

INFORMATION SOCIETY TECHNOLOGIES (IEST) PROGRAMME



Information Society
Technologies



Audio-Visual ConTent search and retrieval in a distributed
P2P repositORY

VICTORY
IST-6-044985-STREP

Profiles and annotation population			
Deliverable No.		D5.2	
Workpackage No.	WP5	Workpackage Title	Search engine components
Task No.	T5.2	Task Title	User profiles and annotation population to the VICTORY community
Authors		Dr. E. Godio, Dr. G. Paravati, Dr. A. Sanna (POLITO)	
Status (F: final; D: draft; RD: revised draft):		RD	
File Name:		VICTORY-POLITO-RD-WP5-V4-D5.2.doc	
Project start date and duration		01 January 2007, 30 Months	

Table of Contents

TABLE OF CONTENTS	2
LIST OF FIGURES	3
LIST OF TABLES	3
ABBREVIATION/TERMS LIST	4
EXECUTIVE SUMMARY	5
1. INTRODUCTION	6
1.1. DELIVERABLE STRUCTURE	6
2. USER PROFILE	7
2.1. ONTOLOGICAL USER PROFILE.....	8
2.1.1. <i>Ontology Definition</i>	8
2.1.2. <i>Related work</i>	8
3. PROFILE MANAGEMENT INTERFACE	10
4. DESKTOP EDGE-PEER USER PROFILE MANAGEMENT INTERFACE	12
5. MOBILE EDGE-PEER USER PROFILE MANAGEMENT INTERFACE	13
6. CONCLUSIONS	15
REFERENCES	16

List of Figures

Figure 1: Desktop edge-peer user profile management interface: to be remarked the presence of both textboxes and comboboxes..... 12

Figure 2: The combobox dynamically created following the definition given as example in Section 2..... 13

Figure 3: Mobile peer user profile management interface 13

List of Tables

Abbreviation/Terms List

2D	Two dimensional
3D	Three dimensional
GUI	Graphics User Interface
XML	eXtensible Markup Language
QoE	Quality of Experience
URL	Uniform Resource Locator

Executive Summary

This deliverable, entitled "Profiles and annotation population", provides a detailed description of the mechanisms, which were developed in WP5, for implementing the user profiling. A user profile is a collection of personal data assigned to a specific user or identity. In the context of VICTORY, user profiles are used both to enhance query performance and to personalize query results. In other words, user profile information are used to improve the "Quality of the Experience" (QoE) acting as an effective and highly flexible *tool* to search and retrieve Multipedia objects, thus providing a more efficient information access.

User preferences and the search context are considered through the notion of ontological user profile which, dynamically updatable over the time, reflects user interests. The user profile contains information encapsulated in an XML format, whose syntax and structure are defined by an ontology that may be variable through time. This means that also the graphics user interface (GUI) that allows the user to visualize and edit his/her profile has not to be static, but dynamically adaptive according to the ontology changes.

It has been chosen to store user profile information locally on client device file systems. This choice forces the user to keep different user profiles for each device he/she uses to connect to VICTORY; moreover, the user has to manually manage the synchronization among different versions of profiles stored on different devices. On the other hand, this choice avoids each privacy/security issue involved in keeping user profile data in a central remote server.

First of all, the VICTORY ontology, used to generate the user profile, is presented. User profile interface features have been then analysed, in order to define requirements of desktop and mobile edge-peer GUIs; relationships between the XML scheme and the ontology are then described. Finally, graphics elements for desktop and mobile edge-peer interfaces are described in detail, as well as the procedures to dynamically compose them in order to build the GUIs.

1. Introduction

In almost all distributed applications, people who connect to the system are identified through a user profile, which contains personal information and some additional attributes that allow a deeper knowledge related to the user's interests or habits. This feature is present also in the VICTORY project. In this case, the user profile information is stored in a file that resides on the client device file system whose content can be both "published" each time the user shares a new 3D model as an attachment of contextual data related to this action and used to improve the search phase of a Multipedia object.

User preferences and the search context are considered through the notion of ontological user profile which, dynamically updatable over the time, reflects user interests. The user profile contains information encapsulated in an XML format, whose syntax and structure are defined by an ontology that may be variable through time. This means that also the interface that allows the user to visualize and edit his/her profile has not to be static, but dynamically adaptive according to the ontology changes, both on desktop and mobile edge-peers.

Since the user profile schema is part of the VICTORY ontology, the profile data can be utilised during sharing and searching for Multipedia objects. In other words, usual user interests can be used both to improve the query and personalise search results. The user profile management implements the required mechanisms in a generic fashion, i.e. later adaptation and further developments of the user profile schema on the ontology side do not require changes on the client side.

In the following Sections features of the user profile management interface will be described as well as the mechanism that allows to build the GUI in a dynamic way.

1.1. Deliverable Structure

The deliverable is organized as follows:

- **Chapter 2** reviews the state of the art related to user profiling and analyses solutions that use ontologies for user profiling
- Interface features are described in **Chapter 3**.
- **Chapter 4** presents the interface for desktop edge-peers.
- **Chapter 5** presents the interface for mobile edge-peers.
- Finally, in **Chapter 6**, conclusions are drawn for this deliverable.

2. User profile

In recent years, as a consequence of the emergence of search metadata-engines, the information acquisition process has provided several kinds of tools to collect information from the Internet. However, the quality of the collected information is still a problem. In the Internet, for instance, millions of users access search engines every day and they have different profiles and access different types of URLs. Based on this, it has emerged the idea of developing systems that are not only responsible for searching information, but also finding "good quality" (i.e. relevant) information to the users.

The need for more effective information retrieval has led to the creation of the notions of the semantic web and personalized information management, areas of study that take advantage of the semantic context of the presented information to facilitate storage and retrieval of the information itself. The notion of user profiling has been introduced in order to record the user context and personalize applications so as to be tailored to the user needs.

In this sense, a user would not spend hours accessing unimportant URLs and could extract URLs which are interesting in relation to its preferences. In the literature, several web-based search information systems have been proposed, using some type of filter techniques, such as in [1] and [2]. However, to improve the quality of these systems, it is expected that the required information has not been only retrieved, but it also has to fulfil the personal needs of a user, through a user profile [3].

Some systems involving user profile to collect information from the Internet can be found in [4] and [5]. Most of the existing systems search for all possible pages in order to detect "relevant" pages for the users. However, even the use of filter techniques may become impractical when a huge amount of pages has to be elaborated.

Moreover, the content and media format nowadays applications can produce is becoming more diverse (e.g., text, image, audio, and video). The variety of users and platforms can take advantage of user profiling as a method of personalizing functionalities and interfaces of many software systems.

In general, a user profile should take into account the following factors: 1) device, media, and application specifics; 2) explicit user preferences and implicit user models; 3) an ease-to-use but secure mechanism for storing and accessing profile data.

Moreover, a set of "fundamental functions" of a user profiling system can be identified [6]:

- user profile creation for a new user;
- user profile deletion for a profile which is no longer needed;
- information updating/writing to a user profile;
- information reading from a user profile;
- information removing from a user profile.

Multi-agent based systems are often used to support flexible/adjustable user profiling mechanisms [2][6][7][8][9]. Moreover, with their special ability to operate disconnected and autonomously, mobile agents are good candidates to operate in wireless access networks; a mobile user can dispatch mobile agents to the fixed network where they operate autonomously in the user's behalf [10].

User profile management has been a task also investigated by the W3C. A proposal called Open Profiling Specification (OPS) [11] has been made to the Privacy Preferences Project (P3P) [12] and it is supported by a number of companies such as IBM. OPS proposes a standard for sharing and exchanging personal information with Web services. Web services that support OPS/P3P are able to share user profiles so that people do not need to provide the repeated information to each service. Microsoft .NET Passport [13] and the Liberty Alliance Project [14] have provided similar services. Two other contributions are W3C's Composite Capability/Preference Profiles (CC/PP) [15] and 3GPP's Generic User Profile (GUP) [16] which both provide a basic user profile framework for sharing user information across different devices and networks.

2.1. Ontological User Profile

2.1.1. Ontology Definition

As defined in [26], an ontology is a formal explicit description of a domain, consisting of classes, which are the concepts found in the domain (also called entities). Each class may have one or more parent classes, formulating thus a specialization/generalization hierarchy. A class has properties or slots (also called roles or attributes) describing various features of the modeled class, and restrictions on the slots (also referred to as facets or role descriptions). Each slot, in turn, has a type and could have a restricted number of allowed values, which may be of simple types (strings, numbers, booleans or enumerations) or instances of other classes. Classes may have instances, which correspond to individual objects in the domain of discourse; each instance has a concrete value for each slot of the class it belongs to. An ontology together with a set of individual instances of classes constitutes a knowledge base.

2.1.2. Related work

Ontologies have been proven an effective means for modeling digital collections and user contexts. They can be a very useful tool, because they may present an overview of the domain related to a specific area of interest, thus ontologies and can be used for browsing and query refinement. Ontologies model concepts and relationships in a high level of abstraction, providing rich semantics for humans to work with and the required formalism for computers to perform mechanical processing and reasoning. Ontologies play a key role also in VICTORY, thus they are strictly related to the user profile definition as described in next Sections.

Using an ontology to model the user profile has already been proposed in various applications like web search [17], [18], and personal information management [19]. However, ontologies modeling user profiles are application-specific, with each one having been created specifically for a particular domain.

Ontologies in the form of hierarchies of user interests have been proposed in [17]. Gauch et al. [20] also proposed a system that adapts information navigation based on a user profile structured as a weighted concept hierarchy. The user may create his/her own concept hierarchy and use it for browsing web sites. A user model can also be built using an ontology schema. Razmerita et al. [21] presented a generic ontology-based user modelling architecture applied in the context of a Knowledge Management System.

In the field of ontology design, efforts have been made by several research groups to facilitate the ontology engineering process, employing both manual and semi-automatic methods.

Semi-automatic methods focus on the acquisition of ontologies from domain texts. A framework is proposed with this objective in [22]; it incorporates several information extraction and learning approaches, in order to face the discovery of relevant classes, their organization in a taxonomy and the non-taxonomic relationships between classes. Comprehensive surveys of existing methodologies can be found in [23] and [24].

Throughout the ontology creation process, the designers may take into account a set of ontology design criteria, such as clarity, coherence, and extensibility [25]. Tazari et al. [27] suggest the following concepts as important for user profiling: user identity, characteristics, capabilities, universal preferences, state of the user, application-specific preferences. Other concepts like current activity, current terminal, location, motion state, and orientation are also mentioned. They also propose groups of parameters concerning personal information (name, birthday, address, bank account, and credit card), general characteristics (physical factors: weight and height, physical disabilities and abilities: reading, speaking and writing), education, occupation, interaction-related information, expertise, and user state.

Interests ([17], [20], [28], [29]) and preferences [30], [29] are considered of particular importance for most applications that incorporate profiles. Interests are in some cases organized in hierarchies of concepts [17], [20]. Abilities, both physical and mental also can be relevant [31]. For example, the ability of a user to mentally rotate two or three dimensional objects can affect the interpretation of a picture [32]. The gender factor also has been proven to affect the performance of different users while interacting with the same system [33]. User expertise, either computer-related or related to another domain is a concept necessary for many profiling applications [29].

3. Profile management interface

A user profile management interface has to perform mainly two tasks:

- visualizing the user profile;
- allowing the user to modify and save the changes made to the profile.

These features must be obviously available for both desktop and mobile edge-peers, this means that two interfaces have to be developed; in particular, two different programming languages have been used: Java for desktop edge-peers and C# .Net compact framework for mobile edge-peers. The two interfaces are very similar, both in appearance and functionalities. The differences that exist between them will be illustrated afterwards.

As mentioned above, the interface has to be dynamically built on the base of an ontology, then GUI fields are populated retrieving data from the user profile: this is made by parsing the content of two files received from the super-peer the edge-peer is connected to:

- “profile.xml”: this file contains the user profile information in an XML format. A sample user profile is:

```
<Agg class="Agg_Profile.V1">
  <AV n="Att_UserName" v="my_username" />
  <AV n="Att_FirstName" v="my_firstname" />
  <AV n="Att_LastName" v="my_lastname" />
  <AV n="Att_DateOfBirth" v="1900-01-01" />
  <AV n="Att_Age" v="" />
  <AV n="Att_Device" v="mobile" />
  <AV n="Att_Intentions" v="buy" />
  <AV n="Att_SuperPeers" v="1.2.3.4" />
</Agg>
```

- “victory.omml”: contains the full VICTORY ontology; only one part of which describes the syntax and the structure of the user profile. A sample definition of the user profile schema is:

```
<ClassHistory name="Agg_Profile">
<Aggregate id="Agg_Profile.V1" super="Aggregate.V1" editable="false">
  <Attribute name="Att_UserName" type="Text.V1"/>
  <Attribute name="Att_FirstName" type="Text.V1"/>
  <Attribute name="Att_LastName" type="Text.V1"/>
  <Attribute name="Att_DateOfBirth" type="Date.V1"/>
  <Attribute name="Att_Age" type="Int_Age.V1"/>
  <Attribute name="Att_Device" type="Txt_DeviceType.V1"/>
  <Attribute name="Att_Intentions" type="Txt_UserIntention.V1"/>
  <Attribute name="Att_SuperPeers" type="Text.V1"/>
</Aggregate>
</ClassHistory>
```

Moreover the ontology file may contain the definition of specific attribute types, for example, in this case, the attribute "Att_Intentions" belongs to the type "Txt_UserIntention.V1", whose definition is:

```
<ClassHistory name="Txt_UserIntention">
  <Atomic id="Txt_UserIntention.V1" super="Text.V1" editable="false">
    <ValueEnumeration>
      <V v="inform"/>
      <V v="design"/>
      <V v="buy"/>
    </ValueEnumeration>
  </Atomic>
</ClassHistory>
```

In the example above, the definition of an enumeration value class is represented, this means that the attribute can assume only values belonging to a restricted pool. One of the simplest ways to allow the user to choose among these values is to use a "ComboBox" control, populating it with items corresponding to each member of the pool. Considering that the ontology may be modified through time, these items cannot be "hard-coded"; on the other hand, the "ComboBox" has to be populated, in a dynamic way, at runtime. The time-invariance is not guaranteed for attributes too, thus there may be more or less attributes to visualize; also the attribute type may vary. Said that, it is really important to have an adaptive dynamically runtime-built interface.

Both for desktop and mobile edge-peers, the interface consists of a central area whose aim is to visualize the profile information, and two buttons, on the bottom, that allow the user to save or reject changes. An accurate description of both interface types follows.

4. Desktop edge-peer user profile management interface

Agg_Profile.V1	
UserName	my_username
FirstName	my_firstname
LastName	my_lastname
DateOfBirth	1900-01-01
Age	NO VALUE
Device	mobile
Intentions	buy
SuperPeers	1.2.3.4

Buttons: Cancel, Save

Figure 1: Desktop edge-peer user profile management interface

The edge peer user interface is a Java-platform based application. For each attribute in the profile schema, it shows the name using a *JLabel* and the related values with a *JTextField* or a *ComboBox*, depending on the type of attribute: in fact, as mentioned above, one of the best ways to show and make editable an attribute belonging to an enumeration class is to populate a *ComboBox*, while for attributes belonging to superclasses like *Text* or *Integer*, a good way is to insert them into an editable *TextField*.

The procedure used to dynamically populate the interface is the following:

- When the application starts, two arrays are created, one of *JLabel* objects, the other of *Component* ones. The attribute names are all inserted in the first array, after creating the related *JLabels*, while the second one will be populated by *JTextFields* or *JComboBoxes*, which both are subclasses of "*Component*", depending on the type of the attribute. The application also searches for the "victory.omml" file describing the ontology definition related to user profile.
- For each attribute found in the "profile.xml" file, the application:
 - Sets the i^{th} label in the label array to display the attribute name.
 - Parses the "victory.omml" file to retrieve the attribute type
 - If the attribute belongs to a superclass like *Text* or *Integer*, a *JTextField* is created, its content is set to the attribute value and then it is added to the *Components* array.
 - Otherwise, if the attribute type is specifically defined, which usually means it is enumerative, its definition is searched and the allowed values are used to populate a *JComboBox*, which will then be the i^{th} *Components* array element.

- If the attribute belongs to a class that defines a minimum and a maximum value, as may happen for example for the *Int_Age* attribute, a *JComboBox* is created and populated with all values belonging to the allowed interval. As before, the *JComboBox* is then inserted at the i^{th} position of *Components* array.
- The new *JLabel* and *Component* are inserted into the visualization layout

Then, the application allows the user to edit his/her profile and decide whether to save or discard changes, using the two buttons placed at the bottom of the window.



Figure 2: The combobox dynamically created following the definition given as example in Section 2

5. Mobile edge-peer user profile management interface



Figure 3: Mobile peer user profile management interface

The mobile edge-peer user profile interface is quite similar to the desktop edge-peer one. The main differences are in the use of C#.NET compact framework instead of Java. This implies that the data visualization can be easily managed using a *DataGrid* control, which is shown in the main form of the program (see Figure 3). The grid contains two columns:

- the first one, which is uneditable, contains the attribute names
- the second one, the content of whose cells, on the contrary, can be edited, retains the attribute values.

When the application starts, the *DataGrid* is filled with the attributes found into the “profile.xml” user profile schema. During this operation, the application also creates a support array which associates to each *DataGrid* row the type of attribute value it contains, searching the “victory.omml” ontology to retrieve this information. After this, the application listens for

user events, in particular for clicks on any of the cells belonging to the second column. When a click is caught, the application dynamically builds a form that allows the user to edit the data contained into the cell. This implies the choice of an editing form layout and its initialization. A complete list of available layouts and required initialisations follows:

- Text-type attributes: the application creates and shows a form containing a *TextField* control initialised to the current value of the clicked cell.



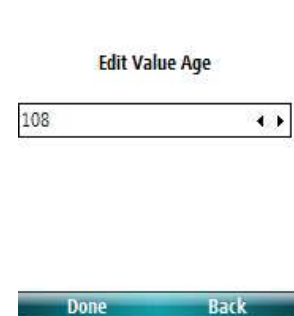
The screenshot shows a mobile-style form titled "Edit Value: UserName". It contains a single text input field with the text "my_username" inside. At the bottom of the form, there are two buttons: "Done" on the left and "Back" on the right.

- Specific (enumerative) attributes: the application creates and shows a form containing a *ComboBox* which is populated, in a similar way as it happens in the desktop edge-peer interface, parsing the data type definition contained in the ontology file and retrieving the enumerative choice pool.



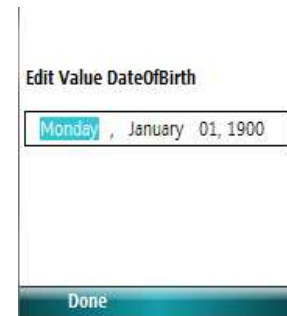
The screenshot shows a mobile-style form titled "Edit Value Device". It contains a dropdown menu (ComboBox) with the text "desktop" and a small arrow on the right side. At the bottom of the form, there are two buttons: "Done" on the left and "Back" on the right.

- Also integer-type attributes belonging to a specific range are made editable using form containing a *ComboBox*, which in this case contains all the range of values, extremes included, declared in the ontology definition. The control is initialised to the current cell value.



The screenshot shows a mobile-style form titled "Edit Value Age". It contains a dropdown menu (ComboBox) with the text "108" and a small arrow on the right side. At the bottom of the form, there are two buttons: "Done" on the left and "Back" on the right.

- Date attributes: the application creates and shows a form containing a *DateTimePicker*, which is a specific control that allows the user to easily manage date variables. Also in this case, the value of the control displayed is set equal to the clicked cell one.



6. Conclusions

The procedures used to dynamically build the interfaces for desktop and mobile edge-peers are able to take into account changes both in the user profile schema and in the ontology used to describe the syntax and the structure of the user profile itself.

Although the two interfaces are designed for, in general, extremely different classes of devices, the user may not feel strong differences while using them, because the way in which data can be edited is very similar.

References

1. J.R Chen, S.R Wolfe and S.D Wragg, S. D. A distributed multiagent system for collaborative information management and sharing. In Proc of 9th Int. Conf. on Information and Knowledge Management, pp. 382-388, United States, 2000.
2. N.G Shaw, A Mian and S.B Yadav. A comprehensive agent-based architecture for intelligent information retrieval in a distributed heterogeneous environment. *Decision Support Systems*, 32(4):401-415, 2002.
3. Gabriel L. Somlo and Adele E. Howe. Using Web Helper Agent Profiles in Query Generation. AAMAS'03, July 14-18, 2003, Melbourne, Australia
4. H. Chen, A.L Houston, R. R Sewell and B.R Schatz. Internet browsing and searching: User evaluations of category map and concept space techniques. *Journal of the American Society for Information Science*, 49(7):582-603, 1998.
5. F Menczer. Complementing search engines with online web mining agents. *Decision Support Systems*, 35(2):195-212, 2003.
6. Z. Ma and D. Silver, E. Shakshuki. User Profile Framework for Agent-based Grid Computing. Workshop on "Knowledge Grid and Grid Intelligence" October 13, 2003, Halifax, Canada.
7. M Chau, D Zeng, H Chen, M Huang and D Hendriawan. Design and evaluation of a multi-agent collaborative web mining system. *Decision Support Systems*, 35(1): 167-183, 2003.
8. C Căndea, M Staicu and C Zamfirescu, C. An agent-based system for modelling the searching process on the web. In Proc of 4th IEEE Int Conf on Intelligence Engineering Systems, 2000.
9. Manuel F Gomes Junior and Anne Magaly Canuto. Carcara: A Multi-agent System for Web Mining using Adjustable User Profile and Dynamic grouping. Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'06).
10. Carsten Pils and Jens Hartmann. The User Agent: An Approach for Service and Profile Management in Wireless Access Systems. *Lecture Notes in Computer Science, Advances in Artificial Intelligence*. Workshop Reader held at PRICAI 2000 Melbourne, Australia, August 28 - September 1, 2000.
11. Submission of OPS to W3C <http://www.w3.org/Submission/1997/6/Overview.html>
12. W3C Privacy Preferences Project (P3P). <http://www.w3.org/P3P/>
13. Microsoft .NET Passport. <http://www.passport.net/>
14. The Liberty Alliance Project. <http://www.projectliberty.org/>
15. W3C CC/PP: A user side framework for content negotiation. <http://www.w3.org/TR/NOTE-CCPP/>
16. 3GPP Generic User Profile (GUP). http://www.3gpp.org/ftp/tsg_sa/TSG_SA/TSGS_13/Docs/PDF/SP-010548.pdf
17. J. Trajkova, S. Gauch, Improving Ontology-based User Profiles, Proc. of RIAO 2004, University of Avignon (Vaucluse), France, April 26-28, 2004, pp. 380-389.
18. S. Lawrence, (2000). Context in web search. *IEEE Data Engineering Bulletin*, 23(3), pp.25-32.
19. V. Katifori,, A. Poggi,, M. Scannapieco, T. Catarci, & Y. Ioannidis (2005). OntoPIM: how to rely on a personal ontology for Personal Information Management. In Proc. of the 1st Workshop on The Semantic Desktop.
20. S. Gauch, J. Chaffee, A. Pretschner, 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, vol. , (2003).
21. L. Razmerita,, A. Angehrn, A. Maedche, Ontology based user modeling for Knowledge Management Systems, Proceedings of the User Modeling Conference, Pittsburgh, USA, Springer Verlag, pp. 213-217, 2003.
 22. A. Maedche, S. Staab, "Mining Ontologies from Text", EKAW 2000, pp. 189-202.
 23. M. Cristani, R. Cuel, "A Survey on Ontology Creation Methodologies", International Journal on Semantic Web and Information Systems, Vol. 1, No. 2, pp. 49 – 69, 2005.
 24. N. F. Noy, C. Hafner, "The State of the Art in Ontology Design, A Survey and Comparative Review", AI Magazine, 18 (3), Fall 1997, pp. 53-74.
 25. C. Fluit, M. Sabou, F. van Harmelen, "Ontology-based Information Visualisation", In Visualising the Semantic Web, Springer Verlag, 2002.
 26. N. F. Noy, D. L. McGuinness, "Ontology Development 101: A Guide to Creating Your First Ontology", Stanford Knowledge Systems Laboratory Technical Report KSL-01-05, March 2001.
 27. R. Tazari, M. Grimm, M. Finke, (2003), Modeling User Context, Proceedings of the 10th International Conference on Human-Computer Interaction (HCI2003), Crete (Greece), June 2003.
 28. Teevan, S. T. Dumais, and E. Horvitz, Personalizing Search via Automated Analysis of Interests and Activities, Proceedings of SIGIR 2005, ACM Press, August 2005.
 29. M. Golemati, A. Katifori, C. Vassilakis, G. Lepouras, C. Halatsis, User Profile Ontology version 1, available at <http://oceanis.mm.di.uoa.gr/pened/?category=publications>
 30. A. Kobsa, User Modelling: Recent work, prospects and hazards, Adaptive User Interfaces: Principles and Practices (Schneider-Hufschmidt, T. Khme, U. Malinowski, eds. 1993).
 31. B. Kules, User Modeling for Adaptive and Adaptable Software Systems, 2000, Available at <http://www.otal.umd.edu/UUGuide/wmk/>
 32. B. Gutkauf, S. Thies, G. Domik, User Adaptive Chart Editing and Presentation - Applied Through User Modeling and Critiquing Available at <http://wwwcs.uni-paderborn.de/cs/agdomik/arbeitschwerpunkte/ucmm/idias/root.html>
 33. G. S. Hubona and G. W. Shirah, The Gender Factor Performing Visualization Tasks on Computer Media, Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) - Track 4, Big Island, HI, p 40097c, 2004.