Scalable Computing: Practice and Experience
Volume 8, Number 1, pp. 29–40. http://www.scpe.org

ISSN 1895-1767
© 2007 SWPS

AGENT TECHNOLOGY FOR PERSONALIZED INFORMATION FILTERING: THE PIA SYSTEM

SAHIN ALBAYRAK*, STEFAN WOLLNY†, ANDREAS LOMMATZSCH‡, AND DRAGAN MILOSEVIC‡

Abstract. As today the amount of accessible information is overwhelming, the intelligent and personalized filtering of available information is a main challenge. Additionally, there is a growing need for the seamless mobile and multi-modal system usage throughout the whole day to meet the requirements of the modern society (anytime, anywhere, anyhow). A personal information agent that is delivering the right information at the right time by accessing, filtering and presenting information in a situation-aware matter is needed. Applying Agent-technology is promising, because the inherent capabilities of agents like autonomy, pro- and reactiveness offer an adequate approach. We developed an agent-based personal information system called PIA for collecting, filtering, and integrating information at a common point, offering access to the information by WWW, e-mail, SMS, MMS, and J2ME clients. Push and pull techniques are combined allowing the user to search explicitly for specific information on the one hand and to be informed automatically about relevant information divided in pre-, work and recreation slots on the other hand. In the core of the PIA system advanced filtering techniques are deployed through multiple filtering agent communities for content-based and collaborative filtering. Information-extracting agents are constantly gathering new relevant information from a variety of selected sources (internet, files, databases, web-services etc.). A personal agent for each user is managing the individual information provisioning, tailored to the needs of this specific user, knowing the profile, the current situation and learning from feedback.

Key words. intelligent and personalized filtering, ubiquitous access, recommendation systems, agents and complex systems, agent-based deployed applications, evolution, adaptation and learning.

1. Introduction.

Nowadays, desired information often remains unfound, because it is hidden in a huge amount of unnecessary and irrelevant data. On the Internet there are well maintained search engines that are highly useful if you want to do full-text keyword-search [1], but they are not able to support you in a personalized way and typically do not offer any push-services or in other words no information will be sent to you when you are not active. Also, as they normally do not adapt themselves to mobile devices, they cannot be used throughout a whole day because you are not sitting in front of a standard browser all the time and when you return, these systems will treat you in the very same way as if you have never been there before (no personalization, no learning). Users who are not familiar with domain-specific keywords won't be able to do successful research, because no support is offered. Predefined or auto-generated keywords for the search-domains are needed to fill that gap. As information demands are continuously increasing today and the gathering of information is time-consuming, there is a growing need for a personalized support (Figure 1.1). Labor-saving information is needed to increase productivity at work and also there is an increasing aspiration for a personalized offer of general information, specific domain knowledge, entertainment, shopping, fitness, lifestyle and health information. Existing commercial personalized systems are far away from that functionality, as they usually do not offer much more than allowing to choose the kind of the layout or collecting some of the offered information channels (and the information within is not personalized).

To overcome that situation you need a personal information agent (PIA) who knows the way of your thinking and can really support you throughout the whole day by accessing, filtering and presenting information to you in a situation-aware matter (Figure 1.1). Some systems exist (Fab [2], Amalthaea [3], WAIR [4], P-Tango [5], Trip-Matcher [6], PIAgent [7], Letizia [8], Let's Browse [9], Newt [10], WebWatcher [11], PEA [12], BAZAR [13]) that implement advanced algorithmic technology, but did not become widely accepted, maybe because of real world requirements like availability, scalability and adaptation to current and future standards and mobile devices.

In this paper we present an agent-based approach for the efficient, seamless and tailored provisioning of relevant information on the basis of end-users' daily routine. As we assume the reader will be familiar with agent-technology (see [14, 15] for a good introduction), we will concentrate on the practical usage and the real-world advantages of agent-technology. After briefly describing the existing systems from which the scientific publications are available, we describe the design and architecture and afterwards depict the system in Section 4.

2. Related work.

The following paragraphs are going to briefly present some of the already mentioned systems (Fab, Amalthaea, WAIR, P-Tango, PIAgent, Letizia, Let's Browse, Newt, WebWatcher and PEA), for which we believe that are related to our work.

*DAI-Labor, Technical University Berlin, Germany (sahin@dai-lab.de)
†High Performance Computing, Zuse Institute Berlin, Germany (wollny@zib.de)
‡DAI-Labor, Technical University Berlin, Germany ({andreas, dragan}@dai-lab.de)

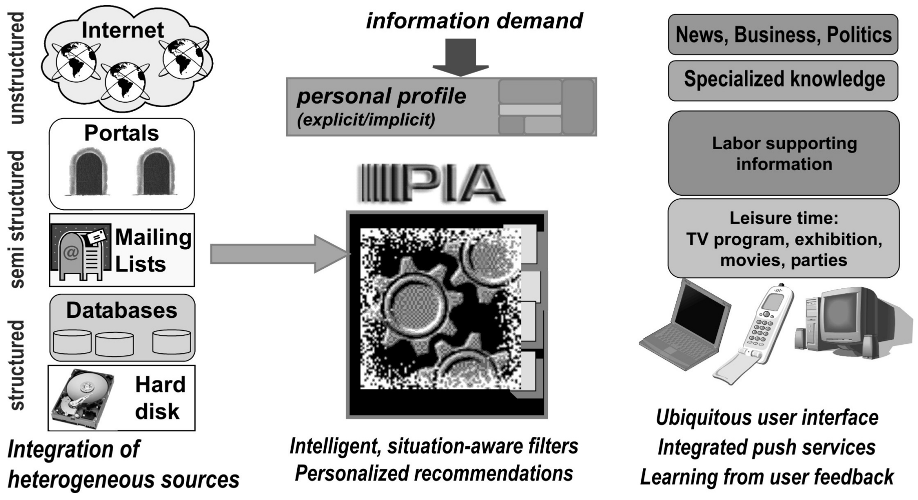
unstructured
semi structured
structured
Internet
Portals
Mailing Lists
Databases
Hard disk
Integration of heterogeneous sources
information demand
personal profile
(explicit/implicit)
PIA
Intelligent, situation-aware filters
Personalized recommendations
News, Business, Politics
Specialized knowledge
Labor supporting information
Leisure time: TV program, exhibition, movies, parties
Ubiquitous user interface
Integrated push services
Learning from user feedback

Fig. 1.1. *Demand for a personal information agent*

Fab [2] is an automatic recommendation service for information retrieval, which is able to over time adapt to its users, who consequently receive increasingly personalized documents. By maintaining both collection and selection agents, Fab system is a good test-bed for trying out different filtering strategies, which either collect documents from the Web that belong to the certain topic, or select some of the collected documents that are suitable for a particular user. The creation of profiles through the content-based analysis, which are afterwards directly compared to find similar users for collaborative recommendations, represents the unique synergy of these two frequently combined filtering techniques. Unfortunately, the usability of the whole system depends on the ability of the content based filtering to generate the usable profiles, being the serious drawback of the Fab system.

Since information discovery and information filtering are proven to be the suitable domains for applying multi-agent technology, a personalized system, named Amalthaea [3] has been developed. It proactively tries to discover from various distributed sources the information that is relevant to a user. The multi-agent technology is applied by maintaining two different types of agents, being information filtering and information discovery ones. The ways how these agents are managing to learn the user's interests and habits, to maintain their competence by adapting to the changes in the user's information needs, and to explore the new domains that may be of interest to a user, depend on evolution programming, being maybe not so applicable for the large-scale information retrieval tasks.

Seeking the state of a user profile, which best represents actual information interests and therefore maximize the expected value of the cumulative user relevance feedback, is formulated in WAIR multi-agent system [4] as the reinforcement learning problem. The insufficiency of explicit user ratings is tried to be overcome by using the classification approach based on the neural network, which exploits different implicit indicators of interests in order to estimate the real relevance feedback values. Unfortunately, the amount of the explicit ratings needed for training that classifier still seems to be too large. This clearly limits the applicability of the WAIR system.

To intelligently deliver a personalized newspaper, which contains only the articles of highest interest that are individually selected everyday for each and every user, P-Tango [5] system proposes a framework for combining different filtering strategies. Although the currently combined strategies are only the content-based and collaborative ones, a proposed framework is significant, by reason of being extendible to any filtering methods. In spite of this extensibility, we believe that the agent-based framework that we propose in this paper, offers better flexibility when the integration and afterwards the usage of new strategies is concerned.

As the information became the one of most significant resources for business and research, both periodically scanning different information sources and pushing the found relevant articles to interested users, have also motivated the development of PIAgent [7]. While a usage of various extractor agents each supporting a particular information source is more or less typical for agent-based filtering systems (and it is also present in our approach), the uniqueness of PIAgent lies in its application of back propagation neural network for separating relevant articles from others. Such a neural network approach has strength in optimistically providing excellent classification accuracy. Unfortunately, its big weakness in often expensive training that practically makes the PIAgent to be hardly applicable for nowadays information retrieval tasks.

The intelligent assistance to the user, who is browsing the Internet for the interesting information, is provided by the autonomous interface agent, named Letizia [8]. It tracks user behavior and uses various heuristics to anticipate, which hyperlinks may lead to the potentially relevant documents, and which should be ignored by reason of pointing to junk or not existing page. The cornerstone property of the Letizia system is in asking the user neither to explicitly state its interests by defining the query nor to provide the explicit feedback about a real relevance of recommendations. Although this explicit communication with the user can speed up learning, the priority in designing the Letizia system has been given to both letting the user to browse without being interrupted and asking for help only when being unsure which link to follow.

The MIT Media Laboratory has also developed an agent, whose job is to choose, from the links reachable on the current Web page, those that are likely to best satisfy the interests of multiple users. The agent is named Let's Browse [9], by reason of providing the assistance to the group of humans in browsing, by suggesting hyperlinks likely to be of common interests. Although this system demonstrates how documents that are good for the group of users and that are in the neighborhood can be found, it generally does not respond to the challenge of finding the data that is located anywhere on the Internet.

The ability to both specialize to user interests, adapt to preference changes and explore the newer information domains makes the foundation of the NewT [10], being one personalized multi-agent filtering system for news articles. As user information interests are modeled as the population of the competing profiles, the used learning mechanisms are both relevance feedback, as well as the crossover and mutation genetic operators. These recombination genetic operators are mainly responsible for the adaptation and exploration issues by creating more fitted future populations. In the meantime, a user profile also learns through the application of the relevance feedback techniques. Taken together these learning mechanisms make the so-called Baldwin effect [24], saying that a population evolves towards a fitter form much faster, whenever its members are allowed to learn during their lifetime. Although the Baldwin evolution seems to be more powerful than the evolution approach used in Amalthaea, it has the same weaknesses which limit its applicability for large-scale information retrieval.

Users may find it difficult both to create the appropriate queries and to locate the information of interest in the case of having no specific knowledge about the content of the underlying document collection. On the one hand, some systems aim to deploy efficient clustering algorithms, which will dynamically produce the table of contents, needed to facilitate the users' browsing activities. The cornerstone idea is to by some means help a user first to get an overview concerning the available content, and then to accurately express its information needs. On the other hand, WebWatcher [11] acts as the tour guide that provides the assistance, which is similar to the guidance of the human in the real museum. It accompanies users from page to page, suggests appropriate hyperlinks, and learns from the obtained experience to improve its advice giving skills. Such a system unfortunately can only locally assist the user, which brings the same drawbacks being present in Letizia and Let's Browse systems.

Personal Email Assistant (PEA) [12] filters incoming mails and ranks them according to their relevance in order to help nowadays users, who easily end up with large part of their working day being spent with reading emails. PEA maintains the personal user model that consists of several profiles and uses the evolutionary algorithms to move that model constantly closer to the current information needs. By doing that PEA aims at assisting users in dealing more effectively with their daily load of emails so that valuable working time is saved for more productive and creative tasks. Even though the evolution strategies seems to be powerful enough for dealing with emails in the PEA system, their usage in the Internet-like environment still remains to be a great challenge.

3. Design of PIA: The Personal Information Agent. To meet the discussed requirements and to support the user in that matter, we designed a multi-agent system composed of four classes of agents: many information extracting agents, agents that implement different filtering strategies, agents for providing different kinds of presentation and one personal agent for each user. Logically, all this can be seen as a classical three tier application (Figure 3.1). Concerning the information extraction, general search engines on the one hand but also domain-specific portals on the other hand have to be integrated. Additional information sources (Databases, Files, Mailinglists etc.) should also be integrated easily at run-time.

Several agents realize different filtering strategies (content-based and collaborative filtering [16], [5]) that have to be combined in an intelligent matter. Also agents for providing information actively via SMS, MMS, Fax, e-mail (push-services) are needed. A Multi-access service platform has to manage the presentation of the results tailored to the used device and the current situation. The personal agent should constantly improve the knowledge about his user by learning from the given feedback, which is also taken for collaborative filtering, as information that has been rated as highly interesting might be useful for a user with a similar profile as well. As users usually are not very keen on giving explicit feedback (ratings), implicit feedback like the fact that the user stored an article can also be taken into account [18].

A keywordassistant should support the user in defining queries even if he is not familiar with a certain domain. PIA provides three strategies for finding adequate keywords and for optimizing existing requests:

1. Keywords predefined by experts for frequently requested topics (or categories) can help the unexperienced user to find the relevant keywords. The suggestions provided by domain experts are usually a good starting point for individual requests.

2. An alternative method for finding interesting keywords is the extraction of words and phrases from interesting papers. This strategy helps the user to identify the key concepts from a paper that can be useful for finding other relevant documents. In contrast to other approaches (like Googles Find similar documents) the keyword extraction gives the user the opportunity to adapt extracted keywords according to the personal interests and preferences.

3. For optimizing existing queries the PIA system suggests keywords from similar requests. For computing the similarities between user requests the systems analyses the overlapping of user profiles (based on stems) and the corelation between the user ratings. Keywords that frequently occure in the requests of similar users are suggested to the user as potentially relevant search terms.

The whole system is designed to be highly scalable, easy to modify, to adapt and to improve and therefore an agent-based approach that allows to integrate or to remove agents even at run-time is a smart choice. The different filtering techniques are needed to provide accurate results, because the weakness of individual techniques should be compensated by the strengths of others. Documents should be logically clustered by their domains to allow fast access, and for each document several models [19] will be built, all including stemming and stop-word elimination, but some tailored for very efficient retrieval at run-time and others to support advanced filtering algorithms for a high accuracy.

If the system notices that the content-based filtering is not able to offer sufficient results, additional information should be offered by collaborative filtering, i. e. information that was rated as interesting by a user with a similar profile will be presented.

With the push-services, the user can decide to get new integrated relevant information immediately and on a mobile device, but for users who do not want to get new information immediately, a personalized newsletter also has to be offered. This newsletter is collecting new relevant information to be conveniently delivered by e-mails, allowing users to stay informed even if they are not actively using the system for some time.

4. Deployment and evaluation.

4.1. Overview. We implemented the system using Java and standard open source database and web-technology and based on the JIAC IV agent framework [20]. JIAC IV is FIPA 2000 compliant [21], that is it is conforming to the latest standards.

It consists of a communication infrastructure as well as services for administrating and organizing agents (Agent Management Service, AMS and Directory Facilitator, DF). The JIAC IV framework provides a variety of management and security functions, management services including configuration, fault management and event logging, security aspects including authorization, authentication and mechanisms for measuring and ensuring trust and therefore has been a good choice to be used from the outset to the development of a real world application.

Within JIAC IV, agents are arranged on platforms, allowing the arrangement of agents that belong together with the control of at least one manager. A lot of visual tools are offered to deal with administration aspects. Figure 3.2 shows a platform, in this case with agents for the building of different models specialized for different retrieval algorithms.

The prototypical system is currently running on Sun-Fire-880, Sun-Fire-480R and Sun Fire V65x, whereas the main filtering computation, database- and web-server and information-extraction is placed on different machines for performance reasons.

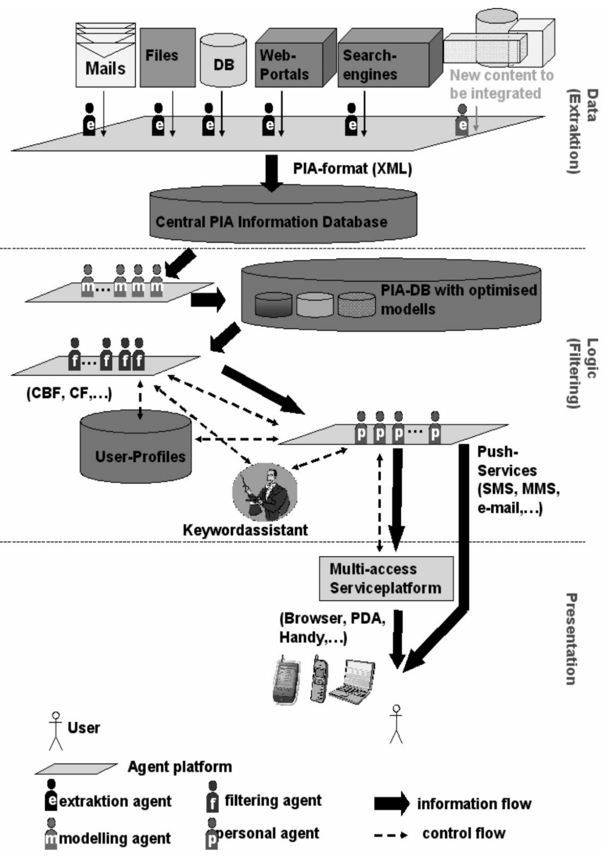

Fig. 3.1. *The PIA System seen as a three tier application*

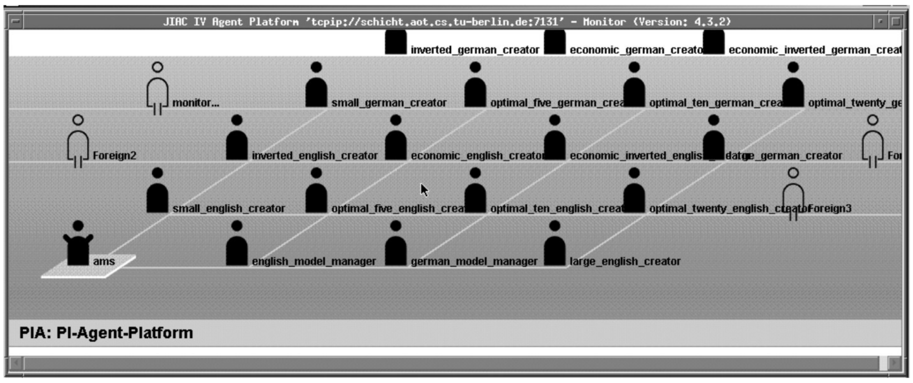

Fig. 3.2. *Several agents are building different models specilised for different retrieval algorithms*

New content is stored, validated and consolidated in a central relational database (update-driven). Information can be accessed by WWW, e-mail, SMS, MMS, and J2ME Clients, where the system adapts the presentation accordingly, using the CC/PP (Preferences Profile) with a tailored layout for a mobile phone and a PDA (see Section 4.6). The personalized newsletter and the push-services are sent via e-mail, SMS or MMS. The user can use self-defined keywords for a request for information or choose a category and therefore the system will use a list of keywords predefined by experts and updated smoothly by learning from collaborative filtering. A combination of both is also possible. The keyword assistant is able to extract the most import keywords of a given article using the term frequency inverse document frequency (TFIDF)-algorithm [22].

4.2. Gathering new information. New information is constantly inserted in the system by information extraction agents, e.g. web-reader agents or agents that are searching specified databases or directories. Additional agents for the collection of new content can easily be integrated even at runtime, as all that is necessary for a new agent is to register himself at the system, store the extracted information at a defined database table and inform the modeling-manager agent about the insertion. As a file reader-agent is constantly observing a special directory, manual insertion of documents can be done simply by drag-and-drop and an e-mail and upload-interface also exists. Source can also be integrated by Web services. New Readers can be created using a easy-to-handle tool and another tool is enabling to conveniently observe the extraction-agents, as this is the interface to the outside that might become critical if for example the data-format of a source is changed.

4.3. Pre-processing for efficient retrieval. The first step of pre processing information for efficient retrieval is the use of distinct tables in the global database for different domains like e.g. news, agent-related papers, etc. Depending on the filtering request, tables with no chance of being relevant can therefore be omitted. The next step is the building of several models for each document. Stemming and stop-word elimination is implemented in every model but different models are built by computing a term importance either based only on local frequencies, or based on term frequency inverse document frequency (TFIDF) approach. Furthermore number of words that should be included in models is different which makes models either more accurate or more efficient. Created models are indexed either on document or word level, which facilitate their efficient retrieval. The manager agent is assigning the appropriate modeling agents to start building their models but might decide (or the human system administrator can tell him) at runtime to delay latest time-consuming modeling activity for a while if system load is critical at a moment. This feature is important for a real-world application, as overloading has been a main reason for the un-usability of advanced academic systems.

4.4. Filtering technology. As the quality of results to a particular filtering request might heavily depend on the information domain (news, scientific papers, conference calls), different filtering communities are implemented. For each domain, there is at least one community which contains agents being tailored to do specific filtering and managing tasks in an efficient way. Instead of having only filtering agents (they will be described in Section 4.5), each and every community has also one so-called manager agent that is mainly responsible for doing coordination of filtering tasks, as well as cooperation with other managers.

The coordination is based on quality, CPU, DB and memory fitness values, which are the measures being associated to each filtering agent [23]. These measures respectively illustrate filtering agent successfulness in the past, its efficiency in using available CPU and DB resources, and the amount of memory being required for filtering. A higher CPU, DB or memory fitness value means that filtering agent needs a particular resource in a smaller extent for performing a filtering task. This further means that an insufficiency of a particular resource has a smaller influence on filtering agents with a higher particular fitness value.

The introduced different fitness values together with the knowledge about the current system runtime performance can make coordination be situation aware (see also [23]) in the way that when a particular resource is highly loaded a priority in coordination should be given to filtering agents for which a particular resource has a minor importance. This situation aware coordination provides a way to balance response time and filtering accuracy, which is needed in overcoming the problem of finding a perfect filtering result after few hours or even few days of an expensive filtering.

Instead of assigning filtering task to the agent with the best combination of fitness values in the current runtime situation, manager is going to employ a proportional selection principle [24] where the probability for the agent to be chosen to do actual filtering is proportional to the mentioned combination of its fitness values. By not always relying only on the most promising agents, but also sometimes offering a job to other agents, manager gives a chance to each and every agent to improve its fitness values. While the adaptation of quality fitness value can be accomplished after the user feedback became available, the other fitness values can be changed immediately after the filtering was finished through the response time analyses. The adaptation scheme has a decreasing learning rate that prevents already learnt fitness values of being destroyed, which further means that proven agents pay smaller penalties for bad jobs than the novice ones [17].

The underlying coordination activities, essentially responsible for the mentioned optimal exploitation of available system resources, are represented on Figure 4.1, giving the simplest possible selection scenario. Under the assumption that everything goes perfectly, the scenario starts with a job creation and ends with a result usage, being done by the User agent (U). The real coordination activities, being performed in a meantime by the chosen Manager (M), are first resource estimation, and afterwards strategy selection. After the selected Filtering agent (F) that encapsulates the particular searching algorithm (deployed filtering strategies are described in Section 4.5), is produced results, the manager M can adapt fitness values based on the measurement of the response time. The found filtering results are finally returned back to the user agent U, and this simple scenario ends.

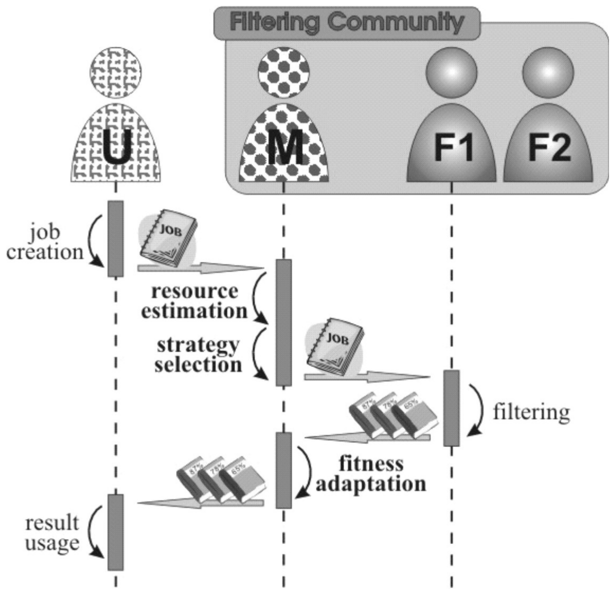

Fig. 4.1. *System architecture illustrating agent communication for resource-aware coordination*

In the case where the received filtering task cannot be successfully locally accomplished usually because of belonging to unsupported information domain, manager agent has to cooperate with other communities. While coordination takes place inside each and every filtering community between manager and filtering agents, cooperation occurs between communities among manager agents (see also Figure 4.2). The cooperation is based on either finding a community which supports given domain or in splitting received task on sub-tasks where for each sub-task a community with good support exists.

The information is usually scattered around many different, more or less dynamic, distributed sources. Two cornerstone challenges therefore become both finding which sources should be consulted for resolving the particular request, as well as putting the found results together. While the challenge of searching for sources is known as the database selection problem, the composing of a final result set is often simply referred as the information fusion. One light cooperation approach, already published in [25], and which is based on the application of the intelligent cooperative agents, is going to be briefly illustrated in the rest of this sub-section.

The fundamental cooperation idea is based on the installation of at least one filtering community around each database, as well as on setting up the sophisticated mechanisms, which enable that these communities can efficiently help each other while processing the incoming requests. Although the filtering request can be sent to any filtering community, the most suitable communities will be autonomously found, and the request will be then dispatched to them. The found results will be finally collected, and only the best ones will be returned to the sender of the filtering request. The most appealing property behind these cooperative processing is that everything is done transparently for the user, being not any more forced to manually think where the request should be sent, and which obtained filtering results are really the best ones.

Example (Coordination & Cooperation) Figure 4.2 presents a high level overview of the filtering framework being composed of three different filtering communities (FC), where each community has one filter manager agent (M) and different number of specialized filtering agents (F). There are two different databases (DB) with information from different domains, and they are accessed at least by one community. On the figure cooperation is illustrated as a circle with arrows which connect manager agents.

4.5. Filtering strategies. The cornerstone of the PIA system is in offering a framework that facilitates the integration of different filtering strategies. Although this paper is not dealing with any particular filtering

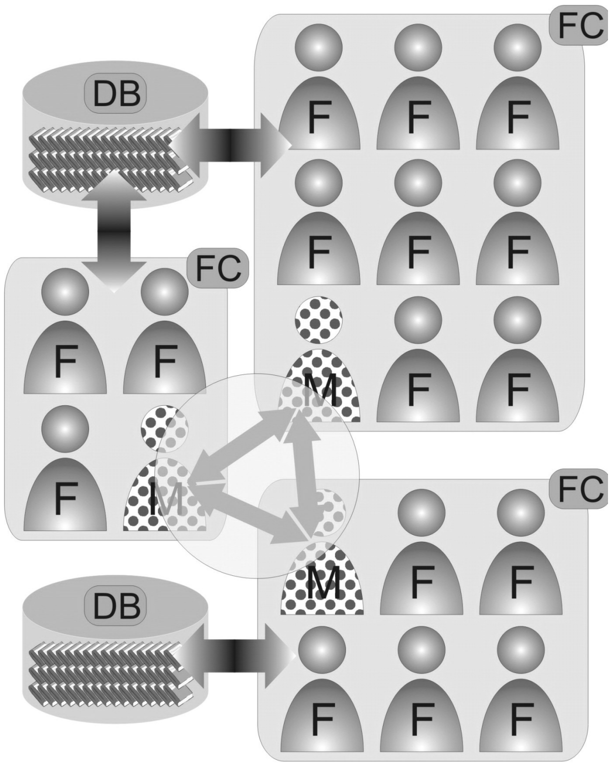

Fig. 4.2. *Filtering framework with three different communities*

strategy, their short descriptions will be given in the following paragraphs in order to make this paper self-contained and to clear the roles of the agents on Figure 3.2.

By using the term frequency inverse document frequency scheme, the importance values of different words can be computed, and each and every document can be modelled by a corresponding weighted vector. While the so-called *Large Filtering Strategy* will always build a model with all words from a document, *Optimal Twenty*, *Optimal Ten*, and *Optimal Five Filtering Strategies* will take into consideration only twenty, ten, and five most important words, respectively. The models, being created by these optimal strategies, are thus smaller, and consequently can be faster both loaded into memory and compared with a filtering request. As they are omitting many words, they might be at the same time potentially less accurate, and the coordination engine has a chance to decide which one in the given situation can be the best solution.

Since the examination of every single document for each request becomes infeasible even for a collection with the modest size, two different indexing filtering strategies have been also implemented. The first one, named *Inverted List Filtering Strategy*, creates for every word the list of documents having that word. The inbuilt simplification, tending to dramatically reduce a size of inverted lists, is made by not storing the positions of words in the corresponding documents. While a strategy due to such a design decision becomes more efficient, it loses its ability to support requests with a phrase. The second *Position Filtering Strategy* will not utilise such a simplification regarding not storing the positions, and thus will be able to accurately find documents with requested phrases. As this second strategy is naturally more expensive, the trade-off, between providing the accurate results and responding quickly, becomes evident and unavoidable for requests with phrases.

The property of fuzzy clustering [24], to assign documents to multiple clusters together with specifying a degree to which a particular article belongs to a given cluster, has been used as the inspiration for a realisation of a dedicated *Fuzzy Filtering Strategy*. While its strength is in keeping short cluster summaries in the high speed memory, its greatest weakness lies in a used simplification to cluster documents in advance fixed clusters. The few different versions of this fuzzy filtering strategy are finally implemented by limiting the amount of a memory that is utilised for cashing the cluster summaries, having as the implication that different trade-offs between the response time and the memory requirements are possible.

Every mentioned filtering strategy is also exploited for creating its appropriate clone, which will take into account only words from a manually created dictionary. By limiting the vocabulary to few thousands instead to more than half a million, underlying models are much smaller, and thus the underlying strategies become more efficient. Unfortunately, the paid price lies in the lost of a support for all requests with words that are not pre-selected, resulting in the potentially lower quality of found filtering results. These clone strategies finally provide even more fruitful playing ground for both cooperation and coordination mechanisms, which should prove their capabilities while resolving the mentioned trade-off problems.

4.6. Presentation. As one of the main design principles has been to support the user throughout the whole day, the PIA system provides several different access methods and adapts its interfaces to the used device (Figure 4.3). To fulfill these requirements an agent platform (Multi Access Service Platform) was developed that optimizes the graphical user interface for the access by Desktop PCs, PDAs and smart phones.

If the user wants to use the PIA system, the request is received by the Multi Access Service Platform (MASP). The MASP delegates the request to an agent, providing the logic for this service. In the PIA system the requests are forwarded either to login agent or the personal agent. The chosen agent performs the service specific actions and sends the MASP an abstract description of the formular that should be presented to the user. For this purpose the XML based Abstract Interaction Description Language (AIDL) has been defined. Based on the abstract description and the features of the used device the MASP generates an optimized interface presented to the user. The conversion is implemented as a XSLT transformation in which the optimal XSLT style sheet is selected based on the CC/PP information about the user's device.

The Multi Access Service Platform provides a generic infrastructure for providing device optimized interfaces for a big number of devices. The basic idea of MASP is to separate the application logic from the concrete interface design. So the application developer does not have to cope with the specific characteristic of the each relevant device and can concentrate on the application specific data flow and interaction logic.

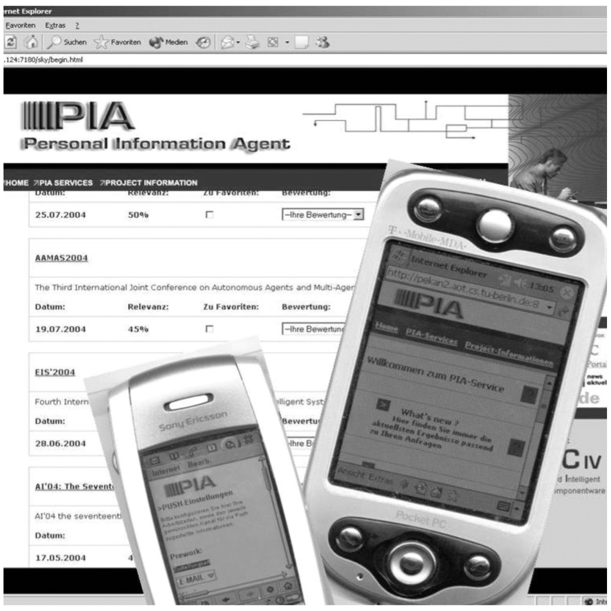
PIA
Personal Information Agent
HOME PIA SERVICES PROJECT INFORMATION
AAMAS2004
EIS'2004
AI'04: The Seventh
Willkommen zum PIA-Service
PUSH-Einstellungen
Sony Ericsson
Pocket PC

FIG. 4.3. *Information accessed by browser or tailored for presentation on a PDA or a mobile phone*

For defining the interface of an application the XML based Abstract Interaction Description Language (AIDL) has been defined. The definition of a user interaction (typically one web page) is structures as a tree of predefined user interface elements (e.g. label, input field). An exemplary page description is shown in Program 1.

The abstract interface description can be easily transformed into HTML, PDA optimized HTML or WML. If the user wants to have a voice interface, a style sheet for converting the abstract user interface description into VoiceXML has to be added to the MASP. Additional changes at the application are not needed. In general, the support for new devices can be added without changing or shutting down the application.

Program 1 The abstract interaction description of the PIA login page

```
<?xml version="1.0" encoding="UTF-8"?>
<scenario name="loginPage">
  <input>
    <UIElement>
      <list name="rootNode">
        <UIElement>
          <pageSetting name="user_language">
            <value>de</value>
          </pageSetting>
        </UIElement>
        <UIElement>
           <label name="login__piaLoginQXQ25">
             <value>PIA-Login</value>
           </label>
        </UIElement>
        <UIElement>
          <list name="login__data">
            <UIElement>
              <label name="login__userName">
                <value>Benutzername:</value>
              </label>
            </UIElement>
            <UIElement>
              <fieldValue name="login__userName_default">
                  <value>andreas</value>
               </fieldValue>
            </UIElement>
          </list>
        </UIElement>
         ...
        <serviceLink name="createAccountServiceLink">
          <provider address="tcpip://127.0.0.1:7325" name="PIAgent"/>
          <service name="MAPPresent"/>
          <param name="scenario">createAccount</param>
        </serviceLink>
      </list>
    </UIElement>
  </input>
</scenario>
```

The transformation of the abstract interface description is done using Extensible Stylesheet Language Transformations (XSLT). A XSLT transformation is typically written by a designer who is an expert for creating optimized user interfaces for a device considering the preferences of the respective audience. For simplifying the building of XSLT transformations, the MASP provides a set of generic rules for transforming the frequent elements of the abstract user interface descriptions into basic HTML or WML. Based on these rules more complex and device optimized XSLT transformations can be defined.

An important feature of the utilised MASP is the support of Composite Capability/Preference Profiles (CC/PP). Considering the specific features and properties (e.g. screen size, supported css version, supported image formats) the user interface designer can optimize the interfaces to the properties of the respective device. For converting media data into a device adequate format, the MASP provides a component for scaling and converting images and videos.

The components and interfaces of the Multi-Access-Service Platform are shown in Figure 4.4. Users who want to use the PIA service interact with the Multi Access Point. The MAP contains components for interaction with the respective device (e.g. web server or voice server) and components for rendering the application interface optimized for supported devices. Approved rendering components for HTML, WML and VoiceXML based user interfaces exists; components for applet based components are under development. For the device independent interface description the MASP uses the Abstract Interface Description Language (AIDL) that is use as interface between interface designer and application developer. The bridge between the application and the Abstract Interface Description is provided by the Alter Ego Agent that contains the interaction description and specific representation rules. Additionally the Media Cache component provides the media content as well as connectivity to external media providers.

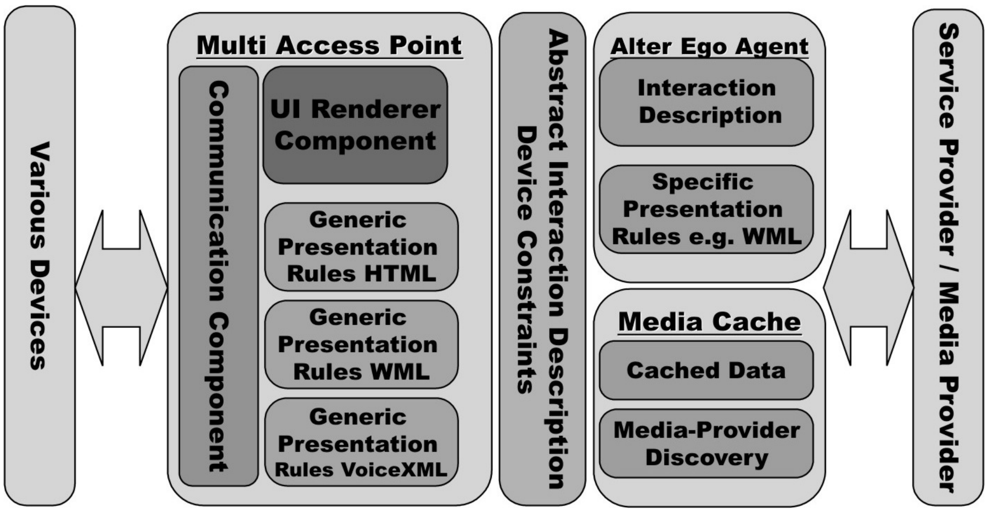
Various Devices
Multi Access Point
Communication Component
UI Renderer Component
Generic Presentation Rules HTML
Generic Presentation Rules WML
Generic Presentation Rules VoiceXML
Abstract Interaction Description Device Constraints
Alter Ego Agent
Interaction Description
Specific Presentation Rules e.g. WML
Media Cache
Cached Data
Media-Provider Discovery
Service Provider / Media Provider

FIG. 4.4. *The architecture of the MASP*

Beside of the features provided by MASP the design of the user interface must create an easy to use system even on devices with a tiny screen and without a keyboard. That is why the PIA interface provides additional navigation elements on complex forms and minimizes the use of text input fields. New results matching a defined request are presented first as a list of short blocks containing only title, abstract and some meta-information (as this is familiar to every user from well-known search-engines). This information is also well readable on PDAs or even mobile phones. Important articles can be stored in a repository. This allows the user to choose the articles on his PDA he wants to read later at his desktop PC.

Depending on the personal options specified by the user, old information found for a specific query may be deleted automatically step by step after a given time, that is, there is always up to date information that is presented to the user (we call this *smart mode*). This is for example convenient for getting personalized filtering news. The other option is to keep that information unlimited (*global mode*) for a query for e.g. basic scientific papers.

For the *push-services* (that is, the system is becoming active and sending the user information without an explicit request), the user specifies his working time (e.g. 9 am to 5 pm). This divides the day in a pre-, work, and a recreation slot, where the PIA system assumes different demands of information. For each slot an adequate delivering technology can be chosen (e-mail, sms, mms, fax or Voice). If you decide to subscribe to the personalized newsletter, new relevant information for you will be collected and sent by e-mail or fax once a day with a similar layout and structure for convenient reading if you have not seen it already by other pull- or push services. Therefore you can also stay informed without having to log into the system and if you do not want to get all new information immediately.

5. Conclusion and future work. The implemented system has an acceptable runtime performance and shows that it is a good choice to develop a personal information system using agent-technology based on a solid agent-framework like JIAC IV. Currently, PIA system supports more than 120 different web sources, grabs daily around 3.000 new semi-structured and unstructured documents, has almost 500.000 already pre-processed articles, and actively helps about fifty scientists related to our laboratory in their information retrieval activities. Their feedback and evaluation is a valuable input for the further improvement of PIA. In the near future we plan to increase the number of users to thousands, and therefore we plan to work on the further optimization of the filtering algorithms to be able to simultaneously respond to multiple filtering requests. Also, we think about integrating additional services for the user that provide information tailored to his geographical position (GPS), a natural speech interface and innovative ways to motivate the user to give precise explicit feedback, as the learning ability of the system is depending on that information.

REFERENCES

[1] S. Brin and L. Page, *The anatomy of a large-scale hyper textual (Web) search engine*, in Proc. 7th International World Wide Web Conference on Computer Networks, 30(1-7), 1998, pp. 107–117.

[2] M. Balabanovic and S. Yoav, *FAB: Content-Based Collaborative Recommendation*, Communication of the ACM, 40(3), 1997, pp. 66–72.

[3] A. Moukas, *Amalthaea: Information Discovery and Filtering using a Multi agent Evolving Ecosystem*, in Practical Application of Intelligent Agents & Multi-Agent Technology, London 1998.

[4] B. Zhang, and Y. Seo, *Personalized Web-Document Filtering Using Reinforcement Learning*, Applied Artificial Intelligence, 15(7), 2001, pp. 665–685.

[5] M. Claypool and A. Gokhale and T. Miranda and P. Murnikov and D. Netes and N. Sartin, *Combining Content-Based and Collaborative Filters in an Online Newspaper*, in Proc. ACM SIGIR Workshop on Recommender Systems, Berkeley, CA, August 19, 1999.

[6] J. Delgado and R. Davidson, *Knowledge bases and user profiling in travel and hospitality recommender systems*, in Proc. of the ENTER 2002 Conference, Innsbruck, Austria, 2002, pp. 1–16.

[7] D. Kuropka and T. Serries, *Personal Information Agent*, in Proc. Informatik Jahrestagung 2001, pp. 940–946.

[8] H. Lieberman, *Letizia: An Agent That Assists Web Browsing*, in Proc. International Joint Conference on Artificial Intelligence, Montreal, August 1995.

[9] H. Lieberman and N. Van Dyke and A. Vivacqua, *Let's Browse: A Collaborative Browsing Agent*, Knowledge-Based Systems, 12(8), 1999, pp. 427–431.

[10] B. Sheth, *A Learning Approach to Personalized Information Filtering*, M.Sc. Thesis, MIT dept, USA, 1994.

[11] T. Joachims and D. Freitag and T. Mitchell, *WebWatcher: A Tour Guide for the World Wide Web*, in Proc. IJCAI (1), 1997, pp. 770–777.

[12] W. Winiwarter, *PEA—A Personal Email Assistant with Evolutionary Adaptation*, International Journal of Information Technology, 5(1), 1999.

[13] C. Thomas and G. Fischer, *Using agents to improve the usability and usefulness of the world wide web*, in Proc. 5th International Conference on User Modelling, 1996, pp. 5–12.

[14] N. Jennings and M. Wooldridge, *Agent-oriented software engineering*, Handbook of Agent Technology (ed. J. Bradshaw), AAAI/MIT Press, 2000.

[15] R. Sesseler and S. Albayrak, *Agent-based Marketplaces for Electronic Commerce*, in Proc. International Conference on Artificial Intelligence, IC-AI 2001.

[16] P. Resnick and J. Neophytos and M. Suchak and P. Bergstrom and J. Riedl, *GroupLens: An open architecture for collaborative filtering of net news*, in Proc. ACM Conference on Computer-Supported Cooperative Work, 1994, pp. 175–186.

[17] S. Albayrak and D. Milosevic, *Self Improving Coordination in Multi Agent Filtering Framework*, in Proc. IEEE/WIC/ACM International Joint Conference on Intelligent Agent technology (IAT 04) and Web Intelligence (WI 04), Beijing, China, September 2004.

[18] D. Nichols, *Implicit Rating and Filtering*, in Proc. 5th DELOS Workshop on Filtering and Collaborative Filtering, Budapest, Hungary, 1997, pp. 31–36.

[19] D. Tauritz, *Adaptive Information Filtering: concepts and algorithms*, Ph.D. dissertation, Leiden University, 2002.

[20] S. Fricke and K. Bsufka and J. Keiser and T. Schmidt and R. Sesseler and S. Albayrak, *Agent-based Telematic Services and Telecom Applications*, Communications of the ACM, 2001.

[21] *Foundation for Intelligent Physical Agents*, `www.fipa.org` 2004.

[22] L. Jing and H. Huang and H. Shi, *Improved Feature Selection Approach TFIDF in Text Mining*, in Proc. 1st International Conference on Machine Learning and Cybernetics, Beijing, 2002.

[23] S. Albayrak and D. Milosevic, *Situation-aware Coordination in Multi Agent Filtering Framework*, in Proc. 19th International Symposium on Computer and Information Sciences (ISCIS 04), Antalya, Turkey, 2004.

[24] M. Zbigniew and D. Fogel, *How to Solve It: Modern Heuristics*, Springer-Verlag New York, Inc., New York, NY, 2000.

[25] S. Albayrak and D. Milosevic, *Cooperative Community Selection in Multi Agent Filtering Framework*, in Proc. IEEE/WIC/ACM International Joint Conference on Intelligent Agent technology (IAT 05) and Web Intelligence (WI 05), Compiegne University of Technology, France, 2005, pp. 527–535.

Edited by: Marcin Paprzycki, Niranjan Suri
Received: October 1, 2006
Accepted: December 10, 2006