USING SOFTWARE AGENTS TO PERSONALIZE ACCESS TO E-OFFICES

Jarogniew Rykowski
The Poznan University of Economics, Mansfelda 4, 60-854 Poznan, Poland
e-mail:rykowski@kti.ae.poznan.pl

Abstract: In this paper, we propose a framework capable of personalization of both the access and the behavior of an e-office. Our solution is based on ACE software agents, partially prepared by (and for) office supplicants, executed in both user personal devices and the e-office network. The main goal of our framework is to provide a universal system for monitoring and notifying about information changes, on one hand, and user-defined customization of e-office services, on the other hand. The framework enables service personalization, user-defined combining and "pipelining" of services, individual brokerage among services and supplicants, and personalization of information delivery. The framework makes it possible to adjust fixed, closed software environment of an e-office to individual requirements and expectations of different users.

Key words: Software agents, personalization, monitoring, e-administration, e-government

1. INTRODUCTION

Recently, a progress in telecommunication (mainly mobile) and mass and personal computing creates many new interesting areas for tele-informatics [4, 13]. Among others, e-administration seems to be one of the most promising [5]. Recently, many countries stressed on wide introduction of e-administration utilities to citizens [3, 20, 24]. However, comparing with traditional administration, usually what is really changed is a way of access (internet- and telecommunication-based) only. A centralized core of an (e-)office usually remains unchanged, being stable and common (i.e., behaving in the same way) for all the users [6, 7]. Moreover, several "back office" activities are performed in the traditional manner, i.e., manually or at least with the help of a local (inaccessible by supplicants) computer system.

Similar to the "traditional" internal organization of an e-office, a typical way of access to an e-office is realized in the "traditional" (however, this time from an Internet user point of view) manner, as a Web interface. Such access, quite easy for advanced, educated users, may be a great obstacle for some people. Moreover, in contrast to classical mass computer systems (as databases, Web data sources, bank systems, etc.), each access to an e-office is different, due to wide differences in user expectations and requirements, restrictions of an environment, places of interest, etc. Thus, a natural need arises to personalize an access to an e-office by (or at least for) end-users. Soon, this need is extended to the personalization of the behavior of an e-office, monitoring information changes, further combining different e-offices into a consistent, complex, however individual e-service, etc.

So far, little work has been devoted to such personalization of access and behavior of an e-office. In this paper, we propose a single framework capable of solving these problems. Our solution is based on software agents, partially prepared by (and for) office supplicants, executed in both user personal devices and the e-office network. The proposed agent-based framework may be used for (1) mass personalization of e-office information systems, both: a way of access and office behavior, (2) combining e-office services and activities to create "virtual e-services", (3) monitoring different office activities and notifying users about "interesting" facts, (4) formatting and sending information to the users via different communication channels (SMS/MMS, e-mail, WAP/WWW, etc.), and (5) taking special profits of mobile devices and mobile communication. Due to special techniques used for preparing, storing, and executing agents, the proposed framework may be used even in closed highly secured e-administration systems.

The remainder of the paper is organized as follows. In Section 2, main rationale for using software agents to contact e-offices is presented. In Section 3, the ACE framework is described being a basis for our proposal. In Section 4, overall system architecture is presented, and basic agent classes are described. In Section 5, typical scenario is given of creating and using ACE agents for contacting an e-office. Section 6 concludes the paper.

2. MAIN RATIONALE FOR USING AGENTS TO CONTACT E-OFFICES

In the traditional approach, the supplicants have to accommodate themselves to the organization of an office (time, place, form, etc.). Even if modern technologies offer several new possibilities to access an e-office, such restriction is still a must. A supplicant is forced to use certain communication channels, fixed forms and documents, given authorization

schema (login, password, PKI keys), etc. However, user expectations are growing. As possibilities of computer and telecom networks grow, a need arises to incorporate new technologies in our everyday work, including efficient, modern ways of contacts with e-administration. A natural trend is to *personalize the access* to an e-office, to adjust to the possibilities of the end-user device, on one side, and user individual requirements and restrictions, on the other side (personal data, national language, etc.).

Users tend to communicate with an e-office in the same manner as for the traditional human staff. In today's systems, however, due to automatization and fixed organization, methods of access to an e-office are quite restricted. Usually, a specialized Web page is used as the main tool for access to an e-office. From a user point of view, however, it is not efficient and comfortable. First, information and forms from the page are fixed, even if generated dynamically from some templates, fixing the way of access to the e-office. Second, there are serious limitations while trying to personalize Web-based access, both technical and economical [18]. Third, a Web page is usually devoted to an on-line access – the results of a user demand are accessible immediately, and may be immediately displayed. However, manual "back office" and essential clerks' activities introduce temporary or longer delays that force the access to be changed to the off-line form. To this goal, such tools as e-mail (and even SMS/MMS) seems to be a better choice. However, users cannot send requests by e-mails, mainly due to the lack of formatting and automatic reception of natural-language-based information by the e-office computer system. As a result, a user is forced to use a Web page for registering a request, and e-mail or similar tool for getting the results. The meantime correspondence is also performed in this mixed style.

The above WWW/e-mail mixed interface is quite tiresome. Besides, it is quite natural that people tend *to use modern communication technologies*, as IM, SMS/MMS, and voice gateways (like PTT), previously devoted to human-human communication, as basic communication channels to e-administration, in both directions – to and from an e-office. However, as stated above, it is quite difficult to provide automatic syntax/semantic analysis for natural-language messages [2, 19], unformatted and with fuzzy meaning. This task is traditionally performed by the human staff of an office. In general, Thus, such simplified solutions as chatterbots [10, 23] are used. However, a chatterbot should be pre-programmed with several keywords to be extracted from a message and further processed. Both the keywords and the algorithm for the extraction should be individualized for particular users, and this is not a trivial task. Thus, so far the chatterbot interfaces are used quite rarely, mainly as generic interfaces to information systems, such as company's front-end, help desk, etc. [11], with limited support for e-office applications.

The trend for individualizing the way of access to an e-office may be broadened to the *personalization of the behavior of the e-office* (to some extent, of course). This covers mainly processing individual cases, and pipelining of several e-services. Even if for a single e-office most of the user cases are similar, there are always some cases that need special attention. However, it is not economically and technically justified to provide a very complex system with large functionality, to deal with any case. Moreover, as one cannot foresee all the possible expectations and demands of all the potential users – it is not possible to satisfy all the users (and the clerks). Instead, a system should be open, with a possibility of incorporating a new functionality once an individual need arises. This also concerns a need for *combining several e-offices into a consistent, single "pipeline"*, when a result created by an e-office is used as an input to another office. So far, such service combining has to be performed manually by the users.

When a user case is processed for a longer time, a supplicant needs to be informed about current state of such process. Thus, a need for *continuous monitoring of the request* arises. Usually, it is needed to contact the office to get the latest info, in a manual, on-line manner. From the user point of view, a phone is the best tool for performing such fast checks. However, from a clerk point of view, unexpected phone calls are serious disruptions of his/her work. A automatic tool capable of getting the "phone-like" natural-language requests and sending the latest info would be very appreciated by both sides.

Last but not least, the *user identification, directly adopted from classical distributed systems, is not efficient enough.* First, the classical approach – user name/password – is easy to crack, especially for non-secured channels as e-mail and SMS/MMS. Traditional PKI cryptography is available mainly from the stationary computers or specialized hardware devices (at least a tokenizer or a specialized mobile phone, so far quite expensive and thus not very popular), and mainly for WWW/WAP connections. However, hardware IDs of end-user devices (e.g., IMEI or subscriber number of a mobile phone) and PKI-based digital signatures may be applied to fix a context (basic user/call identification), to determine both contents and formatting of the messages. Regrettable, so far hardware IDs are not widely used for automatic identification and context detection, especially in e-administration.

We propose to resolve all the above problems by the use of a set of individual, system- and user-defined programs – software agents. These agents are to be executed both at the server-side (i.e., in the internal e-office network), and at the client-side (personal mobile devices of supplicants and private stationary computers). In the next sections we describe our framework based on imperative, user-defined software agents and Agent Computing Environment. We propose specialized architecture of a system capable of performing a brokerage among e-offices and office supplicants.

3. SOFTWARE AGENTS TECHNOLOGY

In our approach, we used a classical definition of a software agent, as presented in [8, 14, 25]. A software agent is a program, executed at a given place, characterized by: (1) autonomy – agents process their work independently without the need for human management, (2) communication – agents are able to communicate with one another, as well as with humans, and (3) learning – agents are able to learn as they react with their environment and other agents or humans. As follows from the above definition, an agent may be programmed by its owner, thus allowing unrestricted personalization of behavior of this agent [2]. Agents may be executed in different places, according to owners' needs and possibilities of the end-user hardware [12].

The ACE framework is based on a set of distributed Agent Servers [15, 16], each of them capable of storing and executing software agents. The agents may be moved among Agent Servers. There are "light" Agent Servers with limited functionality to be executed in a "thin" hardware/software environment (e.g., mobile phones), and "thick", massively used Agent Servers located in stationary network hosts. The "light" servers are mainly used for executing individual agents of an owner of a mobile device, while the "thick" ones are used by many users in parallel, usually to access certain services, external software systems, and public communication channels.

There are two basic classes of ACE agents: public System Agents, and Private Agents. Public *System Agents* SAs are created by trusted users (usually system designers), to be used in a mass manner by many users, providing information in a standardized form and with optimum effort. As overall efficiency is of primary concern, SAs are programmed in Java. A way of usage of a given SA cannot be changed by an ordinary user, however, it may be parameterized during the invocation [17].

The *Private Agents* PAs are created and controlled by their human owners. Unless directly ordered by its owner, the agent cannot be accessed by any other agent. For private agents, the main problem is to achieve a reasonable trade-off between overall system security and a need for remote (i.e., server-side) execution of user-defined, thus „untrusted" (from the local administrator point of view) code. Several restrictions and limitations must be applied to user-defined code, protecting the system from (intentional or accidental) damages. Thus, a specialized language is proposed to program agent behavior, based on XML and equipped with several non-standard mechanisms like run-time monitoring of CPU time and memory allocation [15]. The language is of imperative type, thus allowing much wider personalization of the agent code in comparison with the classical declarative approach. XML-programmed private agents may invoke huge

library of on-site, residential, Java-based system agents: communicators, services, brokers to external software systems, tools and utilities, etc. Usually, a small private agent, being a "light" mobile entity, is able to use (i.e., execute) several system agents, to achieve different goals. From the user point of view, the system is effective and powerful, and even small private agents are "intelligent" enough to fulfill complex requirements. From the system point of view, private agents executed at server-side do not pose a threat to local environment and other agents.

A typical Agent Server is equipped with several specialized system agents, so called input/output gateways, able to communicate with an external world (including other Agent Servers, local and remote software, and humans) via communication channels of different type and purpose. In general, two basic types of human-agent communication gateways are available: textual and Web-based. A *textual gateway* is able to exchange flat text messages among humans and agents. Textual gateways may use such means as an e-mail SMTP/POP3 connection, SMS/MMS connection, a voice gateway, etc. Once sent by a textual message, an ACE agent may act as a chatterbot, analyzing the message via keyword extraction and analysis [19].

Web-based gateways are used to access an agent via a WWW/WAP page, and from specialized ACE applications. For semi-automatic formatting of both contents and presentation of the data to be sent, XSL-T technology was adopted with XSL transformations defined in a personal manner and stored in private agent variables [18]. To improve data presentation, automatic detection of a technical characteristics of the end-user device is applied.

Gateways to external data sources are mainly used for automatic monitoring of information changes. As a change is reported by an external data source, a gateway invokes a selected agent. The agent may next pass the notification about "interesting" changes to user(s), via certain telecommunication gateways. What is "interesting" for the user is programmed by him/her in the code of the private agents [15, 17]. Thus, a set of user's agents is an "intelligent", personalized filter of changes of monitored data.

To facilitate monitoring, so called subscriptions may be used. A *subscription* is a request of automatic execution of (a part of) agent code after introducing certain change(s) in agent's internal variables. A resulting message (subscription alert) is asynchronously (i.e., at a time that cannot be predicted in advance) sent to a subscriber – either agent owner (a human), or another agent. A subscription is set by the agent owner, and at any time may be revoked or changed. The executed agent code is usually responsible for filtering changes towards detecting "critical" updates. The subscriptions permit efficient monitoring of data changes. Once a change is detected, is it checked and eventually reported as an alert. However, as long as there is no change in monitoring information, the agent code is not activated.

4. OVERALL SYSTEM ARCHITECTURE

The ACE framework, described in Section 3, is a generic agent-based environment. As such, this framework should be adjusted to given applications. The adjustment covers building specialized SA agents – gateways and drivers to external systems included – and proposing some prototypes (patterns, skeletons) for PA agents, to be further personalized by the end-users. The SAs are responsible for assuring communication with an e-office (WAP/WWW, e-mail, SMS/MMS, voice gateways, XML-based external applications and systems, etc.), standardization of input/output messages, basic brokerage with services from the internal e-office network, caching and synchronizing the access to the services, etc. There are as well some specialized SAs being system-defined utilities, as for example natural-language parser (programmable chatterbot), national-language detector, XML, HTML and WML formatters, etc. A detailed list of SAs depends on given application, thus here we cannot provide a fixed description. This also concerns user-defined PAs. We describe here a generic system architecture only, assuming that this is a starting point for user-specific applications.

Note that fixing the architecture levels to a mixture of the private and the public layer is not a significant restriction to the personalization of the system behaviour. A set of specialized SAs and user-defined PAs may be changed and extended at any time, both from the system, and from a particular user points of view. Different users observe different behaviour of their individual agents, and the whole system is efficient and scalable.

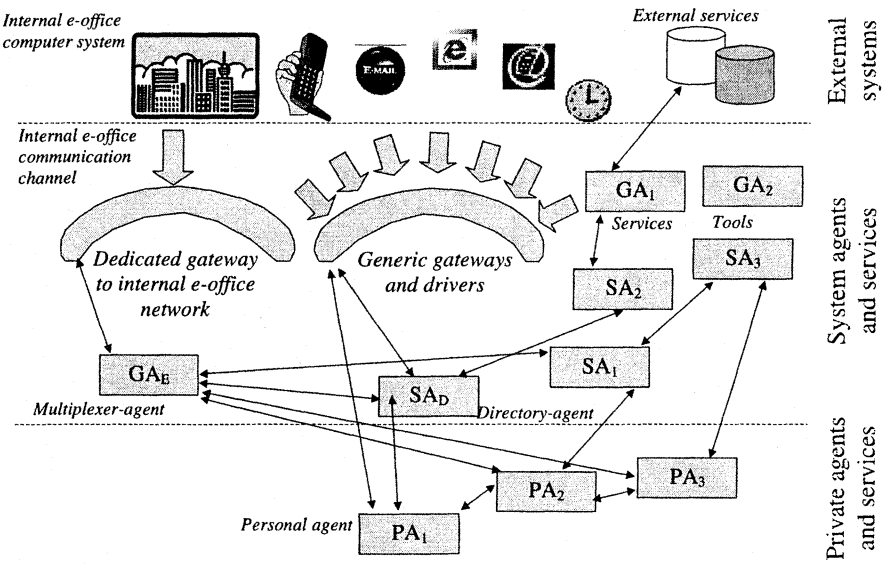

Figure 1. Layered architecture of an ACE-based access to an e-office

The layered system architecture is presented in Fig. 1. There are three basic system layers: (1) connections with external systems and communication channels, (2) basic, predefined system services of an e-office, and (3) private, user-defined agents and services. The presented architecture comprises only logical connections among agents and external systems. The agents may be distributed in the network, in particular, some (private) agents may be executed in end-user device (a palmtop, a notebook, a private stationary host), while some agents (both system and private) – in the local area network of an e-office. Physical connections among distributed agents are realized via specialized agent-agent gateways of distributed Agent Servers, with a possibility of encryption, tunnelling, firewalling, etc. Thus, agent-agent communication is safe, even if the communicating agents are distributed across the network.

Beside typical gateways for basic communication channels (e-mail, WAP/WWW, SMS/MMS, PTT/voice gateway) there are five basic classes of ACE agents: a broker-agent capable of contacting local area network of an e-office, system tools and services, catalogue-agents capable of informing about capabilities of system-defined agents (mainly the above-mentioned system tools and services), user tools and services, and user personal agent.

In general it is not possible to determine detailed characteristics and functionality of a typical agent belonging to one of these classes – this is application- and user-dependent. However, it is possible to describe a general strategy of providing and using agents belonging to all these classes. Such description is given in the next sections of this paper. Note that fixing agent classes and thus (somehow) agent functionality does not restrict introducing new agents and agent classes, in particular monitoring agents, other directory-agents, agent-brokers to/from Web Services and other software systems, agents-gateways for new communication channels, etc.

5. AGENTS CONNECTED WITH INTERNAL E-OFFICE NETWORK

For obvious security reasons, user agents should not access directly services from a local area network of an e-office. Instead, one or more broker-agents should be provided, to act as a firewall/proxy between ACE agents (both system and private) and back-office services and databases. Basic functionality of such agent(s) is the following:

- *Multiplexing and synchronizing requests* generated asynchronously by the population of ACE agents (a proxy for back-office systems).
- *Access standardization* – providing one single standardized way of accessing all the back-office data sources. To this goal, XML should be

used as a basic communication and data storage tool. Agent-broker may be specialized for communication with different data sources accessible by different communication standards and with different data format. Note that a new back-office data source/service may be added at any time, and such modification requires no changes in existing agents.

- *Continuous control over connections* to/from back-office systems, including billing for using the services/information, storing logs of user and system activities, verifying access rights, generating security and missing-information alerts, etc.
- *Caching frequently accessed information* – for slowly-changed information (as most of the information provided by e-offices) for a mass user, an information-cache may substantially reduce network traffic to and from massively accessed back-office systems. For example, such information as law regulations and office rules may be distributed efficiently in a mass manner.
- *Generating user- and source-specific alerts* the ACE agents are waiting for. Important changes in the information provided by back-office data sources may be quickly reported to the agents. To this goal, user-defined subscriptions (cf. Section 3) may be used, to free agents from continuous monitoring and comparison of information changes.

Using broker-agents permits personalized access to back-office data sources with no decrease in the overall security level. In extreme cases, a broker-agent may be linked with a restricted part of the local area network of an e-office by a dedicated hardware and office-specific protocol (e.g., a serial-cable connection capable of sending only text files at request). Note that the details about an implementation of a safe connection between broker-agents and back-office data sources are usually hidden for the ordinary users. Note also that the broker-agent may be able to filter the whole information flow, including blocking given users (user agents), addresses, specific requests, messages containing specific words, etc.

5.1 System defined tools and services

The system defined tools and services are implemented by system agents, usually prepared by e-office staff to be massively used by the population of users. It is not possible to provide here a description of the agents of this layer, as detailed functionality of the agents depend on given application area. Some examples of commonly used system tools and services are the following: wrappers, formatters, cache utilities, data analysers, etc.

Usually, as the ACE framework is mainly used for brokerage rather for implementation of end-user services, the agents of this layer are used as drivers and gateways to some external systems. These are tasks similar to the broker-agent mentioned in previous section, however, addressed to the

systems independent of an e-office. For example, there may be an agent capable of collecting last law regulations from an official government site, some agents providing links with other e-offices, etc. Moreover, one may imagine a situation when a single agent implements an idea of a "virtual office", combining functionality of several branches of an e-office into a single, consistent service. Note that "virtual services" may be personalized by user private agents as well.

An example of an idea of a "virtual service" is given in [17], however, for the domain of Web Services [9, 21]. This idea may be extended to non-standardized Internet services as well, including Semantic Web and its extensions [1, 22] and proprietary solutions of particular e-offices. "Virtual services" permit efficient distribution of component services, as well as information exchange among different e-offices and e-office branches.

Even if system tools and services cannot be programmed by ordinary users, wide usage of subscriptions (cf. Section 3) makes is possible to effectively distribute personalized information about data changes. Once subscribed to a given system service, user private agents are immediately informed about „interesting" changes of the information provided by an e-office. What is "interesting" is programmed in both a subscription (i.e., as a variable of a system agent) and in the way of accepting a subscription alert (i.e., as a code/variable of a private agent).

5.2 Catalogue-agents

Usually, a potential functionality/organization of an e-office is reach enough to be too complicated to an ordinary user. Thus, a catalogue of e-office services would be appreciated. The main goal of such catalogue, being a specialized system agent, is twofold. First, human users may contact the catalogue to collect useful information about services and law/organizational restrictions on using these services. Second, user agents are able to get some formal information about contacting system agents (services) – lists of functions provided, function parameters, etc. Thus, private agents have a possibility of (semi-)automatically adjust to a specificity of a given e-office.

Catalogue-agent is able to generate user-understandable help messages and machine-understandable interface descriptions (meaning and types of invocation parameters) at request. The latter may be used to automatically adjust PAs to the specificity of an e-office. It's obvious that reacting to serious differences in the interfaces to system agents belonging to different e-offices cannot be fully automated, however, critical differences in interfaces may be immediately reported to user agents (mainly the personal agent) to notify "fatal" errors in communication with an e-office. It is up to the user to react to such notifications.

Similar to the system tools and services, the catalogue-agent may be implemented as a broker to an external WS Directory Server, LDAP or a similar service, including proprietary solutions for given e-offices.

5.3 User-defined tools and services

The role of this group of agents is similar to the layer composed of system defined tools and services (Section 4.2) except that this layer is composed of users' private agents. These agents may be executed either at client-side (i.e., in client's mobile devices and stationary PCs), and at server-side (i.e., in the Internet and in an internal local area network of an e-office, depending on application area). Usually, private agents executed at client side are related with data formatting and displaying, while the agents executed at server-side are related with monitoring of changes and user-defined brokerage. Client-side agents are related with (automatic) adjustment of both contents and format of the information to user personal needs and expectations, time, date and place of information delivery, restrictions for end-user hardware and software, characteristics of a communication link, etc. Note that client-side agents may be chosen automatically depending on current user's situation. For example, a user visiting a real office is able to use his/her mobile equipment only, with limited screen&keyboard possibilities, just to receiving very basic alerts and messages. The same user, while back home, is going to use wide screen of his/her stationary PC to get more details about reported e-office activities.

On the contrary, monitoring agents cannot be executed at client-side, as the permanent communication costs would be too high [12]. Thus, such agents should be migrated to a network server, possibly as close to the data source (i.e., e-office local area network) as possible. Working autonomously, user-defined monitors detect "critical" data changes (e.g., by the use of the above-mentioned subscriptions defined in system agents), filtering incoming information and informing owners about "interesting" facts. Note that (1) monitoring agents are strongly personalized, and (2) such agents must be executed at server-side, thus a trade-off arises between security and efficiency. From the system point of view, this trade-off is satisfied with the use of a mixture of ACE private and public agents.

5.4 User personal agent

A personal agent is a main entry point for a particular user, synchronizing all the incoming and outcoming messages, storing user preferences (with reference to preferred communication links, data formats, presentation interfaces, etc.). The agent is used as a main broker/synchronizer for all the traffic between a user (agent owner) and all the other agents working for this user (both system and private agents). Usually, the main task if this agent is

to collect user request via a certain communication channel (i.e., from a certain agent-gateway), standardize this request (usually to the form of an XML message), choose the most adequate private/system agent to serve this request, and finally collect the response and send it back to the user.

The personal agent is also used for sending alerts generated by the user-defined subscriptions. According to current user preferences, an alert is formatted and sent via a certain communication channel – an SMS/MMS, e-mail, IM or a similar chat-based system, a notification waiting in a WWW server, etc. The "best" (at the moment) communication channel may be chosen automatically, taking into account current position of a user (reported by a specialized agent), user's last login (place and time), independent restrictions (e.g., user's availability, scheduled meetings and travels, etc.).

A special interest is taken to assure efficient communication with the use of "handicapped" devices as for example mobile phones. To this goal, a natural-language interface is provided, aiming in a personalized chatterbot conversation [19]. Note that such simplified NL interface may be adjusted to given user(s), thus allowing efficient conversation with automated services of an e-office in a human-to-human manner. This limits a necessity of learning different, fixed, complicated interfaces for different e-offices (to some extent). This also greatly improves an effectiveness of communicating via limited mobile channels, as for example popular SMS messaging, by incorporating user-defined abbreviations and user-specific vocabularies.

6. TYPICAL SCENARIO OF USING ACE AGENTS FOR CONTACTING AN E-OFFICE

In the real life, we must adjust the way of contacting an office to the internal organization of this office: business hours, staff availability, clerk specialization, information workflow, etc. A specificity of an office case / supplicant should be taken into consideration as well. Generic strategy of using ACE agents to facilitate these goals is the following.

1) A supplicant addresses given branch(es) of an e-office, presenting details of his/her office case. In addition, the supplicant determines some personal preferences: mobile and stationary devices used, identifiers (e.g., a phone subscriber number, IMEI device number, PKI signature, IP/DNS address, e-mail account, etc.), preferable communication channels, etc.
2) The office branch proposes a set of predefined SAs and PAs to be installed for the supplicant in the Agent Servers located in the local area network of the office. This set of agents is capable of serving supplicant's case in a predefined way that is common for all the users.

3) The office branch proposes a set of PAs to be installed at client-side by the supplicant, to facilitate contacts with the e-office. It is up to the user to accept this set of agents or not. In the latter case, only the PAs executed in the office network are used. However, once a supplicant accepts PAs agents in his/her private hardware/software environment, he/she may profit in faster notifications (e.g., SMS messages sent to mobile phones), better adjustment to end-user devices, more personalized formatting and presentation methods, and other functionality related with client-side information processing (cf. Sections 4.4 and 4.5). Note that, as the supplicant may inform the office about detailed characteristics of the devices used and preferable communication channels, the generated client-side PAs may be customized, even if the server-side PAs are quite standardized (at the beginning, however, cf. Point 6 below).
4) The SAs and PAs installed are equipped with a set of predefined variables and subscriptions, to deal with the office case of the supplicant. The subscriptions are parameterized by case-dependent information, e.g., office case number, supplicant personal data, etc.
5) The set of accepted agents is installed and activated each time the change of information is detected related with any active subscription. The detected changes are filtered and eventually sent to the supplicant (usually, to the personal agent, and further via given communication channel directly to the end-user device, e.g., a mobile phone).
6) Once the supplicant is not satisfied with a subscription or PA behaviour, he/she has rights to redefine both data and code of his/her agents. For advanced users it is possible to fully personalize the behaviour of the system, both at server-, and at the client-side. For non-advanced users, it is still a possibility to define preferable communication channel, ways and timings of sending alerts, formats and contents of messages generated by the office agents (e.g., small SMS alerting vs. formatted HTML e-mails). Such changes are easily introduced by the office staff (at supplicant's request) or by the users, with the use of specialized agents.

Note that even if the users change the behaviour (i.e., the code) of their private agents executed at the server-side, the overall system security is not reduced due to continuous run-time inspection of "untrusted" (user-defined) agents (cf. Section 3). Note also that private agents are executed only in two "trusted" (from a user point of view) environments – an e-office local area network and private "network" of the user, usually composed of a mobile phone and a stationary home PC. Sending messaging among user agents may be encrypted (e.g., using PKI cryptography), increasing overall confidence level of contacts with the e-office (for particular users).

7. CONCLUSIONS

In this paper, we propose a framework capable of personalization of access and behavior of an e-office. Our solution is based on ACE software agents, partially prepared by (and for) supplicants, executed in both user personal devices and the e-office network. The main goal of our framework is to provide a universal system for monitoring and notifying about information changes, on one hand, and user-defined customization of e-office services, on the other hand. The framework implements an idea of individual "virtual" service, enabling: (1) personalization of behavior of the service, (2) user-defined linking and "pipelining" of services, including services of different e-offices, (3) individual brokerage among services and supplicants, and (4) personalization of information delivery (time, place, form, communication channel, etc.).

The framework makes it possible to adjust fixed, closed software environment of an e-office to individual requirements and expectations of different users. To this goal, one does not need to interfere in e-office internal organization and software systems. Instead, user- and system-defined ACE agents are used as brokers among supplicants and the services offered by an e-office. The agents may be executed both at the client-side (mainly in personal mobile devices), and at the server-side (i.e., in the local area network of an e-office), with no decrease in overall system safety and security. Efficient distribution of agents improves system scalability, and restrict traditional bottlenecks of today's centralized offices.

The whole system is scalable and open for new users and services, including such non-standard functionality as reporting the less-busy business hours, current case state and missing documents, predicted timings, etc. The system is also open for new communication standards and growing functionality of end-user devices, mainly mobile equipment (for example voice gateways, IM and chat communication, etc.). Several natural-language interfaces makes it possible to communicate with the ACE agents as if they were humans, facilitating the usage of the system by non-advanced users.

Although the ACE framework is fully implemented and industry-tested [15÷19], we could not test the proposed application in a real (e-)office. Thus, we developed a simulator, being a set of ACE agents, to model a behavior of a sample e-office and different office supplicants. Even this simplified simulator showed that the system would be very useful, both for the supplicants, and for office staff. The initial implementation costs are quite low (a PDA/PC with Java environment installed, and an SQL-database for storing agents), in comparison with other parts of a local area network of a typical office. We are looking now for an office ready to implement and test our approach with real supplicants and services.

REFERENCES

[1] Benjamins, R., Agents and Semantic Web: a Business Perspective, http://www.agentlink.org/agents-barcelona/presentations/3_RichardBenjamins_final.pdf

[2] Bonett, M., Personalization of Web Services: Opportunities and Challenges, 2001, Ariadne Issue 28, http://www.ariadne.ac.uk /issue28/personalization/intro.html

[3] E-government - Electronic Government, EurActiv: homepage, http://www.euractiv.com/Article?tcmuri=tcm:29-117473-16&type= LinksDossier

[4] E-Government in developing countries, http://www1.worldbank.org/publicsector/egov/

[5] eGovernment leadership: High Performance, Maximum Value, The Government Executive Series, http://a456.g.akamai.net/7/456/1701/5e33f326cdecd2/ www.accenture.com/ xdoc/en/industries/government/gove_egov_value.pdf

[6] eOffice homepage: Virtual Office Management that Transforms Any Traditional Office, http://www.greatshop.com/eoffice20.html

[7] eOffice Services homepage, http://www.eofficeservices.biz/

[8] Franklin S., Graesser A. Is it an Agent, or just a Program? A Taxonomy for Autonomous Agents, 3rd Int. ATA Workshop, Springer-Verlag, 1996

[9] Lombardi, V., Designing for Web Services, April 2002, http://www.newarchitectmag.com/documents/s=2452/new1015627350101/index.html

[10] Marcus P. Zillman's chatterbot resources and sites, http://chatterbots.blogspot.com/

[11] Microsoft Agent home page, http://www.microsoft.com/msagent/default.asp

[12] Milojicic D., Trend Wars – mobile agent applications, IEEE Concurr., 1999, pp. 80-90

[13] Moriuchu, Y., Private Sector Recommendations to government on realization of e-government, in e-Government, http://www.gbde.org/egovernment/

[14] Nwana, H., Software Agents: an overview, Knowl. Eng. Rev. 11-1996-3, pp. 205-244

[15] Rykowski J., Agent Technology for Secure Personalized Web Services, 24th Int. Scientific School ISAT 2003, Szklarska Poreba (Poland), 2003, pp. 185-193

[16] Rykowski, J., Databases as repositories for software agents, in Emerging Database Research in East Europe eds. B. Thalheim and G. Fiedler, Pre-conference Workshop joined with the 29th VLDB Conference, Berlin, Germany, 2003, pp 117-123

[17] Rykowski J., Cellary W., Virtual Web Services - Application of Software Agents to Personalization of Web Services, 6th International Conference on Electronic Commerce ICEC 2004, Delft (The Netherlands), 2004, ACM Publishers; pp. 409-418

[18] Rykowski, J., Juszkiewicz, A., Personalization of Information Delivery by the Use of Agents, IADIS Int. Conf. WWW/Internet 2003, Algarve, Portugal, 2003, pp. 1056-1059

[19] Rykowski, J., Using software agents to personalize natural-language access to Internet services in a chatterbot manner, 2nd Int. Conf. L&T'05, Poznan, Poland, April 2005

[20] US President's Expanding Electronic Government initiative, http://www.whitehouse.gov/omb/egov/

[21] Web Services Activity home page, http://www.w3.org/2002/ws/

[22] Web Ontology Language (OWL), http://www.w3.org/2004/OWL

[23] Weizenbaum, Joseph: Computer power and human reason. From Judgment to Calculation. S.Francisco, 1976

[24] Westholm, H., Aichholzer, G. The impact of eEurope Initiative on public administration in Europe, http://www.prisma-eu.net/deliverables/SG1administration.pdf

[25] Wooldridge, M., Jennings, N.R., Intelligent agents: theory and practice, Knowledge Engineering Review 10-1995-2, pp. 115-152