PACMAS: A Personalized, Adaptive, and Cooperative MultiAgent System Architecture

Giuliano Armano, Giancarlo Cherchi, Andrea Manconi, and Eloisa Vargiu
University of Cagliari
Piazza d'Armi
I-09123, Cagliari, Italy
Email: {armano,cherchi,manconi,vargiu}@diee.unica.it

Abstract— **In this paper, a generic architecture, designed to support the implementation of applications aimed at managing information among different and heterogeneous sources, is presented. Information is filtered and organized according to personal interests explicitly stated by the user. User profiles are improved and refined throughout time by suitable adaptation techniques. The overall architecture has been called PACMAS, being a support for implementing Personalized, Adaptive, and Cooperative MultiAgent Systems. PACMAS agents are autonomous and flexible, and can be made personal, adaptive and cooperative, depending on the given application. The peculiarities of the architecture are highlighted by illustrating three relevant case studies focused on giving a support to undergraduate and graduate students, on predicting protein secondary structure, and on classifying newspaper articles, respectively.**

I. Introduction

Accessing the widespread amount of distributed information resources, such as the World Wide Web (WWW), entails relevant problems (e.g., "information overload" [19]). Moreover, different users are typically interested in different parts of the available information, so that personalized and effective information-filtering procedures are needed. Software agents have been widely proposed for dealing with this kind of information retrieval and filtering problems [13] [8] [15] [25].

From our perspective, assuming that information sources are a primary operational context for software agents, the following categories can be identified focusing on their specific role: (i) *information agents*, able to access to information sources and to collect and manipulate such information [19], (ii) *filter agents*, able to transform information according to user preferences [18], (iii) *task agents*, able to help users to perform tasks by solving problems and exchanging information with other agents [10], (iv) *interface agents*, in charge of interacting with the user such that she/he interacts with other agents throughout them [17], and (v) *middle agents*, devised to establish communication among requesters and providers [7]. Although this taxonomy is focused on a quite general perspective, alternative taxonomies could be defined focusing on different features. In particular, one may focus on capabilities rather than roles, a software agent being able to embed any subset of the following capabilities: (i) *autonomy*, to operate without the intervention of users; (ii) *reactivity*, to react to a stimulus of the underlying environment according to a stimulus/response behaviour; (iii) *proactiveness*, to exhibit goal-directed behavior in order to satisfy a design objective; (iv) *social ability*, to interact with other agents according to the syntax and semantics of some selected communication language; (v) *flexibility*, to exhibit reactivity, proactiveness, and social ability simultaneously [24]; (vi) *personalization*, to personalize the behavior to fulfill user's interests and preferences; (vii) *adaptation*, to adapt to the underlying environment by learning how to react and/or interact with it; (viii) *cooperation*, to interact with other agents in order to achieve a common goal; (ix) *deliberative capability*, to reason about the world model and to engage planning and negotiation, possibly in coordination with other agents; (x) *mobility*, to migrate from node to node in a local- or wide-area network.

In this paper, we present a generic multiagent architecture designed to support the implementation of applications aimed at: (i) retrieving heterogeneous data spread among different sources (i.e., generic html pages, news, blogs, forums, and databases), (ii) filtering and organizing them according to personal interests explicitly stated by each user, and (iii) providing adaptation techniques to improve and refine throughout time the profile of each selected user.

Each agent is autonomous and flexible, and may implement (one or more of) the following capabilities: personalization, adaptation, and cooperation. The overall architecture has been called PACMAS, being designed to support the implementation of Personalized, Adaptive, and Cooperative MultiAgent Systems. The PACMAS architecture can easily give rise to specific systems by (1) identifying the characteristics of the dataflow that occurs from information sources to users (and vice versa), and (2) customizing each involved agent according to its actual role and capabilities.

The remainder of this paper is organized as follows: In Section 2 the Personalized, Adaptive, and Cooperative architecture, called PACMAS, is depicted. In Section 3, three case studies are presented, each one customized for a specific application. Section 4 draws conclusions and future work.

II. The PACMAS Architecture

PACMAS is a generic multiagent architecture aimed at retrieving, filtering and reorganizing information according to users' interests. PACMAS agents can be personalized, adaptive, and cooperative, depending on their specific role.

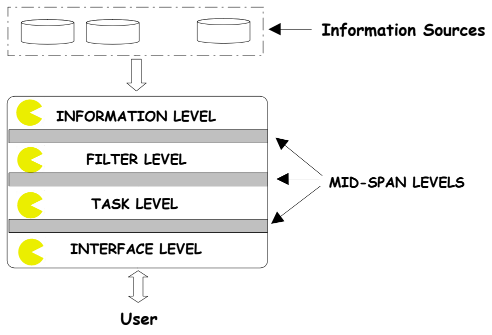


Fig. 1. The PACMAS Architecture.

PACMAS Macro-Architecture

The overall architecture (depicted in Figure 1) encompasses four main levels (i.e., information, filter, task, and interface), each being associated to a specific role. The communication between adjacent levels is achieved through suitable middle agents, which form a corresponding mid-span level.

Each level is populated by a society of agents, so that communication may occur both horizontally and vertically. The former kind of communication supports cooperation among agents belonging to a specific level, whereas the latter supports the flow of information and/or control between adjacent levels through suitable middle-agents.

At the information level, agents are entrusted with extracting data from the information sources. Each information agent is associated to one information source, playing the role of wrapper. Upon extraction, the information is then made available to the underlying filter level.

At the filter level, agents are aimed at selecting information deemed relevant to the users, and cooperate to prevent information from being overloaded and redundant. Two filtering strategies can be adopted: generic and personal. The former applies the same rules to all users; whereas the latter is customised for a specific user. Each strategy can be implemented through a pipeline of filters, since data undergo an incremental refinement process. The information filtered so far is then made available to the task level.

At the task level, agents arrange data according to users' personal needs and preferences. In a sense, they can be considered as the core of the architecture. In fact, they are devoted to achieve users' goals by cooperating together and adapting themselves to the changes of the underlying environment. In general, they can be combined together according to different connection modes, depending on the specific application.

At the interface level, a suitable interface agent is associated to each different user interface. In fact, a user can generally interact with an application through several interfaces and devices (e.g., pc, pda, mobile phones, etc.). Interface agents usually act individually without cooperation. On the other hand, they can be personalized to display only the information deemed relevant to a specific user. Moreover, in complex applications, they can adapt themselves to progressively improve their ability in supplying information to the user.

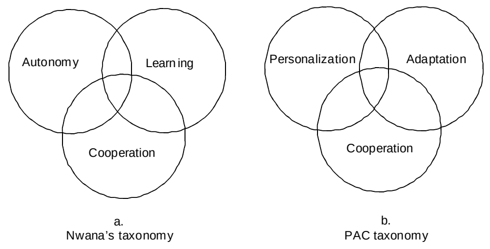


Fig. 2. Agents taxonomies.

At the mid-span level, agents are aimed at establishing communication among requesters and providers. In the literature, several solutions have been proposed: e.g., blackboard agents, matchmaker or yellow page agents, and broker agents (see [7] for further details). In the PACMAS architecture, agents at the mid-span level can be implemented as matchmakers or brokers, depending on the specific application.

PACMAS Micro-Architecture

Keeping in mind that agents may be classified along several ideal and primary capabilities that they should embed, let us first recall the agent taxonomy proposed in [20]. In such taxonomy, three primary capabilities have been identified: autonomy, learning, and cooperation (see Figure 2-a). In our view, agents are always autonomous and flexible, hence we deem that autonomy should not be explicitly listed in a diagram. On the contrary, we claim that personalization should be taken into account as a primary feature while depicting the characteristics of software agents, the resulting taxonomy is depicted in Figure 2-b.

As for personalization, an initial user profile is provided in form of a list of keywords, representing users' interests. The information about the user profile is stored by agents belonging to the interface level. It is worth noting that, to exhibit personalization, filter and task agents may need information about the user profile. This flows up from the interface level to the other levels through the middle-span levels. In particular, agents belonging to mid-span levels (i.e., middle agents) take care of handling synchronization and avoiding potential inconsistencies. Moreover, the user behavior is tracked during the execution of the application to support explicit feedback, in order to improve her/his profile.

As for adaptation, a model centered on the concept of "mixtures of experts" has been employed. Each expert is implemented by an agent able to select relevant information according to an embedded string of feature-value pairs, features being selectable from an overall set of relevant features defined for the given application. The decision of adopting a subset of the available features has been taken for efficiency reasons, being conceptually equivalent to the one usually adopted in a typical GA-based environment [11], which handles also dont-care symbols. The system starts with an initial population of experts, during the evolution of the system further experts are created according to a covering, crossover, or mutation mechanism.

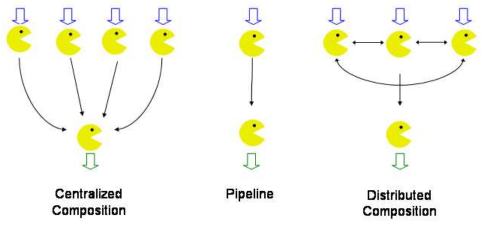


Fig. 3. Agents Connections.

As for cooperation, agents at the same level exchange messages and/or data to achieve common goals, according to the requests made by the user. Cooperation is implemented in accordance with the following modes: centralized composition, pipeline, and distributed composition (see Figure 3). In particular: (i) centralized compositions can be used for integrating different capabilities, so that the resulting behavior actually depends on the combination activity; (ii) pipelines can be used to distribute information at different levels of abstraction, so that data can be increasingly refined and adapted to the user's needs; and (iii) distributed compositions can be used to model a cooperation among the involved components aimed at processing interlaced information. The most important form of cooperation concerns the "horizontal" control flow that occurs between peer agents. For instance, filter agents can interact in order to reduce the information overload and redundancy, whereas task agents can work together to solve problems that require social interactions to be solved.

III. Case Studies

In order to highlight the peculiarities of the architecture, three relevant case studies are presented. The first one is focused on giving a support to undergraduate and graduate students; the second one is concerned with the problem of predicting protein secondary structure; and the third one is devoted to classify newspaper articles.

All the proposed case studies have been implemented using Jade [4] as the underlying framework.

PACMAS for Supporting Students in University Activities

This case study is focused on giving a support to undergraduate and graduate students [1].

Motivation: Let us consider a typical University Department. It generally makes available the information about courses, seminars, exams, professors, and students on different areas: web sites, forums, and news (NNTP) servers. All the relevant information is spread on the department portal, on the web site of each course, and on the personal page of each professor. Furthermore, each professor might activate her/his news and forum service. Some of the information potentially interests all students, such as lesson timetables, exam dates, taxes, and student tutoring. On the other hand, students belonging to different courses are interested in different lessons and exams. For example, a student attending the MSc in Computer Science may be interested in the *Object Oriented Programming Languages I* course rather than in the *Processors and Embedded Systems Architectures* one. Similarly, a student attending the MSc in Digital Microelectronics may be interested in the *Processors and Embedded Systems Architectures* course rather than in *Object Oriented Programming Languages I* one. Typically, a student in search of relevant information about her/his University activities browses web sites, and reads announcements from forum and news services. This is a repetitive and boring task that can be automated. From our perspective, personalization and adaptation represent the added value of such an automated system.

Implementation: Using PACMAS, we developed a system devoted to support undergraduate and graduate students in their University activity at the Department of Electrical and Electronic Engineering (DIEE) of the University of Cagliari. Let us note that supporting students involves several activities: information extraction, information retrieval and filtering, information processing, and results presentation. Each activity corresponds to a suitable level of the PACMAS architecture.

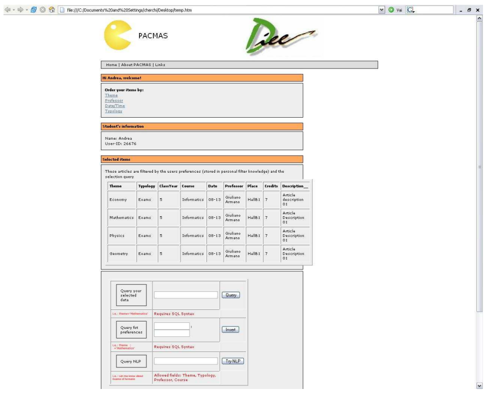
Information Extraction. It is carried out at the information level by information agents that play the role of wrappers, devised to process information sources. Each wrapper is specialized for dealing with a specific information source: e.g., web pages, forums or news services. In the current implementation, information agents are not personalized, not adaptive, and not cooperative ($\overline{PAC}$). Personalization is not supported, since information agents are aimed at retrieving information potentially relevant to all students, regardless of their personal interests and preferences. Adaptation is also not supported, being the system mainly concerned with changes in users needs rather than in the underlying environment[2]. Cooperation is also not supported, cause each information agent is devoted to wrap a different information source.

Information Retrieval and Filtering. It is carried out at the filter level. In particular, this level contains a set of "redundancy filters" (one for each information source), an anti-spam filter agent and a population of personal filter agents (one for each user of the system). Redundancy filters cooperate together to remove the redundancy of data provided by the information sources (throughout the information agents). Redundancy filters are not personalized, not adaptive, and cooperative ($\overline{PA}C$). Similarly to information agents, personalization and adaptation are not required. On the other hand, cooperation is required to prevent the information from being redundant. The anti-spam filter is not personalized, not adaptive, and not cooperative ($\overline{PAC}$)[3]. Being not dependent from a specific student, it filters the same information by removing undesirable contents according to a rule-based mechanism. Personal filters

[1]This work has been partially funded by the Italian Ministry of University and Research under the program PRIN 2003 *Programmi di Ricerca Scientifica di Rilevante Interesse Nazionale*.

[2]In this particular case the variability of the information sources

[3]In the current release of the system anti-spam agents are not permitted to implement adaptation, although in principle this property may be supplied in a future release.



file:///C:/Documents%20and%20Settings/cherchi/Desktop/temp.htm

PACMAS

Home | About PACMAS | Links

Hi Andrea, welcome!

Order your items by:
Theme
Professor
Date/Time
Typology

Student's information

Name: Andrea
User-ID: 26676

Selected items

These articles are filtered by the users preferences (stored in personal filter knowledge) and the selection query

Theme	Typology	ClassYear	Course	Date	Professor	Place	Credits	Description__
Economy	Exams	5	Informatics	08-13	Giuliano Armano	HallB1	7	Article description 01
Mathematics	Exams	5	Informatics	08-13	Giuliano Armano	HallB1	7	Article Description 01
Physics	Exams	5	Informatics	08-13	Giuliano Armano	HallB1	7	Article Description 01
Geometry	Exams	5	Informatics	08-13	Giuliano Armano	HallB1	7	Article Description 01

Query your selected data		Query
i.e.: theme="Mathematics"	Requires SQL Syntax	
Query fot preferences	:	Insert
i.e.: theme : ="Mathematics"	Requires SQL Syntax	
Query NLP		Try NLP
i.e.: Let me know about Exams of Armano	Allowed fields: Theme, Typology, Professor, Course	

Fig. 4. JSP graphical interface.

are personalized, adaptive and not cooperative ($PA\overline{C}$). As for personalization, they are sensible to any explicit change imposed by the corresponding student or to a change that occurs in the curriculum of the student. As for adaptation, they are able to progressively adapt their filtering capabilities according to the choices performed by the corresponding student during the lifetime of the agent. Cooperation is not supported; in fact, in the current release of the system, only a specific support for implementing voting policies according to the guidelines of GA-based systems is supplied.

Information Processing. It is carried out at the task level, where agents are devoted to perform different tasks according to the requirements imposed by the corresponding user. In particular, each task agent is customized for a specific task (e.g., lessons timetable, seminars, and exams scheduling). Agents belonging to the task level exploit a model centered on the concept of "mixtures of experts", each expert being implemented by an agent. The system supports each user with a specific population of experts, handled in accordance with the basic guidelines of online systems, expecially the ones that characterize evolutionary environments. Task agents are personalized, adaptive, and cooperative (PAC). Personalization is required since different behaviors are associated to different students. Adaptation is required since they adapt themselves to the needs of the corresponding student through a GA-based feedback mechanism. Cooperation is required since they usually need other task agents to successfully achieve their own goals.

Results Presentation. It is carried out at the interface level, through agents aimed at interacting with the users. Agents and users interact through a suitable graphical interface that can be run on several devices, including mobile phones. A different interface agent has been associated to each device. In the current implementation, the system embodies a graphical interface that runs on several devices, including MIDP 1.0 compliant devices, and JSP web pages (as the one shown in Figure 4)[4].

Interface agents are also devoted to handle user profile and propagate it by the intervention of middle agents. Furthermore, any feedback provided by the user can be exploited by the adaptive mechanism to improve the user profile. Interface agents are personal, adaptive, and not cooperative ($PA\overline{C}$). Personalization is required to allow each student the customization of her/his interface. Adaptation is supported, since an interface agent must adapt to the changes that occur in the preferences and interests of the corresponding student. Cooperation is not supported by agents that belong to this architectural level.

[4]Available at: http://iascw.diee.unica.it/PacmasWWW

PACMAS for Predicting Protein Secondary Structures

In this section we briefly describe an application concerned with the problem of predicting protein secondary structure using PACMAS (for further details see [2]).

Motivation: Difficulties in predicting protein structure are mainly due to the complex interactions between different parts of the same protein, on the one hand, and between the protein and the surrounding environment, on the other hand. Actually, some conformational structures are mainly determined by local interactions between near residues, whereas others are due to distant interactions in the same protein. Moreover, notwithstanding the fact that primary sequences are believed to contain all information necessary to determine the corresponding structure [1], recent studies demostrate that many proteins fold into their proper three-dimensional structure with the help of molecular chaperones that act as catalysts [9], [12]. The problem of identifying protein structures can be simplified by considering only their secondary structure; i.e. a linear labeling representing the conformation to which each residue belongs to. Thus, secondary structure is an abstract view of amino acid chains, in which each residue is mapped into a secondary alphabet usually composed by three symbols: alpha-helix (α), beta-sheet (β), and random-coil (c).

Implementation: Keeping in mind that the PACMAS architecture encompasses several levels, each one hosting a set of agents, in the following, we illustrate how each level supports the implementation of the proposed application.

At the information level, agents play the role of wrappers, which –in our view– can be considered a particular kind of filters, devised to process information sources. Each wrapper is associated to one information source: (i) the selected training set (the TRAIN database), (ii) the test set (the R126 database), and (iii) a database containing information about the domain knowledge (the AAindex database). Datasets are briefly summarized in Table I. In the current implementation, information agents are not personalized, not adaptive, and not cooperative (shortly $\overline{PAC}$). Personalization is not supported at this level, since information agents are only devoted to wrap the datasets containing proteins. Adaptation is also not supported, since information sources are invariant for the system and are not user-dependent. Cooperation is also not supported by the information agents, since each agent retrieves information from different sources, and each information source has a specific role in the chosen application.

At the filter level, agents embody encoding methods. Let us briefly recall that encoding methods play an important role

TABLE I
INFORMATION SOURCES FOR PREDICTING PROTEIN SECONDARY STRUCTURES

Dataset	Description
TRAIN	It has been derived from a PDB selection obtained by removing short proteins (less than 30 aminoacids), and with a resolution of at least 2.5 Å. This dataset underwent a homology reduction, aimed at excluding sequences with more than 50% of similarity. The resulting training set consists of 1180 sequences, corresponding to 282,303 amino acids.
R126	It has been derived from the historical Rost and Sander's protein dataset (RS126) [22], and corresponds to a total of 23,363 amino acids (the overall number has slightly varied over the years, due to changes and corrections in the PDB.)
AAindex	It contains information about hydrophobicity, dimension, charge and other features required for evaluating the given metrics. In the current application eight domain-specific metrics have been devised and implemented. A sample metrics is: Check whether hydrophobic amino acids occur in a window of predefined length according to a clear periodicity, whose underlying rationale is that sometimes hydrophobic amino acids are regularly distributed along alpha-helices.

in the prediction of protein secondary structures. In fact, they describe the chemical-physics properties of aminoacid deemed more interesting for the prediction. Several populations of filter agents have been implemented, each of them performing a different encoding techniques: one-shot, substitution matrices, multiple alignment algorithms, and a techique that combines the specificity of the multiple alignment technique with the generality of the substitution matrices. Personalization is not supported by filter agents, since they always embody the same encoding methods for all users. Adaptation is also not supported either, since encoding methods do not change during the application. Cooperation is supported by filter agents, as some implemented encoding methods brings together several algorithms (e.g., the encoding method that combines multiple alignment with substitution matrices).

At the task level, a population of task agents, which are the core of this case study, perform the protein secondary structure prediction. The "internals" of each task agent is based on the micro-architecture proposed for the NXCS-Experts [3]. In its basic form, each NXCS expert *E* can be represented by a triple $\langle g, h, w \rangle$, where: (i) *g* is a "guard" devised to check whether an input *x* can be processed or not, (ii) *h* is an embedded predictor whose activation depends on *g(x)*, and (iii) *w* is a weighting function used to perform output combination. Hence, the output of *E* coincides with *h(x)* for any input *x* "acknowledged" (i.e., matched) by *g*, otherwise it is not defined. Typically, the guard *g* of a generic NXCS classifier is implemented by an XCS-like classifier, able to match inputs according to a set of selected features deemed relevant for the given application, whereas the embedded predictor *h* consists of a feed forward ANN, trained and activated on the inputs acknowledged by the corresponding guard. In the case *E* contributes to the final prediction (together with other experts), its output is modulated by the value *w(x)*, which represents the expert strength in the voting mechanism. It may depend on several features, including *g(x)*, the overall fitness of the corresponding expert, and the reliability of the prediction made by the embedded predictor. It is worth noting that matching can be "flexible", meaning that the matching activity returns a value in [0,1] rather than "true" or "false". In this case, only inputs such that $g(x) \geq \sigma$ will be processed by the corresponding embedded predictor (σ being a system parameter). Task agents are not personalized, adaptive, and cooperative (shortly $\overline{P}AC$). Personalization is not required, since task agents exhibit the same behaviors for all the users. Adaptation is required, since each expert is suitably trained through a typical evolutionary behavior. Cooperation is required, since they usually need other task agents to successfully achieve their own goals.

At the interface level, agents are aimed at interacting with the user. In the current implementation, this kind of agents has not been developed. Nevertheless, we are investigating how to implement a flexible behavior at the user side. In particular, a suitable web interface is under study. We envision an interface personalized for each user, in which the user can input a protein to be predicted also being given the possibility of selecting the encoding technique to be applied. The resulting information agents will be personalized, adaptive, and not cooperative (shortly $PA\overline{C}$). Personalization will be required in order to allow each user to customize the user interface. Adaptation will be required, since agents could adapt themselves to the changes that occur in the user preferences. Cooperation will not be required by the agents belonging to this architectural level.

As for the mid-span levels, the corresponding middle agents exhibit a different behavior depending on the mid-span level that they belong to. In particular, let us recall that, in the PACMAS architecture, there are three mid-span levels, one between information and filter levels (in the following, IF level), one between filter and task levels (in the following, FT level), and one between task and interface levels (in the following, TI level). In this specific application personalization and adaptation are not supported by middle agents, since they are only devoted to connect together agents belonging to adjacent levels. Cooperation is supported by agents belonging to the IF and the FT levels, since in the training phase they are used to verify the prediction.

PACMAS for Newspaper Articles Classification

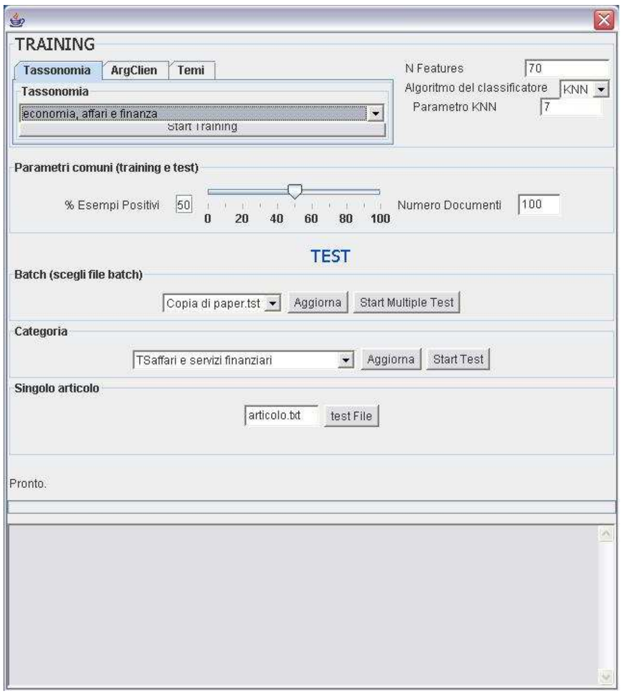
In this section we briefly describe the case study concerned with the problem of classifying newspaper articles using PACMAS (see [6] for details).

Motivation: All the information sources belonging to the WWW make it hard for users to choose the most suitable according to their interests. Finding useful information of personal interest has become difficult for Internet users. Ideally, users should be able to take advantage of the wide range of available information while being able to find the one she/he is interested in. In particular, manually selecting newspaper articles is quite difficult or not feasible within the time constraints common for most users also considering that the results could not perfectly fit with the user interests. Some systems try to perform that task automatically, performing content-based filtering. In particular, software agents have been widely proposed for retrieving information from the web ([23], [16], and [5]).

Implementation: At the information level, agents play the role of wrappers, each one being associated to a different information source. In particular, in the current implementation a set of agents wraps databases containing italian newspaper articles [5]. Furthermore, an agent wraps the proposed taxonomy that is a subset of the one proposed by the International Press Telecommunications Council [6]. Information agents are not personalized, not adaptive, and not cooperative (shortly $\overline{PAC}$). Personalization is not supported at this level, since information agents are only devoted to wrap information sources. Adaptation is also not supported, since we assume that information sources are invariant for the system and are not user-dependent. Cooperation is also not supported by the information agents, since each agent retrieves information from different sources.

At the filter level, a population of agents manipulates the information belonging to the information level through suitable filter strategies. First, a set of agents removes all non-informative words such as prepositions, conjunctions, pronouns and very common verbs by using a standard stop-word list. After stop-words removal, a set of agents performs a stemming algorithm [21] to remove the most common morphological and inflexional endings from words. Then, for each class, a set of agents selects the features relevant to the classification task according to the information gain method [7]. Filter agents are not personalized, not adaptive, and cooperative (shortly $\overline{PA}C$). Personalization is not supported at this level, since the adopted filter strategies are user-independent. Adaptation is also not supported, since the adopted strategies do not change during agents activities. Cooperation is supported by the filter agents, since agents cooperate continously in order to perform the filtering activity.

At the task level, a population of agents have been developed, each one embedding a *k*-NN classifier [8]. Each agent has been trained in order to recognize a specific class, and it is also devoted to measure the classification accuracy according to the confusion matrix [14]. Task agents are not personalized, adaptive, and cooperative (shortly $\overline{P}AC$). Personalization is not supported at this level, since, in the current implementation, the adopted classification strategies are user-independent. Adaptation is supported by the task agents since they learn the classification rules during their life. Cooperation is supported by the task agents, since agents sometimes have to interact


TRAINING

Tassonomia: economia, affari e finanza

N Features: 70

Algoritmo del classificatore: KNN

Parametro KNN: 7

Parametri comuni (training e test)

% Esempi Positivi: 50

Numero Documenti: 100

TEST

Batch (scegli file batch): Copia di paper.tst

Categoria: TSaffari e servizi finanziari

Singolo articolo: articolo.txt

Pronto.

Fig. 5. Interface for the newspaper articles classifying system.

each other in order to achieve their goals.

At the interface level, agents are aimed at interacting with the user. In the current implementation, agents and users interact through a suitable graphical interface that runs on a pc (see Figure 5). Interface agents are also devoted to handle user profile and propagate it by the intervention of middle agents. Interface agents are personal, not adaptive, and not cooperative (shortly $P\overline{AC}$). Personalization is required to allow each user the customization of her/his interface. In the current implementation adaptation is not supported, but in general an interface agent might adapt to the changes that occur in the preferences and interests of the corresponding user. Cooperation is not supported by agents that belong to this architectural level.

Discussion

The peculiarities of the architecture have been highlighted by depicting three relevant case studies. Table II shows agents and their capabilities for the proposed case studies. In particular, the added value of the proposed approach is that PACMAS agents are polymorphic in the sense that they can exhibit a different behavior depending on the specific application in which they operate.

IV. Conclusions and Future Work

In this paper a generic architecture designed to support the implementation of applications aimed at managing information among different and heterogeneous sources has been presented. Information is filtered and organized according to personal interests explicitly stated by the user. User profiles are improved and refined throughout time by suitable adaptation

[5]In general they may wrap any web sites containing newspaper articles (e.g., online newspapers).

[6]http://www.iptc.org/

[7]It measures the number of bits of information obtained for category prediction by knowing the presence or absence of a term in a document.

[8]The *k*-nearest neighbor is a classification method based upon observable features. The algorithm selects a set which contains the *k* nearest neighbours and assigns the class label to the new data point based upon the most numerous class with the set.

TABLE II
AGENTS CAPABILITIES

Agents	**Case study 1**	**Case study 2**	**Case study 3**
Information	$\overline{PAC}$	$\overline{PAC}$	$\overline{PAC}$
Filter	Redundancy: $\overline{PA}C$ Anti-spam: $\overline{PAC}$ Personal: $PA\overline{C}$	$\overline{PA}C$	$\overline{PA}C$
Task	PAC	$\overline{P}AC$	$\overline{P}AC$
Interface	$PA\overline{C}$	$PA\overline{C}$	$P\overline{AC}$
Middle	$\overline{PAC}$	IF: $\overline{PA}C$ FT: $\overline{PA}C$ TI: $\overline{PAC}$	$\overline{PA}C$

techniques. The overall architecture has been called PACMAS, being a support for implementing Personalized, Adaptive, and Cooperative MultiAgent Systems. PACMAS agents are autonomous and flexible, and can be personalized, adaptive and cooperative depending on the implemented application.

As for the future work, we are investigating how to improve the intelligent capabilities of agents with more complex forms of personalization, adaptation, and cooperation. Moreover, the possibility to implement further intelligent applications using PACMAS is currently under study.

V. ACKNOWLEDGEMENTS

We would like to thank Andrea Addis for participating in the implementation of the prototype.

REFERENCES

[1] C. Anfinsen. Principles that govern the folding of protein chains. *Science*, 181:223–230, 1973.

[2] G. Armano, G. Mancosu, A. Orro, M. Saba, and E. Vargiu. Biopacmas: A personalized, adaptive, and cooperative multiagent system for predicting protein secondary structure. In *AI*IA 2005: Advances in Artificial Intelligence, 9th Congress of the Italian Association for Artificial Intelligence (AI*IA 2005). LNAI 3673, Springer*, September 2005.

[3] G. Armano, A. Murru, and F. Roli. Stock market prediction by a mixture of genetic-neural experts. *Int. Journal of Pattern Recognition and Artificial Intelligence*, 15(16):501–526, 2002.

[4] F. Bellifemine, A. Poggi, and G. Rimassa. Developing multi-agent systems with jade. In *Eventh International Workshop on Agent Theories, Architectures, and Languages (ATAL-2000)*, 2000.

[5] R. Carreira, J. M. Crato, D. Gonalves, and J. A. Jorge. Evaluating adaptive user profiles for news classification. In *IUI '04: Proceedings of the 9th international conference on Intelligent user interface*, pages 206–212, New York, NY, USA, 2004. ACM Press.

[6] G. Cherchi, A. Manconi, E. Vargiu, and D. Deledda. Text Categorization Using a Personalized, Adaptive, and Cooperative MultiAgent System. In *Workshop dagli Oggetti agli Agenti, Simulazione e Analisi Formale di Sistemi Complessi (WOA 2005)*, November 2005.

[7] K. Decker, K. Sycara, and M. Williamson. Middle-agents for the internet. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI 97)*, pages 578–583, 1997.

[8] O. Etzioni and D. Weld. Intelligent agents on the internet: fact, fiction and forecast. *IEEE Expert*, 10(4):44–49, 1995.

[9] S. J. Gething, M.J. Protein folding in the cell. *Nature*, 355:33–45, 1992.

[10] J. Giampapa, K. Sycara, A. Fath, A. Steinfeld, and D. Siewiorek. A multi-agent system for automatically resolving network interoperability problems. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1462–1463, 2004.

[11] D. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.

[12] F. Hartl. Secrets of a double-doughnut. *Nature*, 371:557–559, 1994.

[13] C. A. Knoblock, Y. Arens, and C.-N. Hsu. Cooperating agents for information retrieval. In *Proceedings of the Second International Conference on Cooperative Information Systems*, Toronto, Ontario, Canada, 1994. University of Toronto Press.

[14] R. Kohavi and F. Provost. Glossary of terms. *Special issue on applications of machine learning and the knowledge discovery process, Machine Learning*, 30(2/3):271–274, 1998.

[15] J. Kramer. Agent based personalized information retrieval, 1997.

[16] H. Lieberman. Letizia: An agent that assists web browsing. In C. S. Mellish, editor, *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 924–929, Montreal, Quebec, Canada, 1995. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA.

[17] H. Lieberman. Autonomous interface agents. In *Proceedings of the ACM Conference on Computers and Human Interface (CHI-97)*, pages 67–74, 1997.

[18] E. Lutz, H. Kleist-Retzow, and K. Hoernig. Mafiaan active mail-filter-agent for an intelligent document processing support. *ACM SIGOIS Bulletin*, 11(4):16–32, 1990.

[19] P. Maes. Agents that reduce work and information overload. *Communications of the ACM*, 37(7):31–40, 1994.

[20] H. Nwana. Software agents: An overview. *Knowledge Engineering Review*, 11(3):205–244, 1996.

[21] M. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[22] B. Rost and C. Sander. Prediction of protein secondary structure at better than 70% accuracy. *Journal Molecular Biology*, 232:584–599, 1993.

[23] B. Sheth and P. Maes. Evolving agents for personalized information filtering. In I. Press, editor, *9th Conference on Artificial Intelligence for Applications (CAIA-93)*, pages 345–352, 2003.

[24] M. Wooldridge and N. Jennings. *Intelligent Agents*, chapter Agent Theories, Architectures, and Languages: a Survey, pages 1–22. Berlin: Springer-Verlag, 1995.

[25] J. Yang, V. Honavar, L. Miller, and J. Wong. Intelligent mobile agents for information retrieval and knowledge discovery from distributed data and knowledge sources. In *IEEE Information Technology Conference*. Syracuse, NY, 1998.