AmI-Blocks'08, European Conference on Ambient Intelligence 2008 manuscript No.
(will be inserted by the editor)

Association: Unobtrusively Creating Digital Contracts with Smart Products

Daniel Schreiber, Melanie Hartmann

Received: date / Accepted: date

Abstract Many business models for smart products, like pay-per-use, require that the smart product can digitally verify whether the user has a contract with the smart product and should be granted access to privileged functionality. Traditional means to do so, e.g. password login, are very obtrusive and can thus not be applied for smart product scenarios. In this paper, we present the mechanism of *association*. Associations represent the abstract concept of a digitally checkable contract on the middleware level. Associations use a service for digitally representing the user that performs the tedious parts of creating a digitally checkable contract automatically. Thus, the interaction can be established unobtrusively. As this service acts on behalf of the user, the user must trust this service. We address this issue in two ways: the service is executed on the *personal trusted device* of the user and the user can control and inspect the actions of the service via a user interface.

Keywords smart products · ubiquitous computing · natural interaction

1 Introduction

With cheaper hardware costs and smaller form factors, it has become possible to embed computing power into everyday products, turning them into *smart* products. We define a smart product P as a physical product ($\mathtt{P_{phys}}$) augmented with a digital service linked to the product (P_{dig}) (see the left block in figure 1).

Recently the economic potential of smart products has gained interest, e.g. pay-per-use pricing with smart products has been proposed by [19]. These new market models require that the smart product is able to check whether access to privileged functions should be provided, that is, if the user has a valid contract with the product provider. To realize such a scenario the juristic concept of a contract needs a digital counterpart that can be automatically checked by the smart product. Note that juristic contracts can be established very unintrusively, e.g. they can be *implied by the acts of the parties* or very formal, e.g. by signing a written document. The middleware mechanism should be designed to reflect this flexibility.

In this paper, we introduce the new concept of a digitally checkable contract or short *digital contract* C_{dig} between the user and a smart product P. For that purpose, we introduce a digital representation of the user U_{dig} that can create the contract C_{dig} with the smart

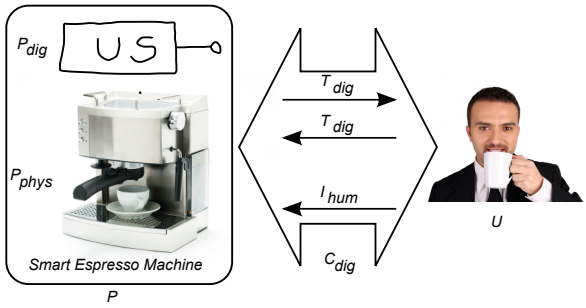

Fig. 1 A smart product P is modeled as physical product $\mathtt{P_{phys}}$ and a digital service P_{dig}. The P_{dig} is here realized as US in the Mundo platform. The prerequisites for creating a *digital contract* C_{dig} between the user U and P are asymmetric. We can unequivocally assume the intent of P for accepting contracts as this is its purpose. P can only use digitally accessible properties and actions of U to determine if U is trustworthy (T_{dig}). U may also check digitally accessible properties (T_{dig}) to determine whether P is trustworthy and should be offered a contract. Further, the user must have the intention I_{hum} to offer a contract to P.

TU Darmstadt
Telekooperation
Tel.: +49-6151-164752
E-mail: schreiber,melanie@tk.informatik.tu-darmstadt.de

product's digital representation P_{dig} after it has been authorized by the user to do so.

A middleware providing such a mechanism lends itself for creating personal smart environments. The user should be able to connect different smart products contracted by her to form a personal smart environment. Therefore, the middleware must make sure smart products contracted by a user provide their priviledged functionality also to other smart products contracted by the user. This could e.g. be used to form copying machine out of a contracted printer and scanner.

2 Digitally Verifiable Contracts

To serve as a complement for juristic contracts in smart products scenarios, C_{dig} may only be created after three prerequisites are fulfilled (see figure 1):

1. P must find the user trustworthy by checking digitally accessible properties ($(P, user) \in T_{dig}$).
2. The user may also use digitally accessible properties of P to judge whether P is qualified ($(user, P) \in T_{dig}$).
3. C_{dig} can only be created when the user *intends* to do so ($(user, P) \in I_{hum}$).

Thus, the user has a digital contract C_{dig} with P ($(user, P) \in C_{dig}$) if these three properties hold:

$$(user, P) \in Q \Leftrightarrow (P, user) \in T_{dig} \wedge \quad (1)$$
$$(user, P) \in T_{dig} \wedge \quad (2)$$
$$\wedge (user, P) \in I_{hum} \quad (3)$$

The procedure for establishing mutual T_{dig} may vary from use case to use case, e.g. the user needs to provide credit card data to P and P must provide some certificate to the user and so on. This procedure can be automatically performed by the digital representation of the user and the smart product. In contrast, automatically detecting the user's intent is almost impossible.

Establishing C_{dig} must be unobtrusive for the user as it needs to be performed potentially multiple times per day. Ideally, creating C_{dig} would be at least as unobtrusive as creating a juristic contract inferred from acting. Existing mechanisms that allow to digitally check whether a juristic contract exists or should be created are often much more obtrusive, e.g. creating a rental contract for a DVD requires the user to provide a password as opposed just taking the DVD from the shelf and leaving the store, which would be enough to legally create a rental contract in a DVD shop.

2.1 Association Mechanism

To unobtrusively ensure all three properties necessary for C_{dig}, we rely on a service U_{dig} for digitally representing the user, which actively performs the necessary steps to establish mutual T_{dig} whenever I_{hum} is given. We call the mechanism by which C_{dig} between a user and P can be created *association*. *Association* is performed following an abstract procedure that can be used with different

- *authorization procedures*, for signaling to U_{dig} that I_{hum} for a P exists and
- *qualification procedures*, for establishing mutual T_{dig}.

Figure 2 shows an overview of the *association* process. Particularly, *association* allows to establish a full C_{dig} relationship in reaction to real world actions of the user, e.g. performing a *qualification procedure* in response to touching an object (the *authorization procedure*). This is less intrusive than traditional means for establishing C_{dig}, e.g. providing a password. Thereby, the available *qualification procedures* may depend on the the *authorization procedures* used and P. If at any time one of the properties $(1) - (3)$ is no longer given, the *association* must be terminated.

A P that is *associated* with a user should grant the user access to privileged functions and other Ps *associated* with the same user to build up a smart environment. However, for the connection between the different P_{dig}s, it is not necessary that $(P_1, P_2) \in T_{dig}$ as sufficient trust is given by the associations with the user. These *connections* between different smart products have already been considered in [6,8], therefore we focus on *association* in this paper.

2.2 Digitally Representing the User

Association requires a service U_{dig} for digitally representing the user. During *association*, U_{dig} performs the necessary steps to establish mutual T_{dig} with P. To do so, U_{dig} can use different *qualification procedure*, e.g. provide a password to P and conversely check the certificate of P. To perform such procedures automatically, the user must store some personal information in U_{dig}. Further, the user must accept that U_{dig} automatically acts on her behalf. To let the user be comfortable with this, the user must feel in control of her U_{dig}. We address this by two means:

- executing U_{dig} on a *personal trusted devices* of the user (explained in section 3.1.1), and
- visualizing the actions of U_{dig} in a user interface for inspection by the user (explained in section 4).

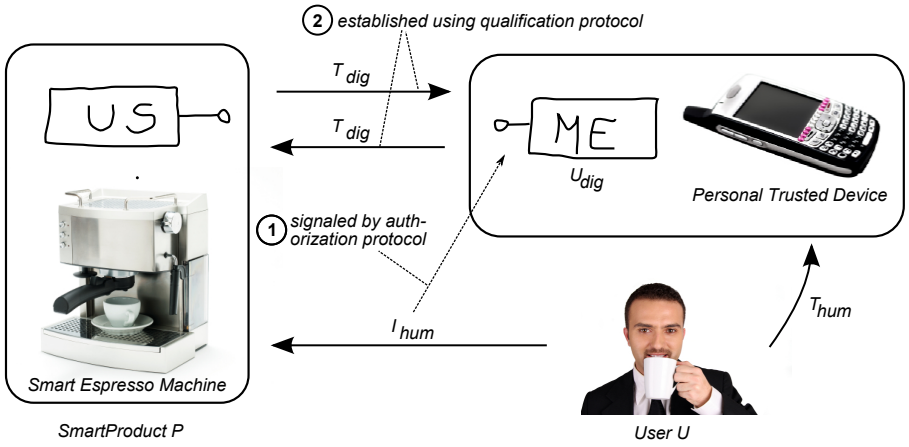

Fig. 2 ME is the name of the U_{dig} service implementation in the Mundo platform. Mutual T_{dig} is established following a *qualification procedure* between U_{dig} and P_{dig}. The ME only performs this when signaled I_{hum} according to a *authorization procedure*. The mobile phone plays the role of a *personal trusted device*.

2.3 Rationale for *Authorization Procedures*

Traditional means to establish C_{dig} required explicit user interaction, like login with passwords. Thereby the user would perform these only if I_{hum} is given. Completely automating the process and relying on T_{dig} alone would require that digital properties could also be used to determine whether I_{hum} exist. This is problematic as methods for determining that the user's intend to perform *association* are error prone and will therefore lead to an unsatisfied user. As I_{hum} cannot be detected directly, we rely on *authorization procedures*, i.e. actions that the user must perform and after which U_{dig} may assume I_{hum} for a P is given. Likewise, the user can use an *deauthorization procedure* to advise her U_{dig} to terminate an *association*.

3 Smart Products with the Mundo Platform

This section describes our implementation of the general concepts introduced in the preceding section with the Mundo platform.

3.1 The Mundo ME as implementation of U_{dig}

The *personal trusted device* running a dedicated service for representing the user has already been proposed in [9]. The Mundo Platform supports the usage of a *personal trusted device* called *Minimal Entity* (ME) [1] that can be used as U_{dig}. However, none of the existing prototypes makes a distinction between the *personal trusted device* and the U_{dig}. We think, such a distinction should be made, because, although representing the user is the primary task of the *personal trusted device*, it is possible for the *personal trusted device* to run other applications as well. For example, the user interface for inspecting actions of U_{dig}, described in section 4 runs on the *personal trusted device*.

3.1.1 Smart Phone as Personal Trusted Device

A very strong means to give the user a feeling of control, is to let her physically control the hardware executing U_{dig}. This serves two purposes. First, as the hardware platform is under control of the user, manipulations become less likely. Second, the user builds up a trust relationship to her device, contributing to her feeling of control over her U_{dig}. Although assuming an emotional relationship between a human and a device may seem far fetched, [24] showed emotional attachment between users and their mobile phones exists.

Modern smart phones already run software dealing with very personal data, like messages or calendars. As the user thus trusts the device, it is an ideal candidate for hosting the U_{dig}. Further, modern smart phones provide ample resources in terms of computing power and network capacity to run a U_{dig} service.

3.2 The Mundo US as implementation of P_{dig}

Before going into details of our implementation of *association*, we briefly explain how a P_{dig} is implemented in the Mundo smart products platform. For this paper, we assume that the digitial representation P_{dig} of the smart product is actually executed on the hardware provided by $\texttt{P}_{\texttt{phys}}$ as this avoids many technical difficulties. Discussions about the various technical challenges that arise from this assumption can be found in [8,12,17]. The MundoCore [2] library, which serves as the basis for our implementation of P_{dig}, is however

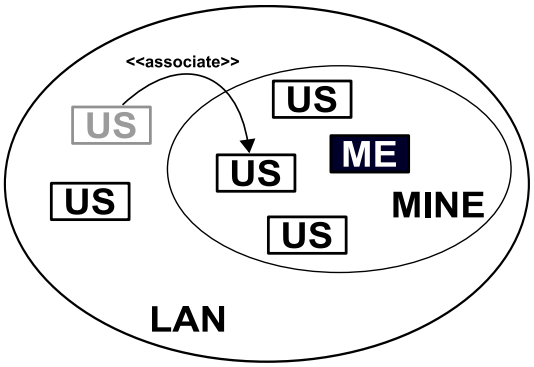

Fig. 3 Association transfers an US into the user's MINE. The MINE is a privileged zone provided by MundoCore. Transferring an US into the MINE zone may result in offering the user privileged functionality of $\texttt{P}_{\texttt{phys}}$, e.g. providing coffee, or of P_{dig}, e.g. providing newspaper article contents. US in the $MINE$ zone grant access to privileged functionality not only to the user or her ME but also to other US in the $MINE$ zone, allowing the creation of smart environments.

very portable and runs on Java, Java CLDC, x86 and ARM platforms. MundoCore has language bindings for C, C++, .NET, Java, and Python. P_{dig} is implemented on top of MundoCore as a Mundo service. A Mundo service is a process that communicates with his environment by exchanging MundoCore messages. Services can communicate regardless of platform and implementation language.

The key feature of a P_{dig} is that it is *associable*. For that reason, every P_{dig} must implement the Mundo interface for User aSsociable objects, short US. This interface provides two functions important for association and two functions for connecting associated US:

```
//Associate this US with the specified ME.
public boolean associate(ME me);
//Deassociate this US from the specified ME.
public void deassociate(ME me);
//Connect this US with another US.
public boolean connect(US us);
//Disconnect this US from another US.
public void disconnect(US us);
```

These functions are passed remote objects, that point to US or ME services running on other devices as actual parameters. The concrete implementation of these methods is up to the US developer, but must adhere to the properties defined above.

3.3 *Association* between ME and US

In this section, we go into detail for the various procedures that are necessary for establishing an association between a ME (as our implementation for P_{dig}) and an US (as our implementation for U_{dig}).

3.3.1 Mundo Association Procedure

Initially an US/P_{dig} is publicly accessible in the so called LAN zone. Using the MundoCore discovery mechanism, a ME can discover those US/P_{dig}s. Upon successful *association* the US/P_{dig} gets transferred from LAN into the ME/U_{dig}'s Individual Network Environment, short $MINE$ (see figure 3). US/P_{dig} in this $MINE$ zone cannot be discovered by any Mundo services outside this zone. Further, messages passed among members of this zone will be protected against eavesdropping by future versions of the MundoCore middleware. The *association* procedure described below ensures that access to the $MINE$ is only granted to eligible US/P_{dig}. In the Mundo platform *association* of the user with P is performed as follows:

1. The user decides if $(user, P) \in I_{hum}$.
2. She signals this to her ME/U_{dig} using an *authorization procedure* described in section 3.3.2. Thereby she instructs her U_{dig} to perform the rest of this procedure automatically.
3. The ME/U_{dig} calls the `associate` method of the US/P_{dig} it wishes to *associate* and passes itself as parameter.
4. ME/U_{dig} and US/P_{dig} follow a *qualification procedure* to establish mutual T_{dig}. *Qualification procedures* usually involve that the user provides something (information or money) to prove trustworthy to the US/P_{dig}. Conversely, the US/P_{dig} may also provide something to prove trustworthy to the ME/U_{dig}. *Qualification procedures* are further described in section 3.3.3.
5. If the *qualification procedure* succeeds, the US/P_{dig} is transferred into the $MINE$ zone and enables its privileged functions for all members of this zone. Additionally, the $\texttt{P}_{\texttt{phys}}$ belonging to the US/P_{dig} may grant access to privileged functions.
6. If the *qualification procedure* fails, e.g. if the US/P_{dig} rejects the association, this is signaled to the user.

The *association* can be terminated at any time by the US/P_{dig}, the ME/U_{dig}, or the user if anyone decides that the other party is no longer trusted. This is a unilateral decision, and our association semantics just foresee that the other party is notified.

3.3.2 Authorization Procedures

Using an *authorization procedure* the user signals her I_{hum} and T_{hum} for a P to her U_{dig}. By performing this, the user instructs her U_{dig} to negotiate mutual T_{dig} with P and *associate* the US/P_{dig}. This procedure should be as unobtrusive as possible, as the user should

seamlessly interact with smart products. Several *authorization procedures* can be implemented:

Static List The simplest form is to keep a list of trusted US/P_{dig} in the ME/U_{dig}. This list could for example contain personal items of the user like his MP3 player. Whenever an US/P_{dig} from the list is encountered, the ME/U_{dig} automatically tries to *associate* the US/P_{dig}.

Authorizing Real World Actions Another option is to employ techniques like SyncTap [15] or [10]. *Association* with a coffee machine could for example be performed when a cup is placed under the machine and a button on the ME/U_{dig} is simultaneously pressed. To do so, the *personal trusted device* must be equipped with a button and the US/P_{dig} must be equipped with a sensor for detecting cups. This option is slightly less convenient than automatically associating the US/P_{dig} but provides more control to the user. Also, ME/U_{dig} and US/P_{dig} must agree on the protocol used, e.g. SyncTap, to legitimate such a request. Future versions of the ME/U_{dig} will support different protocols. The used protocol can also determine the allowed *qualification procedures*, e.g. the user may allow her ME/U_{dig} to pay only up to 1 €for associations authorized via the SyncTap protocol.

Digital User Interface The last option is to provide a special trusted application with a user interface for performing associations. Operating on P_{dig} only via a user interface may seem a step backward from the goals of ubiquitous computing, however often the convenience of interaction on P_{dig} is preferred over the intuitiveness of interaction on $\mathtt{P}_{\mathtt{phys}}$. For example, when $\mathtt{P}_{\mathtt{phys}}$ is out of reach. To support direct operations on P_{dig}, we present a prototypical implementation of a user interface in the section 4. Also, confirmation in a user interface may be required for risky transactions.

3.3.3 Qualification Procedure

The role of the *qualification procedure* is to establish mutual T_{dig}. Several options for implementing this on the side of the US/P_{dig} and the ME/U_{dig} exist. Usually the US/P_{dig} will require some prove for the trustworthiness of the user. The ME/U_{dig} is responsible for delivering this prove of trustworthiness, if it has been authorized by the user. In turn, the ME/U_{dig} may require additional proves of trustworthiness from the US/P_{dig}.

Parameterizing Qualification Procedures The user may e.g. wish to only provide *valuable information* over secure connections. Thus, US/P_{dig} that do not support this cannot be *associated* with a protocol requiring such information. Also, parameters of the *qualification procedure* may depend on the *authorization procedures* used or the product P, e.g. *associations* requested by the user via the user interface may use higher amounts of money than *associations* requested by real world actions or the user may allow the ME/U_{dig} to give away her password to a certain P. We assume future US/P_{dig} and ME/U_{dig} provide a number of supported *qualification procedures* and determine automatically which can be used. The user should be able to specify these settings, however reasonable default settings need to be provided.

Example Qualification Procedures The most obvious qualification procedure for the US/P_{dig} is to create C_{dig} with every *paying user*. This can be easily implemented for prepaid scenarios. The user deposits an amount of money in her account, which is accessible by the US/P_{dig}. Every time the user wishes to associate the US/P_{dig}, it checks whether the funds in the account suffice. Another option is to *request secret information*, e.g. username and password or to *request valuable information* e.g. credit card data.

Incorporating Real World Action All these implementations of *qualification procedures* only check digital properties. For some use cases, this may be sufficient and e.g. physical access of the user to $\mathtt{P}_{\mathtt{phys}}$ must also be assured. For example, some secure *qualification procedures* may require the transmission of data via out-of-bounds channel, e.g. a *location limited* IR channel [4]. As the US/P_{dig} has access to the hardware of $\mathtt{P}_{\mathtt{phys}}$, it may detect e.g. button presses or access IR an IR detector. If using such an out-of-bounds channel requires user interaction, this can also be done at the stage of the *authorization procedure*.

4 MINE Manager

As stated before, the user should be able to inspect and control the current state of her $MINE$ network. For example, that she is able to check which US/P_{dig}s have been associated by her ME/U_{dig} or to end *association* with no longer trusted US/P_{dig}. Therefore, we propose to implement a *MINE manager* application, which serves as equivalent of a window manager known from desktop computing. Thereby, the MINE manager has two purposes:

- Visualizing the current state of the MINE to the user and
- Serving as a legitimate source of association and deassociation requests to the ME/U_{dig}

Fig. 4 Screenshots of the MINE manager interface running on the iPhone as an example of a *personal trusted device*. The left screen shows the list of available US/P_{dig} in the LAN zone. The middle screen shows the associated US/P_{dig} in the MINE. The right screen shows the details for a single smart product.

4.1 Prototype

The interface of the MINE manager is designed to be easily understandable for the end-user. Currently it is possible to browse the US/P_{dig}s in the LAN (left screenshot of figure 4) and MINE zone (middle screenshot of figure 4). The US/P_{dig} are presented in a simple list. US/P_{dig} in the LAN can be associated. A double click on the US/P_{dig} initiates *association*. The ME/U_{dig} has a special *authorization protocol*, which grants the highest priority to association requests performed in this way. As the MINE manager runs on the *personal trusted device* of the user along with her ME/U_{dig}, such a request cannot be faked.

Upon completion of association the ME/U_{dig} sends a message to the MINE manager, which results in listing the US/P_{dig} in the MINE part of the interface. Thereby, this message from the ME/U_{dig} may be due to *associations* initiated by other *authorization protocols*, e.g. through real world actions.

The right screenshot in firgure 4 shows the detail view for a single smart product. So far this is only a static text. However, we plan to extend the capabilities of the interface e.g. by introducing alerting functions, so that a smart product may raise the attention of the user, e.g. to signal that a coffee is ready.

5 An Associable Coffee Machine as Example For a Smart Product

Products with *associated* US/P_{dig} grant access to privileged functions. This can happen in the digital world, e.g. allowing the download of content after payment, or in the physical world, e.g. by providing coffee from a smart coffee machine only after *association*. To continue this example, we present multiple possible implementations of *association* for a smart espresso machine, of which one has been implemented.

Fig. 5 A coffee machine turned into a smart product. Sensors and communication electronics were added inside the machine's case.

5.1 Smart Coffee Machine Tally List Implementation

The tea kitchen of our department hosts a coffee machine. Users mark every consumed coffee with a tick on a tally list (the coffee account). With a smart coffee machine ticking the tally list is no longer necessary, as the machine can take count of the number of cof-

fees consumed. Implementing this on the basis of an *association protocol* is simple: A fixed list of employees represented by their ME/U_{dig}s are trusted. Conversely, the ME/U_{dig}s are set up to trust the coffee machine. This setup relies on a social protocol to make everybody pay the consumed coffee and to avoid faking of ME/U_{dig}s.

A prototype of such a machine was deployed in our department [7], see figure 5. The Mundo version used in this prototype did not yet provide the *association* mechanism and we could not deploy ME/U_{dig} services on personal trusted devices. Nonetheless, a form of *association* was implemented using the digital door transponder as a surrogate for the ME/U_{dig}. The *authorization protocol* was to press the transponder, which then transmitted its ID via an RF signal, while a cup was placed under the machine. *Association* was performed automatically. After serving the coffee, the machine terminated the *association*, so that it could be used by the next user.

5.2 Smart Coffee Machine Credit Card Scenario

A more complex coffee vending machine in a public space might require access to the users' credit card information. Depending on the choice of coffee, an amount is deducted by the smart coffee machine. This setup is more technologically advanced, as the coffee machine must communicate with a credit card clearing provider. However, this is already common place for online shopping. It is possible that an US/P_{dig} accepts multiple ways in which a ME/U_{dig} can qualify, e.g. members of the department staff are automatically qualified like above, whereas foreigners have to provide credit card data.

5.3 Association as a Locking Mechanism

Association can also just serve as a locking mechanism. A coffee machine may be configured to associate with any ME/U_{dig}, however once associated it is no longer available for others. In the case of the coffee machine, the machine cannot be used by others until the coffee is collected by the first user.

6 Related Work

In this section, we review related work to our framework for creating C_{dig}. First, we review how establishing C_{dig} is supported in existing frameworks and middleware for smart products. Then we examine existing approaches for the unobtrusive establishment of C_{dig} from other areas with regard to their applicability in a smart products scenario.

Existing frameworks and standards for the implementation of P_{dig}, like UDDI [21], UPnP [22], OSGi [13] do not provide a mechanism for establishing C_{dig} comprising all three aspects (mutual T_{dig}, T_{hum}, and I_{hum}). They do not foresee a special service for representing the user that could be used for facilitating creation of C_{dig}. Implementing U_{dig} as mobile code, e.g. with JINI [18] or OSGi contradicts the desire of the user to control the hardware platform executing her U_{dig}. UDDI does only specify the discovery of P_{dig} and does not provide a mechanism for association.

The GAIA plattform [16] or iROS [11] do not envision a U_{dig}, able to act on behalf of the user. A mechanism for creating C_{dig} is not provided. The mechanisms and concepts described in this paper can be used to implement a mechanism for establishing C_{dig} on top of these middlewares.

ReWiRe [23] represents users digitally. However, it does not explicitly foster trust between the user and her digital representation. ReWiRe does also propose the idea of a user interface for controlling all services currently connected to the user, which we think is important to let the user inspect her environment of smart products. ReWiRe allows connections between users and devices, however these are not used to provide the user with access to privileged functions.

Want et al.[25] proposes a *personal server*, carried by the all the time. As the users digital identity resides on a piece of hardware she controls, her trust in her digital identity is high. However, [25] only supports access to data on the *personal server* via resources in the environment and is not able to represent the user in qualifying for access to smart products.

The mechanism of *association* generalizes zero interaction authentication (ZIA) presented in [5]. ZIA works well for few often used products, like a personal laptop. In a smart products scenario C_{dig} to unknown smart products must be created frequently. Performing the bind procedure of ZIA is then too complicated. In *association* the bind procedure can be flexibly replaced with less obtrusive *authorization procedures*.

Qualification procedures in online transactions usually requires explicit interaction of the user, e.g. provision of a password. The browser can store the passwords, thereby playing the role of a U_{dig}. However, the browser is not always available when the user interacts with smart products. Other *qualification procedures*, more suitable for smart products scenarios like [14] or [20] could be integrated into the system. The benefit of *association* over any system relying on one

procedure, is that different procedures can be used according to the situation. From the viewpoint of the smart product developer *association* provides a single interface to all these mechanisms.

Storing this information on a central server is an approach followed by the MobileME [3]. Though, the MobileME is not able to react to physical actions of the user. Further, as the MobileME resides on foreign servers, the users' feeling of control is limited.

7 Summary and Outlook

We presented the mechanism of *association*, which establishes a *digitally checkable contract* between a smart product and a user. To do so in an unobtrusive way, we introduced a digital representation of the user U_{dig}, that can perform the tedious part of *association* on behalf of the user. The user can advise U_{dig} to perform association by performing unobtrusive physical actions, e.g. pressing a button on a smart product and on her *personal trusted device* simultaneously.

Our implementation of U_{dig}, the ME emphasizes the importance of executing U_{dig} on a *personal trusted device* of the user, to address the trust considerations of digitally representing a user. For further improving the users' feeling of control, we introduced a $MINE$ manager interface, that allows the user to manage and inspect her *associations* to smart products.

Currently, we assume P_{dig} is executed on the hardware of $\mathtt{P}_{\mathtt{phys}}$. Relaxing this requirement would allow many more products to become smart products. However, ways to maintain the link between $\mathtt{P}_{\mathtt{phys}}$ and P_{dig} must be found. An easy solution is to run the P_{dig} service on the computers of the manufacturer, accessible over the internet and to embed an ID and a pointer to the service in $\mathtt{P}_{\mathtt{phys}}$, e.g. via RFID. This raises some questions, e.g. whether it should be possible to interact with P_{dig} while one has no access to $\mathtt{P}_{\mathtt{phys}}$ or how to change $\mathtt{P}_{\mathtt{phys}}$ as a result of operations on P_{dig}.

References

1. E. Aitenbichler, J. Kangasharju, and M. Mühlhäuser. Talking Assistant: A Smart Digital Identity for Ubiquitous Computing. In *Advances in Pervasive Computing*, 2004.
2. E. Aitenbichler, J. Kangasharju, and M. Mühlhäuser. MundoCore: A Light-weight Infrastructure for Pervasive Computing. *Pervasive and Mobile Computing*, 2007.
3. Apple MobileME. http://www.apple.com/mobileme/.
4. D. Balfanz, D. K. Smetters, P. Stewart, and H. C. Wong. Talking to strangers: Authentication in ad-hoc wireless networks. In *Proc. of NDSS*, 2002.
5. M. D. Corner and B. D. Noble. Zero-interaction authentication. In *Proc. of MobiCom*, 2002.
6. W. K. Edwards, M. W. Newman, J. Sedivy, T. Smith, and S. Izadi. Challenge: recombinant computing and the speakeasy approach. In *Proc. of MobiCom*, 2002.
7. J. Kangasharju G. Austaller, E. Aitenbichler. Interaction with a smart espresso machine. In *Demo at IEEE PerCom*, 2006.
8. H. Gellersen, G. Kortuem, A. Schmidt, and M. Beigl. Physical prototyping with smart-its. *IEEE Pervasive Computing*, 3(3), 2004.
9. A. Hartl, E. Aitenbichler, G. Austaller, A. Heinemann, T. Limberger, E. Braun, and M. Mühlhäuser. Engineering Multimedia-Aware Personalized Ubiquitous Services. In *IEEE MSE)*, 2002.
10. L. E. Holmquist, F. Mattern, B. Schiele, P. Alahuhta, M. Beigl, and H.-W. Gellersen. Smart-its friends: A technique for users to easily establish connections between smart artefacts. In *Proc. of UbiComp*, 2001.
11. B. Johanson, A. Fox, and T. Winograd. The Interactive Workspaces project: experiences with ubiquitous computing rooms. *Pervasive Computing*, 1(2), 2002.
12. J. C. Lee, D. Avrahami, S. E. Hudson, J. Forlizzi, P. H. Dietz, and D. Leigh. The calder toolkit: wired and wireless components for rapidly prototyping interactive devices. In *Proc. of DIS*, 2004.
13. OSGi Alliance. http://www.osgi.org.
14. Shwetak N. Patel, Jeffrey S. Pierce, and Gregory D. Abowd. A gesture-based authentication scheme for untrusted public terminals. In *Proceedings of the 17th annual ACM symposium on User interface software and technology*, pages 157–160, Santa Fe, NM, USA, 2004. ACM.
15. J. Rekimoto. Synctap: synchronous user operation for spontaneous network connection. *Personal Ubiquitous Comput.*, 8(2), 2004.
16. M. Roman, C. Hess, R. Cerqueira, A. Ranganathan, R. H. Campbell, and K. Nahrstedt. A Middleware Infrastructure for Active Spaces. *Pervasive*, 1(4), October 2002.
17. M. Strohbach, G. Kortuem, and H.-W. Gellersen. Cooperative artefacts-a framework for embedding knowledge in real world objects. In *Proc. Ubicomp Workshops*, 2005.
18. Sun Microsystems. Jini architecture specification version 1.2, http://www.sun.com/software/jini/specs/jini1.2html/jini-title.html.
19. F. Thiesse and M. Köhler. An analysis of usage-based pricing policies for smart products. *Electronic Markets*, 18, 2008.
20. Julie Thorpe, P. C. van Oorschot, and Anil Somayaji. Passthoughts: authenticating with our minds. In *Proceedings of the 2005 workshop on New security paradigms*, pages 45–56, Lake Arrowhead, California, 2005. ACM.
21. Universal Description, Discovery, and Integration Consortium (2000). Uddi version 3.0.2, http://uddi.org/pubs/uddi_v3.htm.
22. UPnP Forum. Upnp basic device specification, http://www.upnp.org/standardizeddcps/basic.asp, 2002.
23. G. Vanderhulst, K. Luyten, and K. Coninx. ReWiRe: Creating Interactive Pervasive Systems that cope with Changing Environments by Rewiring. In *Proc. of IE*, 2008.
24. J. Vincent. Emotional attachment and mobile phones. *Knowledge, Technology, and Policy*, 19(1), 2006.
25. 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. of UbiComp*, 2002.