A personal agent application for the semantic web *

Subhash Kumar†, Anugeetha Kunjithapatham, Mithun Sheshagiri,

Tim Finin, Anupam Joshi, Yun Peng, R. Scott Cost
Computer Science and Electrical Engineering
University of Maryland Baltimore County
Baltimore MD 21250 USA

Abstract

The Semantic Web is a vision to simplify and improve knowledge reuse on the Web. It is all set to alter the way humans benefit from the web from active interaction to somewhat passive utilization through the proliferation of software agents and in particular personal assistants that can better function and thrive on the Semantic Web than the conventional web. Agents can parse, understand and reason about information available on Semantic Web pages in an attempt to use it to meet users' needs. Such personal assistants will be driven by rules , axioms and the internal model or profile that the agents have inside them for the user. An intrinsic and important pre-requisite for a personal assistant or rather any agent is to manipulate information available on the Semantic Web in the form of ontologies, axioms, and rules written in various semantic markup languages. In this paper, a model architecture for such a personal assistant dealing with real-world semantic markup is described. The agent reasons with semantic markup written in DAML+OIL, using the Java Expert System Shell (JESS) as the reasoning engine. This software assistant views information providers on the Semantic Web as recommender agents that have a limited view of the user's preferences and provides a improved notion of personalization by collaborating with peer personal assistants (what are referred to as buddy agents) within communities that the user has identified as trusted parties to exchange information with. Collaboration is achieved through simple solicitation and recommendation of information with these buddy agents.

Introduction

The Web provides an infrastructure for people to access documents and services on the Internet. Today's methods require human intelligence and are still faithful to the original way in which the web was conceived and developed. The interface to services is represented in web pages written in natural language which must be understood and acted upon by a human. The Semantic Web is a vision to augment the current web with formalized knowledge and data that can be processed by computers thereby shifting the focus away from a human-centered interaction. Efforts are underway to define the format and meaning of the language of such a Semantic Web. The structured data on the Semantic Web could serve both humans and computers, while a part of it will be formalized knowledge and will be used only by machines. The EU-NSF strategic workshop report on the semantic web(EU-NSF 2001) identifies two key applications that are enabled by the semantic web:

- Applications for the organizations such as the development of ontology based marketplaces for business-to-business electronic commerce, or the bioinformatic knowledge grid in which biological data and knowledge bases are seamlessly interconnected and computing resources are available.
- Applications for the masses such as intelligent personal assistants gathering and filtering relevant information and composing it into a coherent picture with regard to the user's preferences (like being the travel assistant).

The Semantic Web will benefit the simple web user because it will support them in their day-to-day work, leisure and interaction with organization and because it will help them to enforce the degree of control they want (over their personal data, preferences, etc.)(EU-NSF 2001).

To realize the real power of such a Semantic Web, programs that collect Web content from diverse sources, process the information and exchange the results with other programs need to be created. The effectiveness of such software agents will increase exponentially as more machine-readable web content and automated services (including other agents) become available.

Software agents, characterized by their sense of autonomy, the agent's ability to control its own behavior to a certain degree and other social abilities such as the ability to exchange data with other agents, respon-

*This research was supported in part by DARPA contract F30602-97-1-0215.

†Current address: IBM T.J.Watson Research Center, 19 Skyline Drive, Hawthorne NY 10532

siveness to the environment, and pro-activeness are expected to perform roles on the Semantic Web similar to what an average user performed on the conventional web. For example, agents could help humans to cope with supposed information overload and to assist users in performing repetitive, common tasks(Hoyle & Lueg 1997).

A personal Software Assistant equipped with a model of its user's preferences operating on the Semantic Web can address a wide range of activities that it can help to automate. This could range from identifying content on the web useful to the user(from recommender agents) to managing the user's calendar. The means by which the assistant comes up with a model for the user is an area of vast research with proposed techniques ranging from explicit stating of preferences to complete implicit learning techniques and other hybrid approaches.

Personalization techniques have been greatly explored for tackling the issue of information overload and for targeting appropriate information or products to the end user on e-commerce sites. Such systems are driven by user profiling techniques that track user preferences, thereby identifying the appropriate content for the user. Personal agents offer a means for bringing the notion of personalization to the user's side with the ability to identify data directly from the Semantic Web and from other agents operating over the Semantic Web based on its internal model for the user.

An important pre-requisite for such (personal) agents operating over the Semantic Web is the ability to process and manipulate the semantic markup, maintain an internal brain of knowledge and draw inferences from the accumulated knowledge. The principle of "inference" is to be able to derive new data from data that is already known. Inference is one of the driving principles of the Semantic Web, because it will allow the creation of software applications that derive a use from the Semantic Web data.

In this work, a Personal Agent architecture for the Semantic Web is described and its operation over the Semantic Web interacting with recommender agents and peer Personal Agents dealing with real world semantic markup is explored. The Personal Agent and the other agents in the multi-agent system are implemented using the Java Agent Development Environment(JADE)(Fabio Bellifemine 1999). JADE is a software framework to develop agent applications in compliance with the FIPA specifications. The Personal Agent uses the Java Expert System Shell(JESS) for reasoning over the Semantic Web knowledge. JESS (Friedman-Hill 2002) is a rule engine and scripting environment written entirely in Sun's Java language . The application of the Personal Agent is discussed in conjunction with the 'Xtalks' Semantic Web portal. The Xtalks (Cost *et al.* 2002) system is a real-world, fielded application on the Semantic Web which supports user and agent interaction in the domain of talk discovery where talk announcements are marked up using DAML. Scenarios involving collaboration between peer Personal Agents or 'buddy' agents are envisioned and described. The Personal Agent model allows a de-centralized, distributed, peer-peer type of an architecture in the place of conventional web-based information providers and recommender systems that act like centralized systems for disseminating information.

Subsequent sections describe the notion of Personal Agents and Semantic Web in more detail, describe the functioning of the Personal Agent and the other agents in the multi-agent system, provide key implementation insights regarding the DAML reasoner using JESS. The final section summarizes the work and touches on future work ahead in this space.

Background and Related Work

What is a Personal Assistant ?

A Personal Assistant(PA) is a software agent that acts semi-autonomously for and on behalf of a user, modelling the interests of the user and providing services to the user or other users and PAs as and when required. It is unobtrusive but ready when needed and rich in knowledge about the users and their areas of work (FIPA d). This is the generalized notion of a Personal Agent from the agents standards body, Foundation for Intelligent Physical Agents (FIPA) (FIPA c).

The functions of a Personal Agent can be as varied as carrying out one or more of the following activities: Managing a user's diaries, filtering and sorting email, managing a user's desktop environment, managing a user's activities, plans and tasks, locating and delivering multimedia information, recommending entertainment, purchasing desired items, and, planning travel. The reference architecture of such a Personal Agent as described by FIPA is shown in Figure 1

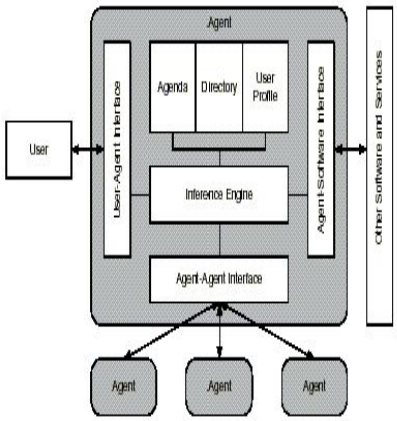

Figure 1: **FIPA Personal Assistant Reference Model**

Personal Assistant Applications

Learning Personal Agents have been used for the information filtering from the WWW (Lang 1995), (Armstrong *et al.* 1995), (Pazzani, Nguyen, & Mantik 1995). In case of the WebWatcher (Armstrong *et al.* 1995) and the agent described in (Pazzani, Nguyen, & Mantik 1995) the agent tries to find an "interesting" link in a Web Page that has already been pre-selected by a user.Similarly in News-weeder the user is subscribed to news-groups which are of interest to the user and have a large proportion of relevant articles.

In (Pannu & Sycara 1996) the authors investigate how a Personal Agent could be structured to acquire a user profile, which enables it to distinguish between relevant and irrelevant documents in text form on the WWW. This user profile is then used to accomplish the task of notifying users about conference announcements and requests for proposals that match their research interests. WebMate (Chen & Sycara 1998) is a personal software agent that accompanies a user when he browses and searches and provides intelligent help. It learns user interests incrementally and automatically provides documents that match the user interests

DAML and the Semantic Web

EXtensible Markup Language (XML) and the Resource Description Framework (RDF) form the underlying basis for representing semantic content. XML allows users to add arbitrary structure to their documents but says nothing about what the structures mean. Meaning is expressed by RDF, which encodes it in sets of triples, each triple being rather like the subject, verb and object of an elementary sentence. The third basic component of the Semantic Web comprises collections of information called ontologies. In philosophy, an ontology is a theory about the nature of existence, of what types of things exist. An ontology, in the context of the Semantic Web, is a document or file that formally defines the relations among terms (Tim Berners-Lee & Lassila 2001). The different layers of the semantic web as adapted from (Berners-Lee 2000) are shown in Figure 2

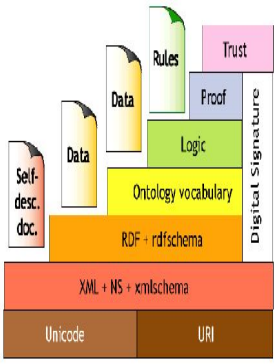

Figure 2: **Layers on the Semantic Web**

The DAML language is an extension to XML and the Resource Description Framework (RDF). The language provides a rich set of constructs with which to create ontologies and to markup information so that it is machine readable and understandable. It leverages and extends the expressability of RDF and RDF-Schema (RDFS) (Staab *et al.* 2000).

Agents for the Semantic Web

The RETSINA (Reusable Environment for Task-Structured Intelligent Networked Agents) Calendar Agent(RCAL) (Payne, Singh, & Sycara 2002), works symbiotically with Microsoft's Outlook 2000 and the Semantic Web. It can parse and reason about schedules, such as conference programs or recurring appointments that are marked up on the Semantic Web. RCAL can import and store schedules within Outlook 2000 and refer to these events to check if they have been updated, or to see if the user is free at a given time slot.

ITTalks (Cost *et al.* 2002) is a web portal offering access to information about talks, seminars and colloquia related to information technology (IT). It is organized around domains, which typically represent event hosting organizations such as universities, research laboratories or professional groups, and which are represented by independent web sites. ITTalks utilizes DAML for its knowledge base representation, reasoning, and agent communication. With information denoted in a semantically machine-understandable format like DAML, the computer can deduce additional information, a task which is difficult in a traditional database system.

The agent system described in this work derives and complements the work described in (Cost *et al.* 2002). While (Cost *et al.* 2002) describes the notion of Semantic Web portals and agents improving the utility of each other, this work describes a Personal Agent architecture in the context of Semantic Web portals and its notion of being a semantic recommender residing at the user side and performing tasks on the user's behalf including collaboration with peer Personal Agents.

Rule Based Systems for Agents

(Terveen & Murray 1996) describes an end-user programming system that makes it easy for users to state rules for their agents to follow. The system automatically determines conflicts between rules and guides users in resolving the conflicts. The authors propose that much of the knowledge that such an agent needs can be expressed as rules of the form when these conditions are true, take these actions. They represent rules using CLASSIC, a description logic which permits users to create structured descriptions of sets of objects (known as concepts) and individual objects.

DAMLJessKB (Kopena) facilitates reading DAML files, interpreting the information as per the DAML language, and allowing the user to query on that information. The software leverages the existing RDF API (SiRPAC) to read in the DAML file as a collection of RDF triples. It uses Jess (Java Expert System Shell)

as a forward chaining production system, which carries out the rules of the DAML language.

The basic flow of this library is as follows as stated in (Kopena):

- Read in Jess rules and facts representing the DAML language
- Have RDF API read in the DAML file and create SVO triples
- Take triples and assert into Jess' rete network in VSO form, with some slight escaping of literals and translation
- Have Jess apply the rules of the language to the data
- Apply the agent's rules, queries, etc
- Serialize relevant facts back to DAML

System Design

The design of the Personal Agent, which is the focus of this work has been done in a highly modularized fashion with the key insight being that the functionality of the Personal Agent should be easily augmentable to perform activities beyond the scope of this work, for example, travel arrangement, meeting scheduling etc., The Personal Agent provides the basic infrastructure for manipulating the user's schedule, an internal representation of the agent's knowledge in its so-called 'brain' and reasoning with information in its brain obtained from sources like the semantic web, peer Personal Agents and other forms of recommender agents. The utility of such a Personal Agent is in making the knowledge available through such sources of direct value to the end-user.

To demonstrate the utility and working of such a Personal Agent, complex scenarios involving recommender agents from the semantic web and peer Personal Agents have been designed. In the current model of the web, various types of recommender systems are prevalent that recommend different things ranging from research papers (Middleton) to TV content (Kurapati *et al.*). These systems are limited by the amount of information that the user provides voluntarily. It would be reasonable to assume that an user would not want to divulge his complete set of preferences in a particular domain for want of privacy and other security considerations such as the amount of trust the user places on such a recommender system.

One good solution to this problem would be to introduce the notion of a Personal Agent that resides at the side of the user, is trust-worthy and has a more complete model of the user's preferences in a particular domain making it much more capable of delivering the correct recommendations to its user. The model here would be that a third-party recommender system is not aware of specific user preferences but works with a more general model with another level of filtering being performed by the user's personal agent. A simple example of such a situation would be a talk-recommender system like "Xtalks" (ITT) being aware of the fact that the user is very interested in talks in the area of wireless computing but not the fact that the user never likes to attend talks by Mr. Foo Bar on the subject.

Also, conventional web-based information providers and recommender systems act like centralized systems disseminating information. The Personal Agent model allows a de-centralized, distributed, peer-peer type of an architecture. For instance, a system like 'Xtalks' would recommend talk announcements only to registered users. But a peer-peer multi-agent model would provide capabilities for even unregistered entities to receive the information.

The multi-agent system designed and implemented to demonstrate these ideas consist of the following agents - the user's personal agents, recommender agents like the 'Xtalks' Agent and information agents like the 'Mapquest' Agent.

The Xtalks Agent

This agent acts as a commercial third-party recommender agent that recommends talk announcements to registered users. Users register their Personal Agents with the Xtalks agent to have them receive talk announcements. Also, users can express their interests, schedule and location constraints through a DAML profile. The Xtalks agent monitors the Xtalks (ITT) system for new talk announcements. When new talk announcements get added to the system, the Xtalks agent informs registered Personal Agents about the talk announcement. This step is preceded by interaction between the Xtalks and the Mapquest agent to verify that the talk location is accessible from the home location of the user.

The Mapquest agent

The 'Mapquest' agent built around the Mapquest (Map) system provides information about distance, driving time and driving directions between two addresses. The required information is scraped from the Mapquest (Map) system, massaged into a form suitable for inter-agent communication and sent to the requesting agents. The Mapquest agent exposes its service using the FIPA Request Interaction Protocol(FIPA e). In the future, one could assume that web-pages from Mapquest (Map) are augmented with semantic markup or the information being available as a web-servicee thereby eliminating the need for an intermediary agent to convert web-page content into agent-friendly representations.

The User's Personal Agent

The central entity of the system is the user's Personal Agent which reasons with all the knowledge input from different sources and arrives at meaningful conclusions on behalf of the user. The Personal Agent is equipped with a 'brain' that essentially stores various types of information in the form of triples and a rule-based reasoning engine that manipulates the information in

the brain to draw meaningful conclusions. In addition to interacting with recommender systems like Xtalks, the personal agent also collaborates with peer Personal Agents both recommending and receiving recommendations about talk announcements and arriving at a conclusion based on the following premises.

- User's interest in the talk announcement topics
- User's schedule constraints (both in terms of availability and feasibility to attend the talk)
- Decisions of other Personal Agents the user would like his Personal Agent to interact with in arriving at a decision

Each of these pre-conditions are examined in more detail.

User's interest in the talk: The user agent is assumed to have a good model of the user's preferences for the particular domain, in this case, topics in Computer Science. It is available to the Personal Agent through an explicit DAML profile in which the user's interests are represented as topics in the ACM topic hierarchy. The talk announcements are also available as DAML URIs with the topics marked up from the ACMtopic hierarchy. Rules specifying the following statements are loaded in the 'brain' which help the agent in determining the interest of the user in that talk

- If the user is interested in a particular topic in the hierarchy, he is also interested in all the sub-topics of that topic
- If the user is interested in a particular topic and the topic of the talk happens to be the same, then the user is interested in the talk.

To determine all the sub-topics of a topic in the ACM hierarchy, a rule of the following form in inserted in the reasoner.

- The sub-topics of a sub-topic of a (parent) topic are also sub-topics of the (parent) topic

User's schedule constraints: The following conditions are checked to determine the feasibility of scheduling the talk on the user's calendar.

- Whether the user has an empty slot in his calendar for the period of that talk
- Whether the talk location is reachable in available time from the location of the previous appointment of the user
- Whether the next appointment of the user (after the talk) is reachable from the location of the talk in available time.

The time calculations are computed through interaction with the Mapquest Agent.

Peer-Personal Agents Interaction The notion of interaction between peer Personal Agents draws its roots from the concepts of instant-messaging and its popularity in helping to form online communities. It often occurs in day-to-day life that a users' decisions are influenced by the decisions of their friends/buddies. This agent interaction scenario tries to mimic the real world phenomena of forming buddy lists and engaging in group messaging. The user specifies the set of so called 'buddy-agents' which represent the group of Personal Agents of the user's buddies.

As in any instant-messaging/buddy-list application, the formation of buddy-list is preceded by a set of subscription and reply messages that establish the identity of the buddies with each other. The user could specify his list of buddies through publicly accessible DAML-URIs that specify agent details such as names and transport addresses. A Personal Agent can locate and subscribe to peer Personal Agents through the DAML URIs thereby expressing their willingness to both recommend and accept talk recommendations from the peer agents.

Thereafter, Personal Agents can exchange recommendations and decisions among themselves. To ensure tractability of the system, the following agent responses are identified as the set of all possible responses that an agent could send in reply to a request from another Personal Agent regarding a decision to attend a particular talk.

Resp. Value	Intended Meaning
0	Talk does not match interest.
1	Talk matches interest but schedule Conflict.
2	Talk matches interest and passes Buddy Recommendation Test(*).
3	User confirmation obtained about his willingness to attend talk.

Table 1: Buddy Agent Interaction Constants

(*) The buddy recommendation test succeeds if average score received from buddy agents exceeds a threshold. In this case the threshold could be the median value of 1.5.

There is a possibility of an occurrence of a dead-lock. A simple scheme employed to avoid deadlocks is, whenever an agent receives a query about a talk that it itself is waiting for replies from buddy-agents, then it immediately returns a special value of 1.5 (middle of all levels). This might also be used when the agent receiving the recommendation is not aware of the talk that is being recommended and hence does not really have a decision. Note that this request for a decision could itself act as a talk recommendation when it happens that the receiving agent is not aware of the talk.

In addition to multi-agent interactions to arrive at a decision in scheduling a talk, the Personal Agent can also be loaded with special rules to over-ride the default behavior. Such rules could b of the following form

- If the talk is on a specific topic, schedule the talk irrespective of other considerations.

- If the speaker of the talk is a specific person, schedule the talk irrespective of other considerations.
- If a specific buddy of mine decides to attend the talk, schedule the talk if the talk is of interest to me, irrespective of what my other buddies decide, provided there are no schedule conflicts.

These are merely examples and are in no way an exhaustive set of rules that can possibly be derived in such a situation. These examples are just to show the richness in adopting such an approach.

Another capability of the user's Personal Agent is to adopt a pro-active behavior in addition to all the reactive kinds of behaviors in responding to talk recommendations. Such a recommender system is limited by the initial set of filtering criteria specified by the user when submitting a profile. For example, if the user had specified his home location to be Baltimore, a system like 'Xtalks' would be recommending talks to the user in and around Baltimore oblivious of a user's travel plans. In such situations, the Personal Agent adopts pro-active approach by monitoring the user's calendar for the following conditions.

- The user has an appointment scheduled on the calendar that is away from the default home location, indicating that the user is travelling.
- Cancellation of a particular appointment from the user's calendar indicating that a potentially interesting talk could have not been scheduled because of earlier schedule conflicts.

In the first case, the Personal Agent would have never received a talk recommendation for a talk happening away from the user's home location. In such a case, the Personal Agent queries the 'Xtalks' system for talks that could be happening at the new location. In the second case, presence of an appointment on the calendar indicates the possibility of a previous talk announcement not getting scheduled because of conflicts on the user's calendar. Given the model of the Personal Agent, the decision of not scheduling such a talk because of schedule conflicts resides in its brain. So, the Personal Agent could request for information from the 'Xtalks' agent for that particular talk. Alternately, the Personal Agent could adopt a lazy approach and request for talks during the period of the cancelled appointment and try to schedule it on the user's calendar. Such functionalities are modelled as additional plug-ins that augment the capabilities of the personal agent.

Key Implementation Insights

All the agents are implemented using the Java Agent Development Environment (JADE). JADE (Fabio Bellifemine 1999) is a software framework to develop agent applications in compliance with the FIPA specifications for interoperable intelligent multi-agent systems. JADE can be considered an agent middle-ware that implements an Agent Platform and a development framework. Inter-Agent communication is through the FIPA-ACL (FIPA a) which specifies a standard message language by setting out the encoding, semantics and pragmatics of the messages. Each agent resides in its own individual platform with agent communication enabled using the Internet Inter-ORB Protocol (IIOP) and communicates with agents in other platform using the IIOP transport mechanism provided by JADE.

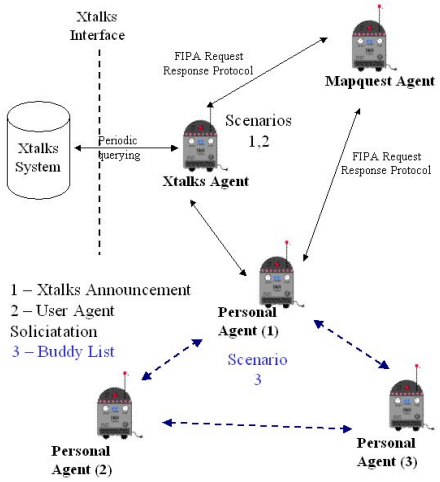

Figure 3: **Multi-Agent Scenario and Interactions**

The Xtalks agent also exposes a query interface through an ontology, which defines the set of supported queries that an agent could pose to the Xtalks agent. Different Personal Agents exchange talk recommendations as defined by the buddy list ontology. Messages could be either talk recommendations, which are also used by the Xtalks agent or a 'query-if' in FIPA-ACL language act, asking the other agent whether (it believes that) a given proposition is true (FIPA b). The sending agent is requesting the receiver to inform it of the truth of the proposition. In this case, the proposition being the sending agent's belief regarding the interest of the other agent (user) in the talk.

A Personal Agent can acquire gather information from its peer Personal Agents through by querying them. The querying is done using a general peer to peer querying mechanism. The DAML Query Language (DQL)(dql a) is used as query description language. Queries are framed using the an ontology(dql b) developed for DQL and these query descriptions are packed into FIPA ACL messages. The interaction between the agents is modeled after the FIPA Request Interaction Protocol. The result of the query is sent as a FIPA INFORM message.

The Personal Agent Architecture

The design of the Personal Agent was driven by the follwoing ideas

- The functionalities should be independent
- The functionality of the Personal Agent should be

easily extensible, with new capabilities being added seamlessly

- The whole Personal Agent can be easily packaged and provided as an easily configurable utility.

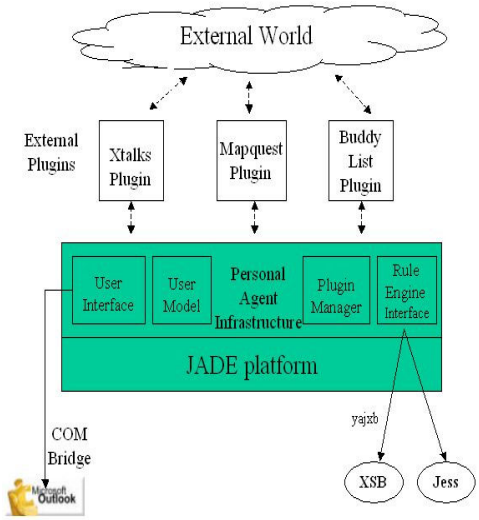

Figure 4: **Personal Agent Architecture**

DAMLJessKB is used to read in DAML file and interpret its contents. The original version of DAMLJessKB represents the interpreted facts in the (PropertyValue subclass man person) format. In our system, an alternative representation of the facts has been introduced, for the reasons as given below. Ordered facts like (property subclass man person) are actually equivalent to an unordered fact like (property (_data subclass man person)), i.e., the slot data is all stored in one multi-slot. Matching data in multi-slots is less efficient than matching data in normal slots because, there exists an extra indirection through an array, and also, the Rete network will contain an extra pattern-network test for the length of the multi-slot. Since the lengths would always be 3, this test would be unnecessary.

The choice of Jess as the reasoning engine for the Personal Agent is driven by these design decisions. Jess, with the whole engine being written in Java, integrates easily with the Personal Agent and is easy to be packaged and deployed. An ideal Personal Agent architecture could resemble something close to the one represented in Figure 4. The key idea with this architecture is that user could download and add components developed by third-party vendors that the user trusts which can augment the functionality of the Personal Agent. For example, a Personal Agent with components designed for interaction with a 'Xtalks' agent could be augmented tomorrow with an 'Amazon' component that helps it to interact with an 'Amazon' agent to receive different kinds of recommendations like books, movies etc., The most rudimentary way of envisioning a

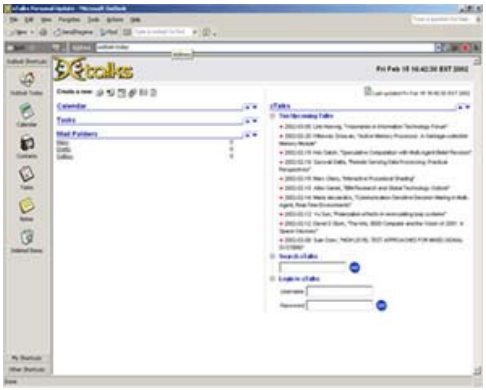

Figure 5: **Outlook Today Extensions**

plug-in would be as a JADE behavior that the agent can pick-up and execute at runtime. In such a situation, a plug-in manager would simply search a pre-configured directory for new behaviors and add it to the list of behaviors currently executed by the Personal Agent.

The Personal Agent interacts with the user through Microsoft Outlook. The user's calendar is assumed to be stably and consistently stored on Microsoft Outlook. Also, any appointment that the Personal Agent schedules for the user is manifested on the Microsoft Outlook calendar for the user. The java-based agent interacts with Microsoft Outlook using Bridge2Java (Bri). Bridge2Java is a tool that allows Java programs to communicate with ActiveX objects. It allows easy integration of ActiveX objects into a Java Environment. Using the Java Native Interface and COM technology, Bridge2Java allows an ActiveX object to be treated just like a Java object.

Another useful user-interface extension explored and developed with the Personal Agent through Microsoft Outlook is the 'Outlook Today Extensions for Xtalks' (Kumar). This downloadable component offers an interface as shown in Figure 5. The view displaying both the list of most recent talks announcements available from Xtalks and those scheduled by the Personal Agent in the user's calendar serves to show the filtering done by the Personal Agent through its internal reasoning process.

Conclusion

The Semantic Web is here for good and is here to stay. Various technologies including agents, ontologies and information management are currently being developed to make the Semantic Web a reality. We have presented an example architecture for a Personal Agent working on the Semantic Web collaborating with recommender agents and peer Personal Agents. The Personal Agent operates using a rule-driven brain operating on Semantic Web data and the user profile. The multi-agent system interaction through DAML naturally extends the language for knowledge representation to being the language for communication. The system though far from

being a real world prototype, demonstrates the notion for such future systems that operate on the Semantic Web and integrate well with day-to-day software that the user utilizes.

Typically decision making in agents is based on a set of general rules and the user's profile and preference. In the real world a user's profile and preferences keeps changing although mostly at a gradual rate. An agent should therefore monitor and model this change to ensure high quality decision making. To this effect, a feedback mechanism is being designed that captures the user's actions as feedback to refine the user's existing model. We cannot expect the user to provide explicit feedback all the time. Alternate sources of indirect feedback can be obtained from the user's use of his/her machine. For example, browser bookmarks and cookies can be used to obtain useful information about users interests.

Life span of the agent also forms an important criterion in the design of personal agents. Building and refining the user model typically takes a lot of time. It is therefore essential to keep the agent running as long as possible to make good use of its knowledge of the user in decision making. An implication of building life-long agents is the problem of managing a large number of facts that are gathered continously by the agent. It is also desirable to have the agent running all the time.

These requirement bring up several issues:

- Formalisms that identify facts that need to be retained and for how long.
- Clever compression mechanisms that store facts that are not of immediate need but nonetheless might be required.
- Mechanisms that recover an agent's state and re-start the agent if the agent terminates due to software bugs or machine failure.

The notion of intelligent Personal Agents never really took-off in the context of the web as it stands today. But the emergence of the Semantic Web promises to change that, for, the concepts of intelligent agents and the Semantic Web are a synergy. Effective and private access to user's desires, preferences, and habits coupled with information garnered from the Semantic Web promises to offer potent personal assistants such as the one described in this paper, that are bound to make life simpler for their masters.

References

Armstrong, R.; Freitag, D.; Joachims, T.; and Mitchell, T. 1995. Webwatcher: A learning apprentice for the world wide web. In *AAAI Spring Symposium on Information Gathering*, 6–12.

Berners-Lee, T. 2000. The semantic web vision.

Bridge2java http://www.alphaworks.ibm.com/tech/bridge2java.

Chen, L., and Sycara, K. 1998. WebMate: A personal agent for browsing and searching. In Sycara, K. P., and Wooldridge, M., eds., *Proceedings of the 2nd International Conference on Autonomous Agents (Agents'98)*, 132–139. New York: ACM Press.

Cost, R. S.; Finin, T.; Joshi, A.; Peng, Y.; Nicholas, C.; Chen, H.; Kagal, L.; Perich, F.; Zou, Y.; Tolia, S.; and Soboroff, I. 2002. Ittalks: A case study in the semantic web and daml. In *proceedings of the International Semantic Web Working Symposium*.

Daml query language http://www.daml.org/2002/08/dql/.

An ontology for representing daml queries http://gentoo.cs.umbc.edu/anu/dqlontology.daml.

EU-NSF. 2001. Research challenges and perspectives of the semantic web - report from the joint european commission and national science foundation strategic workshop on the semantic web.

Fabio Bellifemine, Agostino Poggi, G. R. 1999. Jade-a fipa-complient agent framework. In *Proceedings of PAAM'99*, 97–108.

FIPA. Fipa acl message structure specification.

FIPA. Fipa communicative act library specification.

FIPA. Fipa http://www.fipa.org/.

FIPA. Fipa personal assistant specification.

FIPA. Fipa request interaction protocol specification.

Friedman-Hill, E. J. 2002. Jess, the expert system shell for the java platform.

Hoyle, M., and Lueg, C. 1997. Open sesame: A look at personal assistants.

Xtalks http://www.ittalks.org/.

Kopena, J. Damljesskb http://plan.mcs.drexel.edu/projects/legorobots/design/software/damljesskb/.

Kumar, S. Outlook today extensions for xtalks http://daml.umbc.edu/download/.

Kurapati, K.; Gutta, S.; Schaffer, D.; Martino, J.; and Zimmerman, J. A multi-agent tv recommender.

Lang, K. 1995. NewsWeeder: learning to filter netnews. In *Proceedings of the 12th International Conference on Machine Learning*, 331–339. Morgan Kaufmann publishers Inc.: San Mateo, CA, USA.

Mapquest http://www.mapquest.com/.

Middleton, S. E. Exploiting synergy between ontologies and recommender systems.

Pannu, A., and Sycara, K. 1996. A learning personal agent for text filtering and notification.

Payne, T. R.; Singh, R.; and Sycara, K. 2002. Calendar agents on the semantic web.

Pazzani, M.; Nguyen, L.; and Mantik, S. 1995. Learning from hotlists and coldlists: towards a www information filtering and seeking agent.

Staab, S.; Erdmann, M.; adche, A. M.; and Decker, S. 2000. An extensible approach for modeling ontologies in rdf(s).

Terveen, L. G., and Murray, L. T. 1996. Helping users program their personal agents. In *Proceedings of ACM CHI 96 Conference on Human Factors in Computing Systems*, volume 1, 355–361.

Tim Berners-Lee, J. H., and Lassila, O. 2001. The Semantic Web. *Scientific American*.