The Virtual Secretary Architecture for Secure Software Agents

G. Hartvigsen[¥1], S. Johansen[¥], A.Helme[§2], R.A.Widding[¥], G.Bellika[¥], W.Cao[¥]

[¥] Department of Computer Science, Institute of Mathematical and Physical Sciences, University of Tromsø, N-9037 Tromsø, Norway

[§] Faculty of Computer Science/SPA, University of Twente, 7500 AE Enschede, The Netherlands

Abstract

The Virtual Secretary project focuses on the construction of an environment for secure software agents. As a research vehicle, i.e., to enable full-scale experiments in realistic settings, some major secretarial tasks have been chosen. Our environment for secure software agents includes propagation mechanisms, mechanisms for authentication and control, and common user software. A key element in this environment is the propagation of the Virtual Secretary's user model (i.e., non-executable authenticated control software). This paper gives an overview of the Virtual Secretary system architecture.

1 Introduction

In the last decade, mobile computers have introduced new problems and challenges to distributed systems and computer networks. Through wide area networks, computations can be distributed to and completed on computers in different geographical locations. Information stored in a global network can be fetched or updated in a few seconds. Mobile computers could connect to the network and services from nearly any geographical area in the world. In addition, the variation of computer usage will continue to increase – ubiquitous computing would affect the way of using computers for most user groups.

These opportunities demand tools and system paradigms that exploit their capabilities and reduce their problems. One such tool could be a software agent, i.e., a worm-like program, which effectively uses the network and remote processing sites (see, e.g., Riecken, 1994; Wooldridge and Jennings, 1995). A key issue in the utilization of a network of worm-like programs is the security. Most people do not want to let worm-like programs enter their own secure computer environment. Especially horrifying is the idea of "intruders" carrying out unauthorised tasks. To construct more than a toy program for an open research network, security issues need to be seriously treated.

The problem definition for the Virtual Secretary project is: How can a environment for secure software agents for secretarial tasks in a global network be constructed? The problem definition is based on the assumption that security can be improved through the transmission of only an

1. This work was partly supported by the Research Council of Norway, grant no. 100425/410.
2. This work was supported by Xerox EuroPARC, Cambridge, Olivetti Research Laboratory, Cambridge, Digital Equipment Corporation, and by the European Communities' ESPRIT Programme through BRA project 6360.

authenticated user model which specifies the user's current task and his/her most reasonable way of behave when a specified problem may occur. This approach implies that no executable code is transferred. The assumption requires that a general version of the Virtual Secretary (i.e., body process) already is present on the remote host. If not, given that security can be guaranteed, minimum code should be transferred. This means that the problem definition can be divided into the following subproblems: (1) How can a user model for propagation of processes (i.e., mobile agents) be constructed?; (2) What are the security issues that have to be met to propagate user models in a secure way?; (3) How can a control language for propagation of agents and tasks through user models be designed?; and (4) Based on the results from 1-3; How can a Virtual Secretary that handles (i) file retrieval from remote hosts in a global network, (ii) electronic mail, and (iii) propagation of the user's environment, be constructed? The tasks in point 4 do only involve the user's Virtual Secretary (located on accessible hosts in the network), and not contact with other users' Virtual Secretaries.

This paper gives an overview of the Virtual Secretary software architecture, which is being constructed at the Department of Computer Science, University of Tromsø.

2 The Virtual Secretary's Tasks

The first version of the Virtual Secretary focuses on tasks that do not involve communication with other users' Virtual Secretaries, i.e., cooperation with other users' agents. Currently, we focus on three groups of secretarial tasks: local information filtering, global information filtering and propagation of the user's environment. Local and global information filtering include file retrieval from remote hosts in a local and global network and administration of electronic mail. Propagation of the user's environment includes export of the user environment to a remote host.

We consider the tasks above to be some of the most important and most frequently performed tasks this kind of secretary needs to take care of. The selection criteria of tasks are the tasks' need for and utilization of the user model. The tasks provide a testbed for the Virtual Secretary system architecture.

The file retrieval tool is used when the user wants a file or several files from one of his workstations given that he does not remember the file name and possibly also not the name of the workstation (i.e., file server). Today, this task might be completed through a remote login followed by a browse through most files. To be able to examine the file, parts of (or the hole file) need to be copied (in one way or another) to the user's current location. This again may require allocation of a huge amount of network resources. In addition the user needs to spend the time it takes to locate the wanted file. This knowledge includes a specification of current tasks. The Virtual Secretary carries with her a certain amount of knowledge of the user and, if necessary, uses this knowledge in the location of the wanted file(s). In this way the correct file and the user model will be loaded on the network. The user will also be able to work with other tasks when one clone of the virtual secretary is occupied with the file retrieval. Obviously, the file retrieval tool should also take care of normal file retrieval (where the file name and host name are known).

Global access to files forces the file location mechanism to be as supportive as possible to minimize the network traffic. Especially multimedia files are expensive to copy from one site to another. This calls for a sophisticated mechanism to minimize the network traffic. Of special importance is the support given at place to find the file needed. As mentioned above, the problem occurs if the user does not remember the desired file name, and does not manage to choose the right file when the file list is presented to him. This problem could be handled by a kind of adaptive software that contains a user model that guides the system in the seek of the correct file. This user model could be copied to the remote machine as part of the Virtual Secretary.

The mail handling tool includes automatic forwarding of "important" mail, e.g., by filter distribution lists and "garbage" mail. A scenario would be that a Virtual Secretary propagates a couple of hours ahead of the user and installs his wanted features on the remote host. The mail system should be adaptive in the way that the user model is updated and when appropriate it changes the user interface. This might be done through adapting some ideas from adaptive user interfaces (Bundy, 1984; Rissland, 1984; Greenberg and Witten, 1985; Cleal et al, 1988), and in particular the Ratatosk project (Danielsen et al., 1990).

The the tool for propagation of the user's environment transfers the user's own graphical user interface, including menus, access to (as many as possible of) his basic applications. This tool is wanted when the user move, permanently or temporary, to another place.

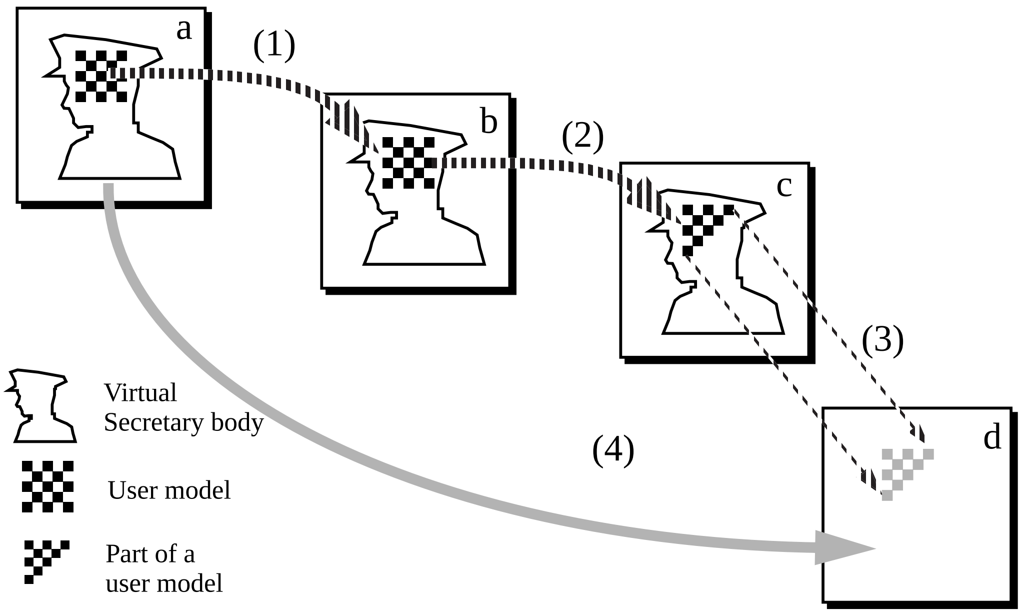

FIGURE 1. The propagation of a user model in a Virtual Secretary setting. Assumptions: (1) Propagation. (2) Mission to a host holding a Virtual Secretary body process. (3) Mission to a host not holding a Virtual Secretary body process. (4) Copy of a Virtual Secretary body process. (From Hartvigsen et al., 1995)

Figure 1 illustrates how the Virtual Secretary propagates to a remote host. In (1) we have that the hosts a) and b) both hold a Virtual Secretary body process. In addition, host a) holds a user model. Given that the user moves to host b) only the user model needs to be propagated. In (2), we have a similar situation as for (1) except that now the Virtual Secretary is sent on a mission to perform a specific task, e.g., information gathering. Now, only part of the user model might need to be propagated. Arrow (3) illustrates the situation that the Virtual Secretary, in order to complete its mission, needs to propagate to host d) which does not hold a Virtual Secretary body process. Since the allowance of this kind of operation heavily increases the risks, it should only be allowed after a thorough security control. If host c) is a portable computer it might be more convenient to get the copy of the Virtual Secretary body from, e.g., host a) (as shown in arrow 4) and the user model from host c).

3 Approach

The ordinary way to deal with security problems is to adopt a *security policy* that defines appropriate levels of security for the system's activities, and enforced by a set of security mechanisms. This approach implies no additional precautions to meet new security challenges from software agents. The problem of this approach with respect to software agents is that we do not have any guarantees for what might happen when an active program is allowed to enter the system.

Another approach to the construction of software agents, taken by the Virtual Secretary project, is to exclude as many security risks as possible through the adoption of a different architectural approach to system design, especially the propagation scheme. In this approach an environment for secure software agents is created, including:

- propagation mechanisms,
- authentication and control mechanisms, and
- common user software.

To reduce the security problem we have chosen to propagate passive code as authenticated user models. This approach is based on the assumption that a Virtual Secretary body process (i.e., a Virtual Secretary excluding a user model) exists on the target host, and that a description of this process is well known. A user model contains a description of the mission, the user (including the user's history, current tasks, preferences, etc.), administrative and control data, etc. The user model, or, if appropriate, part of the user model is initialized by the body process on the remote host to continue its mission. Finin (1989) argues that "a user model is that knowledge about the user, either explicitly or implicitly encoded, which is used by the system to improve the interaction." Authentication and control mechanisms ensure that traditional security issues such as authentication of principals, integrity of communication, confidentiality, etc., are obtained.

Common user software, i.e., that all users may get a complete description of the Virtual Secretary process and use the secretary himself/herself, de-mystifies the software. The description of the Virtual Secretary's tasks will be available for all users. When a user needs to send its Virtual Secretary on a mission in the network, the propagation will take place through the copying of the user model or part of the user model.

Another feature we expect to gain from the proposed architecture is reduced network costs involved in the process propagation. Given that the propagation is necessary, our approach is expected to reduce the amount of software necessary to be transmitted in order to propagate the process. This feature is of special importance for mobile computers. Although the network bandwidth in wireless networks increases rapidly, still the question of power consumption (i.e., battery usage) strongly indicates that the use of wireless network interface in mobile computers should be minimized.

4 The Virtual Secretary System Architecture

The Virtual Secretary system architecture is mainly based on the requirement put forward by the user model. The architecture is illustrated in Figure 2.

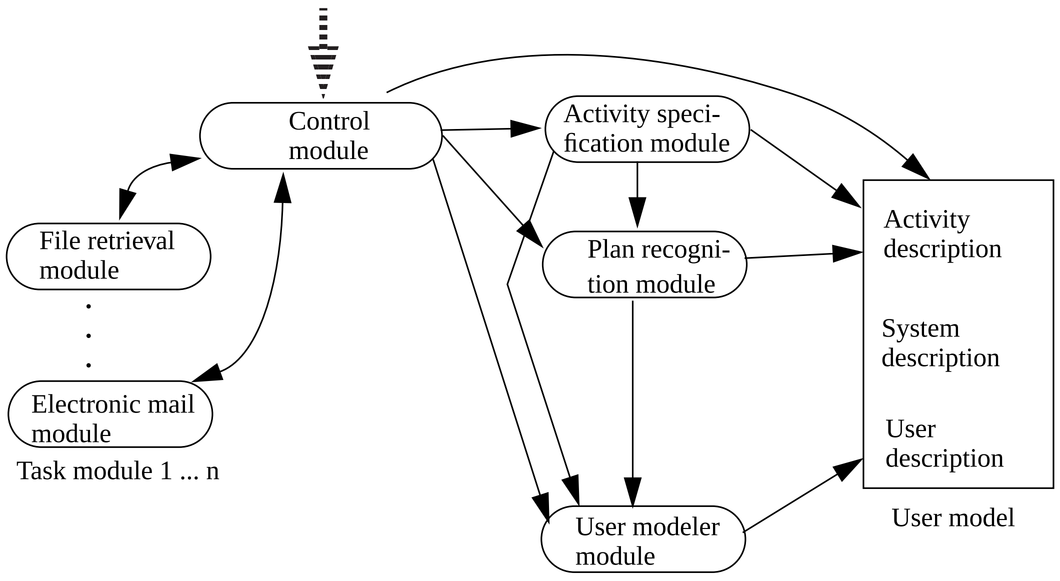

FIGURE 2. The Virtual Secretary system architecture.

The Virtual Secretary consists of the following modules (Figure 2):

Activity specification module	This module offers manual specification of the job that is going to be conducted. E.g., when the user wants the Virtual Secretary to fetch information from a remote host he can specify the task manually.
Plan recognition module	The plan recognition module tries to recognize any pattern in the user's interaction with the Virtual Secretary. A recognized pattern will be restored and used by the tutoring and explanation modules.
User modeler module	This module uses information from the activity specification and plan recognition module to update the user module.
Control module	The control module constitutes the central part of the Virtual Secretary. All input and output (I/O) are organized in this module. This also means that security issues are handled by the control module. The control module calls other modules when needed.
Task module 1 .. n	These modules conduct the offered services from the Virtual Secretary.

The plan recognition module is only briefly addressed in the first version of the Virtual Secretary.

4.1 Security

The key issue in the Virtual Secretary project is security. In the first version of the Virtual Secretary, security is obtained through the construction of a environment for secure software agents, including

- propagation mechanisms,
- authentication and control mechanisms, and
- common user tools.

In the Virtual Secretary, we use body processes to enable certification in advance. The body processes are willing to accept requests over the network and can act as bodies of worm-like programs, because the requests can be propagated further.

TABLE 1. Tasks available at service Level 1 and service Level 2. ✔ denotes generally available. (✔) means available only if explicitly stated by the system manager.

Task	**Level 1**	**Level 2**
Read News	✔	✔
Web search	✔	✔
Read diary	(✔)	✔
Update diary	(✔)	✔
Read shared files	✔	✔
Read user (private) files	-	✔
Read user (private) mail	-	✔

As illustrated in Table 1 and Figure 3 the user can access services at two service levels (Hartvigsen et al., 1995): Level 1 (Public services) and Level 2 (Private services). Level 1 enables a general user to communicate with the global Virtual Secretary demon and access cite-specific information from WWW, News, and files accessible at this level. Level 2 gives access to private data, e.g., electronic mail, diary information, files etc. This level requires user authentication.

Level 1 tasks (public services) are handled by a general body process through a protocol with a limited set of commands. No agent code is directly executed, but is transferred as a user model and executed as part of the body process. For this purpose the body process has a Virtual Agent shell (vsh) at its disposal. This shell is part of the body process and cannot be directly accessed by external users, i.e., we have no remote shell facilities. Still the same effect is achieved through the commands that constitute the body process protocol. The net result is that only a fragment of the agent code has to be propagated to the target host as safe commands to the body process. (Most of the agent code is already present as part of the body processes.)

Access to private data (Level 2) is done through another configuration of the body process, which let the user exploit the full potential of the Virtual Secretary. At this level, the body process has to use methods for authentication and control. Level 2 uses a similar shell as the public services, but the access to personal data gives an other configuration. Note that a new agent generates a new copy, i.e., all agents are separated. The motivation for designing our own query language is to be able to handle many types of data in a uniform way, and to simplify future extensions.

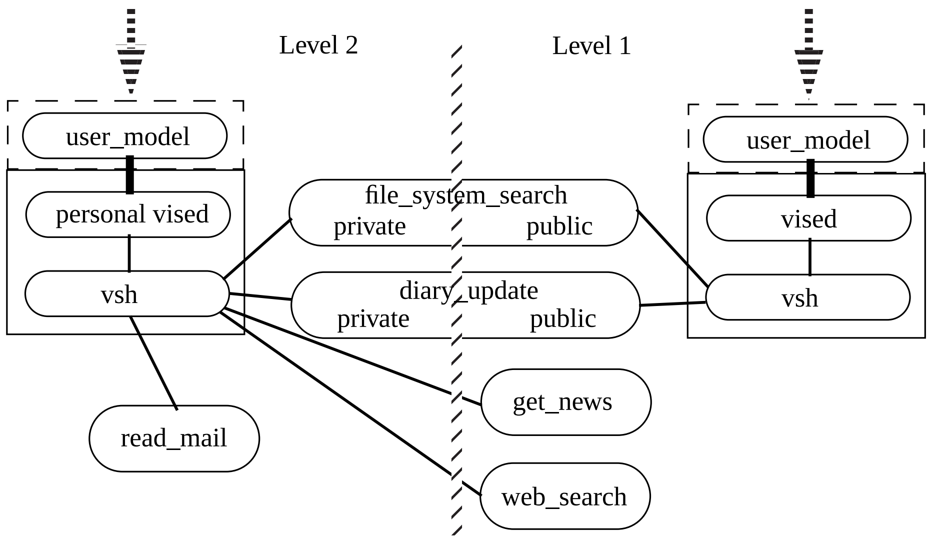

FIGURE 3. Access levels (Hartvigsen et al., 1995).

The Virtual Agent shell (vsh) offers three basic services (Hartvigsen et al., 1995):

1. Information filtering and retrieval
2. Agent support
3. General programming concepts

The information retrieval part specifies what data to retrieve and what to filter out. The result of a query is a net list of documents with a short description and a link to the document. A list of the documents that vsh filtered out can also be produced. Please note that vsh only acts as a tool for implementing the requests from Virtual Secretary modules. The vsh is by no means a remote shell. All external access has to go via the body processes.

The vsh handles data from different sources by contacting local data providers for the appropriate data types. Current data types are: www (html), news, email, text files, diary, BIBT$_E$X and bibsys (which is the Norwegian University Library system). The reason for dividing the architecture into many modules is that each data type requires different treatment. That is, we convert it all to text, but the communication protocols are different. The body processes can then access all the data through different modules and limit the search to either a few data-types or search them all.

5 Status and Open Problems

The Virtual Secretary is an ongoing project at the University of Tromsø. Currently, we focus on the system architecture, including security issues. We argue that our approach to the construction of software agents, i.e., the transference of passive code and not active programs, the use of authentication and control mechanisms, and the provision of common user software, excludes the major security problems related to worm-like programs. However, still several crucial security-related issues are not worked out.

Current problems being addressed include:

- A secure way to propagate a Virtual Secretary body process to a host not holding such a process.
- Software releases – how are body processes updated or invalidated, and how are earlier releases handled?

It should be mentioned that we still are investigating the system architecture, especially alternatives to the vsh.

Although we in the first version of the Virtual Secretary have focused on a limited number of secretarial tasks, we expect that the Virtual Secretary will be able to take care of most secretary services both locally and when traveling around, including:

- Administrate a personal diary, including scheduling of meeting and resource reservations.
- File and retrieve information from various databases globally located.

An important extension would be to include cooperation mechanisms. This will enable several agents to cooperate in solving a specific task.

To take full advantage of the user model approach, another three models are needed: explanation module, tutor module and diagnose module. The diagnose module uses the results from the plan recognition module to uncover misinterpretations and misunderstandings. Notification of possible errors will be transferred to the explanation module that in turns will present a situation description (status) to the user. The tutor module interrupts the user when a new user skill level is detected. The explanation module offers help to the user. The help is either offered explicitly through user request or because of illegal actions.

6 Concluding remarks

As argued by Riecken (1994), the term *intelligent agent* has become quite popular. Announcement of products like the Apple Newton's agent software and General Magic's messaging agents (White, 1994) are evidence of significant interest in agent research and development. However, there is no consensus on the terminology in the field. One classification scheme for a software agent is whether the agent can move around in a network of computers or is tied to one specific host. In the Virtual Secretary project the agent is expected to propagate in a global computer network. A fundamental problem this kind of software has to face is related to communication: how can an agent migrate or propagate to a remote host? This problem represents the focus of the Virtual Secretary project that has been presented in this paper. The project's objective is to construct mobile agents that in a secure way propagate through the network. The first version of the Virtual Secretary will be capable of handling simple secretarial tasks as file retrieval from remote hosts and forwarding and filtering electronic mail (correspondence).

The security risks are reduced through the construction of an environment for secure software agents, including propagation mechanisms, mechanisms for authentication and control, and common user software. The description of the Virtual Secretary's tasks will be available for all users. When a user needs to send its Virtual Secretary on a mission in the network, the propagation will take place through the copying of the user model or part of the user model. In the normal case, this approach implies that no executable code is transferred. The assumption requires that an authorized general version of the Virtual Secretary already exists on the remote host.

7 References

Bundy, A. 1984. Intelligent front ends. In M.A.Bramer (Ed.), *Research and development in expert systems.* Cambridge, Cambridge University Press, pp. 193-203.

Cleal, D M, Heaton, N O. 1988. *Knowledge based systems: implications for human-computer interfaces.* Chichester, Ellis Horwood.

Danielsen, T., Finnset, W., Flægstad, F., Hartvigsen, G., Steen, R. 1990. RATATOSK – an Adaptive User Interface to Computer-Mediated Communication. In: P.Schicker, E.Stefferud (Eds.), *IFIP WG 6.5 International Symposium on Message Handling Systems and Application Layer Communication Protocols* (3-5 Oct. 1990, Zurich, Switzerland) Amsterdam: North-Holland, pp. 345-354.

Finin, T.W. 1989. GUMS – a general user modeling shell. In: W.Wahlster, A.Kobsa, (Eds.), *User Models in Dialog Systems*, Berlin, Springer-Verlag, pp. 411-430.

Greenberg, S, Witten, I H. 1985. Adaptive personalized interfaces - a question of viability. *Behaviour and information technology, 4*, 31-45.

Hartvigsen, G., Helme, A., Johansen, S. 1995. A Secure System Architecture for Software Agents: The Virtual Secretary Approach. In Proceedings of the Second Broadcast Open Workshop (Grenoble, France, July 5-7, 1995).

Riecken, D. 1994. Intelligent agents. *Communications of the ACM*, 37 (7 July), 18-21.

Rissland, E L. 1984. Ingredients of intelligent user interfaces. *International journal of man-machine studies, 21,* 377-388.

White, J.E. 1994. Telescript Technology: The Foundation for the Electronic Marketplace, General Magic White Paper, General Magic Inc., 1994.

Wooldridge, M., Jennings, N.R. 1995. Agent Theories, Architectures, and Languages: A Survey. In: Wooldridge, M., Jennings, N.R. (eds.), Intelligent Agents. Lecture Notes in Artificial Intelligence No. 890, Berlin, Springer-Verlag, pp. 1-39.