverse operation to the personalization procedure.

In situations where personal data on a mobile device is utilized in a non-manipulative fashion (e.g., when personalizing a mobile phone just for making some phone calls), it may be practical to perform a *read-only* personalization where any personal user data are simply deleted when the device is released. By analogy, if we wish to explicitly state that any modified personal data has to be copied back to the server when the mobile device is released, we speak of a *read-write* personalization. Note, however, that even if personal data has not been deliberately modified by a user, potentially useful meta-data such as the user's call history or email history is not preserved in the case of a read-only personalization.

3.3. User Experience

From the user's perspective, the process of instant personalization of mobile devices is experienced as follows:

A user picks up an arbitrary *blank* device (that is a device devoid of personal user information) that is physically available in the current place at the given time. The user takes temporary ownership of the handheld and logs on, using a personal user name or fingerprint for identification, and a password for authentication. Hereupon the device automatically downloads the user's device-specific individual preferences and settings, the meta data, together with his or her device-dependent user data from a dedicated server in the background infrastructure, using a wireless connection. Thus the user's preferred personal configuration is re-established on the device. Now the user is able to use the device (e.g., the personalized phone, PDA, or e-book) as if it were exclusively owned by him or her, exploiting the capabilities of the device to its fullest, with the personal user data (contacts, appointments, notes, etc.) as well as the personal meta data (e.g., mail server settings, passwords, bookmarks, shortcuts, customized views and program settings, self-contained applications and tools, etc.) installed. When the device is no longer needed, the user simply logs off, upon which the latest modifications of device settings or user data are written back to the back-end server. Finally, all the user's personal user and meta data are wiped off the device. This restores the original uninitialized blank state of the device so that it becomes available again to other users.

4. Conceptual Framework

By means of instant personalization and temporary ownership, we wish to realize the following design goals:

1. Higher *availability* of personal user data by providing anytime, anywhere access;
2. *interchangeability* of handheld devices to increase the *accessibility* of personalized device functionality;
3. support for *disconnected operation*;
4. *periodic data backup*;
5. *recovery* of personal user data and
6. *protection of confidentiality* of personal user data stored on personalized devices that are physically unavailable.

With the provisioning of instant personalization of mobile user devices as a service, the first two goals are implicitly realized: during personalization, user data are copied onto the device, and the personal meta data are automatically installed, which yields the personalized device functionality. Note that device interchangeability and device-independence have different meanings. Device *interchangeability* refers to the fact that I can easily change my device by virtually "moving" my personal user data and meta data from one device to another device of a certain type. In doing so, the device-specific characteristics in terms of usability and functionality are retained. The aim of *device independence*, however, is typically regarded as to provide an abstract, device-independent means of performing a task. In this case, the original qualities of the particular devices are not preserved or considered of secondary importance only. Examples hereof are the virtual network client or the remote desktop access described in Section 2.

The instant personalization of mobile devices anywhere and anytime requires connectivity to the instant personalization server as part of the background infrastructure. However, once the personalization is completed, the personalized mobile device no longer needs to stay connected but can operate in disconnected mode, thus supporting *disconnected operation*. In case of a read-write personalization, connectivity is again required when the device is to be released and data has to be sent back to the instant personalization server.

Portability and usability thanks to a comparably small form factor are some of the key advantages of handheld devices. However, as the number of small personal devices a user relies on grows, it becomes gradually more likely that a certain device is physically unavailable when needed. This may be because the user simply left the device behind, having forgotten to take the mobile phone out of the jacket worn the day before, for instance. A mobile device may even become permanently unavailable in case the user loses it someplace, or if it gets stolen, for example. Apart from the material loss, this leads to two further concerns. One is the *recovery* of personal user data stored on the device, especially if no recent backup of the data exists. Another concern is the *protection of data confidentiality*, as the personalized handheld device may carry private information that should not be revealed to others, or confidential data such as passwords or credit card numbers.

In our concept, these issues are addressed in the following way. First, a personalized device may perform an automatic release (auto-release) after a prolonged period of inactivity, triggered by a user-definable timeout. Second, the release of a personalized device can also be initiated by an external device, such as the remote personalization server or another device personalized by the user. In either case the client device first reconnects to the server to write back recently modified or added user data. As a result, the data of temporarily or permanently unavailable mobile devices is recovered (given that network connectivity is available and the batteries in the device are not depleted). Data confidentiality is preserved by erasing the concerned user data on the handheld device when the latter is released, which prevents the fraudulent use of private data. Additionally, if data confidentiality is paramount, the mobile user device may lock itself automatically after a short period of inactivity, preventing other users from accessing private information before the device completed the release operation.

Finally, user data are implicitly protected by means of data backups whenever a personalized handheld device is released and the modified user data are retransmitted to the server. However, a personalized user device may be continuously used for a longer period of time. If this device breaks down beyond repair during operation, all data that has been modified since the personalization is lost, too. To prevent this, the client can be configured to regularly reconnect to the instant personalization server in order to transmit recent changes in user data and meta data, thus achieving a *periodic data backup*.

4.1. Exclusive and Concurrent Personalization

Instant personalization benefits from the observation that a user typically only utilizes one personalized device of a kind, such as one mobile phone, one PDA, or one laptop, for example. In general, it is therefore sufficient to have only one device of a kind personalized in read-write mode at a time, removing the need for complex data synchronization and conflict resolution schemes. Consequently, in our concept, read-write personalization currently is performed as an exclusive operation. This means that a device that has previously been personalized in read-write mode has to be released before the server allows to personalize another device of the same type in read-write mode. Note that the user may concurrently perform as many read-only personalizations on separate devices as desired, as they do not require further assistance on behalf of the instant personalization server. However, read-write is probably the preferable personalization mode since it also preserves additional meta data (such as a call history, for instance) that has been accumulated during the utilization of a personalized device.

It may happen that a user wants to instantly personalize a device at hand even though a previously personalized and currently unavailable device has not yet been released before. Then, instead of waiting for the automatic timeout-triggered release to occur, the instant personalization server can enforce the release of the unavailable device by means of a remote release-request sent via the network. If the remote device is not reachable, the user may choose to perform a read-only personalization for the time being and wait for the remote device to write back its modified data in the meantime. Alternatively, the user can override the read-write personalization lock on the server and enforce a new read-write personalization, upon which the server considers the data on the unreachable device as stale and no longer valid. From the perspective of a device which has been personalized in read-write mode but which is unable to connect to the instant personalization server, there are also two options: it may either postpone the auto-release for a specified amount of time and try to reconnect in the meantime, or perform the release operation anyway, rating the protection of data confidentiality higher than data recovery.

Of course, computing power of the mobile devices and communication bandwidth allowing, data synchronization techniques as discussed by Zhang et al. [22] may also be used to support multiple concurrent read-write personalizations per user and type of device.

4.2. Cross-Platform Personalization

The instant personalization of a mobile device not only includes the transfer of personal user data to the device, but also the installation of the meta-data that is required for the smooth functioning of the particular applications the user expects to work with. However, user and meta data for a device often depend on the specific type of device, applying to concrete versions operating systems (such as Symbian OS for mobile phones, Windows CE or Palm OS for PDAs, for instance) and the standard applications associated with these operating systems. As a consequence, to widen the applicability of personal data, it is necessary to provide abstractions or generalizations for *device-specific* knowledge on how to automatically install or extract a user's data and meta information.

One solution are *standardized interfaces* to applications that are commonly integrated with certain operating systems or types of devices. Such a standardized interface already exists for the Microsoft Windows CE operating system (version 3.0 and later): the Pocket Outlook Object Model (POOM). It provides a generic API for manipulating contact, calendar, and tasks data. As these data make up an integral part of the personal user data and meta data typically used on a PDA, the POOM interface contributes towards realizing a unified personal data management for Windows CE based handheld devices, irrespective of hardware configuration and manufacturer.

Another possible solution is the definition of device-independent *personalization profiles*. Such profiles could then describe the structure and vocabulary of personal user settings and preferences. Once personalization profiles for mobile user devices have been defined and standardized, they provide an abstract and universal interface for manipulating personal user data and meta data across different hardware platforms and operating systems. An example for such a standardization effort are the Composite Capability/Preference Profiles (CC/PP) [5] proposed by the W3C Device Independence Working Group, which have been designed to enable "access to a unified web from any device in any context by anyone". A similar goal is pursued by the development of SyncML. SyncML is intended as a single common data synchronization protocol that aims to deliver an open, industry-wide specification for the universal synchronization of remote data and personal information across multiple networks, platforms, and devices.

Note that even if device-independence is achieved at the software level, problems could still arise from the heterogeneity of manufacturer-dependent hardware components. Although supporting a similar operating system, mobile phones from different vendors may still significantly differ in terms of hardware control elements (such as different button layouts), for example. Thus the perceived ease of use of a successful instant device personalization may be negatively affected, especially if a user has difficulties adjusting to unfamiliar operating controls.

An alternative approach could be to upload a complete virtual machine image instead of performing a fine-grained personalization on the data element level. Such a procedure would implicitly retain all software installations and system modifications performed on the personalized device. However, this scheme has several drawbacks. Firstly, it would typically be necessary to transfer the complete (binary) image even if only a small portion of the user's personal data had been altered, thus increasing the data transfer load and impeding a customized personalization procedure. Secondly, it would no longer be possible to synchronize and combine data modifications from multiple devices since either all changes or no changes could be retained per image. Thirdly, the upload of complete images (including systems software and applications) onto arbitrary devices would presumably conflict with existing licensing policies.

While we are planning to eventually employ a generic personalization profile such as CC/PP or SyncML, our initial prototypes have been developed using a combination of the standardized POOM API, together with a set of operating-system-specific methods not supported by the API (e.g., manipulating the registry under Windows CE).

4.3. System Architecture

Our prototypical system for instant personalization consists of a client component (Instant Personalization Client), which is executed on the mobile devices, and a server component (Instant Personalization Server), which resides in the background infrastructure (see Figure 1). In the following, we describe the two components in more detail.

The *Instant Personalization Client* (IPC) is executed as a *persistent system process* on the mobile device. It is automatically launched whenever the device is started (e.g., after a reboot or reset). The IPC also features a *graphical user interface* (GUI) (see Figure 2 for screenshots of the GUI we implemented for our IPC client prototype). If the device is in the unpersonalized state, the GUI allows the user to log-on to the personalization server (see login screen dialogue), to choose the program modules that have to be personalized (see selection of personalization modules dialogue), and to specify the timeout (in minutes) for the auto-release function. Afterwards, the *module manager* performs the instant personalization for all selected personalization modules. After the device has been personalized, the user can choose the "release now" menu option to actively release it once the device is no longer needed. The user can also wait until the IPC-internal timer starts the auto-release operation (see auto-release notification dialogue). The IPC communicates with the IPS via *sockets* using TCP/IP connections. For the server-initiated release, a separate listener thread on the IPC listens to server requests.

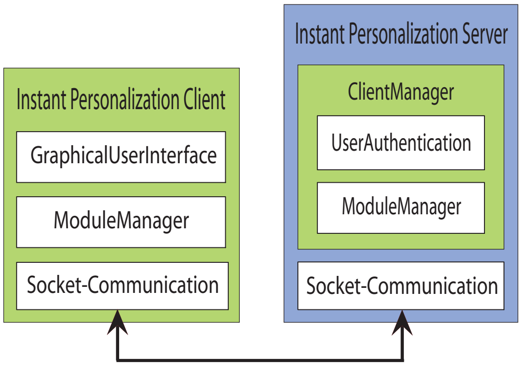
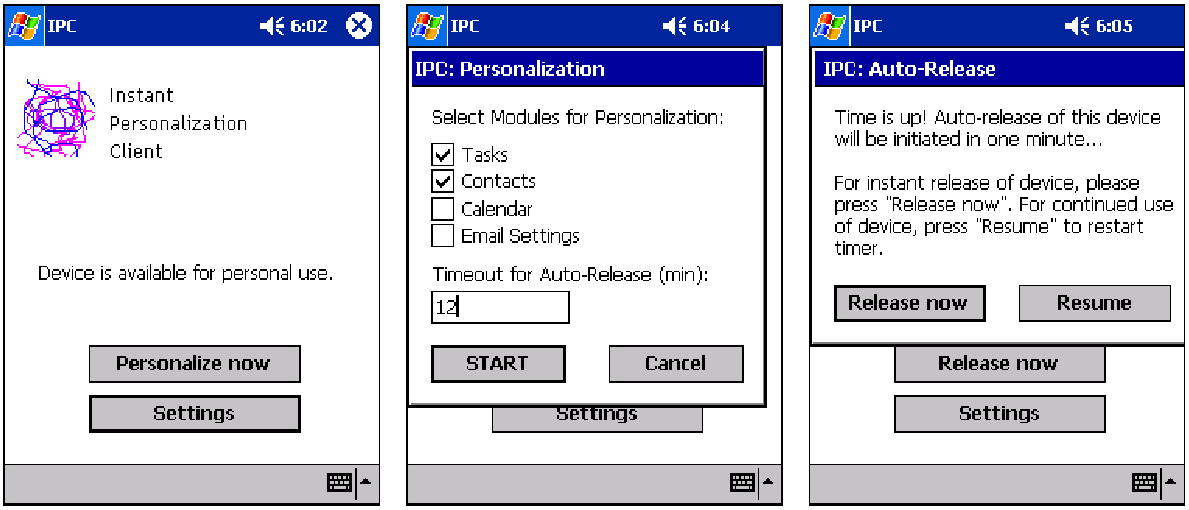


Figure 1. Architecture of the Instant Personalization System.

The *Instant Personalization Server* (IPS) acts as a background service residing in the network. It manages the database which contains the users' specific data needed for the instant personalization of mobile devices. The IPS uses sockets with a fixed port number to listen to IPC requests. Whenever a user takes ownership of a blank device and initiates the instant personalization procedure, the IPC on that device connects to the IPS via a secure channel. The IPS *client manager* identifies the user and authenticates the corresponding password. On successful authentication, the module manager on the user's handheld requests personalization information for the desired personalization modules which are then returned by its counterpart module manager on the IPS. Otherwise, the connection is closed by the IPS. Similarly, if the user's personalized device is to be released after a read-write personalization, then the IPC connects to the IPS, is authenticated, and writes back the modified portions of the user data and meta data. For a release after a read-only personalization, the IPC simply removes the user's personal data from the device. Note that an IPS can typically only serve user data and meta data for devices that share compatible interfaces for personalization (such as POOM for Windows CE based devices, for instance). The actual scope of devices that can be handled by an IPS therefore largely depends on the availability of suitable interfaces for cross-hardware and cross-platform personalization as discussed in Section 4.2.



Instant Personalization Client

Device is available for personal use.

Select Modules for Personalization:

☑ Tasks
☑ Contacts
□ Calendar
□ Email Settings

Timeout for Auto-Release (min): 12

Time is up! Auto-release of this device will be initiated in one minute...

For instant release of device, please press "Release now". For continued use of device, press "Resume" to restart timer.

Figure 2. GUI of the prototypical Instant Personalization Client (from left to right): login screen dialogue, selection of personalization modules dialogue, auto-release notification dialogue.

5. Discussion

There are a number of benefits and challenges concerning a practical large-scale deployment of an instant personalization infrastructure, which we discuss in the following.

5.1. Sharing and Pooling of Mobile Devices

A person can employ several devices of a kind (e.g., one in the office, one in the car, and one at home), and conveniently switch between those devices by means of instant personalization. And since a device is always stripped of the user's personal and potentially confidential information when it is released after use, a user can temporarily lend devices out to friends and strangers alike without having to worry about the protection of private data.

For this reason, the concept of instant personalization is particularly suited for the *sharing* and *pooling* of mobile user devices in general. While handhelds have been a mainstay in the business world for several years, they are recently also adopted on a larger scale in other areas such as hospital environments [1] or education [6, 19]. For instance, the University of South Dakota became one of the first universities to implement a full-scale PDA program, giving faculty members an opportunity to study how the devices can be integrated into college teaching and learning [16]. In such environments, the use of handheld devices can greatly benefit from an instant personalization infrastructure: instead of dealing out mobile devices on a per person basis, each device being permanently owned and exclusively utilized by one user, it becomes feasible to provide a shared pool of devices out of which one can pick any device, instantly personalize it on demand, and use it just as long as needed. Such an approach is not only resource-efficient, lowering the number of devices that have to be bought and maintained, but it also increases the ease and flexibility of device utilization, as a user no longer relies on his or her own personally owned device but is free to use any available device.

Instant personalization of mobile devices is also advantageous in areas where it is inconvenient or prohibited to take along personally owned electronic devices. To protect the privacy of guests in places such as swimming baths, for example, it may not be desirable that guests bring along their own personal handheld devices that might be equipped with a digital video camera. Instead, the authorities could place generic devices for instant personalization at the guests' disposal, at a *pay-per-use* basis, for example. Such a short-time leasing of mobile devices for instant personalization can also be to the benefit of the guests, as it removes the need to worry about personally owned devices being stolen while swimming or being damaged when unintentionally exposed to water or sand.

Further, an interesting question is who should be in charge of operating an instant personalization server, and where the server should be physically located. The availability of an instant personalization server may be unsatisfactory if it is located at the user's home, there being affected by power outages, transient failures of the user's network connection, or unskillful maintenance, for example. In this context, a promising option could be to have telecommunications providers offer the instant personalization service bundled with the traditionally provided communication services. Both services go together well, as connectivity is the prerequisite of anytime, anywhere instant personalization.

5.2. Bandwidth Requirements

The availability of a sufficiently high bandwidth may pose a challenge for the practical realization of an instant personalization system. Traditionally, when using a permanently personalized device for remote data access, it is only necessary to regularly synchronize that portion of the data which has been modified either on the device or on the remote server, which may significantly reduce bandwidth requirements.

In contrast, the instant personalization of arbitrary blank devices always requires the complete download and installation of all necessary user data and meta data. Consequently, for instantly personalizing devices with a potentially large amount of personal user data (such as laptop computers, MP3 players, or digital cameras with large storage capabilities), high-speed network connectivity would be necessary for achieving a swift data transfer. Further relevant factors are the cost and reliability of the data transfer.

When focusing on the instant personalization of resource-limited mobile devices, however, bandwidth typically constitutes a minor problem. On the one hand, the amount of accumulated user data and meta data is usually comparably low (the space required for plain-text contact entries on a mobile phone or for calendar entries on a PDA, for instance, is typically in the dimension of a few hundred kilobytes and can be further reduced by means of compression techniques). On the other hand, the data rates of available network technologies such as Wireless LAN (11 Mb/s and beyond), Bluetooth (up to 2 Mb/s), or emerging 3G/4G telecommunication networks (up to 2 Mb/s in 3G networks, and 20 Mb/s and beyond in 4G networks) should be high enough for the realization of a reasonably quick data transfer. The estimated duration of an instant device personalization with regard to different communication technologies, data rates, and amounts of personal user data is displayed in Table 1.[1] We can see that – with the expected emergence of higher bandwidth communication networks – the realization of a truly instant personalization in the range of a couple of seconds or even only milliseconds can be achieved even for comparably large amounts of user data. Once the personalization is completed, the amount of data that has to be transferred back to the server during the release operation is typically uncritical, as it is sufficient to only write back the portion of the data which has actually been modified. In the case of a read-only personalization, the release operation can even be efficiently performed off-line by simply erasing any personal information from the device. In addition, progressive update propagation schemes as suggested by Lara et al. [4] could help to further reduce latencies as the user may already start using the device before the personal data has been completely fetched from the server.

In the long run, if the development of computer networks advances at the current rate, one may argue that we ultimately find a close to perfect network at our hands, with global coverage, nearly unlimited bandwidth, high stability and minimum delay. Such a development would obviously greatly facilitate the instant personalization of mobile devices. At the same time, the availability of a nearly perfect network could, provocatively speaking, even remove the need for storing personal data locally on diverse devices. It may even provide a boost for the concept of virtual network computing as described by Richardson et al. [17], promoting a unified instant remote access to personalized resources residing in the background infrastructure by using dumb virtual terminals for local input and output only.

However, we think that a complete future shift towards network computing is questionable for several reasons. Firstly, in the past, similarly optimistic prophecies regarding an expected breakthrough of the thin-client approach repeatedly proved to be false and unrealistic. This was for technical and economical reasons, even in the face of a considerable increase of communication bandwidth, or simply just because the underlying concept itself consistently failed to meet the customers' expectations. Secondly, we think it is doubtful that there will ever be such a (close to) perfect network available. Already today network bandwidth for Internet connections or UMTS communication channels, for instance, is predominantly provided on a best-effort basis, as network providers are generally interested in maximizing the traffic load and keeping excess capacities to a minimum in order to keep down costs and to maintain their competitiveness. Consequently, the quality of service characteristics of such communication networks should be far from perfect, particularly in peak times. Thirdly, a solution where computations and data processing are performed locally typically scales better and is more robust against interferences or denial of service attacks than a centralized server approach for which a stable network connection and a remote data transfer is required for each device and each single operation.

5.3. Trust and Security

In our system, user *identification* is performed by means of a user name, and user *authentication* by means of a secret user password. Both user name and password are transmitted to the server via a secured communication channel (e.g., using SSL [15]). Alternatively, the user can choose to identify him or herself conveniently via fingerprint, removing the need for manually typing a user name or for using an extra identification badge or tag. This was a feasible option

[1] For calculating the amount of typical personal data used with a PDA or smart phone, we took the following data as a basis: 340 contacts at 1.5 KB each on average, 20 tasks at 2 KB each, and 50 future appointments at 1.2 KB each, yielding approx. 600 KB.

Table 1. Duration of instant personalization with respect to typical communication technologies, net data rates, and different amounts of data (contacts, appointments and task list for a PDA or mobile phone; e-books; digital photos; complete mailbox including email attachments).

Communication	Bandwidth	Bandwidth	PDA	e-Book	Digicam	Email
Technology	(nominal)	(net)	(∼600 KB)	(∼2 MB)	(∼20 MB)	(∼200 MB)
GPRS (4 time slots)	57.6 Kb/s	48 Kb/s	1 min 40 s	5 min 33 s	56 min	9 h 16 min
GPRS (8 time slots)	115.2 Kb/s	96 Kb/s	50 s	2 min 47 s	28 min	4 h 38 min
UMTS (global cell)	144 Kb/s	144 Kb/s	33 s	1 min 51 s	19 min	3 h 5 min
UMTS (micro/macro cell)	384 Kb/s	384 Kb/s	13 s	42 s	7 min	1 h 9 min
Bluetooth	2 Mb/s	1 Mb/s	4.8 s	16 s	2 min 40 s	26 min 40 s
UMTS (pico cell)	2 Mb/s	2 Mb/s	2.4 s	8 s	1 min 20 s	13 min 20 sec
WLAN 802.11b	11 Mb/s	5.5 Mb/s	870 ms	2.9 s	29 s	2 min 52 s
WLAN 802.11a, Hiperlan/2	54 Mb/s	32 Mb/s	150 ms	500 ms	5 s	50 s
4G Networks	20-300 Mb/s	100 Mb/s	50 ms	160 ms	1.6 s	16 s

since the handheld devices we used for prototyping featured a built-in fingerprint sensor.

Initially, we intended to use fingerprints in place of passwords. However, fingerprint sensors only constitute a secure means for user authentication when embedded in trusted hardware where personal fingerprint information is protected from illegitimate access and tampering [10]. As this is not the case with most fingerprint hardware that comes with today's of-the-shelf handheld devices, an impostor may, with moderate effort, bypass the physical fingerprint sensor and insert another user's fingerprint sample (fingerprints are in general easily available from objects a particular user has previously touched – they then only need to be digitized by the impostor and transformed into the typically publicly known format used by the device-specific fingerprint software).

This raises the question of *trust* in general. When a user possesses several devices of the same kind, or if devices are shared in a closed group (e.g., among friends or colleagues), trust is not an immediate concern. However, am I willing to entrust my private data to a device of unknown origin that may have been tampered with and therefore be potentially malicious and untrustworthy? A publicly available device may be spying on me, secretly stealing personal passwords or disclosing confidential information. It is therefore of prime importance that a user is in a position to clearly assert that any given device has not been tampered with and can be considered trustworthy. A promising attempt to tackle this issue is the Trusted Platform Module (TPM) [20] technology promoted by the Trusted Computing Group.

Alternative methods of secure authentication are one-time authentication schemes, using one-time passwords as first described by Lamport [11], or utilizing trusted hardware tokens carried by the user, including smart cards [13] or hardware tokens similar to the ones described by Corner and Noble [3]. Another possibility for achieving secure authentication is to use challenge-response mechanisms, such as providing distorted facial images of persons known to the user as a challenge for which he or she has to provide the correct names, or asking the user to recognize a known face out of a selection of otherwise unfamiliar faces, as performed by the Passfaces[2] system, for instance.

Another challenge is the protection of data confidentiality with respect to unauthorized recovery of personal user data: confidential user data that has been deleted during the release-phase of a temporarily personalized device should not be recoverable, or only at high cost. Gutmann [7, 8] describes the problems and potential solutions in greater detail. Here, the availability of a trusted and tamper resistant hardware module (such as TPM) can also be used to protect a user's personal secrets, by providing a secure storage area which can be completely flushed on demand and which cannot be inspected using memory viewing tools, for example.

6. Prototype Implementation Status

On the client side, we used HP iPAQ handheld devices of the H5450 series with PocketPC 2002 installed. The client software was programmed in Visual C++ for Embedded Ver. 3.0. On the server side, we used an Intel-based desktop computer running Windows XP, and Microsoft Visual C++ Ver. 6.0 for programming. The HP iPAQ H5450 features a built-in Wireless LAN network interface, which we used for connecting wirelessly to the instant personalization server. It also contains a built-in fingerprint sensor. For programming the fingerprint hardware om the iPAQ, we used the Biometrics API[3] which is freely available as part of the iPAQ Pocket PC Developer Program.

[2] Passfaces™ by Real User, www.realuser.com/

[3] http://devresource.hp.com/drc/technical_papers/Bioapi.jsp

We have implemented an initial prototype of the instant personalization system, meeting the basic design goals described in Section 4. Currently, the user can choose among four different modules for the instant personalization of the mobile device: personal *tasks*, *contacts*, *calendar entries*, and personal *email settings* (for remote email access using IMAP). For the personalization of tasks, contacts, and calendar, we used the Pocket Outlook Object Model (POOM) as a standardized means of accessing personal user settings and user data on the handheld device. For the adjustment of the email client settings, we had to directly manipulate the Windows CE Registry where all the data about applications, drivers, user preferences, etc. are stored. Authentication is performed by means of user name and password. The integration of the fingerprint sensor (especially the processing of fingerprints on the server side) is still in an experimental stage. Another open task is the integration of compression and synchronization techniques into the module manager to reduce the communication load during data transmission.

7. Conclusion

Mobile user devices such as mobile phones or PDAs are proliferating in everyday life, turning into basic *commodities* that are no longer exclusively sold by specialist stores only, but increasingly offered in supermarkets and fashion stores alike.

As mass-produced handheld devices become available in large quantities and at moderate prices, the concept of instant personalization of mobile devices presents an opportunity to reduce the dependence on single personal devices we permanently possess. Instant personalization can help to increase the accessibility of specialized functionality provided by personalized handheld devices, improve the availability of personal user data, facilitate periodic data backup and recovery, and support data confidentiality when devices are lost or stolen.

In this paper, we have presented the goals and requirements of instant personalization and temporary ownership for mobile user devices, and described an initial prototype we have developed that supports our core concepts on Windows CE devices. The next steps will be to evaluate the system and eventually to employ the personalization profiles described in Section 4.2 in order to support a more diverse set of handheld user devices.

References

[1] H.-B. Bludau and A. Koop, editors. *Proc. 2nd Conf. on Mobile Computing in Medicine, Workshop of the Project Group MoCoMed, GMDS-Fachb. Med. Informatik & GI-Fachausschuss 4.7, Heidelberg*, volume 15 of *LNI*. GI, 2002.

[2] P. Chandrasekaran and A. Joshi. MobileIQ: a framework for mobile information access. In *Proc. 3rd Int. Conf. on Mobile Data Management*, pages 43–50, Jan. 2002.

[3] M. Corner and B. Noble. Zero-interaction authentication. In *Proc. 8th Annual Int. Conf. on Mobile computing and networking*, pages 1–11. ACM Press, 2002.

[4] E. de Lara, R. Kumar, D. Wallach, and W. Zwaenepoel. Collaboration and multimedia authoring on mobile devices. In *Proc. of MobiSys 2003*, San Francisco, USA, May 2003.

[5] Device Independence Working Group (DIWG). Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies 1.0. W3C Recommendation, Jan. 2004.

[6] M. Fallon. Handheld Devices: Toward a More Mobile Campus. *Syllabus Magazine*, Nov. 2002.

[7] P. Gutmann. Secure deletion of data from magnetic and solid-state memory. In *Proc. 6th USENIX Security Symposium, San Jose, California, USA*, July 1996.

[8] P. Gutmann. Data remanence in semiconductor devices. In *Proc. 10th USENIX Security Symposium, Washington, D.C., USA*, Aug. 2001.

[9] D. Hilbert and J. Trevor. Personalizing shared ubiquitous devices. *interactions*, 11(3):34–43, 2004.

[10] A. Hohl and A. Zugenmaier. Safeguarding Personal Data with DRM in Pervasive Computing. In *Proc. Security and Privacy Workshop at PERVASIVE 2004*. Kluwer Academic Publishing, 2004.

[11] L. Lamport. Password authentication with insecure communication. *Comm. of the ACM*, 24(11):770–772, 1981.

[12] M. Lankhorst, H. van Kranenburg, A. Salden, and A. Peddemors. Enabling technology for personalizing mobile services. In *Proc. 35th Annual Hawaii Int. Conf. on System Sciences (HICSS 2002)*, pages 1464–1471, Jan. 2002.

[13] M.-H. Lin and C.-C. Chang. A secure one-time password authentication scheme with low-computation for mobile communications. *SIGOPS Oper. Syst. Rev.*, 38(2):76–84, 2004.

[14] B. Myers and M. Beigl. Handheld computing. *IEEE Computer*, 36(9):27–29, Sept. 2003.

[15] Netscape Communications. Secure Sockets Layer (SSL) 3.0 Specification, Nov. 1996.

[16] D. Peterson. Implementing PDAs in a College Course: One Professor's Perspective. *Syllabus Magazine*, Nov. 2002.

[17] T. Richardson, Q. Stafford-Fraser, K. Wood, and A. Hopper. Virtual Network Computing. *IEEE Internet Computing*, 2(1):33–38, 1998.

[18] Z. Sahinoglu, F. Matsubara, K. Peker, and J. Cukier. A mobile network architecture with personalized instant information access. In *Digest of Technical Papers. Int. Conf. on Consumer Electronics (ICCE 2002)*, pages 34–35, June 2002.

[19] E. Soloway, C. Norris, P. Blumenfeld, B. Fishman, J. Krajcik, and R. Marx. Log on education: Handheld devices are ready-at-hand. *Comm. of the ACM*, 44(6):15–20, 2001.

[20] Trusted Computing Group (TCG). TCG TPM Specification Version 1.2 (Revision 62), Oct. 2003.

[21] R. Want, T. Pering, G. Danneels, M. Kumar, M. Sundar, and J. Light. The Personal Server: Changing the Way We Think about Ubiquitous Computing. In *Proc. UbiComp 2002*, pages 194–209. Springer-Verlag, 2002.

[22] J. Zhang, A. Helal, and J. Hammer. Ubidata: ubiquitous mobile file service. In *Proc. 2003 ACM Symposium on Applied Computing*, pages 893–900. ACM Press, Mar. 2003.