Individual and Social Recommendations for Mobile Semantic Personal Information Management

Wolfgang Woerndl, Georg Groh, Aleksandar Hristov
Institut für Informatik
Technische Universitaet Muenchen (TUM)
85748 Garching, Germany
{woerndl,grohg,hristova}@in.tum.de

***Abstract*— We present an approach for personal information management for mobile devices like PDAs based on the Semantic Desktop. The main objective is to design and realize a recommendation system to identify interesting items (e.g. messages or documents) based on the current context (time and location) and a user's personal ontology. To do so, our algorithm uses an evaluation function to traverse the graph of resources and rank nodes. Relevant resources and other items such as points-of-interests can also be displayed on a map on the mobile device. The ideas have been implemented and (rudimentarily) tested in the "SeMoDesk" application. Furthermore we introduce the extension of this application with the respect of social computing by a social information item filtering approach. The social filtering allows for the integration of other user's information spaces and makes use of a special bootstrapping algorithm for the integration of heterogeneous ontological perspectives.**

Keywords-personal information management, semantic desktop, social filtering, mobile, context

I. INTRODUCTION

Personal information management (PIM) is intended to support the activities people perform to organize their daily lives through the acquisition, maintenance, retrieval, and sharing of information [1]. Examples for personal information include documents, (Email) messages, contact data, appointments and also references to other information items. The collection of these items is called the personal space of information (PSI) [2]. The notion of personal information management first appeared in the 1980ies [3]. Since then, PIM tools have been developed and used. Due to increasing information overload of users during the last 15 years, the interest in PIM has been boosted very much.

Organizing data and having access to relevant information is particularly important in a mobile scenario, e.g. field staff meeting customers. To support these tasks, PDAs (personal digital assistants) and other mobile devices are available. However, organizing information on mobile devices is even more difficult when compared to a desktop setting. This is mostly due to the fact that mobile devices have limitations in network bandwidth, storage capacities, displays and input capabilities. For example, users cannot browse though many search results on the small screen of a mobile device. Therefore, it is very important to adapt information access to the current user needs and context in a mobile scenario.

Yet most of current PIM research is not geared towards mobile and ubiquitous information access. Therefore, the goal of this work is to support the user in mobile personal information management. More precisely, we want to design and implement a recommendation system to recommend resources to a user that are of current interest to her in a given context. For that purpose, the rest of the paper is organized as follows. First, we describe the background of our work, namely the Semantic Desktop. In Section III, we explain the main ideas behind our recommender. We also present the user interface of the prototype and give some implementation details. We also explain how to use the infrastructure to improve the context- and location-awareness of mobile PIM. In Section IV, we present the extension of the application to allow for including information spaces of other users in the personal social network (possibly in the near vicinity of a user's own mobile device) into the context sensitive item filtering. In Section V, we discuss related work. Finally, we conclude with a brief summary and outlook.

II. BACKGROUND: THE SEMANTIC DESKTOP

One solution to deal with mobile PIM is building onto the *Semantic Desktop*, an approach aiming to integrate desktop applications and the data managed on desktop computers using semantic web technologies [4]. The main idea is to assign meta data to all data objects that a user uses on her computer. Thereby, relations between resources can be defined with the goal to integrate desktop applications and enhance finding relevant information.

Semantic Desktop approaches rely on ontologies to formalize relationships between resources and define a concept hierarchy that can be utilized for information retrieval. For the Gnowsis project, the "Personal Information Model" (PIMO) ontology was designed [5]. We have based our application on the PIMO ontology. The overall goal of PIMO is to define a concept hierarchy allowing a single user to formulate her view on tasks, contacts, projects, files and other resources.

In PIMO, one basic idea is to distinguish between "Thing" and "ResourceManifestation". "Thing" is a superclass of abstract concepts and physical objects, with the goal of representing them on a conceptual level. "ResourceManifestation" is a class to represent the actual documents on a computer system [5]. All objects in PIMO can be connected to each other using relationships, which we explain in more detail in Section III.

While there are Semantic Desktop implementations and related systems like the aforementioned Gnowsis available for desktop computer use, there is little for mobile environments. Therefore, we have designed and implemented SeMoDesk, which is a realization of the Semantic Desktop idea for PDAs [6]. The main design goals were to account for the restricted resources of PDAs, to build a stand-alone application (i.e. not a client of a Semantic Desktop server solution), because of possible network limitations, and adaptation to and usability on the mobile device. For example, phone calls and SMS messages are integrated which is not the case in related, desktop based approaches.

To assist the personal information management, the classes and instances of the personal ontology can be browsed in SeMoDesk. For example, all messages or calls with one person, or all resources such as document or appointments that are associated with a project, can be displayed with one tap on the touchscreen of the mobile device. However, browsing the ontology is not enough, as the following example illustrates. If a user is in a meeting right now, she might not only be interested in documents that are directly related to this meeting, but also messages that are related to a project or a person that is related to the meeting, or contacts that are concerned with a relevant topic, and so on. The goal of this work was to design and realize this kind of recommendation method which will be explained in the next section.

III. INDIVIDUAL RECOMMENDATION OF ITEMS

In this section, we first briefly describe how to manage the ontology, and then we explain the details of the recommender system that proposes resources to users based on the current context (location and time).

A. Managing the personal ontology

After starting up SeMoDesk, the user has the options to manage and browse her ontology, recommend resources of current interest, or display items on a map (cf. chapter III.C.). For the first task, users can and ought to define concepts such as projects, topics or subclasses of a “person” concept (e.g. “personal”, “work”) based on the PIMO ontology. Fig. 1 shows the top level of the ontology for browsing.

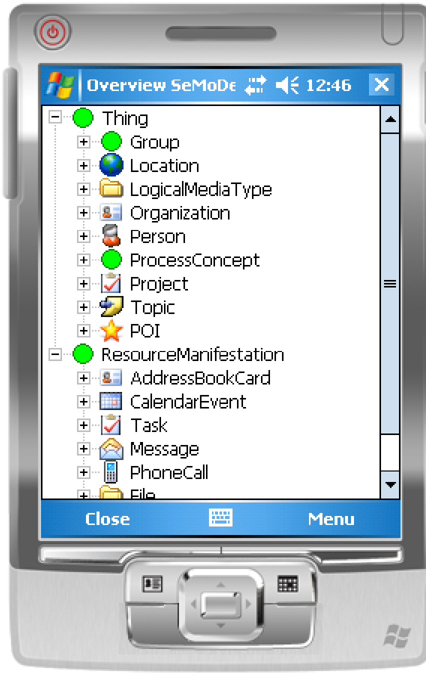

Figure 1. Browsing the ontology

Fig. 2 is example of the details of a location instance. Showing and editing items is possible by either taping on an item in the ontology (a context menu will then open), or by using the “Menu” soft key.

Figure 2. Displaying details of an instance

Afterwards, users can define relationships between the concepts and resources on the mobile device, for example stating that an appointment is related to a project. SeMoDesk was designed to assist the user in this task. In the example of a phone call (Fig. 3), if the caller's phone number can be found in the address book, a relationship between the phone call resource and the person concept is created automatically. The relations can then be easily retrieved. Fig. 4 shows the direct relations for a person.

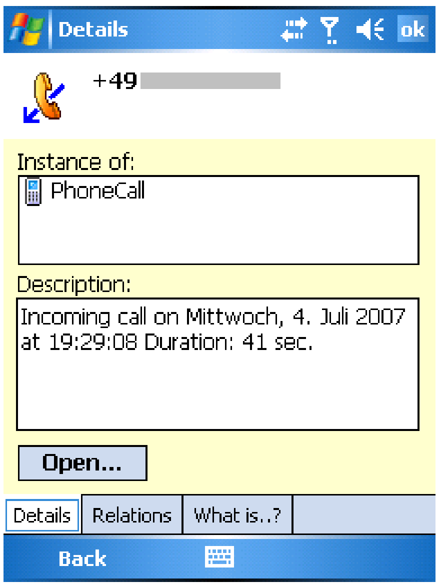

Figure 3. An incoming phone call

Figure 4. Displaying existing relations of an instance

The part of SeMoDesk application for managing the ontology from a user perspective, and the underlying system design, are explained in more detail in [6].

B. Recommender algorithm

1) Overview: The personal ontology we explained above forms a graph with a Thing and a ResourceManifestation as the root nodes. To find recommended resources in this graph, we first have to find appropriate starting nodes for our search. Thus, our basic idea for the recommender consists of the following two steps:

1. Finding current resources (Fig. 5), i.e. resources that are of interest for the user right now
2. Recommending other items, starting from the instances found in step 1

Figure 5. Finding current resources

For step one, our system offers different options. First of all, the user can manually select concepts or resources. To do so, the user can browse the "Concept" or "Resources" parts of the ontology (see the taps in the lower part of Fig. 5). For example, the user can select a topic or a document that is of current interest to her. In addition, the system proposes items in this first step based on the current date and time (button "On Schedule" in Fig. 5, and location (button "In Area"). For location-awareness, the user can define locations (e.g. an address) and assign appointments or other resources to these locations (Fig. 6).

Figure 6. Assigning appointments to locations

As a result of this first step, the system displays a list of resources which are of current interest to the user (middle part in Fig. 5). This list may already contain relevant resources, but the nodes in the list mainly serve as possible starting nodes for the more advanced search in step 2. The user can now select one node and start the recommendation process. The second step, finding additional relevant resources from the specified starting node, will be explained in more detail later in this section.

We like to underline the fact that the search algorithm only uses the relations between entities the user has defined. The system does not analyze the items themselves. However, relations between entities in SeMoDesk could be proposed by the system based on content analyzed, e.g. matching text in documents. Our algorithm searches for resources (e.g. appointments, documents, messages), not concepts. However all nodes and relations in the graph including concepts are used in the inference process.

2) Relations and relation types: As explained above, the personal ontology represents a directed graph. Thereby, it is necessary to distinguish between different relation types. We are using the following types as an extension and refinement of PIMO:

- Related: entities having any (weak or strong) relationship
- HasSubClass / IsSubClassOf: a class that extends the given class or is an extension of a super class
- HasPart / IsPartOf: for example, a person can be part of a project
- HasInstance / IsInstanceOf: any object is an instance of a class
- HasOccurrence / IsOccurrenceOf: if a person is mentioned in an article, for example, we say that this person has an occurrence in the document
- HasMade / IsMadeBy: a person has made a phone call, or a phone call is made by a person
- HasTopic / IsTopicOf: a given document or a project for instance can be assigned a topic
- HasCard / IsCardOf and HasBusinessCard / IsBusinessCardOf: denotes that a person or an organization has an address card

3) Finding a meaningful path: The main difficulty in our concept is how to traverse the directed graph of concepts and resources from a selected starting node. For doing so, it can be observed that not all edges are equally meaningful. Therefore it is important to apply heuristics to find out interesting edges to follow. The problem is illustrated in the example depicted in Fig. 7. In this example, we have two appointments which are instances of the CalenderEvent concept. We start with appointment 1 (“Meeting with Alex”) and seek related nodes. “Meeting with Alex” is related to a project that is also “occurrence” of a topic. This topic is discussed in our second appointment “Meeting with Vladimir” in the example as well. Because of this path between appointment 1 and 2, the system should infer that both appointments are related and could recommend appointment 2 if the user is currently in appointment 1.

But there is also a shorter, but less significant path from appointment 1 to appointment 2 because both are instances of CalenderEvent. We don’t want the system to follow this path because then all appointments would be related to each other and a recommendation is likely to be rather pointless. Therefore we have to put different weights on different relations (cf. III.B.5) thus penalizing the ones which are expected to be less meaningful (In this case the Instance-Of edges).

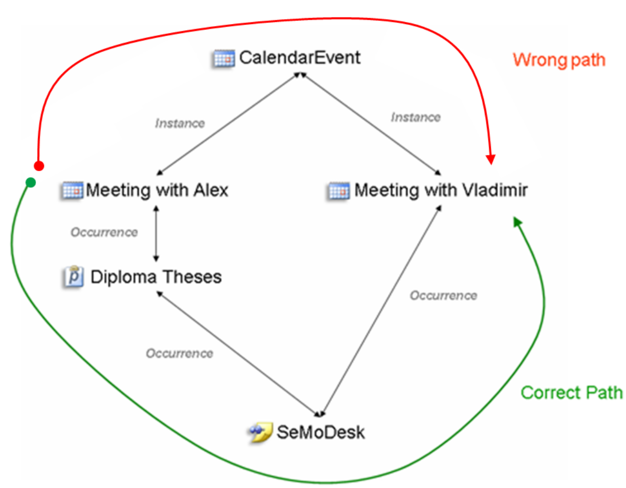

Figure 7. Finding a meaningful path

4) Search algorithm: For the purpose of finding an efficient search algorithm, we have examined common algorithms used in graph theory and Artificial Intelligence research [7]. As is well known, the algorithms can be roughly grouped into informed and uninformed (or blind) approaches. The latter just traverse the whole graph until satisfying results are found, e.g. depth or breadth first search. Informed algorithms apply heuristics to predict the distance to the target node e.g. A* algorithms. For reasons discussed in the previous paragraph we introduced different weights for edges. But for weighted graphs, simple BFS does not yield shortest paths. We could use a "semi-informed" Dijkstra algorithm (time complexity O(n log n+ m)) to find the shortest weighted path form a starting node to a target node that is currently beeing taken into consideration but the complexity is critical for a mobile application if the information space becomes larger.

As a compromise due to the limited resources on the mobile device and potentially many nodes in the ontology graph, we have decided to nevertheless use breadth first search (O(n+m)) in combination with an evaluation function to rate the quality of the expanded nodes. By doing so, we can avoid putting too much effort in following rather meaningless nodes as motivated above in III.B.3 by using BFS to traverse the graph and examining the weight of each node with our evaluation function. Our BFS does not expand nodes whose evaluation value is below the threshold and thus implements an approximation to the target behavior described in III.B.3. The evaluation function allows for incorporating further heuristics than just relation weights (see below).

The evaluation function returns the goodness on a node, in relation to the starting node. That's why our approach relies on specifying one starting node to trigger the search. Our algorithm terminates after all relevant nodes have been analyzed. Neighbors of "bad" nodes – i.e. nodes with an evaluation result below a threshold – are ignored (bad nodes are not expanded) and therefore the amount of expanded nodes is significantly reduced. If search time is a crucial criterion however, the algorithm could easily be modified to use an iterative deepening approach, which would allow termination after a maximal search time duration has elapsed.

5) Evaluation function: Our evaluation function is not only looking at the node that is being analyzed and its neighboring relations, but also at the shortest unweighted path from the starting node found by BFS. We have defined the evaluation function f for this analysis as follows:

$$f = a * depth + b * concept + c * relation$$

a, b and c are parameters to weigh the three factors:

- depth: is computed from the length of the (shortest) unweighted path from the starting node to the current node by weighting each edge by a fall-off coefficient. The fall-off coefficient can be configured in the system, for example we start with 1 and divide by 2 in every further step. Thus depth would be 1 * 0.5 * 0.25 for a path with three edges.
- concept: weight of the node itself, depending on the type of the concept or resource, e.g. less common resources have a higher weight
- relation: Summed up weight of the edges of the shortest unweighted path to the node, where different relation types (corresponding to the edges) can have different weights, as motivated above in chapter III.B.3.

The higher a node is evaluated by the function f, the more likely this node is relevant in the current context. All parameters can be configured in the user interface of our prototype implementation. However, an ordinary user is not supposed to configure the parameters herself, but use a predefined set of reasonable parameters based on the application scenario. For example, in a scenario with a lot of messages but fewer other items, message nodes may have a lower weight.

The algorithm starts examining the nodes with an evaluation value of 1. For each node, the algorithm loops through the whole path back to the starting node, and updates the value of the evaluation using the coefficients for each level. The result of this calculation is then returned to the traversing algorithm, which decides whether to put the node in the result set and whether to expand the node at all.

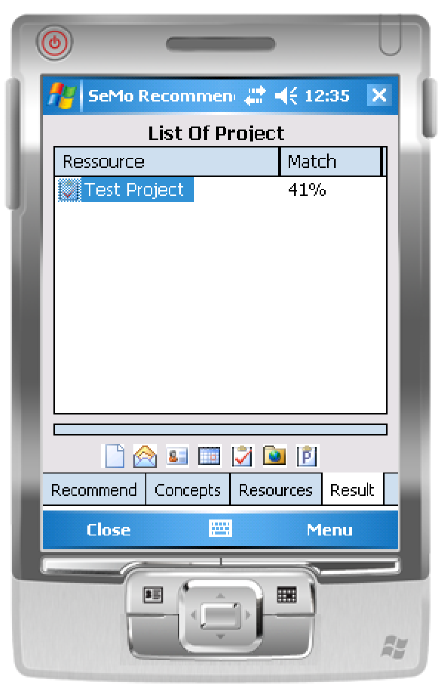

Figure 8. Search result

6) Results and their explanation: After searching the item space as explained above, a result list is shown to the user (Fig. 8). The result set is ranked by how relevant the items are, according to the search process and evaluation function ("match"). Only items with a match above a configurable threshold are given as results. In the example in Fig. 8, only one resource with a match of 41% is recommended.

In the lower half of the result screen, there are seven icons to toggle the type of items a user is currently looking for. The available options in our implementation are: Documents, Messages, Contacts, Calendar, Events, Tasks, Bookmarks and Projects. This selection is in addition to the different weights that nodes have in the evaluation function (see III.B.4). While messages may have been assigned a lower weight in the evaluation function in general, the user is still able to search for relevant messages, for example.

Figure 9. Relation path

The user has the option to display the relation path by taping on a result item (Fig. 9). Thus, the user can comprehend why this particular item was recommended. We consider this explanation an important part of the user interface. In general, it is desirable to explain results to the user in personalization and recommender systems, as studies have shown (e.g. [8]).

C. *Using the personal ontology to recommend additional items based on location and time*

Until now, the explained concepts allow searching the item space in the personal ontology. But the ontology can also be utilized to recommend additional resources, items that are not explicitly managed by the user. The application scenario is that users are looking for points-of-interests (POIs) in the current geographic vicinity to perform certain tasks. For this purpose, we have extended the PIMO ontology by a POI concept with sub concepts such as "cinema", "restaurants", "shop" etc. The user can then relate tasks or any other resources to POI types, as shown in Fig. 10. In addition, appointments (or any other resources, in theory) can be related to addresses (see above Fig. 6). When the user starts the mapping feature of SeMoDesk, information about relevant POIs are shown on a map (Fig. 11), together with the location of upcoming appointments.

Figure 10. Associating a task with a POI concept

There are several possibilities to retrieve the current user position to select the appropriate map segment and center the map on the user position. More and more mobile devices are equipped with GPS, for example. While the rest of SeMoDesk runs as stand-alone application on the PDA without any server, an Internet connection is required for the mapping feature.

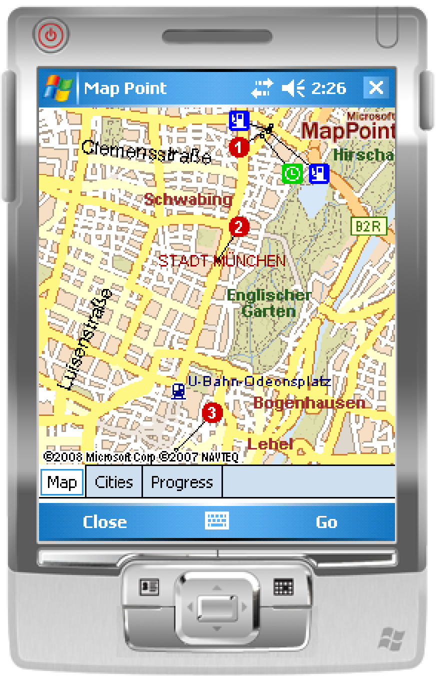

Figure 11. Displaying the map

D. *Indoor location-awareness using RFID infrastructure*

Another goal is to improve the indoor context-sensitivity of the system. We have worked with a RFID infrastructure to be able to locate the user indoors more precisely (e.g. displaying relevant resources when she enters a meeting room) [9].

To accurately model a user's position, a system requires a detailed location model where even small places can be distinguished. This results in the demand for a location model with a diversified granularity. As an extension to the ontology, we hierarchically designed the following classes for the location ontology:

- Country
- City
- Area
- Street
- Place
- Building
- PartOfBuilding
- Level
- Floor

This wide range of granular diversity allows us to model huge areas as well as small spots inside a building. The location taxonomy can be implemented by using HasPart – IsPartOf relations. Additional relation types like IsNearby or IsOnThe-RightSideOf/IsOnTheLeftSideOf can easily be added in the future. In addition, we needed to model location sensors and receivers respectively identification tags. To do so, we have designed a superclass "SensorTags" and subclasses, depending on the technologies used for positioning [9]:

- Sensor: Represents a stationary sensor (e.g. RFID)
- IP: IP entities are linked to Sensor entities and hold an attribute with the IP address to reach a specific sensor on a network (e.g. WLAN).
- RFID Tag: RFID tags represent the devices or objects they are linked to in the ontology. They hold a numerical ID.
- Bluetooth Tag: Bluetooth combines sender and receiver in a single unit that is built-in in electronic devices. Unique numerical tags identify those units.
- GPS Coordinates: Entities of this class hold coordinates for a certain place and are linked to either a Place or a Building entity.

A user is identified by a device with an RFID tag in our model. For reasons of simplicity we decided not to separate user and device positioning. All sensor tags shall be set in direct relation to the user whose devices they are attached to.

The button "In Area" (Fig. 5) is the user interface to search for a certain area. This integrates our location system with the resource recommender that was explained in chapter III.B. Hence, the recommendation system is extended with location awareness. When the user clicks on "In Area", a list of currently available sensors is shown to user. The user can then select one of them, for example a sensor assigned to a meeting room she is interested in right now. What the system does is provide a list of tags, a list of persons in a certain area and also a list of all the resources related to those persons. How this infrastructure can be used to find resources related to a position will be shown below in an example use case (cf. III, F.).

E. *Implementation details*

Our implementation was done using Microsoft's IDE Visual Studio 2008. The programming language is C# and the runtime environment is .NET Compact Framework 3.5. The application was tested using a HTC P3600 PDA phone and some other similar devices. SeMoDesk should run on any Windows Mobile 5 or 6 PDA with a touchscreen interface. For the POI search and mapping feature (chapter III.C.) we are using Microsoft's MapPoint. Other similar services could be integrated easily.

Fig. 12 gives an overview over the main layers of the system design [10]. To store all data, our application uses a SQL Server Mobile Edition as backend database. The main parts of SeMoDesk are components needed for the graphical user interface (GUI), a representation of the PIMO objects and corresponding data provider classes for the database access. Every item in our approach has a GetRelations() method that retrieves all corresponding relations in an efficient manner, for example. The "AI" package contains all the classes of the search and recommendation algorithm as explained above.

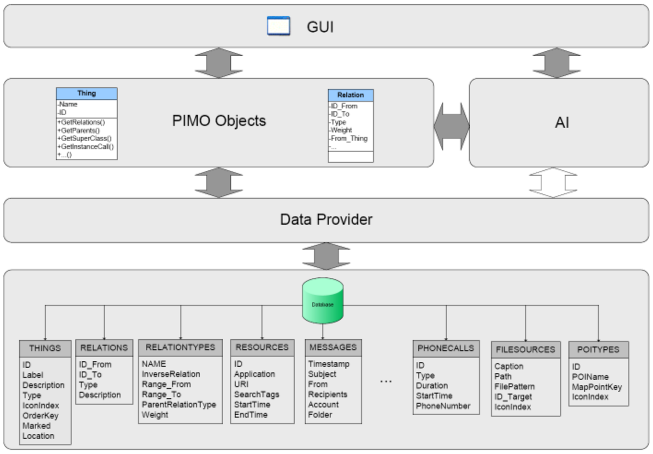

Figure 12. Architecture overview

F. *Use Case: Displaying resources that are related to a room*

Finally, we want to show how the explained infrastructure including the RFID part can be used to put the following use case into practice: "A user enters a meeting room and want to find resources that are related to this room". We have used the short-range HF RFID reader Tricon Starter Kit 100 to test this scenario [9].

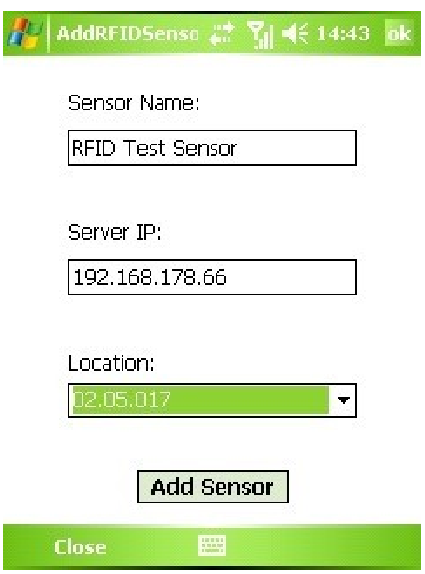

Figure 13. Setting up a new RFID sensor (left)

For this purpose, we set up our ontology to model the sample hierarchy of our university. An RFID sensor was added (Fig. 13) and associated with room 02.05.017.

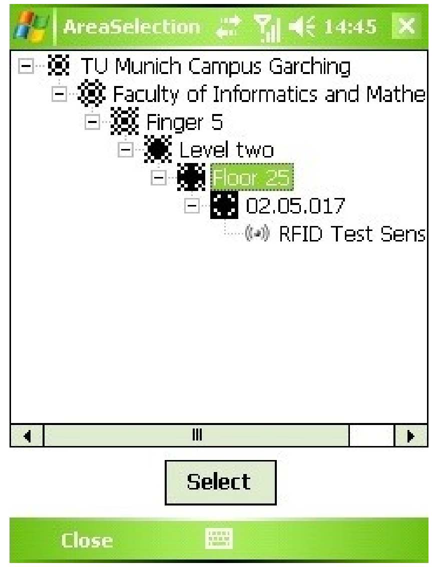

Figure 14. Selecting a floor in the location hierarchy (right)

Then, we started the location-based recommender using the aforementioned "In Area" button. In the appearing AreaSelection window (Fig. 14) we chose the higher location node "Floor 25" which included all sub nodes, i.e. our room with the RFID reader. SeMoDesk connected to the server linked to that reader and received a list of tags in its range (Fig. 15). The one tag that was transmitted turned out to be associated with contact Diane in our ontology. By traversing the relation graph the recommender then determined all entities directly related to the contact (Fig. 16), namely the task "Thesis" and also the associated RFID tag.

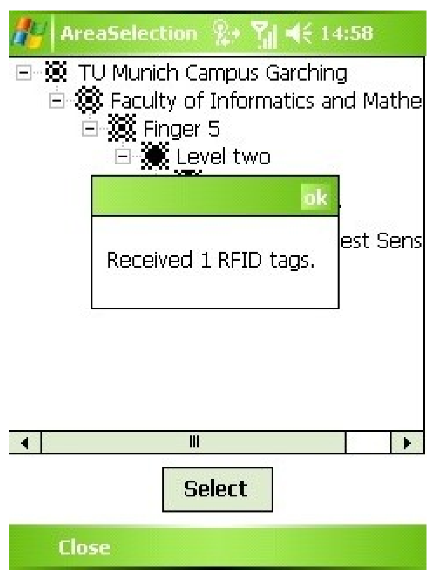

Figure 15. Receiving RFID tags (left)

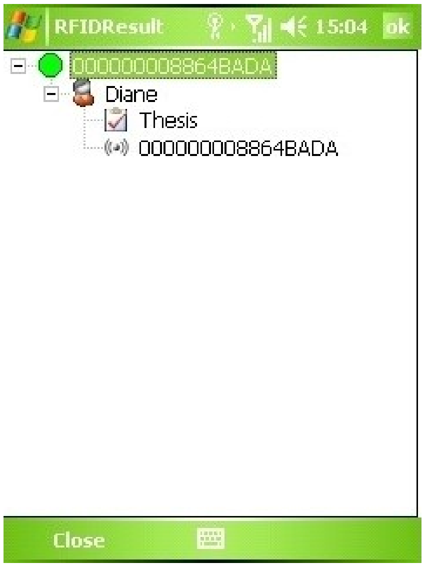

Figure 16. displaying recommended entities (right)

IV. SOCIAL RECOMMENDATION

In this Section IV we explain the extension of our recommendation function with social filtering.

A. *Overview*

Modeling social relations and using the resulting network models for social computing services has become a major trend in Web 2.0 [10]. Although several attempts have been made to apply the social networking paradigm to mobile interaction scenarios, truly convincing applications in this field are still largely missing. We aim at contributing to the research for better mobile social networking applications by investigating possible extensions to our existing mobile personal information management application. One promising option is to extend SeMoDesk with social item filtering, towards realizing the Social Semantic Desktop idea [11][12].

The meaning of the term Social Recommender can be twofold: On the one hand, it can designate systems that recommend social structures that maybe useful for a user. This includes friend-recommenders or team-recommenders [13]. On the other hand, it can designate approaches where the neighborhood from which recommendations are generated is not chosen by e.g. selecting the n other users with the most similar rating behavior, but rather chosen based on the social network of a user [14]. Groh and Ehmig found that social recommendations in the latter sense worked better than state of the art collaborative recommendations in taste-related domains [14]. Follow-up experiments indicated that for more fact-related domains the approach might not work as well. However, in both cases standard cross–validation evaluation methods for recommenders were used that are not able to value one key advantage of social recommenders: Horizon broadening recommendations [14]. By that we mean recommendations made by the close social network of a user, that the user might not like at first sight but that e.g. may help him in "complying to tendencies in his peer group" or that he may like "on second sight" considering the social relations to those people whose ratings have mainly contributed to this recommendation having been made. These horizon broadening effects can make those recommendations useful.

The key idea of our social filtering approach is to extend the information space, which is subject to the information item filtering, by including parts of the information spaces of other users. To do so, we use the social network composed of the community's individual's contacts, already present in the basic application. The resulting network thus has directed edges, which can optionally be weighted (automatically or via an additional module that assigns weights by counting the relative number of communication acts with the respective person).

The overall process of information sharing, which can be mediated by all available network infrastructures (Bluetooth, WiFi Ad-Hoc-Mode etc.) is then very simple: An "inquirer" asks for related nodes to a specified node A from his own information space. While we generally limit the possible set of "inquired" persons to those that have a mutual social relation to the inquirer, we provide two more special modes for the determination of the actual set of "inquired" persons (besides the option to ask all mutual contacts): The first mode inquires all mutual contacts that are additionally located in the physical neighborhood. The idea behind limiting the possible set of to-be-inquired nodes to the immediate physical neighborhood is that this mode supports an additional social control about who might be inquired or who a user is inquired by. In the second mode all persons with which the mutual relationship is either of type business relation or personal relation are inquired. Depending on the type of current network access only a subset of the three basic types of inquiry may be supported.

In order to realize the incorporation of foreign information spaces we extend the PIMO ontology. While the problem of providing or collaboratively constructing an agreeable ontology of social relations is still at least partly open, we propose a basic intermediate solution by extending the PIMO ontology by introducing sub-concepts of "contact", namely "business contact" and "private contact". Thus by effectively assigning these relation-types to the corresponding edges we create a partition of the set of contacts that is applicable to the vast majority of social relations. It is nevertheless planned for later versions to further enhance the portfolio of relation types.

A second extension of the ontology regards the question, which elements of the personal information space are made "publically" available. In order to allow the user to control this option in more detail, we introduce Boolean attributes "socializable" (with sub-attributes "business socializable" and "privately socializable") for every element of the personal information space with a default setting of FALSE.

The filtering (or recommendation) process delivering related items to a specified item from the personal information space can be managed by the inquirer by additionally specifying whether she wants to include other user's information spaces at all and, if so, whether to include only business-contacts or only private contacts or both. Furthermore it can be specified whether only the local physical neighborhood is inquired or whether all contacts are inquired.

Since it cannot be generally assumed that many inquired users are willing to open their information spaces to the general public, we assume in our first version, as has been explained before, that the opening is confined to direct contacts. A more elaborate alternative would be to allow each user (in addition to the per-item-“socializable” attributes) to specify as a policy whether the opening is confined to the set of direct contacts (“restricted”), to the general public (“public”), or to a network of path length at most n away from her (“intermediate”(n)). In order to implement the last policy we have to include information about the social network path into the inquiry-element of the agent interaction protocol.

B. Bootstrap approach

After having determined the set INQ of inquired persons, the social extension of the filtering or recommendation process then follows a “bootstrap” approach: Assume that person X seeks items related to her own item A in the information spaces of the persons in INQ. Assume further that person Y is in INQ . Then the sequence is as follows:

1. If Y does not turn down X’s request, on request of agent X, Y virtually includes X’s item A into his own information space. Virtual inclusion encompasses all “agreeable” semantic item relations from the common PIMO ontology that are present for A in X’s information space. “Agreeable” relations have targets and types that are present in both information spaces.

2. Y then computes a set of related items from his information space with the algorithm described in the previous section with (virtual) start node A. With that step we find related items to A from Y’s “perspective”. After the computation, Y deletes the virtual node from her information space.

3. Y communicates the result set (with those parts of paths of the results which are agreeable) back to X

4. X virtually includes the result items from Y and other agents into her own information space and runs another instance of the filtering/ recommendation algorithm (restricted to those “foreign virtual nodes”) with start node A. This step yields related items to node A from X’s “perspective”. Overall we thus realize a common “perspective” of X on the one side and Y and the other agents on the other side.

Step 4 of the algorithm is optional and ensures that really a common perspective of relevance is established. The Application of Person X can also be configured to omit this step and trust Y’s perspective.

C. Example

The approach is illustrated with an example in Fig. 17. Person X seeks for related items to his “Eigner”-node. Shown in red are Root concepts from PIMO, shown in Blue are our PIMO Extensions. Red edges denote “is-a”-relations, blue edges denote “isInstanceOf” relations. In the example we assume that Y is organizationally related to X and has the same project “OSMOZIS” in his graph. After the virtual insertion, Eigner’s agreeable relations (in Fig. 17 one such relation is indicated as a dashed edge) are virtually included into Y’s information space. Then Y’s application applies the recommendation algorithm and sends the results to X’s application. These results represent Y’s view on what relates to “Eigner” in his information Space and should be recommended. For example this could be the “Palin” node of a person also working in his part of the “OSMOZIS-project”. “Palin” is sent together with the path <“Eigner” → “Project OSMOZIS” ← “Palin”>. If X’s application is configured to perform step 4, Person X’ application will then insert “Palin” under his “Project OSMOZIS” and re-run the recommendation process, which might or might not yield “Palin” as a relevant node depending on the structure of X’s information space.

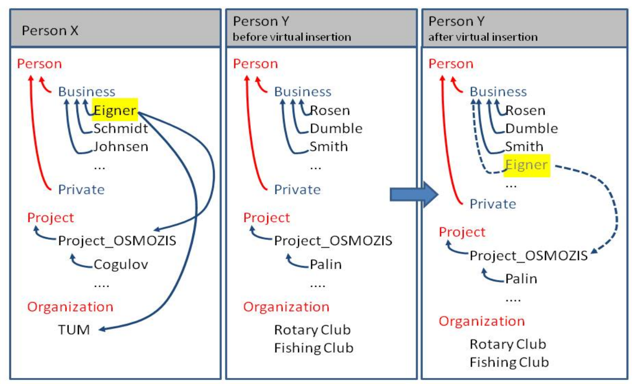

Figure 17. Example for the virtual insertion procedure

Actual access to foreign information items resp. nodes is subsequently implemented by a separate process, if demanded by user X.

In the first version, we use the unaltered local filtering / recommendation algorithm with local evaluation function described in the previous section, thus treating foreign virtual nodes and respective relations and own nodes and relations on equal footing. An enhanced version will include an additional relevance score-element, which differentiates between own and foreign nodes and configurably may give nodes from users a higher weight if the relation to that user is strong or a higher weight if the relation is weak, following Granovetter’s paradigm on the importance of weak ties and deduced importance of information items from the “social fringe”.

D. Implementation

In order to implement the variant of the service where only those mutual contacts are inquired which are in the physical neighborhood of the inquiring user, we rely on Bluetooth for detecting those users. GPS would be less suitable for this task because we assume that a substantial fraction if not the majority of such interactions take place indoors which renders GPS useless. Furthermore, besides its function for detecting nearby other nodes (users, devices), Bluetooth obviously can also serve as a channel to handle the communication between devices. In order to handle the problems associated with all various Bluetooth stacks we use a serial port emulation.

V. RELATED WORK

In this section, we discuss related work. One example for a Semantic Desktop implementation on a desktop computer is the already mentioned Gnowsis system [4]. Gnowsis consists of two parts, the Gnowsis server which performs the data processing, storage and interaction with native applications; and the graphical user interface (GUI) part, implemented as Swing GUI and Web-based interfaces. External applications such as Microsoft Outlook or Web browsers are integrated using standardized interfaces. There are other similar systems for personal computers or servers such as IRIS [15]. However, a Semantic Desktop approach tailored towards mobile devices comparable to SeMoDesk does not exist, as far as we know.

Integrating SeMoDesk with an existing desktop application would be possible rather easily, because the data model of SeMoDesk is based on the PIMO model, but is out of the current scope of our work. A reasonable real life scenario is that the user defines her PIM ontology on the desktop computer and imports and manages resources on the mobile device, while being able to browse and search items, and occasionally add sub concepts on the PDA.

Other earlier related work in PIM research includes the Haystack project which aims at connecting application data and let people manage their information using personalization [16]. However the Haystack client is a rather complicated and extensive application that is not usable for mobile devices. “Stuff I’ve seen” is another interesting desktop application [17]. It allows creating an index of content, including Microsoft Outlook resources, files, and Web pages in the browser cache.

Niu and Kay present a recent approach to utilize a personal con-text ontology called PECO [18]. The ideas are similar to the Semantic Desktop idea with respect to formalizing a user’s personal view on things. PECO is created semi-automatically and then applied for personalization. Their focus is solely on location concepts such as buildings and rooms though. The approach could possibly be integrated into our system to provide a more sophisticated model of location in PIMO.

The idea of the Social Semantic Desktop paradigm was introduced by Stefan Decker which also aims at integrating Social Computing and Semantic Web [11][12]. Gruber aims at the same integration, explaining promising approaches and techniques using the example of a collaborative travel information space [19]. Our system is another example for the integration of social computing and semantic desktop application with the added aspect that it is aimed at mobile social interaction. Völkl et al. formulate requirements for personal knowledge management, reviews some existing approaches and introduces an approach for adding semantic richness to personal knowledge management, which is clearly related to the personal semantic desktop paradigm [20]. Agosto et al. emphasize the decentralized P2P nature of social information exchange on the Web, by introducing an example application [21].

The special role of data-management in ubiquitous computing environments is emphasized in [22]. Two applications for social data exchange are discussed which show that social data exchange and sharing is a key application in mobile environments. Mobile social data sharing is also a natural application in e-learning environments [23].

While there a lots of reviews on recommender systems and information item filtering, Peis et al. attempt to review the activities in the field of semantic recommender systems, a field, which our application is also contributing to [24]. In the area of mobile recommender systems, [25] is an example for a decentralized system for recommending images on PDAs. The approach utilizes item-based collaborative filtering and also incorporates public shared displays for group recommendations.

Finally, there is work on semantics based context reasoning in mobile domains (see e.g. [26]). A more sophisticated model of context in addition to location and time could be integrated in our approach.

VI. CONCLUSION

We have presented an approach for mobile personal information management based on the Semantic Desktop idea. Thereby, users can define and manage a personal ontology to structure their information space. This ontology can then be used to recommend items based on the current user context. We have explained the reasoning behind the recommendation process in this paper.

In order to evaluate our approach, we conducted small scale qualitative evaluations in our lab where users where presented the application with a small information space and asked to perform several recommendation-processes. The feedback was very positive. A systematic evaluation design involves a set of users with their own information spaces. In a first step the individual recommendation process would have to be tested by each individual by rating the relevance and usefulness of the first n of the recommended items. We can then compute a precision estimation of the approach as an average over these ratings. The social recommendation process should then be tested in the same way.

Planned future improvements also include learning of relationships based on the content of resources content (e.g. finding similar documents), and also learning relations based on user behavior. For example, if the user works on a certain document when/after interacting with another user, a relationship between the document and the user could be inferred and proposed for addition to the ontology.

REFERENCES

[1] W.Woerndl and A. Hristov, “Recommending Resources in Mobile Personal Information Management”, Proc. IARIA International Conference on Digital Society (ICDS 2009), Cancun, Mexico, 2009.

[2] J. Teevan and W. Jones, Personal Information Management. Combined Academic Publ., 2008.

[3] M. Lansdale, “The Psychology of Personal Information Management”, Applied Ergonomics, vol. 19, no. 1, pp. 55-66, 1988.

[4] L.Sauermann, G.A.Grimnes, M. Kiesel, C. Fluit, H. Maus, D. Heim, D. Nadeem, B. Horak and A. Dengel, “Semantic Desktop 2.0: The Gnowsis Experience”, Proc. 5th International Semantic Web Conference, Springer LNCS 4273, 2006.

[5] L. Sauermann, L. van Elst and A. Dengel, “PIMO – a Framework for Representing Personal Information Models”, Proc. of I-Semantics' 07, 2007.

[6] W. Woerndl and M. Woehrl, “SeMoDesk: Towards a Mobile Semantic Desktop”. Proc. Personal Information Management (PIM) Workshop, CHI 2008 Conference, Florence, Italy, 2008.

[7] S. Russel and P. Norvig, Artificial Intelligence: A Modern Approach, 2nd edition, Prentice Hall, 2002.

[8] J.L. Herlocker, J.A. Konstan and J. Riedl, “Explaining collaborative filtering recommendations”, Proc. of the 2000 Conference on Computer Supported Cooperative Work, pp. 241–250, 2000.

[9] F. Schulze, Context- and Location-aware Retrieval Methods for Personal Information Management on Mobile Devices, Bachelor Thesis, Department for Informatics, Technische Universitaet Muenchen, Germany, 2009.

[10] A. Hristov, A Context-Sensitive Recommendation System for a Mobile Semantic Desktop Application, Diploma Thesis, Institut fuer Informatik, Technische Universitaet Muenchen, 2008.

[11] S. Decker, “The social semantic desktop: Next generation collaboration infrastructure”. Information Services & Use, vol. 26, no. 2, pp. 139-144, 2006.

[12] S. Decker and M. Frank, The Social Semantic Desktop. DERI Technical Report 2004-05-02, Digital Enterprise Research Institute (DERI), Galway and Innsbruck, 2004.

[13] M. Brocco and G. Groh, “Team Recommendation in Open Innovation Networks”, Proc. RecSys09, New York, USA, Oct. 2009.

[14] G. Groh and C. Ehmig, “Recommendations in Taste Related Domains: Collaborative Filtering vs. Social Filtering”, Proc. Group07, Sanibel Island, USA, Nov. 2007.

[15] A. Cheyer, J. Park and R. Giuli, “IRIS: Integrate. Relate. Infer. Share”, Proc. 1st Workshop on the Semantic Desktop, ISWC Conference, Galway, Ireland, 2005.

[16] D.R. Karger, K. Bakshi, K., D. Huynh, D. Quan and V. Sinha, “Haystack: A General-Purpose Information Management Tool for End Users Based on Semistructured Data”. Proc. Conference on Innovative Data Systems Research (CIDR), Asilomar, CA, 2005.

[17] S.T. Dumais, E. Cutrell, J.J. Cadiz, G. Jancke, R. Sarin and D. Robbins, “Stuff I've Seen: A system for personal information retrieval and re-use”, Proc. SIGIR, 2003.

[18] W. Niu and J. Kay, “Pervasive Personalisation of Location Information: Personalised Context Ontology”, Proc. of the International Conference on Adaptive Hypermedia, Springer, 2008.

[19] T. Gruber, “Collective Knowledge Systems: Where the Social Web meets the Semantic Web” Keynote Presentation, 5th International Semantic Web Conference, November 7, 2006, http://tomgruber.org/writing/CollectiveKnowledgeSystems.pdf.

[20] M. Völkel, S. Schaffert and E. Oren, „Personal Knowledge Management with Semantic Technologies“ In Rech / Decker / Ras, Emerging Technologies for Semantic Work Environments: Techniques, Methods, and Applications, chapter 9. Information Science Reference, June 2008.

[21] L. Agosto, M. Plu, L. Vignollet and P. Bellec, “SoMeONe: A cooperative system for personalized information exchange”, in: Enterprise Information Systems V, Springer, 2005.

[22] M. Franklin, "Challenges in Ubiquitous Data Management”, Lecture Notes in Computer Science 2000, Springer, 2001.

[23] G. Singh, L. Denoue and A. Das, “Collaborative Note Taking“, Proc 2nd IEEE International Workshop on Wireless and Mobile Technologies in Education, 2004. pp. 163- 167.

[24] E. Peis, J. Morales-del-Castillo and J. Delgado-López, ”Semantic Recommender Systems. Analysis of the state of the topic”, Hipertext.net, num. 6, 2008. ISSN 1695-5498, http://www.hipertext.net.

[25] W. Woerndl, H. Muehe and V. Prinz, “Decentral Item-based Collaborative Filtering for Recommending Images on Mobile Devices”, Proc. Workshop on Mobile Media Retrieval (MMR'09), MDM 2009 Conference, Taipeh, Taiwan, May 2009.

[26] A. Bikakis, T. Patkos, G. Antoniou and D. Plexousakis, “A Survey of Semantics-Based Approaches for Context Reasoning in Ambient Intelligence”, In: Constructing Ambient Intelligence, Communications in Computer and Information Science series Vol. 11, Springer, 2008.