Task-centered Information Management

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

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, Greece
vivi@mm.di.uoa.gr

Giorgios Lepouras
Dept. of Computer Science And Technology,
University of Peloponnese,
Terma Karaiskaki, 22100,
Tripolis, Greece
gl@uop.gr

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

Abstract

The goal of DELOS Task 4.8 Task-centered Information Management is to provide the user with a Task-centered Information Management system (TIM), which automates user's most frequent activities, by exploiting the collection of personal documents considered as a Digital Library. In previous work we have explored the issue of managing personal data by enriching them with semantics according to a Personal Ontology, i.e. a user-tailored description of her domain of interest. Moreover, we have proposed a task specification language and a top-down approach to task inference, where the user specifies main aspects of the tasks using forms of declarative scripting. Recently, we have addresses new challenging issues related to TIM personalization and user's task inference. More precisely, the first main contribution of this paper is the design of a profiling ontology for the user, that can be used to semi-automatically populate the Personal Ontology. The second contribution is the investigation of task inference theoretical issues. In particular, we show how the use of the Personal Ontology helps for computing simple task inference. The third contribution is an architecture for the system that implements simple task inference. In the current phase we are implementing a prototype for TIM whose architecture is the one presented in this paper.

Categories and Subject Descriptors

H.2 [Database Managment]: H.2.1 Logical design; H.3 [Information Storage and Retrieval]: H.3.7 Digital Libraries; H.5 [INFORMATION INTERFACES AND PRESENTATION]: H.5.2 User Interfaces

General Terms

Design, Algorithms

Keywords

Personal Information Management, Ontology Management, Task Management

1 Introduction

Personal Information Management (PIM) aims at supporting users in the collection, storage and retrieval of their personal information. It is a crucial challenge nowadays; indeed, the collection of digital documents stored within the personal repository of each one of us increases every day, in terms of both size and "personal" relevance. In a sense, this collection constitutes the Digital Library that is closest to the user and most commonly and frequently used. It is often the starting point or reason for wider access to digital resources. Furthermore, the user's personal document collection is daily used to perform routine activities, as booking hotels, and organizing meetings. The focus of DELOS Task 4.8 Task-centered Information Management is precisely to provide the user with a Task-centered Information Management system (TIM), which automates these user activities, by exploiting the collection of personal documents considered as a Digital Library.

At the beginning of our investigation, we have faced the issue of providing the user with a system allowing her to access her data through a Personal Ontology (PO), that is unified, integrated and virtual, and reflects the user's view of her own domain of interest [4][15]. The study of such a system is motivated by the need to assign semantics to the user's personal collection of data, in order to help inferring and executing the most frequent user's tasks that either take such data as input or produce it as output. Then, we have focused on defining a task specification language[5], which was appropriate in our context, in the sense that it was both expressive enough to be effectively helpful to the user, and simple enough to allow tasks to be inferred. Moreover, such a language had to be tailored to use the PO to semi-automatically obtain task input and to possibly update the PO after the task execution, according to the task output and semantics.

Populating and maintaining a PO which indeed reflects the user's view of her domain of interest and allow to assign semantics to the whole collection of her personal data, is already a big issue. Moreover, having the PO tailored to the specific user, means letting the PO reflect the user's profile and preferences, and evolve according them. Thus, personalization and profiling both become crucial. On the other hand, having a task specification language allows to semi-automatically execute tasks that the user previously defined. Our further goal is to have the system *infer* what is the task the user intends to execute next.

This paper addresses the above mentioned issues. Specifically, our main contributions can be summarized as follows:

- First, we introduce a profiling ontology, whose goal is to provide an exhaustive yet extensible user modelling. This requires to specify different kind of user characteristics, depending on whether they correspond or not to timeless aspects of user's life. Then, we have provided techniques that exploit this profiling ontology to manually and semi-automatically populate the PO.

- Second, we cope with task inference in TIM. In particular, we investigate the main task inference theoretical issues. Then we show how to exploit the underlying PO and the features of our task specification language, in order to provide simple task inference in our setting. The main idea here is to suggest appropriate tasks from personal data, based on their semantics.

- Third, we propose a novel architecture for TIM, which implements our solution to the simple task inference issue. More precisely, we propose to integrate into the system an existing web bookmarking tool, called Snip!t, whose distinguishing feature is precisely to "intelligently" suggest appropriate actions to perform starting from a Web page content.

The paper is organized as follows. After discussing related work in Section 2, we briefly introduce in Section 3 the previous main task contributions. In Section 4, we discuss issues related to profiling and to the population of the PO. In Section 5 we discuss task inference related theoretical issues. Then, in Section 6, after introducing Snip!t and presenting its distinguishing features, we propose a new architecture for TIM, and show how this provides a simple task inference. Finally, we conclude by presenting future work.

2 Related work

The issues faced in DELOS Task 4.8 concern with several different areas of research. In what follows we discuss related work in each of the main such areas.

User modelling and ontologies. The use of ontologies in user modelling is not a new concept. Ontologies in the form of hierarchies of user interests have been proposed in [22]. Gauch et al. [11] also proposed a system that adapts information navigation based on a user profile structured as a weighted concept hierarchy. The user may create her own concept hierarchy and use it for browsing web sites. A preference model based on concept graphs is also proposed in [16] for query personalization. Razmerita et al. [21] presented a generic ontology-based user modelling architecture applied in the context of a Knowledge Management System.

Task management. Recently, research efforts on the problem of managing user's tasks have lead to the prototype Activity-Centered Task Assistant (ACTA), implemented as a Microsoft Outlook add-in[2]. In ACTA, a user's task, named "ACTA activity", is represented as a pre-structured container, that can be created inside the email folder hierarchy. It is a task-specific collection containing structured predefined elements called "components", that embody common resources of the task and appear as activity sub-folders. Thus, for example, by creating an ACTA activity to represent a meeting, and by inserting the component "contacts", the user aims at relating the content of this sub-folder, which will essentially be a list of names, with that particular meeting. Moreover, the population of an activity is done semi-automatically, by allowing the user just to drag into the appropriate activity component, the document containing the relevant data, which is afterward automatically extracted and stored. Even though ACTA activities are built relying on user's personal data, their approach is not comparable to ours, since they do not consider tasks as a workflow of actions (e.g. filling and sending the meeting invitation email), which can be inferred and semi-automatically executed.

Task inference. There has been a long history of research into task detection, inference and prediction in human-computer interaction, with a substantial activity in the early 1990s including Alan Cypher's work on Eager [6] and several collections [1]. The line of work has continued (for example [18]), but with less intensity than the early 1990s. Forms of task inference can be found in widely used systems, for example the detection of lists etc. in Microsoft Office or web browsers that auto-fill forms. The first example clearly demonstrates how important it is that the interaction is embedded within an appropriate framework, and how annoying it can be when inference does not do what you want! Some of this work lies under the area of "programming by demonstration" or "programming by example", where the user is often expected to be aware of the inferences being made and actively modify their actions to aid the system. This is the case of [12] where authors present a learning system, called PLIANT, that helps users anticipating their goal, by learning their preferences and adaptively assisting them in a particular long-running application such as a calendar assistant. Other work falls more under user modeling, intelligent help, automatic adaptation or context-aware interfaces where the user may not be explicitly aware that the system is doing any form of inference [3]. Our work lies with the former as we do expect that users will be aware of the inferences being made and work symbiotically with the system in order to create a fluid working environment.

3 Summary of previous contributions

In this section we briefly present the DELOS Task 4.8 previous contributions, namely OntoPIM and the task specification language.

OntoPIM. OntoPIM [15] is a module that allows to manage the whole collection of heterogeneous personal data usually maintained in a personal computer (e.g. contacts, documents, emails), and to access them through a unified, integrated, virtual and yet user-tailored view of her data. This view is called Personal Ontology (PO), since it reflects the user's view of her own domain of interest. It is therefore specific to each user. As for the language to specify the PO, we use the Description Logic called DL-Lite$_A$ [19][20], since besides allowing to express the most commonly used modeling constructs, it allows to answer expressive queries, i.e. conjunctive queries, in polynomial time with respect to the size of the data. This is clearly a distinguishing and desirable feature of such a language, in a context like ours, since the amount of data is typically huge in one's personal computer.

In order to achieve the above mentioned result, OntoPIM proceeds as follows. First it extracts, by means of appropriate wrappers, pieces of relevant data from the actual personal data contained in the personal computer. Then it exploits the so-called Semantic Save module, which (i) stores such data in a DBMS, maintaining also its provenance, and (ii) stores the relationship existing between the data and the PO, as (implicitly) specified by the user. Note that the latter relationship reflects indeed the data semantics according to the user.

Task Specification. In order to be able to semi-automatically execute user tasks, we defined a task specification language [5] having two main features. First, the language is at the same time expressive enough for actually being helpful to the user, and simple enough for being effectively "usable" and "learnable" by the system. Second, the language allows to specify as a part of the task definition, the input/output data mappings, i.e. the relationships existing between the task and the PO. Specifically, the input data mappings specify the query to be posed over the PO in order to obtain the task input, whereas the output data mapping specify the task output as an update (possibly empty) to be computed over the personal data, according to the semantics of both the task execution and the PO. As we will see, the specification of task input/output data mappings is crucial for task inference/detection/suggestion.

Furthermore, we have explored task inference top-down approaches, where the user specifies aspects of the task using forms of declarative scripting. Our proposal was based on the idea of combining task decomposition and a plan language to describe for each complex task, the execution plan of its subtasks. On one hand, a complex task is decomposed into a set of subtasks. This allowed for a comprehensible view of the task. On the other hand, we have propose a plan language limited to sequence, alternatives and repetition.

4 Ontology profiling and PO population

The strength of TIM as a system for managing user tasks greatly relies on the PO. Bearing in mind that each user has her own view of her domain of interest, personalization and profiling are crucial issues here. Indeed, the goal is to exploit user's profile and preferences in order to maintain the PO as "personal" as possible. In this section, we therefore start by discussing the TIM profiling ontology. Then we present techniques that exploit the profiling ontology to populate the PO.

Based on important user characteristics as derived by the surveyed literature (cf. Section 2), a profiling ontology that unifies them in a comprehensive, yet extensible model has been designed. This ontology is available in Protégé and RDF format in [14]. The instance "Ego" of the concept "Person" denotes the user. It is the central one in the profiling ontology, as it incorporates all the user profile characteristics. These may be either (i) simple concept attributes, like "name", "date of birth", "email address", or (ii) complex concept attributes, like "physical characteristics", or (iii) instances of other ontology concepts, like the instance of the concept "Home" denoting the place of user's residence, or the instances of the concept "Person" denoting the user's mother or a particular user's friend.

An aspect that has to be carefully taken into account to make the PO evolve according to the user's life and perspective, is time validity of user's characteristics. In fact, user's characteristics fall into two general categories. On the one hand, there are *time-constrained characteristics* that correspond to aspects of the user's life that are valid in a particular period. Examples of this kind are the attribute "email address" denoting user's email address during her visiting period at I.N.R.I.A., the instance of the concept "Home" denoting the place of user's residence from 1976 to 1989, and the instance of the concept "Friend" denoting a user's university fellow student. Obviously, if these characteristics are instances of concepts, they may themselves have attributes (i.e. slots) and be related to other concepts, providing information on the respective aspects during the same period. On the other hand, there are *timeless characteristics* that correspond to aspects of the user for which it would make no sense to consider any period of validity. Examples of this kind of characteristics are the attributes "name", and "date of birth", or the instance of the concept "Person" denoting the user's mother. Note that the same concept may have both instances denoting time-constrained user characteristics and instances denoting timeless characteristics, as the concept "Person" in our example. In fact, the time validity depends on the relation existing between the user and the characteristics. Thus, the relations "Prefers" and "Interested In", both crucial in the profiling ontology, associate to the instance "Ego" timeless characteristics that can be instances of any profiling ontology concept. By contrast, the relation "Involved In" associates "Ego" to time-constrained characteristics corresponding to user activities.

The PO can be seen as an extension of the profiling ontology, integrating in fact into one structure both the user's profile and the user's view of her domain of interest. When the system is initialized, an appropriate PO will be selected by a library of available ones adjusted to accurately reflect user's characteristics and interests. This can be done both manually (by the user) and automatically (by the system). For the manual population of the ontology a web based profiling extraction application is being developed. The Ontology

Profiler will ask the user a set of questions in order to select the appropriate ontology template(s). Then, the user will be guided to populate the PO with instances, concepts, attributes and relations, by means of a set of questions associated to each element of the PO.

For the automatic population of the PO, user information that is available within the file system may be exploited. User information may include:

- Chosen language and time-zone. These could give information about the user nationality and home country/city. Dialing codes and IP addresses can be also used for determining the user's location as well as information about her parents, friends, etc.

- Current file structure. If the user has created a more elaborate file structure than that already provided by the operating system to store her files, then it could also be a source of new concepts. A dictionary of synonyms could be used here to make the matching more effective. The user could also be prompted to indicate folder structures that contain documents relevant to her interests or activities. For example, if the user has a folder named "Articles" with sub-folders like "cooking" or "gardening" used to further categorize the documents, these concepts may be used to populate the relation "Interested In" of the profiling ontology.

- The system can also scan address books in order to retrieve contact information and populate the "Person" concept with instances.

- Calendars and to-do lists may be used to identify user activities, and therefore to populate the relation "Involved In".

- The web cache and bookmark/favourites structure could also be a possible source for deriving interests and preferences. The user can also be prompted, through an appropriate questionnaire to provide information concerning her personal data, interests, preferences, contacts, etc.

5 Task inference

In this section, we first address task inference theoretical issues. In particular we focus on bottom-up approaches to simple task inference. Then, we show how the use of a PO can help solving simple task inference in our setting.

As discussed in Section 2, the most successful systems have often been within dedicated applications, where there is a single line of activity and detailed semantic understanding of each action. This was for example the case with EAGER and PLIANT. For more generic systems detection and inference is more difficult, particularly where the user trace data is at a low level either keystroke or click-stream data (which often has to be the case where application support is not assumed). In fact, two particular problems for generic systems are:

- *interleaving* – Users are often performing several tasks simultaneously, perhaps while waiting for something to complete, or because they notice and alert, get a telephone call. Before task inference can begin it is necessary to disentangle these, otherwise each sub-task is littered with the "noise" of the others.

- *generalisation* – Where the user entered data is simply sequences of keystrokes, clicks on locations or basic data types, it is hard to generalise without very large numbers of examples.

The use of the PO helps considerably with these problems. Specifically, to face the interleaving problem, we will encourage a drill-down model of interaction where the user either selects previous outputs of actions and then drills forwards (e.g. recent email > sender > Antonella > University > City > Rome > Flights to Rome) or responds to suggested form fills (e.g. from Flight booking form with field 'City' select "Rome because it is the City where Antonella works"). This creates an explicit link either directly between actions or indirectly between them through ontology relationships, which can then be used to separate out interleaved tasks and sub-tasks by tracing lines of dependency, rather like pulling out a string of pearls from a jewellery box (see Figure 1).

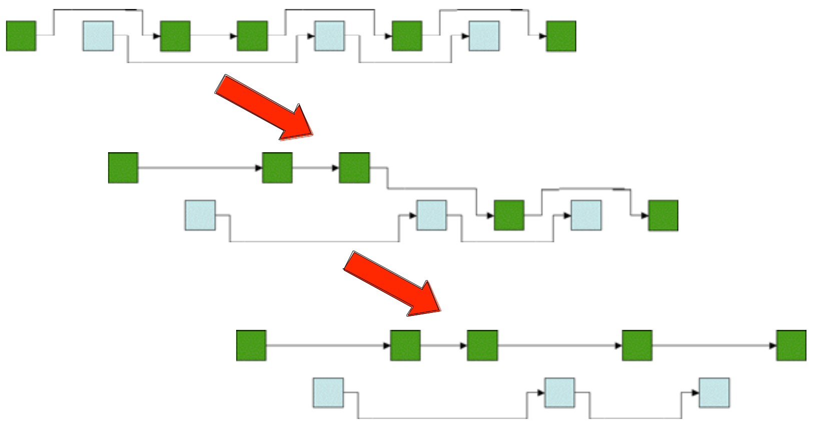


Figure 1: Untangling interleaved tasks using dependencies.

Concerning the generalisation problem, because we have a rich typing of task/action input and output data through the PO, we are in a much better position to generalise. If we only know that a form field requires a character string, then given a future character string we may have many possible previous tasks sequences whose initial actions require a string. In contrast, if we know that a character string is in fact a person name or a city name, then faced with a future person name (perhaps from a directory lookup, or an email sender) it is easier to find past tasks requiring as input a person name. In other words, our generalisation is not based on the representation in terms of letters, but in terms of the elements of the ontology.

Let us now focus on simple task inference, where a simple task can include sequences and/or a repetitions of sub-tasks/actions. Thus, here we do not cope with choices. Hence, the job of the inference here is to suggest the most likely single actions and entire task sequences so as to minimize the user's effort in using the system. Intuitively, we intend to build a series of increasingly complex inference mechanisms, both in terms of our development process and in terms of the users experience. That is, even if we have complex inference mechanisms available, these need to be presented incrementally to the user. In fact, to some extent, even simple type matching can be viewed as a crude form of task inference. However, if this is supplemented by sorting of a few most likely candidate actions/tasks based on past usage, then a form of task sequence comes almost "for free".

For example, suppose that the user has recently (1) taken a phone number, (2) done a reverse directory lookup (in some external web service) to find the person's name, then (3) done an address lookup to find their address and finally (4) taken the address and used a web mapping service to show the location. Now, suppose the user has an email containing a telephone number. The content recogniser finds the number and so suggests a semantic save of the number and also actions using the number. Top of the action list would be likely actions on telephone numbers, and notably the reverse lookup because it was recent. Once this was done, one of the outputs of the reverse lookup would be the person's name and similarly its top option would be the address lookup. So at each stage the previous sequence would be the first option, which means that the task is not totally automated, but the user is required low effort. The next step, which likewise requires minimal "intelligence", is simply to offer the entire sequence of actions as one of the options when the telephone number is encountered. This simply requires that all recent/previous task sequences are stored and so recent or frequently performed task sequences starting with the given type are suggested. The user is also given the option of naming a particular sequence of actions. If the user chooses to do this, the task can be used in future interactions. Note too that selection and more so naming is effectively a confirmation by the user that this is a useful task sequence and so will make it far more likely to be presented in future.

More complex tasks including repetitions of sub tasks can similarly be built bottom-up, for example, if the user chooses to perform an action sequence on an instance that is in some visible collection (e.g. selected from a PO concept, or from table of results) and then proceeds to select a second instance in the collection, one of the options would be not only to perform the sequence on the selected item but on all the collection (or selected members of it).

6 Snip!t and TIM

In this section, we first present an existing tool, called Snip!t, that is a web bookmarking tool, whose distinguishing feature is to suggest appropriate actions to perform starting from a Web page content. Then we propose a novel architecture for TIM, implementing task inference as discussed in Section 5, by integrating Snip!t into the system.

Snip!t [8] is a web bookmarking tool, however unlike most web bookmarks, the user can also select a portion (snip) of the page content and then, using a bookmarklet, this selection is sent to the Snip!t web application. The snip of the page is stored along with the page url, title etc. and the user can sort the snip into categories or share it with others using RSS feeds. In addition, when the selected text contains a recognised type of data such as a date, post code, person's name, etc., then actions are suggested. For example, if the selected text is recognised as a post code and this leads to suggested actions such as finding the local BBC news for a specific area (see Figure 2).

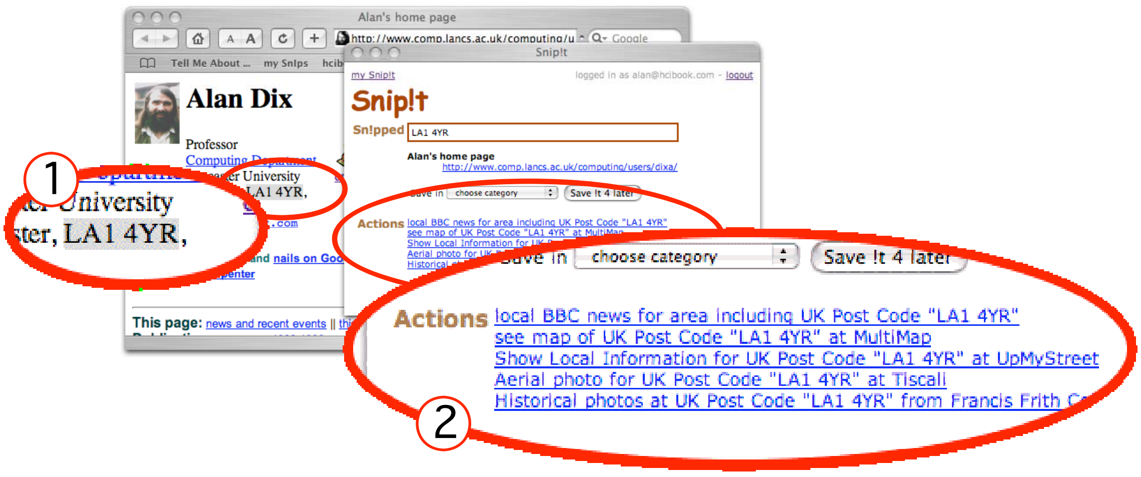

Alan's home page
http://www.comp.lancs.ac.uk/computing/u
Google
Tell Me About ... my Snips hcib
Alan Dix
Professor
Computing Department
Lancaster University
LA1 4YR,
University
ster, LA1 4YR,
This page: news and recent events
Snip!t
my Snip!t
logged in as alan@hcibook.com - logout
Snip!t
Sn!pped LA1 4YR
Alan's home page
http://www.comp.lancs.ac.uk/computing/users/dixa/
Save in choose category Save It 4 later
Actions
local BBC news for area including UK Post Code "LA1 4YR"
see map of UK Post Code "LA1 4YR" at MultiMap
Show Local Information for UK Post Code "LA1 4YR" at UpMyStreet
Aerial photo for UK Post Code "LA1 4YR" at Tiscali
Historical photos at UK Post Code "LA1 4YR" from Francis Frith Co
1
2

Figure 2 : Snip!t in action (actions for a post code)

Snip!t had two contributing origins. In a study of bookmarking organization some years ago [9] some subjects said that they would sometimes like to bookmark a portion of a page. While the study had different aims this suggestion led to the first version of Snip!t in 2003. In addition, this gave an opportunity to use the new platform for data-detector technology originally developed as part of onCue, the key product of an ex-dot.com company aQtive [8]. So onCue combines a way of storing and sorting data (portions of web pages) and intelligent suggestion of actions to perform based on the data. Thus it shares some characteristics with the TIM vision.

Internally the bookmarking side of Snip!t is fairly straightforward with simple hierarchical categorisation scheme and the ability to assign snips to multiple categories. The snipping of the page contents itself is done using a bookmarklet, that is a small piece of Javascript that is placed as a browser bookmark, usually on the browser toolbar. When the user clicks the bookmarklet the Javascript executes extracts the selected content and then creates a HTTP request to the Snip!t server.

The “intelligent” parts of Snip!t use an architecture inherited from onCue. It consists of two main types of agents: recognisers and services (cf. Figure 3). Recognisers work on plain text and discover various data types whilst services take these data types and suggest actions based on them (usually creating invocations of web applications). The recognisers within Snip!t are of several types:

- **basic recognisers:** can be based either on *large table lookup*, e.g. common surnames and forenames, place names, or on *regular expression / pattern matching*, e.g. telephone number, ISBN.

- **chained recognisers:** where the semantics of data recognised by a recogniser is used by another to:

    - look for a wider data representation, e.g. postcode suggests looking for address; these recognisers are used to allow scalability and reduce false positives;

    - look for a semantically related data, e.g. URL suggests looking for the domain name; these recognisers are used to reduce false nagatives;

    - look for inner representation, e.g. from Amazon author URL to author name; these recognisers are also used to allow scalability.

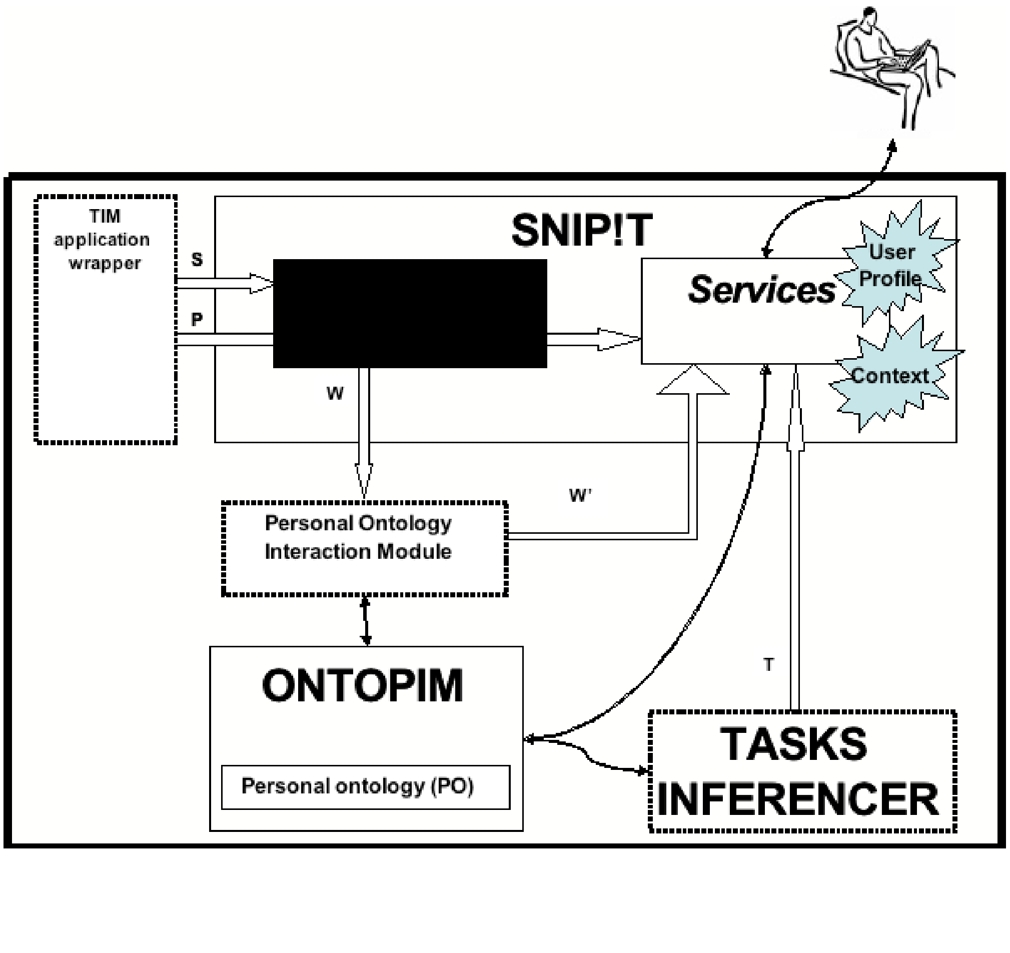
Each agent, recogniser and service, is relatively simple, however the combinations of these small components create an emergent effect that intelligently suggests appropriate actions based on the snipped text.

Let us now turn attention to TIM, and the way the user interacts with the system in order to execute a task. The starting point may be of three kinds:

(i) explicit invocation of a particular task (e.g. selecting it from an application menu),

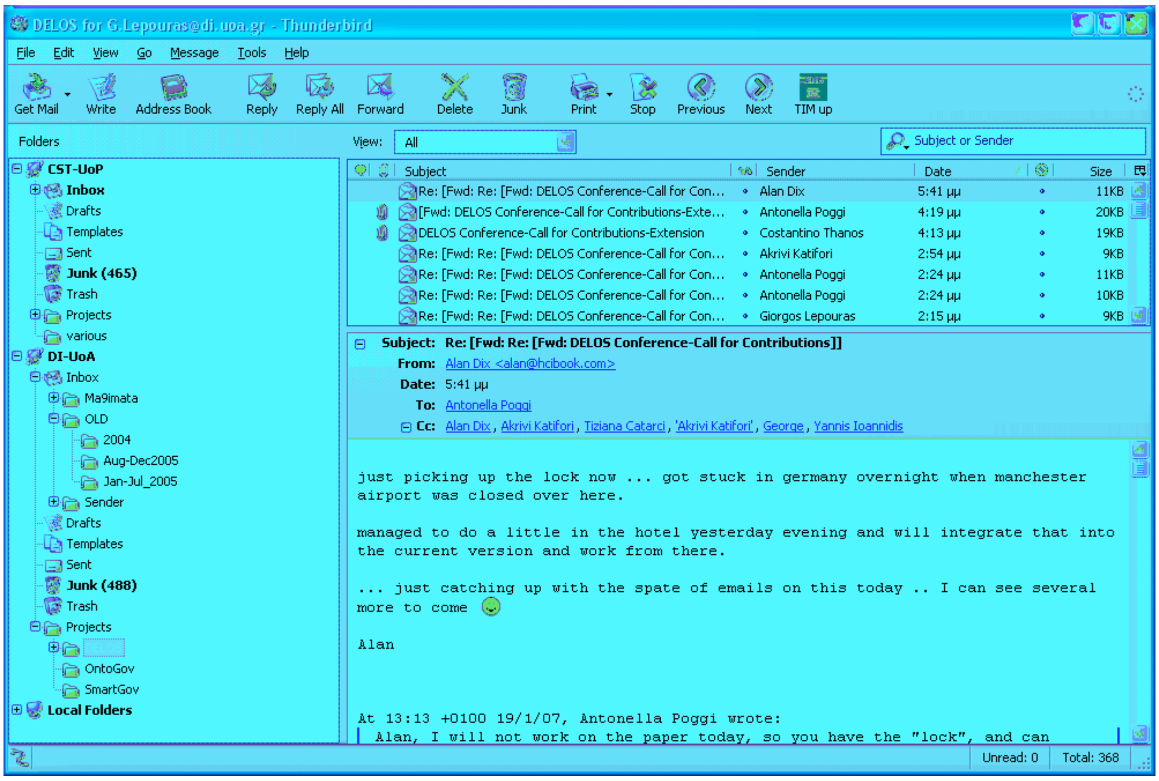
(ii) choosing a data value in a document, email etc. then drilling down into tasks possible from it,

(iii) selecting an item in the PO and then, drilling into available tasks (that is one's with matching types).



Consider now the architecture illustrated in Figure 3. The main idea is to integrate Snip!t within the system. In particular, in case (i) the user will directly indicate among all Snip!t services the one he wants to execute next. In contrast, cases (ii) and (iii) are closer to the typical Snip!t scenario. Indeed, in both cases, given a (collection of) data value(s), the system suggests tasks to perform from it. Apart OntoPIM and Snipt!t, this new architecture includes the three main TIM modules discussed below.

- **TIM application wrapper:** this module, which is dependent on the particular application enables the user to select any piece of document and send it to Snip!t in an appropriate form (cf. snip **S**). As for a demonstration, we have extended the functionality of Thunderbird mail client, so as to allow the user to select the TIM button while reading a message (see Figure 4). A script running on the mail client saves the message, parses the MIME headers filling-in an HTML form which appears in



Subject	Sender	Date	Size
Re: [Fwd: Re: [Fwd: DELOS Conference-Call for Con...	Alan Dix	5:41 μμ	11KB
[Fwd: DELOS Conference-Call for Contributions-Exte...	Antonella Poggi	4:19 μμ	20KB
DELOS Conference-Call for Contributions-Extension	Costantino Thanos	4:13 μμ	19KB
Re: [Fwd: Re: [Fwd: DELOS Conference-Call for Con...	Akrivi Katifori	2:54 μμ	9KB
Re: [Fwd: Re: [Fwd: DELOS Conference-Call for Con...	Antonella Poggi	2:24 μμ	11KB
Re: [Fwd: Re: [Fwd: DELOS Conference-Call for Con...	Antonella Poggi	2:24 μμ	10KB
Re: [Fwd: Re: [Fwd: DELOS Conference-Call for Con...	Giorgos Lepouras	2:15 μμ	9KB

Subject: Re: [Fwd: Re: [Fwd: DELOS Conference-Call for Contributions]]
From: Alan Dix <alan@hcibook.com>
Date: 5:41 μμ
To: Antonella Poggi
Cc: Alan Dix, Akrivi Katifori, Tiziana Catarci, 'Akrivi Katifori', George, Yannis Ioannidis

```
just picking up the lock now ... got stuck in germany overnight when manchester
airport was closed over here.

managed to do a little in the hotel yesterday evening and will integrate that into
the current version and work from there.

... just catching up with the spate of emails on this today .. I can see several
more to come

Alan


At 13:13 +0100 19/1/07, Antonella Poggi wrote:
| Alan, I will not work on the paper today, so you have the "lock", and can
```

a new browser window. The user can then press the submit button to send the message to Snip!t.

- **Personal Ontology Interaction Module (POIM):** given a collection of data **W** each with its associated type as returned by Snip!t recognisers, this module access the PO and returns a collection of data **W'** somehow related to **W** according to the PO. Such data is then used to perform appropriate tasks.

- **Task Inferencer:** this module is responsible for task inference. Intuitively, given a collection of data **W'** from the **POIM** and its provenance, the **Task Inferencer** suggests what is the next task/action to be executed from **W**.

7 Conclusion and future work

In this paper we have investigated new challenges toward a fully equipped TIM system. In particular we have addressed the user profiling and some initial system personalization issues. Moreover, we have addressed theoretical task inference issues, and we have proposed a first implementation providing a solution to simple task inference in one's personal computer provided with a PO.

Many other aspects deserve to be further investigated. Concerning the domain of personal ontologies, an issue that should be taken into account and explored is the concept of time in the context of personal ontologies which needs further research in order to be incorporated successfully in the ontology model and reasoner.

Concerning task inference, we have concentrated on simple tasks, possibly including sequences and repetitions of actions, whereas we have not coped with alternatives, i.e. choices. These may arise either where a selection has to be made from a collection (e.g. 1–m relations in the ontology) or where tasks sequences vary slightly depending on aspects of the context or instance (e.g. compress files larger than 100K). Various machine learning techniques can be applied here, which we plan to study in future work.

References

[1]. Beale R. and Finlay J. 1993. Neural networks and pattern recognition in human-computer interaction. In *SIGCHI Bull.* 25 (2): 25-35.

[2]. Bellotti V. and Thornton J. 2006. Managing Activities with TV-Acta: TaskVista and Activity-Centered Task Assistant. In *Proceedings of the Second SIGIR Workshop on Personal Information Management (PIM)*, 2006.

[3]. Boone G. 1998. Concept features in Re: Agent, an intelligent Email agent. In K. P. Sycara and M. Wooldridge (eds), *Proceedings of the Second international Conference on Autonomous Agents. Minneapolis: Minnesota (United States)*. 10 – 13 May 1998. ACM Press. 141-148.

[4]. Catarci T., Dong X. L., Halevy A. and Poggi A.. 2007. Structure Everything. In Jones W. and Teevan J. (eds), *Personal Information Management.* University of Washington Press UW Press. To Appear.

[5]. Catarci T., Habegger B., Poggi A., Dix A., Ioannidis Y., Katifori A. and Lepouras G. 2006. Intelligent User Task Oriented Systems. In *Proceedings of the Second SIGIR Workshop on Personal Information Management (PIM)*, 2006.

[6]. Cypher A.. Eager: Programming Repetitive Tasks by Example. In *Proceedings of the ACM Conference on Human Factors and Computing Systems (CHI'91)*, 1991.

[7]. Cypher A.(ed), 1993. *Watch What I Do: Programming by Demonstration*. MIT Press.

[8]. Dix A., Beale R. and Wood A. 2000. Architectures to make Simple Visualisations using Simple Systems. In *Proceedings of Advanced Visual Interfaces (AVI2000*), 2000. ACM Press. 51-60.

[9]. Dix A. and Marshall J. 2003. At the right time: when to sort web history and bookmarks. In *Volume 1 of Proceedings of HCI International 2003*. J. Jacko and C. Stephandis (ed.). 2003. Lawrence Erlbaum Associates. 758-762.

[10]. Dix A., Catarci T., Habegger B., Ioannidis Y., Kamaruddin A., Katifori A., Lepouras G., Poggi A., Ramduny-Ellis A. 2006. Intelligent context-sensitive interactions on desktop and the web. In *Proceedings of the International AVI'2006 Workshop on Context in Advanced Interface*, 2006.

[11]. Gauch S., Chaffee J., Pretschner A. 2003. Ontology-Based User Profiles for Search and Browsing, User Modeling and User-Adapted Interaction: The Journal of Personalization Research, Special Issue on User Modeling for Web and Hypermedia Information Retrieval, 2003.

[12]. Gervasio M. T., Moffitt M. D., Pollack M. E., Taylor J. M. and Uribe T. E. 2005. Active preference learning for personalized calendar scheduling assistance. In *Proceedings of the 10th international conference on Intelligent user interfaces (IUI '05)*, 2005. ACM Press. 90-97.

[13]. Golemati M., Katifori A., Vassilakis C., Lepouras G., Halatsis C. 2007. Creating an Ontology for the User Profile: Method and Applications, to appear in *The First International Conference on Research Challenges in Information Science (RCIS). 23-26 April 2007. Quarzazate (Morocco).*

[14].M. Golemati, A. Katifori, C. Vassilakis, G. Lepouras, C. Halatsis 2006. *User Profile Ontology version 1*, available at http://oceanis.mm.di.uoa.gr/pened/?category=publications

[15]. Katifori A., Poggi A., Scannapieco M., Catarci T., and Ioannidis Y. 2005. OntoPIM: how to rely on a personal ontology for Personal Information Management. In *Proc. of the 1st Workshop on The Semantic Desktop*. *2005*.

[16]. Koutrika G. and Ioannidis Y. 2005. Personalized Queries under a Generalized Preference Model. In *Proceedings of 21st Intl. Conf. On Data Engineering (ICDE), 5-8 April 2005. Tokyo. Japan*. 841-852.

[17]. Lepouras G., Dix A., Katifori A., Catarci T., Habegger B., Poggi A., and Ioannidis Y 2006. OntoPIM: From Personal Information Management to Task Information Management. In *Proceedings of the Second SIGIR Workshop on Personal Information Management (PIM)*, 2006.

[18]. Lieberman H. (ed), 2001. *Your Wish is My Command: Programming By Example*. Morgan Kaufmann.

[19]. Poggi A.. Structured and Semi-Structured Data Integration. PhD thesis, Dipartimento di Informatica e Sistemistica, Universit`a di Roma “La Sapienza”, 2006.

[20]. Poggi A., Lembo D., Calvanese D., De Giacomo G., Lenzerini M., and Rosati R. 2007. Linking Ontologies to Data. Submitted to an international journal, 2007.

[21]. Razmerita L., Angehrn A., Maedche A. 2003. Ontology based user modeling for Knowledge Management Systems, In *Proceedings of the User Modeling Conference. Pittsburgh (USA). 2003*. Springer Verlag. 213-217.

[22]. Trajkova J., and Gauch S. 2004. Improving Ontology-based User Profiles. In *Proceedings of RIAO 2004. University of Avignon: Vaucluse (France). 26-28 April 2004*. 380-389.