OntoPIM: From Personal Information Management to Task Information Management

George Lepouras
Dept. of Computer Science and Technology, University of Peloponnese
Terma Karaiskaki, 22100
Tripolis, Hellas
+30 2710 372201
gl@uop.gr

Alan Dix
Computing Department, Lancaster University,
Lancaster, LA1 4YR
United Kingdom
alan@hcibook.com

Akrivi Katifori
Dept. of Informatics and Telecommunications, University of Athens
Panepistimiopolis, 157 84
Athens, Hellas
+30 210 727 5241
vivi@mm.di.uoa.gr

Titziana Catarci
Dipartimento di Informatica e Sistemistica "A. Ruberti"
Universit `a di Roma "La Sapienza"
Via Salaria, 113
00198 Rome, Italy
catarci@dis.uniroma1.it

Benjamin Habegger
Dipartimento di Informatica e Sistemistica "A. Ruberti"
Universit `a di Roma "La Sapienza"
Via Salaria, 113
00198 Rome, Italy
habegger@dis.uniroma1.it

Antonella Poggi
Dipartimento di Informatica e Sistemistica "A. Ruberti"
Universit `a di Roma "La Sapienza"
Via Salaria, 113
00198 Rome, Italy
poggi@dis.uniroma1.it

Yannis Ioannidis
Dept. of Informatics and Telecommunications, University of Athens
Panepistimiopolis,
Athens, Hellas
yannis@di.uoa.gr

1 INTRODUCTION

Personal Information Management (PIM) aims to support users in the collection, storage and retrieval of their personal information. In such a framework the focus is mainly on how better to handle the information collected. Task Information Management (TIM) on the other hand adopts a more user-centric view and aims to support users in performing their tasks. Possible ways of achieving this are to automatically find data users need in order to perform a task, to automate the execution of tasks and the synthesis of new more abstract tasks by identifying tasks users carry out often. PIM and TIM can be seen as complimentary since an efficient organization of personal information can help in the discovery of data relevant to a task. To this end, it is necessary to adopt a holistic approach in the design and implementation of a Task Information Management System.

2 TIM: SOME EXAMPLES

We will give some examples that illustrate how TIM can help users perform everyday tasks. Tasks such as searching for a flight, booking a hotel room or filling an expense claim report are nowadays often performed online. In such tasks users are requested to fill in forms with personal, context-specific information. In the simplest case the user fills in more or less static data such as the name, surname and email address. Some web browsers propose default values selected from previous similar situations.

However, this is a static approach which does not take into consideration other user actions. For example, the user may have received an email with subject DELOS and in the message body identified keywords "meeting", "Athens" and "21st May".

An envisaged Task Information Management system would offer the user the possibility to search for a hotel or for flights in these dates and for the corresponding location[1]. During the search the system may offer to complete other values (probably gathered during older searches) like the type of room or the Frequent Flyer Membership card number. If at the end of search task, the user books a hotel or a flight online, cost and other necessary information can be kept and proposed when the user fills in the expense claim form. The system could also offer to make an entry in the user calendar for the meeting. Finally, if the received email had an attached file called "meeting-agenda" when the user saves it, the system could keep extra information derived by the user's actions.

In the described scenario the TIM system can identify, store, correlate and re-use information gathered during user interaction. This requires a system that can monitor and model user actions, parse documents and associate collected pieces of information, store them semantically and retrieve them when needed.

The work described is being developed in the framework of DELOS Network of Excellence as collaboration between University of Rome La Sapienza, University of Lancaster and National and Capodistrian University of Athens.

[1] Although Microsoft's Smart Tags allow the recognition of a particular type of data and the execution of user selected actions, a smart tag does not take into consideration all related data. Furthermore, such functionality works inside a certain application and not system-wide.

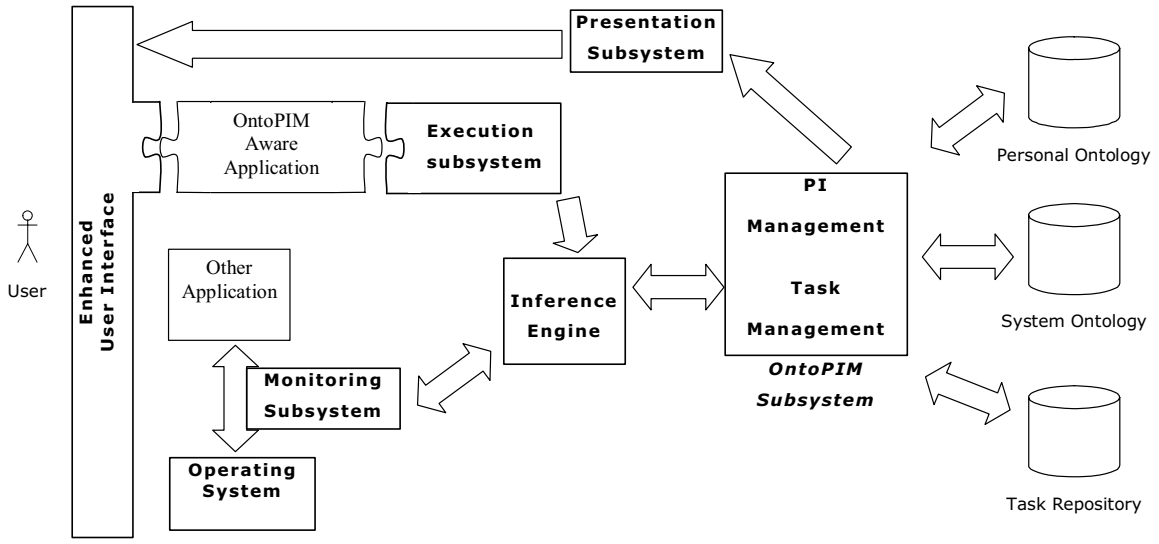

Figure 1. The architecture of Task Information System

3 RELATED WORK

Recently, especially the last year, there have been some attempts aiming to improve the usability of everyday tasks in windows based platforms. From these research efforts, two are most closely related to our approach. The first is the one proposed by Schwarz and Roth-Berghofer [1]. Similarly to our proposed system, they envisage a system that will observe the user's work as well as his ways of information handling and automatically learn and identify goals, intentions, structures, task ontology, and work processes. The second [2] aims at employing semantic web techniques to enhance the searching functionalities of the user's desktop. Haystack [3] is a somewhat older approach, similar to the second one. It proposes the use of a semantic network of information about the corpus of user documents. This network is constantly enriched through user observation and used to facilitate search and retrieval.

Another recent system CREO [4] uses semantic web technology mined from various web sites and databases to facilitate web-based tasks. When the user views a web page the text is analysed and compared to its large user-independent ontology. Where text matches some concept this can be used to trigger pre-defined or user-defined web-based tasks. CREO is an example of data-detector technology and the TIM team have been involved in previous data-detector systems onCue and Snip!t [5]. However, whilst most data-detector work is focussed on a short-term interaction around each recognition event, in the proposed TIM system the recognition of, for example, a date in an email, is used to bring unstructured text into a long-term personal semantic store and facilitate a continuing interaction.

4 AN APPROACH FOR A TIM SYSTEM

In this section is described an architecture proposal for a Task Information System. It comprises a set of basic components that can cater for the different user needs and application capabilities in a task supported environment. The architecture consists of a monitoring subsystem, execution subsystem, inference engine, Personal Information Management and Task Management subsystem (ontoPIM), a personal ontology, a system ontology, task repository and presentation subsystem. Interrelations between these subsystems are shown in Figure 1. In this architecture we can envisage two broad categories of applications: ontoPIM aware applications and other applications. The first category includes applications built with ontoPIM compatibility. These applications can be directly queried by the execution subsystem to retrieve context-dependent information, and the presentation subsystem can access the application's user interface to provide support information. For other applications the monitoring subsystem can employ a number of options to observe the user–application interaction and send the necessary information to the inference engine. Solutions such as the monitoring of user-triggered events or inclusion of suitably modified proxy servers have been discussed. Communication between the inference engine and the monitoring subsystem can be two way in case the inference engine needs to query the monitoring subsystem for more detailed information.

The inference engine aims to abstract from the data received by the monitoring and execution subsystems to obtain personal information and also to elicit user actions. The Inference Engine in conjunction with the Personal Information Management and Task Management updates the personal ontology and task repository.

The ontoPIM subsystem will:

- Aid the user in performing tasks
- Detect the data user's need to perform a task
- Synthesize new tasks from logs of previous tasks

Presentation of information is carried out by the presentation subsystem. A number of alternative methods for presenting information have been considered. If the aim is to provide context sensitive support, then the information should not be presented independently of the user interaction flow (as it would be the case of employing a separate floating window).

Solutions such as the use of a different modality (for example voice messages) may be also problematic.

Two solutions appear as the most prominent. The first is the integration of ontoPIM information in the UI of the application, while the other involves the use of right-click pop-up menus. Integrating information in the context of the application's user interface may offer a seamless working environment for the user where the ontoPIM is transparent. In such a case the user interface of applications is being enhanced with additional information retrieved by the ontoPIM. For example, if the user after reading the email (described in the previous example) regarding the meeting opens a web page to search for flights, ontoPIM may automatically fill in the necessary values (dates, place) previously gathered.

However two things may hinder the adoption of this solution. One is that if an application is not compatible to ontoPIM, user interface integration can not be always guaranteed. A second problem may arise from the fact that the integration alters the interface of the original application. Apart from copyright concerns, users may also object to such a modification and prefer to be able to control the flow of (personal) information.

For the latter, use of pop up, context menus can help. In most windowing environments, this type of menus is invoked by right clicking the mouse. Selections appearing in the menu are then related to object the user clicked on. In the previous example, if the user right clicks on an input field ontoPIM may propose values that can be typed in. Additionally, the user can use the pop up menu to initiate actions related to the object.

Finally, complementary to these solutions other less obtrusive approaches for the presentation of information may be employed. For example, if the system notes that the user is often performing the same series of tasks a tray notification or balloon help dialog can be initiated to ask the user if a new, composite task should be introduced to encompass them and facilitate user performance. The user may then decide to ignore the message or take some action.

The presented architecture has the benefit of allowing the implementation of different levels of ontoPIM awareness. In a case where ontoPIM has a direct access to the user interface, listening to events and updating the interface becomes almost trivial. In the case where the application is not ontoPIM aware, more subtle actions may be required such as listening to system events, probing the file system for new files, or adding a proxy between client and server applications, adding an independent popup generating application, etc.

5 ONTOPIM SCENARIO

To better illustrate (Figure 2) how ontoPIM would operate we will present a usage scenario. In this scenario the user receives an email with an attachment named *Agenda.doc*, title *DELOS* and message body with keywords such as *meeting, Athens, May 21st*. When the user views the email ontoPIM will scan it for information relevant to the user's personal ontology. The system will recognize keywords in the message body and if appropriate actions are related to them will present (probably in the right click pop menu) actions that can executed. For example by identifying the concept *MEETING* the system can include in the pop-up menu the task *FindHotel*.

Furthermore, since from the message the inference engine can deduce that a meeting is planned for the 21st of May in Athens, a new instance of the concept meeting can be created and linked to the message. The concept will have attributes (date, place, participant, reason) populated by the message information. If the user performs a semantic save [6] an instance of the agenda concept and an instance of the word document (in the personal and system ontology respectively) will also be created and linked to the document in the user's information space.

Since the ontology now includes a new instance of the meeting concept with the necessary attributes, if the user performs the task *FindHotel* the system can propose to automatically fill-in the necessary values for the execution of the task (the date and place). Moreover, the system can also help the user synthesize new tasks. For example if the user usually performs after the *FindHotel* a *FindFlight* task, the system can propose the composition of a new task which will allow the execution of both tasks as one

6 CONCLUSIONS

We have outlined an architecture for a TIM system which we feel can support the user manage personal information as well as everyday tasks. This work is in progress and a number of issues have yet to be clarified including the most suitable method of presenting support and its level of transparency from the user's point of view and methods for monitoring user interaction with a variety of non compliant applications.

7 REFERENCES

[1] Sven Schwarz and Thomas R. Roth-Berghofer. Towards Goal Elicitation by User Observation, Proceedings of the FGWM 2003 Workshop on Knowledge and Experience Management

[2] Alexandru Chirita, Rita Gavriloaie, Stefania Ghita,. Wolfgang Nejdl, and Raluca Paiu. Activity based metadata for semantic desktop search. In Proceedings of the 2nd European Semantic Web Conference, Heraklion, Greece, May 2005

[3] Eytan Adar, David Karger. Haystack: Per-User Information Environments, Conference on Information and Knowledge Management 1999.

[4] A. Faaborg and H. Lieberman (2006). A Goal-Oriented Web Browser. Proceedings of CHI 2006. ACM Press. pp. 751–760

[5] A. Dix, T. Catarci, B. Habegger, Y. Ioannidis, A. Kamaruddin, A. Katifori, G. Lepouras, A. Poggi, D. Ramduny-Ellis (2006). Intelligent context-sensitive interactions on desktop and the web Proceedings of workshop on Context in Advanced Interfaces at AVI2006.

[6] Vivi Katifori, Antonella Poggi, Monica Scannapieco, Tiziana Catarci, and Yannis Ioannidis. OntoPIM: how to rely on a personal ontology for Personal Information Management. In Proceedings of the First Workshop on the Semantic Desktop. November 2005.

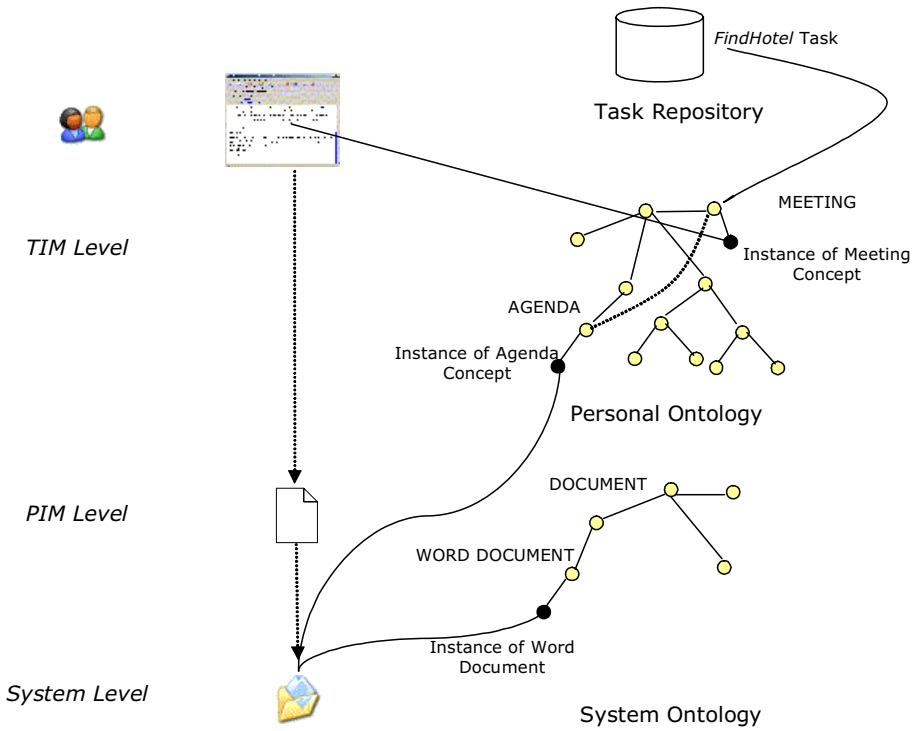

Figure 2. OntoPIM Execution Scenario