Agents for Support of WAP-based Services

Mihhail Matskin and Runar Bell
Department of Computer and Information Science, Norwegian University of Science and Technology,N-7491 Trondheim, Norway
e-mails: {Mihhail.Matskin,Runar.Bell}@idi.ntnu.no

Key words: Intelligent agents, mobile communication services, WAP, WML

Abstract: We consider an application of agent-based methods to development of WAP-based services. Special attention is given to supporting off-line autonomous processing and personalizing WAP-based services. A developed Java-based WML-Agent system is used as an example of utilization of agent technology in WAP-based communications.

1. INTRODUCTION

WAP (Wireless Application Protocol) is a new rapidly growing area of mobile communications. Because of its promising as a standard the current tendency that more and more companies support WML–pages for their customers will be continued in future. This tendency is also a basis for fast growing number of available WAP-based services. At the moment most of efforts are directed to transferring information from HTML to WML-oriented view and this is a practical need because of without available information there is no basis for advanced services. The problem is that specific features of WAP-based communications often are not enough taken into account in new implemented services. The WAP/WML is a new field and there should be a new original way of utilization of its features. Of course there is a similarity between utilization of HTML-pages and WML-pages. However the traditional solutions for WWW cannot be cloned for WAP/WML directly. The basic differences as we see them are as follows:

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35522-1_37

H. R. van As (ed.), *Telecommunication Network Intelligence*

- The users of WAP will not surf through pages not knowing what they are looking for but rather they will surf in order to cover their needs
- It may also be assumed that WAP users will not want to surf the Internet at all but just have a direct access to a specified information

The last assumption is, maybe, too limited from our point of view because of anyway users may need search of desired information. However a definitely important requirement from the WAP perspective is a high level of personalization in the WAP-based services. We see the following reasons for that:

- The first reason is a high cost of mobile communications
- The second reason is that most of WAP users assumed to be non-experienced Internet users (which is, maybe, too limited view because of knowledge of Internet is increased very fast)
- The third reason is a limited expressive capability of WAP devices

It is possible that the last limitation will be relaxed in the future but at the moment we should take it into consideration. A consequence of the first reason is that the mobile user connection time to the network for requesting information should be minimized and that as much as possible work for surfing should be done off-line. Another consequence from the first reason is requirement to precision of delivered information – scrolling of a huge amount of information usually given by WWW search engines may require a lot of time (and money as a duration of mobile connection can be quite long).

In order to be able to get information from WWW the user can make use of several methods:

- It is possible to make search using some of the Internet search engines (AltaVista, HotBot, Infoseek etc.)
- It is possible to browse through one of many catalogue listings (Yahoo etc.)
- It is possible to guess a hyperlink to a place where the user may find the information s/he is looking for or just go browsing through hyperlinks

These ways of obtaining information can be applied for WAP-based services only partly because of the above-mentioned reasons. However in the case of WAP the user may better know what s/he is looking for and where to find it and this can be a basis for efficient solutions.

Some support for customers of WAP-devices is provided by operators of mobile networks who specify available services on special homepages. Definitely this branch will be developed in future. However our goal is to

allow user to choose an information s/he is interested in, to make the chosen information available, updateable and adaptable and to make such service flexible and dynamic.

In order to reach the goal we apply an agent-based approach. In particular we would like to give the WAP user an agent-based tool for selection, personalization, finding and updating information to be accessed via WAP devices.

The rest of the paper is organized as follows. First we consider briefly WAP features and intelligent agent characteristics. Then we present our vision of utilization of agents in WAP-based communications. Next we describe a Java-based WML-Agent system which we develop for providing agent-based services in WAP-based communication. Then we consider some related works. In the conclusions we consider a summary of results and future works.

2. WAP – WIRELESS APPLICATION PROTOCOL

Here we consider very briefly only basic features of the WAP.

The basic idea of WAP [7,18] as a standard for wireless communications is to extend communication ability of wireless telephony. In other words cellular phones are no longer just phones but rather communication devices capable of running applications and capable to communicate with other devices and applications over a wireless network. The standard takes into account the limitations of wireless data network compare to wired networks (for instance, less bandwidth, more latency, less connection stability and less predictability) and limitations of mobile handsets compare to personal computers (for instance, small screen size, complicated text input, limited memory, slow CPU).

The WAP standard specifies end-to-end application protocol and an application environment based on a browser as two essential elements of a wireless communication. The application protocol is a layered communication protocol which is embedded in each WAP-enabled user component. The network side includes a server component which implements the other end of the protocol that is capable of communicating with any WAP component. The server component often takes the role of a gateway for routing requests from user component to an application server and the gateway can be physically located in a telecom network building a bridge between wireless and computer networks.

A "microbrowser" integrated into a handset allows to display information to a user and to accept requests from the user. The basis for information representation is WML – a tag-based document language which is an XML-based language. The basic unit of WML is a card which specifies a single user interaction. Navigation occurs between cards which are grouped into decks. A deck is the top-most element of a WML document and it presents possible alternatives for user interactions.

It is assumed that when the user logs on to a WAP gateway s/he supplies the address of a starting page (WML deck), a portal, and then performs interactions according to the deck description. It is desirable for user to be able to find all the information s/he would ever need linked directly or with as few links as possible from the starting deck. It is also preferable that favorite decks/cards which the user visits more often can be reached through a fewer set of links than decks/cards which are used once for a while.

3. INTELLIGENT AGENTS

We apply an agent-based approach to support of WAP-based communication. Here we give only a brief description of the basic notions related to agents (for more detailed overview of basic agent properties we refer, for example, to [1,2,3,12,20]).

"Agents" is a new approach to building software in a distributed decentralized environment. In spite of different views to agents in different research societies there increases an agreement of what are basic agent characteristics. Here we summarize some essential characteristics by adopting a definition from [20] as the most representative one from our point of view. According to that we consider an agent as a software entity which represents somebody (a human or another agent) in computer environment and enjoys the properties of autonomy, pro-activity, reactivity and social ability (they are referred as weak agent properties in [20]). By autonomy an ability to operate without intervention from outside is assumed. By pro-activity a goal-oriented behavior and ability to take initiative is understood. Reactivity assumes perception and affecting the environment and social ability assumes communication with other agents in order to achieve own goals.

Among other possible agent characteristics we should also point out mobility [20] and we consider mobility as a very useful agent feature. However, we treat mobility rather as an optimization of agent communication than a

necessary agent property. This is why agent mobility was left outside the scope of the paper.

In this work we mostly employ autonomy, pro-activity and reactivity of agents while for supporting social ability we refer to [8,9] where architecture for agents cooperative work support is developed. The above-mentioned agent characteristics are very suitable for achieving our goals because of they allow us to support personalization of behavior, off-line activity and environment monitoring as specific requirements of WAP-based communication which we referred in the Section 1.

4. WHAT DO AGENTS HAVE TO DO IN WAP-BASED APPLICATIONS?

As we mentioned in the Section 1, our concern is to provide user with a means for description of required services and for making such services available via WAP-device (WAP-phone) in a flexible way.

The way of how WAP-services are now supported by service providers is proposing some predefined set of services and making these services available via WAP-handset menu. A typical set of services includes weather forecast, news, banking and finances, newspapers, travel information, currency exchange rates, access to catalogues etc. These services are quite reasonable selection but they can't satisfy all customers of course and, what is more important for us, they are not enough personalized. By personalization we understand not only supporting static profiles but also dynamic adaptation. How to personalize the set of services and select/add new services?

Let us consider some possible scenarios of user WAP-based communication.

The ideal starting point for user communication with WAP-based device is a WML deck which always reflects what the user wants to see at that precise moment. In this way it should be as easy as possible to retrieve the information without typing advanced queries or long hypertext links in order to get to a certain destination. For example, if the user likes to read the international news or weather forecast on his way to work in the morning, then the best way would be to get an overview of all the latest, and still not read, news from the favorite news agencies as either a starting card or as a card close to the starting card in the hierarchy. Of course, during the day user preferences or requirements for information could be changed to, for example, more business-oriented news in daytime or TV-programs in the

evening on the way home. Taking this into account the starting card should reflect such shift of focus of interest and it should allow to get access to preferable information as easy as possible. At the same time the previous focus of interests could be not important anymore and it should not correlate with the current focus.

Some another scenario may reflect a permanent need for awareness of important events and newly available information of great importance for the customer. As a typical example we can consider an information from stock exchange market, especially, awareness of changes in particular stocks quotas beyond a predefined threshold. Another scenario may require customer to search for some important information which can't be presented as an element in his card before hand (or can't be clearly defined and may require some filtering) and to notify him about its availability.

We consider only few different possible scenarios for a user. Each of these scenarios could be specialized for different customer. We think that it would be impossible to create a WAP portal which is accommodated for all the different needs a user might have. It also may be very hard and time consuming work to manually create lots of portals which are meant to be updated regularly and reflect different user needs. From the user point of view, the best way to reach the documents of interest as easy as possible could be make them available through own personalized portal. Since the users are interested in calibrating their portal not very often somebody is needed to keep it constantly up to date. This is a job which is just perfect for a software agent.

Software agent can be an autonomous constantly running piece of software. At period when it is not necessary to do anything, it will hibernate but when a change in the source for portal or some event occurs, the agent awakes and updates the information on the WAP portal. When the user connects to personal portal s/he will get a pre-generated and up-to-date documents without waiting for completion of queries at servers around the world. As we refer to agent features in the Section 3 autonomy, reactivity and pro-activity should play a key role in implementing such agents. Pro-active behavior should rely on a customer profile presented in a system. In a more advanced case agents learning ability can be developed and monitoring of customer's actions may be a source for deciding about a preferred time for availability of particular information. In the same way a pro-active reasoning using available sources about customer (for example, from user profile, from electronic notebook etc) can be quite advanced with using logical rules and deductive methods.

5. A JAVA-BASED WML-AGENT SYSTEM

As an example of utilization of agent technology in WAP communication we developed a Java-based WML-Agent system.

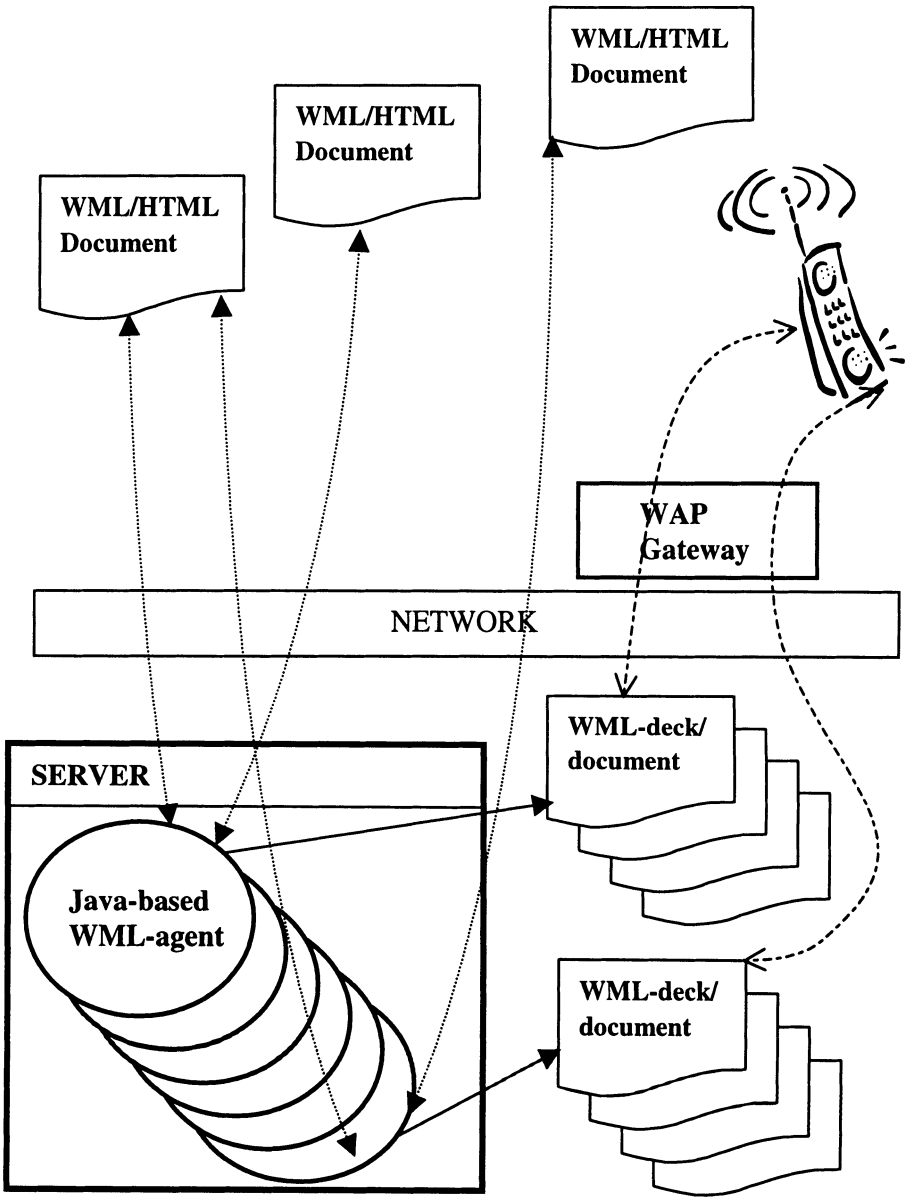

Figure 1: Java-based WML-Agents

The main idea behind Java-based WML-Agent is creating an intelligent agent which is

- highly adaptive to extracting interesting information from various WML/HTML-pages
- able to locate, collect and present information in a WAP device oriented view
- able to gather information from pages which are commonly updated without having customer to do that himself
- able to reorganize an order and priority of information to be presented to the customer
- able to employ a simple pro-active behavior
- able to process user requests off-line

The Java-based WML-Agent should inhabit the Internet, collect information from different sources in the Internet and provide WML-decks which the customer can access via WAP gateway as it shown in Figure 1 (this figure also shows that there can be many agents corresponding to different customers or profiles).

Each user should be able to design his or her starting page telling Java-based WML-Agent what information the user would like to have and, possibly, where to find it. Since most non-experienced customers probably may find this a hard and time consuming task, it should be possible use Java-based WML-Agent by a service provider in order to create certain starting pages targeted to large group of users. This could be, for example, a creation of profile with the target group being customers interested in sports and fashion in the age of 18 to 24 years. Some other profiles would cover target group of peoples interested, for instance, in technology, economy and politics. To combine these two options (different individual and group profiles) it should be possible to let the user create his own profile by combining several profiles together. In our case the user could, for example, indicate that s/he is interested in sports and technology but not in fashion, economy and politics.

In a general case of Java-based WML-Agent it should be possible for the customer to make registration (profile presentation) for a service both using computer-based and WAP-handset based interfaces as it shown in Figure 2 (however in the current version of Java-based WML-Agent only a computer-based interface for registration for a service is implemented).

The result of registration is a record in customer database and a generated personal WML-agent who may start running on a server of a service provider.

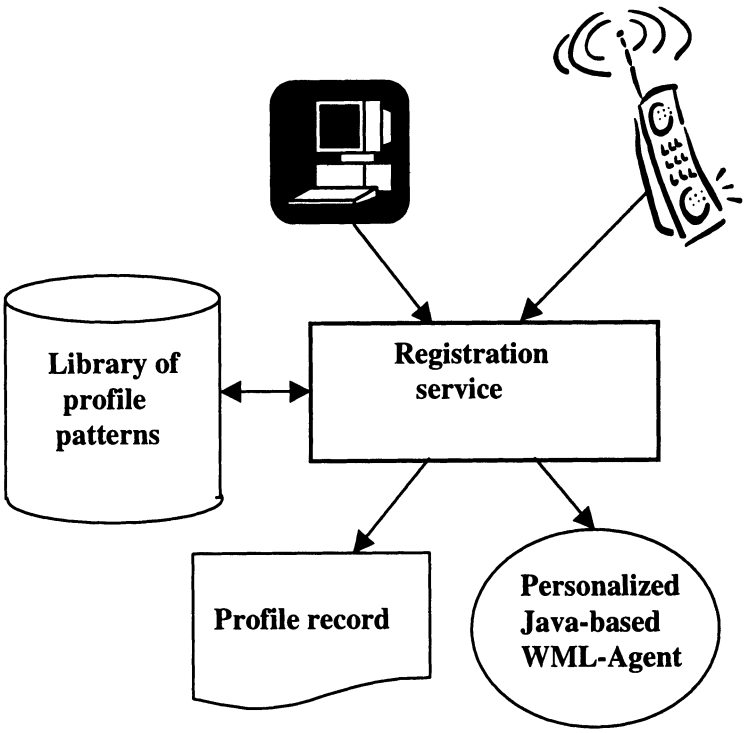

Figure 2. Registration service

As a summary of the scenario requirements from the Section 4 we can say that ability to dynamic shift of focus of required information and dynamic reorganizing the WML-decks (as a basic means for WAP-based interaction) are of a great interest. There are actually two parts of the problem:

- Description of customer's needs and keeping updated focus of customer's interests
- Organizing visualization of required information

These parts were main focus of the work for implementation of the Java-based WML-Agent system.

6. A BRIEF OVERVIEW OF THE JAVA-BASED WML-AGENT DESIGN

Here we briefly describe some basic features and functionality of the Java-based WML-Agent system.

The Java-based WML-Agent system may be divided into two parts:

- One which takes care of the Graphical User Interface (GUI) and
- One which lies behind the GUI and does all the processing

The GUI part provides means for creating, updating and customizing user cards and requirements and the processing part does search, adaptation and generation of WML decks and cards.

An example of GUI for Java-based WML-Agent is presented in Figure 3.

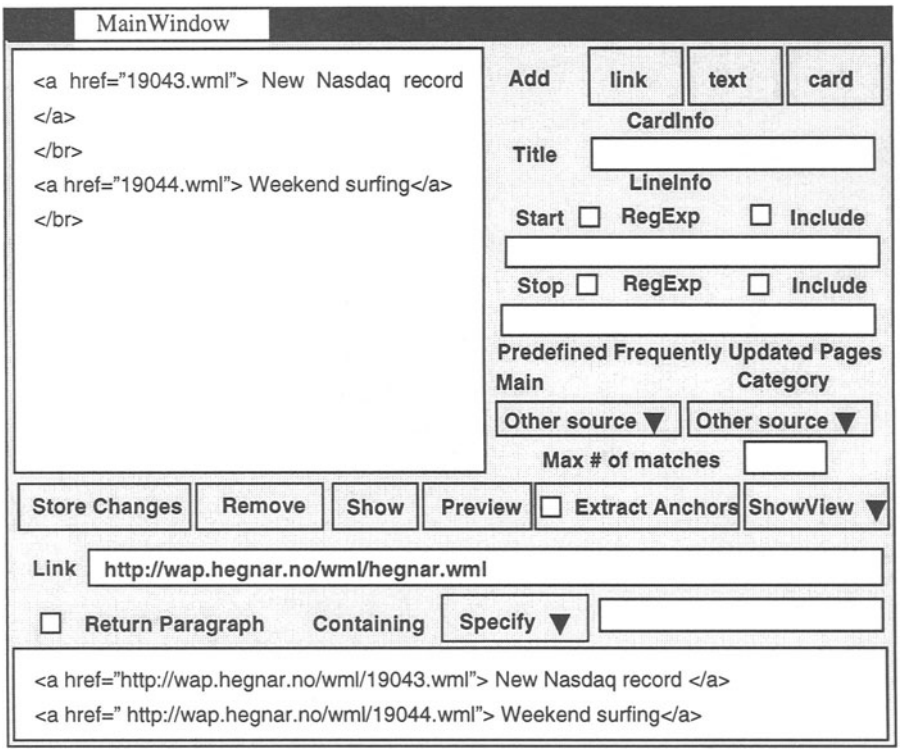

Figure 3. System GUI

Some examples of functionality provided by the GUI are as follows:

- Creating new cards, new fields in already existing cards, new links and text
- Visualizing control of input/output to/from the application
- Presentation and selection of a set of predefined resources. These resources may be easily updated and customized
- Displaying information about the current card
- Defining frequently updated pages. This allows to define WML-pages of special interest which should contain up-to-date information

- Defining limitation for the amount of deliverable information
- Defining filters for information filtering from predefined pages

The processing part of Java-based WML-Agent allows to display the required information according to customers needs, to process queries from the customer off-line and to present results of processing in the form of WML-cards.

A basic loop of an agent processing part can be very generally described as follows:

1. Receive an event from the customer, environment or other agent components
2. Process the event by event handler and call query processing, filtering or pro-activity handling components. All the above components use customer profile record.
3. After processing the event by query processing, filtering or pro-activity handling components generate or update corresponding WML-deck
4. If there are more events left in the queue then go to the first step, otherwise go to "hibernate" state

Events can be generated externally (for example, by clock or customer) or internally (for example, by pro-activity handling component).

Another information retrieval feature that is provided by the Java-based WML-Agent is information filtering. User can describe what particular kind of information from predefined WML-decks and HTML-pages (usually they are some pages provided by newspapers or news agencies) s/he is interested in. This can be described by a regular expression and the agent will filter updated information in the pages according to the expression. Such filtering allows more personalized and precise search that may decrease the amount of deliverable text. This is especially useful with a limited size of screen of a WAP-device.

The Java-based WML-agent system employs basic agent features like autonomy, reactivity and a simple pro-activity. In the current version we restricted pro-activity to ability to a simple reorganizing WML-decks according to results of processing by putting the hottest news on the top. However further development of this feature will be our prioritized work in a new Java-based WML-Agent version.

7. RELATED WORKS

To the best of our knowledge we didn't find yet works on intelligent agents for WAP-based services. However there are works on development of News Agents for the Internet and Web-pages. Some of them are as follows.

Reference.com [13] searches for any information in over 150,000 Usenet newsgroups and publicly accessible lists. The results are either emailed daily to the customer or available at personal WWW site.

RoboBear [14] performs automated searching, downloading and decoding the Usenet newsgroup articles. RoboBear is an application that must be downloaded to the user computer before it is of any use. Then it must be set up to search the newsgroup articles, using the news server's built-in search capabilities.

MyCNN (CNN Custom News) [15] is a site where the user is able to highly personalize his starting page at CNN. This includes having constantly updated weather forecast for cities of interest, stock quotas, sports, showbiz, world news etc.

Infoseek News [16] is a personalized news service that allows to express personal news interests and to create own customized news page.

My Yahoo! [11,19] delivers customized headline news, sports scores, stock quote, weather reports, web resources, personalized version of the Yahoo! directory and many other things. It allows users to find website determined by their interests and people with the similar interests using Automated Collaborative Filtering (ACF) technology in partnerships with Firefly Network, Inc. (for more information about ACF see, for example, [4]).

Paperboy [17] is a free news ticker which compiles a personal newspaper from daily newspapers based on topics. The result is a list of web links which can be delivered daily by email. Customer can also specify which articles s/he wants by giving the system some keywords. Complex queries are also possible with syntax similar to well-known search engines.

Newsweeder II [5] is a service that regularly searches for Web pages and Netnews articles that are interesting to the customer. It applies advanced machine learning techniques to pages and articles which the customer liked in the past.

NewT [6] is a USENET news filter that can be "trained" by giving it a series of examples that show in which kind of articles the user is interested in. From this the NewT agent can search all news articles trying to find other articles which are similar to those initially indicated by the user. When the agent presents the other articles that it has found the user gives feedback according to their applicability; thus the NewT agent can widen or restrict its searching next time.

All the above news services are very interesting and inspiring but they do not exploit pro-activity in the sense we considered it previously and, what is more important for us, they are not designed to be used in WAP-based communication. In other words they don't put enough attention to the special WAP communication features we mentioned in the Section 1.

Several mobile communication suppliers provide WAP start-portals for their customers. However some of them, at the moment, lack the ability to personalize the content of the portal while even those who allow describe personal WML-decks do not support their dynamical changing and pro-active updating.

8. CONCLUSIONS AND FUTURE WORKS

The main goal of this work is to demonstrate advantages of application of agent-based methods to WAP-based communication support. The general benefits of agents in WAP supported communication services are based on their ability to

- off-line processing via autonomous behavior
- personalization of services via pro-activity and adaptation
- relaxation of limitations of current WAP-based devices via precision of delivered information

Another important advantage of the agent-based approach is possible decentralization of the agent-based services and, as a result, a better scalability of service providing in a case of very large number of subscribers.

In this paper we don't consider the issues related to the costs of providing the agent-based services. However we understand that these issues become important in developing future versions of the system.

With a new technology such as WAP their are expected a lot of expansions in the near future both regarding usability due to better equipped terminals

and higher transfer rate due to improvements in the telecommunication infrastructure. In addition to these general improvements we see the following extensions to our Java-based WML-Agent as our next works:

- Taking into account a context (for example, time) of request. This may increase pro-activity of the services and it may not require special support from a mobile network
- Push-services. The WAP specification is going to define a "push" mechanism which will allow any Internet server to send information to the user.

Another future work we see in increasing a social ability of Java-based WML-Agents by allowing them to communicate and negotiate to each other for better satisfaction of the customer requests. In this part we refer to our work on co-operative work support by Agora platform [8,9,10].

ACKNOWLEDGEMENTS

This work is partially supported by the Norwegian Research Foundation in the framework of the Distributed Information Technology Systems (DITS) program and the ElComAg project.

REFERENCES

[1] J. .M. Bradshaw (Editor), *Software Agents*, AAAI Press/The MIT Press: Menlo Park, CA. 1997, p. 291-316.

[2] M. N. Huhns, M. P. Singh, L. Gasser (Editors), *Readings in Agents,* Morgan Kaufmann Publishers, 1998, 523 p.

[3] N. R. Jennings and M. J. Wooldridge (Eds.), *Agent Technology: Foundations, Applications and Markets*. Springer Verlag, 1998, 323 p.

[4] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon and J. Riedl, *GroupLens: applying collaborative filtering to Usenet news*. Communications of the ACM, 40(3), 1997, pp. 77-87

[5] K. Lang, *NewsWeeder: Learning to Filter Netnews*. Armand Prieditis and Stuart Russell (eds.). Machine Learning, Proceedings of the Twelfth International Conference on Machine Learning, Tahoe City, California, July 9-12, 1995, San Francisco, CA : Morgan Kaufmann Publishers, 1995.

[6] P. Maes, *Agents that Reduce Work and Information overload.* Communications of the ACM, 37(7), 1994, pp. 31-40

[7] S. Mann, *Programming Applications with the Wireless Application Protocol: The Complete Developer's Guide*. John Wiley & Sons, 2000, 256 p.

[8] M. Matskin, M. Divitini and S. A. Petersen. *An Architecture for Multi-Agent Support in a Distributed Information Technology Application.* International Workshop on Intelligent Agents in Information and Process Management on the 22nd German Annual Conference

on Artificial Intelligence in Bremen (KI-98), TZI-Bericht Nr. 9, Germany, September 15-17, 1998, pp.47-58.

[9] M. Matskin, *Multi-Agent Support for Modeling Co-operative Work.* In: T. Yongchareon, F. A. Aagesen, V. Wuwongse (Eds.) Intelligence in Networks. The Fifth International Conference SMARTNET'99, 22-26 November 1999, Thailand, Kluwer Academic Publishers, 1999, pp. 419-432.

[10] M. Matskin, O. J. Kirkeluten, S. B. Krossnes and Øystein Sæle. *Agora: A Multi-Agent Platform and its Implementation.* Proceedings of the 2000 International Conference on Artificial Intelligence (IC-AI'2000), June 26-29, Las Vegas, Nevada, USA, CSREA press, 2000, 7 p.

[11] M. Miller, *The Yahoo!* Publisher: Que, 2000, 339 p.

[12] H. S. Nwana, *Software Agents: An Overview*, Knowledge Engineering Review, 11(3), 1996, 40 p.

[13] URL: http://www2.reference.com/

[14] URL: http://www.flashpoint.com.au/

[15] URL: http://www.mycnn.com/

[16] URL: http://infoseek.go.com/

[17] URL: http://www.paperboy.de/

[18] URL: http://www.wapforum.org/

[19] URL: http://my.yahoo.com/

[20] M. Wooldridge and N. Jennings, *Intelligent Agents: Theory and Practice.* The Knowledge Engineering Review, 10(2), 1995 p. 115-152.