HyperNews: the Implementation of the Prototype and Assessment: Working Paper

Morin, Jean-Henry

HyperNews: the Implementation of the Prototype and Assessment

Working Paper

Jean-Henry Morin

Abstract

This paper reports on the implementation of a prototype hypermedia electronic newspaper environment called HyperNews. The HyperNews project is based on agent technology and it's goal is dual: enable the information provider to commercialize multimedia news articles in terms similar to those of a printed newspaper, while helping the information consumers to drastically reduces time spent in searching for and retrieving "interesting" information. The system enforces the copyright control and payment on a "pay per use" basis. The retrieval task is handled by the HyperNews system according to the interests of the reader. Moreover, HyperNews establishes intra- and cross-newspaper hyperlinks to information topics and present the information consumer a personalized electronic hypermedia newspaper through a traditional Java based World-Wide Web browser.

1 Introduction

The HyperNews[1] (hypermedia electronic newspaper) project, part of the MEDIA project [3], aims at developing an electronic newspaper environment based on agent technology. This environment will offer the information providers of newspapers, magazines and alike the means to commercialize electronically the information they hold under similar conditions as the printed versions [7]. It will also offer the information consumers the means to reduce the time spent in retrieving the information for which they have expressed interest from different information providers [2].

The information consumers and providers run an agent execution platform on which the different components of the hypernews system execute as software agents. The mobile agents are used to encapsulate the data in hypernews article agents and to update specific software agents bound to the information providers. The resulting article agents can be distributed and duplicated at will since content access is bound to payment (i.e., superdistribution [27]). This is achieved through a scheme for the commercial distribution of electronic documents that enforces copyright control and payment at the time the information is accessed for reading. Such an environment offers the newspaper and magazine publishing industry the means to commercialize news articles in a way that enforces copyright control and revenue collection. Moreover, it offers the information consumer an interface which is both *vendor independent* and tailorable according to the users needs.

1. This project is supported by the Swiss Federal Government with the FNRS SPP-ICS projects MEDIA (5003-045332) and HyperNews (5003-045333)

In this paper we describe the implementation of the HyperNews prototype. Section 2 summarizes the requirements of a commercial electronic newspaper environment. Section 3 describes the technology used for the implementation of the prototype. Section 4 presents an overview of HyperNews while section 5 describes the agent infrastructure used. Sections 6 and 7 concentrate on the description of the core environment and the HyperNews layer respectively. In section 8 we discuss some security related issues. Section 9 presents an assessment of our work based on a public demonstration of the HyperNews prototype. Finally, in section 10 we present the current state of this work, open issues, future work and draw conclusions from the prototype implementation.

2 Summary of the Requirements for a Commercial Electronic Newspaper Environment

The requirements for a commercial electronic magazine can be seen from two points of view: the information consumer's (the reader), and the information provider's (the publisher). For an in-depth description and understanding of these requirements, we refer the reader to [5]. In summary, the reader of an electronic magazine should be able to:

- choose his information providers and specify his information interests independently for every provider
- define the presentation structure of his electronic-newspaper or magazine
- be notified when information updates are available on issues of interest
- access the historical evolution of an article
- retain full anonymity when reading an article
- pay only for the articles he chooses to read
- hold and distribute articles to other readers without breaking copyright laws
- access, without having to pay again, articles he has already paid for
- publish new articles or commentaries embedding articles of other providers together with his own added value

The information provider (publisher of an electronic magazine, content, added value), on the other hand should be able to:

- produce on demand copies of electronic articles in a cost effective way
- easily transform existing electronic material into self contained electronic articles
- collect revenue from his electronic publishing activity and protect his intellectual property and copyrights against illegal access and use
- accommodate and customize marketing policies and article prices
- offer high service availability to information consumers

3 Enabling technology

The HyperNews design and implementation started in early 1996. The technolgy choices upon which HyperNews is based are strongly interdependent, rely heavily on the Java language, and are based on the state of the art technology available at the beginning of the project (second quarter of 1996). In this section we present the technology choices we made and describe how the current sate of the art will affect them in the forthcoming migration of the system to today's (mid 1998) technology.

The programing language

Very early in the project definition phase the Java language [8] [9] [10] [11] was beginning to establish itself as a choice language for rapid prototyping and platform independence. The whole implementation was carried out using the Java Development Kit Version 1.0.2 (JDK 1.0.2) together with early releases of the RMI (Remote Method Invocation) and Object Serialization packages.

With the currently available versions of the JDK (i.e., JDK 1.2 beta 3), most of the technology described bellow comes for "free", in the sense they are either part of the Java distribution or have been taken into account for easy integration within the Java environment. For example Remote Method Invocation and Object Serialization, JDBC [12], Security [13] [14] [15], Java Beans [16], Java Foundation Classes (JFC) [17] [18], Swing GUI Components [19], etc.

Currently we are working on the migration of HyperNews to JDK1.2 in order to take full advantage of these integrated features and drop on the way most of the external packages we discuss bellow.

The Agent Execution Platform

At the time the project started very few agent environments were available. *Mole* [20], developed at the University of Stuttgart IPVR, was one of the few existing environments that was fully implemented in Java. This was a strong advantage with respect to the time critical aspect of the project, portability and the limited man power. During the implementation, another agent platform was made available by Object Space: Voyager [22]. It is also fully implemented in Java and will be considered for further implementations of HyperNews. This needs however further investigation. Current release of Voyager is Version 2, beta 2.

The cryptographic package

Since the first release of the JDK did not include cryptographic or security packages, we adopted a free external cryptographic library implemented in Java. Namely Cryptix from Systemics Ltd. Version 1.1 [23] was used throughout the implementation due to it's availability at the start of the project. For efficiency reasons, this version used native libraries. Binary distributions of these libraries are available for most architectures such as Windows 95, Windows NT, Solaris, Linux and IRIX among the major ones. However this limitation disappears with release 2.2 and above which are fully implemented in Java. Current available release of Cryptix is version 3.0.3.

User interface issues

The limited set of high level widgets provided by the Abstract Window Toolkit (AWT) in the first release of the JDK was to some extent a limitation to the rapid development of the prototype in research conditions. This is the reason for which we choose to use an external widget package. Namely, Objective Blend [24] and Objective Grid for Java [25], from Stingray Software Ltd., were chosen for the project. These packages provide a useful set of pre-built classes for high level widgets such as trees, grids, tabs, etc.

From the point of view of information formatting and rendering, we decided to take advantage of existing technology. It would have been counter productive to build yet another formatting language and rendering tool. Thus choosing HTML as the formatting language and a plain Java-enabled Web browser as the interface for consuming the information appeared to be a sound choice.

Database issues

The Informix Illustra Object Oriented database was used in the prototype due to it's availability within the KryPict companion project of the MEDIA project. A Java package (ITG) was available for interfacing the database from within Java. This was used for integrating the database aspect in the prototype from the information providers' view point. Needless to mention that interfacing with any other database vendor is now a simple implementation issue thanks to the Java Database Connectivity (JDBC) [12]. As a matter of fact, our industrial partner, L'Hebdo in this project chose the Oracle database during the project.

4 Overview of HyperNews

The agent based HyperNews system is composed of three layers shown graphically in Figure 1. The lowest gray shaded layer is the *agent layer* representing the underlying agent infrastructure (Java Virtual Machine and agent execution platform Mole) over which the HyperNews system is implemented. The second layer, the *core environment*, is the basic building block of the HyperNews system. It enables the safe and secure distribution and exchange of electronic documents. The top layer, the *HyperNews application layer*, implements the various tools, applications and user interface bound to the HyperNews system. This layer has two components depending on the role played by the user, either as an information consumer or as an information provider. A detailed description of the HyperNews architecture can be found in [6] [7].

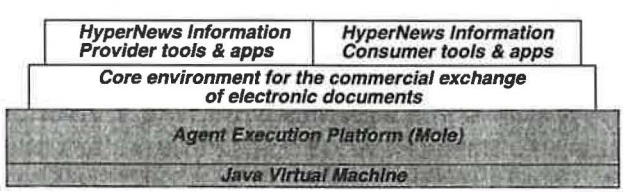

Figure 1 Layered representation of the hypernews architecture

5 The agent infrastructure

The Mole agent system [20] was chosen by the ASAP project to be extended in order to meet the security requirements identified in the MEDIA project. Namely to enforce authoritativeness and untampering of content during migration between locations. It has thus been used as the agent infrastructure over which HyperNews is implemented.

In the Mole system *Locations* represent the places where agents are created and run. Agents migrate between locations. All the locations together represent the agent system as a whole. This enables the necessary abstraction to represent the distributed agent system as a whole, spanning across a global network such as the Internet. Mole offers two types of agents: *User Agents* and *System Agents*.

User Agents are, from a privilege management point of view, *low privileged* agents. They are granted very limited rights to undertake any action in the system and have no rights to act outside the agent system unless they acquire prior authorization from higher privileged agents. User agents can communicate with other agents, migrate and create new user agents.

System Agents have more privileges than user agents. They are intended to be used as an interface to system resources outside the agent system (i.e., database, file system, etc.). These agents must be loaded at platform start-up time, they exist throughout the platform lifetime and can not migrate to other platforms.

Migration involves stopping an already running agent and sending it to the remote location to resume its execution. However Mole does not account for strong mobility but only for week mobility [26] in the sense that executing threads cannot be resumed remotely at the exact point they were stopped. From the point of view the HyperNews implementation this is not a important restriction but rather an advantage for stability and consistency of the migrating agents that need to reestablish consistent links to local resources upon migration, since the environments are likely to be very different.

Inter-agent communication in Mole can be achieved in two different ways: remote procedure call (RPC) and message passing. Remote procedure call is a synchronous communication mechanism which implements a method call to an agent either on a local or a remote location (Figure 2). Message passing can be both synchronous or asynchronous, and allows transfer of messages between agents (Figure 3).

```
Object[ ]       paramArray      = new Object[ #ofParameters ];
RPCMessage      rpcm            = null;
Object          result          = null;
...
//Initialization of the paramArray Objects
...
rpcm = new RPCMessage (     callerAgentName, callerLocation,
                            calleeAgentName, calleeLocation,
                            errorSemantics, RPCmethodID, parramArray);
result = (Object) getCurrentLocation().call(rpcm);
```

Figure 2 RPC call in Mole

```
String[ ]                paramArray     = new String[ #ofStringParameters ];
UnformattedMessage       msg            = null;
...
//Initialization of the paramArray Strings
...
msg = new UnformattedMessage(    callerAgentName, callerLocation,
                                 calleeAgentName, calleeLocation,
                                 errorSemantics, parramArray);
getCurrentLocation().message(m);
```

Figure 3 Message passing in Mole

The general behavior of a mobile agent can be described the following way. Mobile agents are dynamically loaded on the local platform. Then depending on their "behavior" they can die or migrate either on their own behalf or be forced to do so. Agents are characterized by a unique agent name (8 integer values) and a description string. Upon arrival of an agent on a location an init() and a start() methods are called. Before leaving a location, a stop() method is called. For further details on the Mole agent system, we refer the reader to the following articles which provide in-depth description [20][21].

6 The core environment

In order to handle efficiently the security requirements of the system the core environment is decomposed into three main areas holding agents, according to the privileges and roles of the agents: the *entry-point area*, the *restricted area* and the *system area*. The general idea of this decomposition is that any incoming or outgoing agent must first pass through an entry-point area before it is granted either access rights or clearance for leaving. Once a foreign agent has been granted access rights, it is relocated in the restricted area. In the restricted area an agent is not allowed to initiate any action (i.e. not allowed to execute) without having acquired prior authorization from an agent of the system area. Foreign agents can never reside into the system area. Finally, all user interaction and local system resources are managed by agents residing in the system area. Figure 4 illustrates the complete architecture of the core environment with the agents of the HyperNews specific application. In this section we describe the implementation of the core environment used by HyperNews. A more detailed description of the architecture and it's requirements can be found in [5][6][7].

The core environment was designed towards a framework for commercial distribution and exchange of electronic documents over open networks such as the Internet. Thus, the necessary abstraction to turn this work into such a framework is provided by our architecture. In the scope of this prototype, some of these features have been intentionally "hacked" due to resource limitations.

6.1 The system area

The main characteristic of agents belonging to this area is that they are system agents and as such they have access to system resources outside the platform and hypernews resources on the plat-

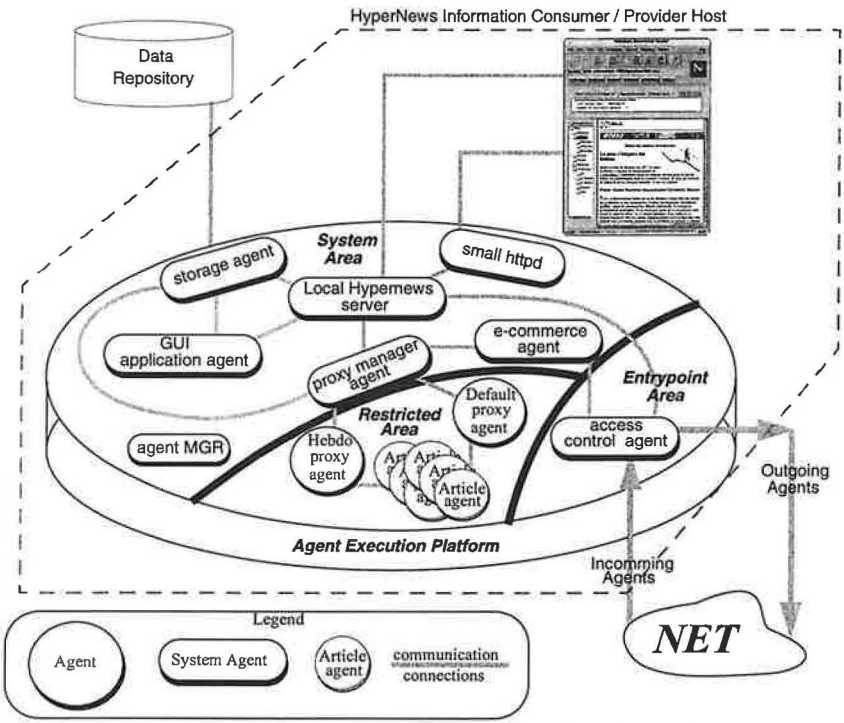

Figure 4 The architecture of the HyperNews system based on agents

form. Moreover, due to their nature (i.e., system agents) they can not migrate. The system agents that lie in this area represent the core-elements of the hypernews system.

The agent-manager agent

The agent-manager agent is responsible for the management of the whole agent execution platform and the monitoring of all the agents that evolve on the location it controls. It provides the core environment with a standard interface to both platform and agent management and monitoring. It is the first system agent to be loaded on the platform and it's initial task is to make sure that the HyperNews system is correctly loaded and started. It does so by managing a catalog referencing all the agents of the platform. Each agent that is loaded must register itself for services it can provide. Similarly, agents must unregister services they provide before being removed from the platform. To do so, each agent implements the following two methods goUp() and shutdown() which are called respectively upon platform start-up and removal.

Throughout the existence of the agent execution platform, the agent-manager is also in charge of what we have called a "*janitor*" service. The janitor is responsible of checking, at regular time intervals, that all the agents evolving on the platform are not idle (i.e., doing something useful). This is inherent to the agent paradigm where an agent platform holds agents in a similar

way a community has members. A member or agent in our case can be apparently idle but it is actually waiting for an event. A user agent must at all time maintain an "activity state" information on whether it is active (i.e., doing something or waiting for some event) or not. In case the user agent is idle and has no further activity, the janitor will ask the agent to remove itself from the platform. If this is not successful, the janitor will then simply kill the reluctant user agent without further notice. This scheme has the advantage of keeping the number of simultaneous active agents on the platform (i.e., load) as low as possible. Theoretically, such situations should not occur because each user agent is associated with an idle activity timer. Thus, if a time-out occurs for a user agent and it is idle, it will remove itself from the platform after completing some necessary cleanup actions. This issue will also be further discussed later in this paper.

The GUI application agent

The GUI application agent is responsible of everything dealing with graphical user interfaces. It embodies the necessary abstraction to acquire and release GUI resources such as windows in order to be able to instantiate application specific controls. In the HyperNews prototype, this agent is in charge of the application's main control panel, the interactive article packager GUI, the database GUI and the wallet GUI.

Moreover, this agent is also in charge of managing the HyperNews reader which is a java enabled Web browser such as Netscape for example. In fact, the environment distribution includes a version of Netscape 3.0, but the user has the choice of using his own browser. It is also anticipated to use the HotJava been Web browser in order to enforce consistency and stability of the browser and the system.

The storage agent

The storage agent is a general purpose interface to storage subsystems such as the Illustra database in our case or the file system. It implements methods to read or write a file from the file system and to execute database queries.

The electronic-commerce agent

The electronic-commerce agent is intended to be used as a general purpose agent responsible of implementing the methods required in order to accommodate the commercial document distribution scheme designed within the project [4].

It is anticipated to use the electronic-commerce agent as a place holder for wide spread electronic-commerce systems and protocols. It was not the purpose of this work to devise yet another (micro)payment protocol or system but rather to be able to accommodate currently available payment systems be they cash, credit or debit type. Ultimately, the user is expected to use different commercial electronic payment systems in connection to the financial institutions he collaborates with. It would be unreasonable to force the user into proprietary business models. Rather the user should be left to choose and express his own preferences according to personal needs and situations through an open architecture (electronic commerce framework).

In the scope of this prototype, a fake cash-like implementation is provided where users are expected to withdraw amounts from different information providers or credit institutions. This

cash can be spent anywhere among the HyperNews information providers. When a user requests to access an article, the required amount of the transaction is "reserved" from the users electronic wallet until the transaction is successfully completed. At this point, the user holds a transaction receipt and the article content is displayed. The transaction is committed and user can see he has been debited with the corresponding amount in his wallet.

The small-httpd agent

The small-httpd agent implements an http daemon for the sole sake of the interaction between the user's HyperNews reader (any Java enabled Web browser) and the HyperNews environment. This server implements a multi-threaded http server that will only accept http requests from the local host for security reasons. It listens for requests on port 8088 of the local host. The URLs have the form: http://localhost:8088/hypernews-request, where hypernews-request identifies the request to be processed by the http daemon in order to undertake the corresponding action. There are five different request types:

- *Applet commands* used to communicate between navigation applets (i.e., context selector, sections selector) and the HyperNews environment. These requests are used for example to set the UDP ports, acquire the contexts and their sections.
- *HyperNews reader environment requests* used to acquire the applets themselves and related resources such as images, widgets and other Java classes, etc.
- Requests with the ".html" suffix used for *container page requests* holding both the context and the section identifiers.
- Requests with the ".hna" suffix used for *HyperNews article requests* holding the article agent identifier.
- Other requests are considered as article embedded elements such as an image for example. A cookie is used to uniquely identify the corresponding article agent to which the request is to be passed.

The local-HyperNews-server agent

This agent plays a central role in the architecture acting as a relayer of most application level requests. It is in charge of dispatching the requests it receives to their correct destinations and also to format container pages according to the received results. It is also in charge of processing application level requests coming from the outside, like for example when receiving an update request from an information consumer, a request for a specific article, etc.

The proxy-manager agent

This agent is responsible of managing the different information provider proxy agents. It also determines the right proxy to contact for relaying requests it receives. If a proxy of a given information provider can not be found locally the request is passed to the Default Proxy which is the general purpose proxy able to handle the requests.

6.2 The entry-point area

Although security is taken into account at the agent platform level, a second level of security needs to be introduced from the HyperNews system point of view to enforce application level security policies. The core system provides mechanisms to master and control agent traffic between the platform and the outside world. This is done by the entry-point area. The role of this area is to be a placeholder of incoming and outgoing agents before they acquire authorization for either entering or leaving the platform. In addition, this area holds a unique system agent: the access-control agent.

The access-control agent

The access-control agent is the equivalent of a fire-wall providing migration authorization to outgoing agents and credential and authorization control of incoming agents. The first action an agent must undertake when arriving on an agent execution platform is to contact the access-control agent with proper identification. After being checked according to access control rights, the agent is either granted access rights and thus relocated in the restricted area or simply killed in case access was denied. Similarly, among the last actions an agent must undertake before leaving an agent platform is to acquire authorization for migrating (i.e., leave the agent platform).

The access-control agent also takes care of immediately processing key management agents such as requests for session keys and public keys. The reason for this is that there is no need to host a foreign agent for such requests. They are thus immediately answered and the agent can leave without further notice. The issues on key management will be presented in further details in the next section.

An agent at any given time belongs to an area of the core environment (as described at the beginning of this section) depending on the rights it has been granted. The types of area restriction are:

SYSTEM_AREA

This area restriction is never granted dynamically by the core environment for security reasons. It is the default area restriction of any system agent.

RESTRICTED_AREA

This area restriction type is granted by the access-control agent to any foreign agent after it has been identified and registered on the agent platform. From this point on, a foreign agent with this area restriction will be able to interact with the system in order to complete the task it was assigned.

ENTRYPOINT_AREA

This area type is assigned to any incoming or outgoing agent before requesting the right to stay or the right to leave the agent platform.

DEADZONE_AREA

This is the default area type assigned to any agent before it is granted any kind of right on the agent platform or just before leaving an agent platform.

6.3 The restricted area

The restricted area is the placeholder for all the foreign user agents when granted access rights exception made for the Default Proxy Agent which is a user agent bound to the local agent execution platform (i.e., it is neither mobile nor foreign). The user agents have restricted rights and most of the actions they undertake have to be accepted or processed by system agents. A common characteristic of the user agents is that they all reside in the restricted area since they are considered foreign agents.

Following is an exhaustive enumeration and brief description of the user agents as currently implemented in the prototype. Some of which will be further described later in this paper. There are five "mobile" HyperNews agents which are sub classes of the UserAgent class. Each agent is assigned a type according to the specific role it is to have. Namely:

- the *BillingAgent* used for all commerce transactions such as *withdraw*, *payment* and *receipts*.
- the *HyperNews Request Agent* (*HNRequestAgent*) used as a general purpose request-reply transaction agent such as *public key request*, *session key request*, *article request*, *information request* and *fragmented information* replies for large objects, *general request*, etc.
- the *HyperNews Article Agent* (*HNAA*) is the actual agent wrapper of an article. It can be a plain *article*, a *container* for a set of articles or both an *article and a container* thus allowing for multiple levels of added value.
- the *Default Proxy* and the *Information Provider Proxy*. The former is the general purpose interface to HyperNews articles, while the latter implements the information provider's specific behaviors to achieve fine grained policy dependent issues such as special pricing policies, frequent buyer advantages, value added services, etc. The prototype does not currently implement an example of this information provider proxy agent. Moreover, the Default Proxy Agent is also a UserAgent but it is not mobile. It is started at platform launch time and is intended to be the default information provider proxy when such a proxy is not available locally. This agent is described in further details in the next section
- the *Proxy Update agent* to update information provider Proxy Agents. This agent has not been implemented yet but is part of the design and will be implemented in the next release.

6.4 The commercial electronic Document Distribution Model

In the scope of the HyperNews project, we have devised a scheme for the commercial distribution of electronic documents [4] that satisfies the defined security and distribution requirements. It is based on public key encryption and requires a trusted third party between the information consumers and providers which may be for example a credit institution or a bank. Both parties trust the credit institution to authorize the unlocking of the article against payment from the information consumer which is credited to the information providers account. Upon successful payment to the credit institution, the article key is released and a receipt is given to the informa-

tion consumer for subsequent access. This receipt is issued only for the information consumer that purchased the article. Thus the receipt is *nominative*. However this can also be bound to what ever commercial policy the providers wish to use. A general overview of the model is given in Figure 5.

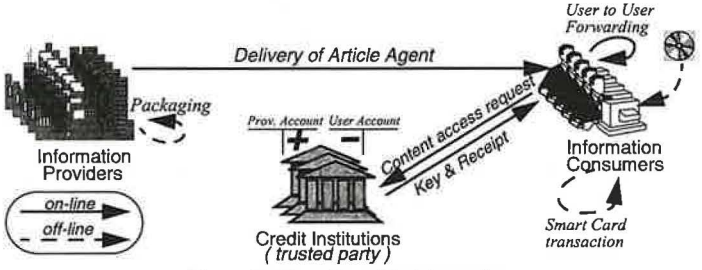

Figure 5 HyperNews Model overview

The operational overview of this scheme and it's implementation is described below. There are basically two aspects to it: packaging and usage.

The *packaging* consists of preparing the electronic document agent with it's encrypted content and attributes for full public distribution. The content, considered as a binary large object (BLOB), is encrypted with a symmetric key (K). This key is itself encrypted with the public key of the accredited credit institution. This is done as many times as the number of credit institutions the information provider is willing to support. An article information string (AIS) is added to the agent providing the necessary public (i.e., free) information about the content such as title, authors, price, abstract, etc. Finally, both the encrypted key and the AIS are signed by the information provider with his private key. This process is shown graphically in Figure 6.

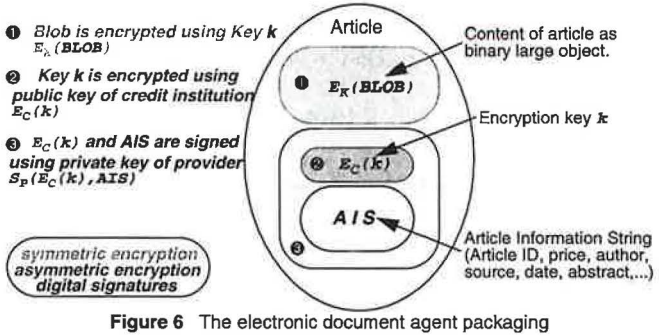

Figure 6 The electronic document agent packaging

The *usage* defines the process of both the access and the subsequent accesses to the content (i.e., unlocking). In both cases there are two steps to cover. The first step is to acquire a session key for the further secure communication with the accredited entity (i.e., credit institution or alike) which will process the access request and release the document key. The second step is the actual content access request and acquisition of the article key and receipt as shown in Figure 7 and Figure 8. The request is formed by extracting from the article agent the encrypted key cor-

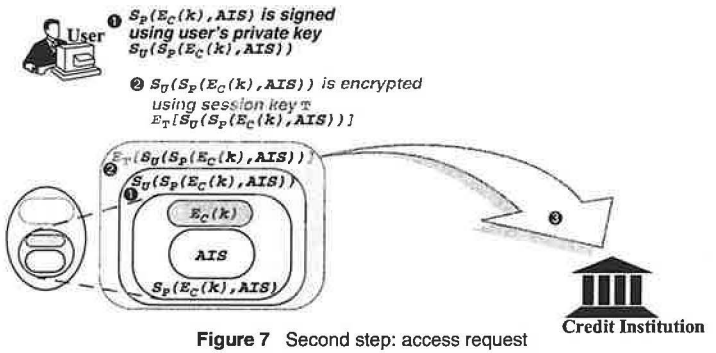

Figure 7 Second step: access request

responding to the credit institution and the article information string. This is then signed by the user and the result is encrypted using the session key previously acquired in the first step. Upon receiving such a message, the credit institution will be able to decrypt it knowing the previously issued session key, verify the signatures of both the user and the provider and thus reveal the AIS and the encrypted key. At this point, the billing occurs and if it is successful, the article key K will be decrypted with the private key of the credit institution and a signed receipt generated for this transaction. Finally, the article key and the signed receipt are encrypted with the session key and the result sent back to the user (i.e., to the article agent responsible for releasing its content).

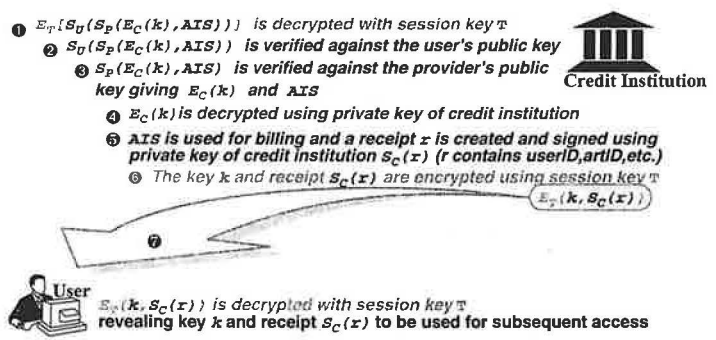

Figure 8 Second step: article key and receipt

Subsequent access to the article content is done exactly the same way except that a receipt is appended to the request of the second step. Upon successful verification of the receipt by the credit institution (i.e., verification of receipt issuer signature and user identification match between receipt holder and requestor), the article key will be returned to the user but no receipt needs to be returned unless a special commercial policy requires it.

6.5 Key management and acquisition

The only keys that need to be exchanged between the participants (i.e., information providers and consumers, and credit institutions) are their public keys and the session keys between the

information consumers and the credit institutions. For the time being, no use is made of certification authorities for public key acquisition. However this can be integrated easily in future implementations. The session key acquisition is secured by using asymmetric cryptography for encryption and signatures.

Every participating entity knows it's own private key. The credit institutions need to know the public keys of both the information providers and consumers, which is a reasonable assumption for a trusted third party. The information consumers need to know the public keys of the information providers and the credit institutions. However, the information providers only need to know the public key of the credit institutions. Finally, the information providers know the document/article symmetric key that served for content encryption, which is also a reasonable assumption since they own their content/added value. This is summarized in Figure 9. From a key management point of view, the major advantage of this document distribution scheme resides in the fact that there is no overhead for document key exchange or replication since the document key is encrypted with the public key of the accredited institutions and held by the article agent itself. Thus, even in case of information provider bankruptcy the content can still be accessed through one of the credit institutions.

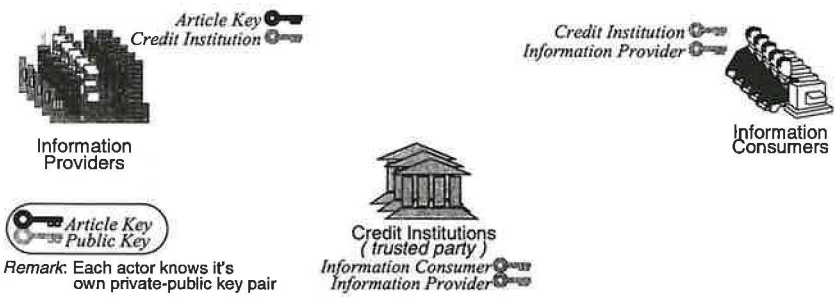

Figure 9 Key management: who knows what

6.6 The HyperNews Article Agent

The HyperNews article agent is a subclass of UserAgent implementing the MobileAgent interface allowing it to migrate between agent platforms and the TimerCallback interface for inactivity time-outs. Since the main idea behind the design of the article agent is to allow it's free distribution without copyright infringement (i.e., superdistribution [27]), the article content is stored encrypted together with the article encrypted key within the agent and the article agent is responsible for it's own security.

The HyperNews Article Agent is the Agent wrapper of the article content. It provides a set of instance variables:

- the article content stored in an instance of the Article class which is serialized, encrypted and stored in a byte array.
- the Article Information String (AIS class)
- the signed MD5 hash of the Article Information String

- the article key encrypted with the public key of the credit institutions (CIKey class). One per credit institution stored in an array.
- the signed MD5 hash of the all the CIKey objects stored in an array.
- a optional array of embedded HyperNews Article Agents thus providing a recursive encapsulation of different levels of added value. This way an article agent can become a container together it's own added value.
- state reflecting the agent type, the area restriction code, whether it is idle or not and information about the creation location, the previous location, the current location, source and destination locations when migrating.
- the action to undertake upon expiration of the article agent timer. Namely, save before being removed, simply remove, restart the timer or do-nothing.
- information whether a Provider Proxy exists locally or not and if the case arises, it's agent name.

together with the following interface:

- various constructors to be used according to the desired type of article agent (i.e., article, container or both).
- an initialization method init(), which is called automatically after migration or when reloaded on a local agent execution platform.
- a start method start(), which is automatically called right after the initialization method.
- a stop method stop(), automatically called to complete clean up actions before migration or local storage.
- the timer expiration callback method hasExpiered(), which is called automatically upon expiration of the article agent timer.
- the migration method moveTo(LocationName ln), which is called to send an article agent to a destination location.
- the standard Mole RPCMessage dispatcher dispatch(RPCMessage rpcm), returning a result of type Object. It calls two private methods of the article agent: openHNArt() and getEmbededFile(). In both cases the returned Object is an instance of a byte array of the content which is returned as the result of an http request to the browser by the http daemon system agent of the local environment. The first one is called when the user requests access to the article content. The second is called to retrieve the article embedded files of an article currently being displayed to the user.
- other conveniency methods to operate on the article agent attributes.

6.7 HyperNews protocols

From a protocol point of view at the HyperNews core environment level, communication is achieved with *two-state return agents*. Two-state in the sense that at any given time such agents are only in either of two states, namely outgoing for remote service request or returning after

service completion. In other words, agents are created at a given location with a specific mission to accomplish. The agents are then sent to their destination location where they will request the service(s) and hopefully return to their initial location holding the result(s). Upon completion, the agents are disposed. The term *hopefully* for the return of the agents was used intentionally because anything can happen to the agents during their journey to, from or even at the destination location. There is thus no guarantee regarding their return. For this reason, every such two-state return agents, prior to leaving, is associated with time-out mechanisms at the source location. That is, a semaphore-like object is created with a unique identifier to be used for notification upon return of the agents. It is combined with the time-out mechanism so that the waiting threads are notified accordingly and the resources relinquished.

There are three types of communicating parties: the information / service providers, the information / service consumers and the credit institutions which is trusted by both the providers and the consumers. It is to be noted that the roles of the consumer and provider can be combined for example in case of article forwarding among consumers, the sender then holds the role of a provider and reciprocally. The different types of agents that migrate among the communicating parties are summarized in Figure 10.

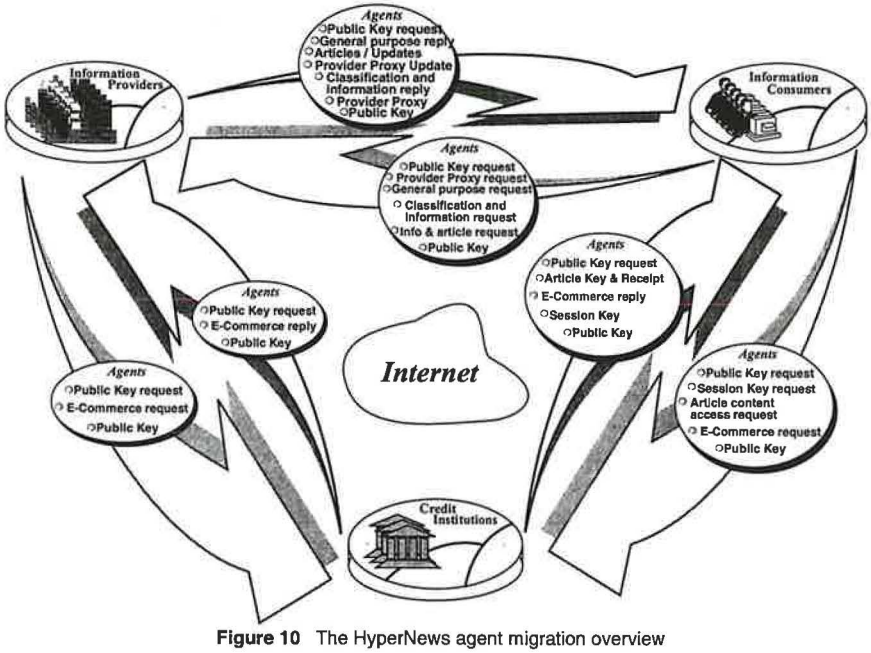

Figure 10 The HyperNews agent migration overview

7 The HyperNews layer

The HyperNews layer implements all the HyperNews application specific components and logic, which are reflected through user interfaces.

Both the information consumers and the information providers run the same environment with slight differences depending on their role. This is expressed by a HyperNews property defining whether a user is an information consumer, an information provider or eventually both. This is specially useful when an information consumer wants to be able to republish material he does not own thus becoming an information provider of his own added value together with other information provider's content. All this being totally legal since access to content is bound to successful payment or presentation of a valid proof of purchase (i.e., receipt). Based on these properties, the HyperNews environment at launch time will be tailored accordingly. The main HyperNews user interface in collapsed mode is shown in Figure 11.

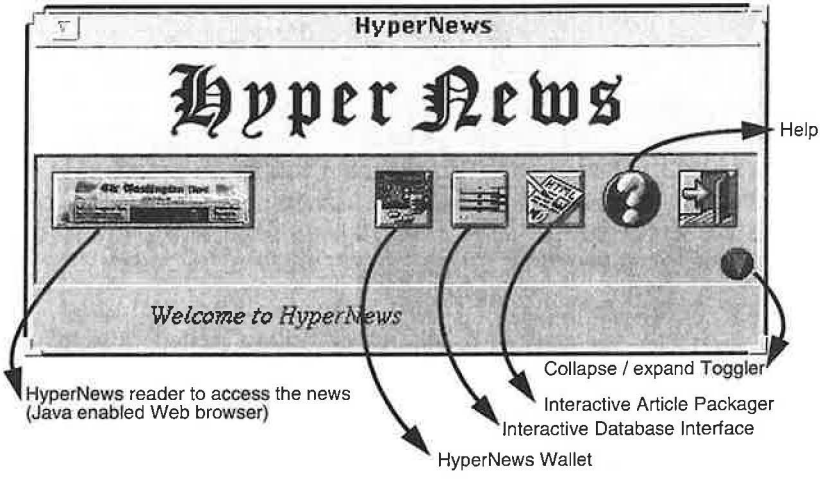

Figure 11 The main HyperNews user interface (collapsed)

The HyperNews environment can be tailored in a number of ways through various properties allowing the user to define in a persistent way his preferences such as the viewing mode, browser to use, filtering attributes, etc.

7.1 Information Consumer Tools

In order for the user to be able to define, customize and access his personal electronic newspapers (i.e., contexts) the following tools are provided:

- the *Context Manager*, to manage personal electronic newspapers
- the *Information Profiler*, to specify for each context both where the information is to come from and what is of interest to the reader
- the *Presentation Profiler*, to specify for each context the layout or in other words how the retrieved information is to be presented to the reader
- the *HyperNews Wallet*, to manage all the electronic commerce related tools.
- the *HyperNews reader*, to access the hypermedia electronic newspaper system

7.1.1 The Context Manager

The context manager allows the user to manage his personal electronic newspapers or information contexts in a similar way to a "news stand". With this tool the user can create, remove and edit his information contexts. These are stored on the users host in a specific directory of the HyperNews system. Throughout the implementation, contexts are represented graphically as "tabs" or "tabed panels". An example of the context manager with three information contexts (projects, private and work) is shown in Figure 12.

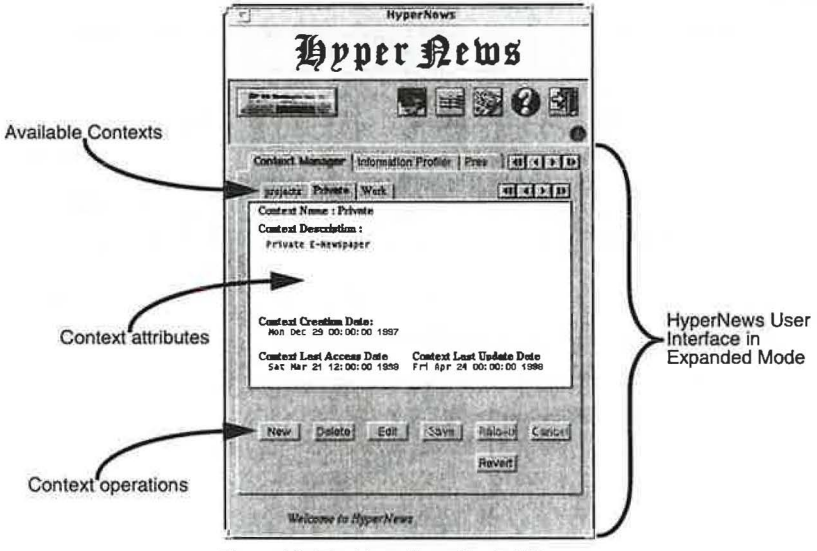

Figure 12 The HyperNews Context Manager

7.1.2 The Information Profiler

Once the user has defined one or more contexts, for each one of them he will have to specify his interests using the information profiler. This is done in two steps. First by selecting among all the available information providers the ones from which information is requested. Second, for each requested information provider by selecting from the provider's classification the elements he is interested in.

An example of the information profiler for the sample "work" context is shown in Figure 13. In this example all three available information providers are selected (left hand side). The selection is identified by a "tic" over the folder image. On the right hand side, the whole classification of the currently selected information provider (L'Hebdo) is displayed and two classification items are currently selected (Laboratoire and Editorial). The selection of an information provider on the left hand side triggers the display of it's corresponding classification on the right hand side from which the user may select the desired items he is interested in.

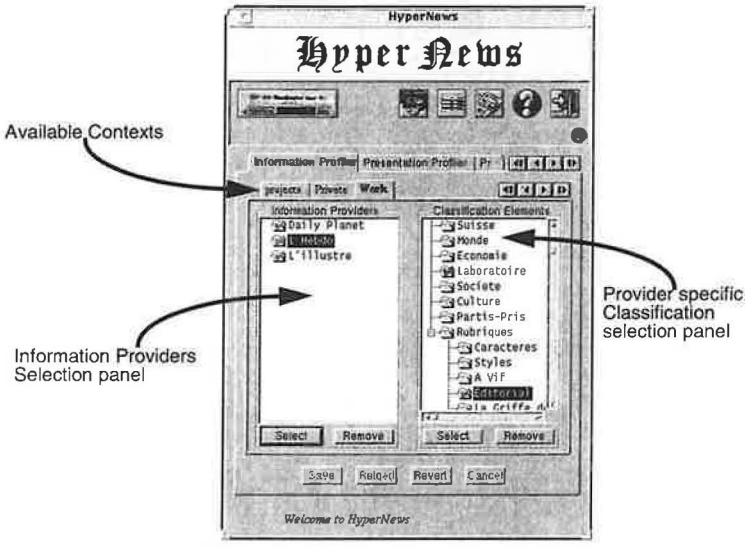

Figure 13 The HyperNews Information Profiler

7.1.3 The Presentation Profiler

The last step to fully define a users context is achieved with the presentation profiler. For each information context, the user defines the layout (i.e., structure) of his personal electronic newspapers and inserts the previously selected classification items from the requested information providers into this structure.

An example of the presentation profiler for the "work" context is shown in Figure 14. In this example, the top panel (Selected Classification) shows for each chosen information provider the selected classification items available. These items are available for insertion in the user defined structure of the bottom panel (Sections and Content). The bottom panel shows the "root" page of the work context in which the user chose to put all the "Editorials" of the selected information providers. As a result, when the user will access this page he will be presented only articles that match these classification items. Namely all the editorials of the selected information providers.

7.1.4 The HyperNews Wallet

This tool is the user's interface to the electronic commerce components. In the current implementation of the HyperNews prototype, it only accounts for "fake" cash like transactions. However other type of payment systems and protocols have been anticipated be they cash, debit or credit. Ultimately, the user should be able to accommodate for any third party electronic commerce system in a transparent way and express his preferences in order for the system to take these into account in the commercial transactions. This tool is also the user's interface to receipt management. As shown in Figure 15, the user can have access to all the receipts currently owned.

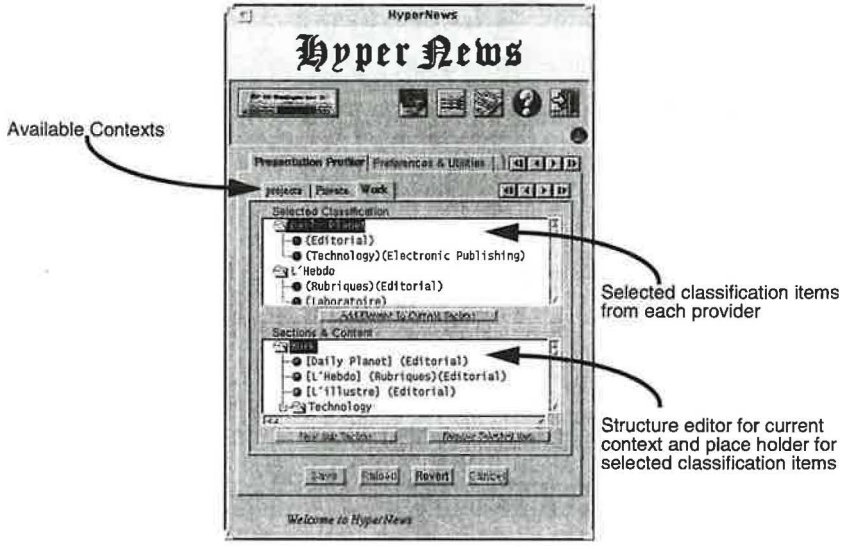

Figure 14 The HyperNews Presentation Profiler

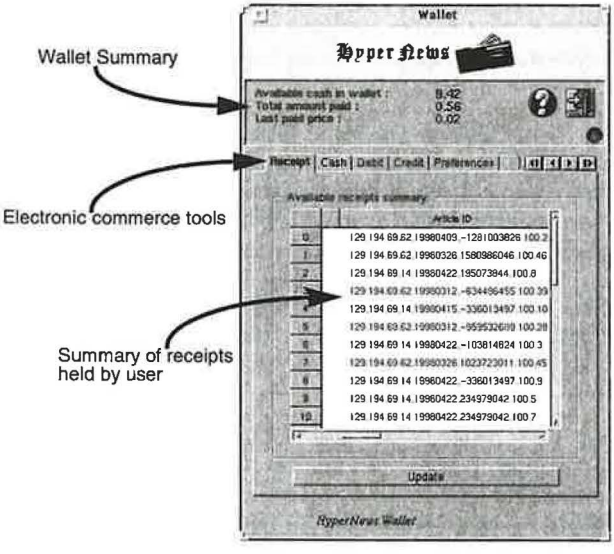

Figure 15 The HyperNews Wallet (receipt summary)

7.1.5 The HyperNews Reader

Finally, the user accesses his personal electronic newspapers through the HyperNews reader which can be any Java enabled Web browser. A default version of netscape 3.0 is included in the HyperNews distribution The basic structure of the interface is composed of three elements each of which is in a separate frame. The first two hold applets whereas the last one is a simple frame in which resulting data is displayed in HTML. Communication between the applets is implemented using a the UDP protocol. Each applet implements a UDP server that interprets the received messaged and take actions accordingly. This is also used by the HyperNews environment to communicate with these applets when needed. The http protocol is first used to exchange information such as the port number between the applets and the environment. Following is a description of these three elements:

- The *context selector*: is a Java applet in the top horizontal frame. It allows the user to skip through his contexts (i.e., electronic newspapers) using the Tabs labeled with the context names. Each context panel displays summary information available such as the last time the context was accessed, the number of available updates, etc. It also provides a button labeled "Update Now" allowing the user to trigger an update process for the given context. Selecting a context via a Tab triggers a change in the sections selector to display the corresponding sections of the newly selected context. This is achieved by sending a message (UDP datagram) to the section selector applet in order for it to retrieve the sections of this new context.

- The *sections selector*: is also a Java applet in the bottom left vertical frame. It shows at all times the sections of the currently selected context in a tree structure. The selection of a section triggers an http request to the HyperNews system. It's result is redirected in the viewing area. Such a request corresponds to requesting the summaries (as desired by the user) of all articles available locally for the current section of the current context filtered according to the users preferences.

- The *viewing area*: the remaining frame (bottom right frame), shows any of the following depending on the user's action:

  — *article summary container page*: showing all the HyperNews articles that match the selected section of the current context. Matching criteria can be customized according to user preferences (e.g., all the articles, only new or old articles since or until last update, already read or unread articles, articles published before and / or after given dates). For each article summary, the user can choose which attributes should be displayed (e.g., source information, image of logo if available, authors, abstract, publish date, expiry date, price, receipt existence and options such as article annotations, send or forward article, delete article). The article title is always shown as it is the hyper-link to access it's content. The retrieval of article summaries is initiated by the user when selecting a section of the current active context in the sections selector applet. An example is shown in Figure 16.

  — *a HyperNews article*: is shown when the user has requested to access it's content. Upon successful payment or receipt verification, the article content is displayed in

the viewing area. HyperNews hyperlinks are identified by an "HN" chain logo. When selecting such links the HyperNews system will propose the corresponding summary information to the user for acceptance. If the article can not be found locally, it will be fetched from it's source. The other hyperlinks are normal Web links that can be browsed in a transparent way by the user. An example of a HyperNews article is shown in Figure 17.

— *a plain Web page*: can be accessed directly from any HyperNews article containing plain Web URLs to access external Web based referenced material. This is an advantage of using standard viewing and rendering tools.

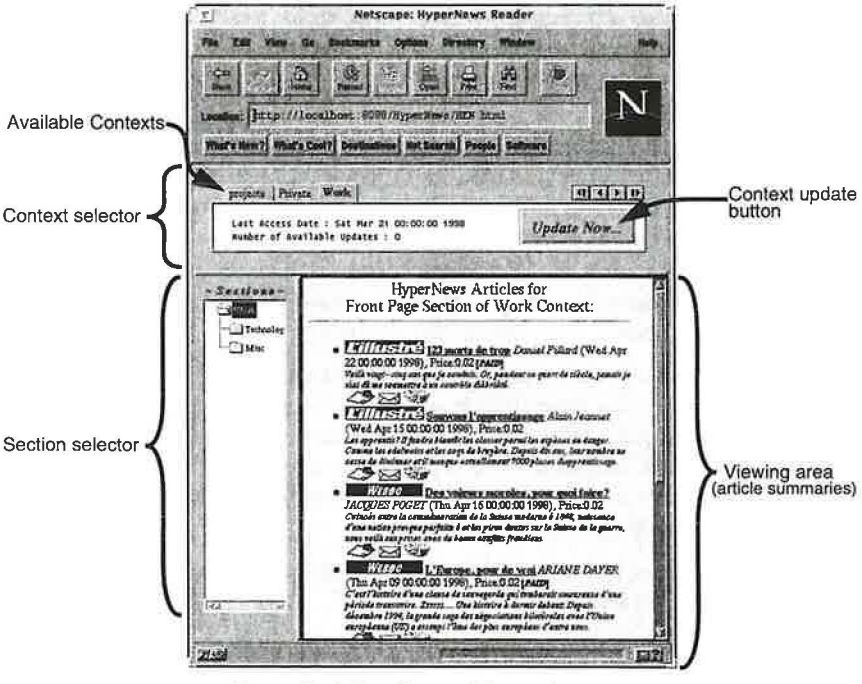

Figure 16 A HyperNews article container page

7.2 Information Provider Tools

In order to publish content, information providers need tools for creating the secure agent based HyperNews articles together with their attributes. From this point of view, an important requirements was the ability for the provider to reuse existing material in electronic form such as ready made HTML formatted Web pages. Two ways of packaging these articles either interactively or in batch mode are provided. Both ways rely on the http protocol for retrieving the content as well as it's embedded elements such as images, applets, etc. This offers the advantage of a unique way of accessing the "raw" data through a URL (i.e., http, ftp, file, etc.). The structure of the HyperNews article is discussed in further details later in this section.

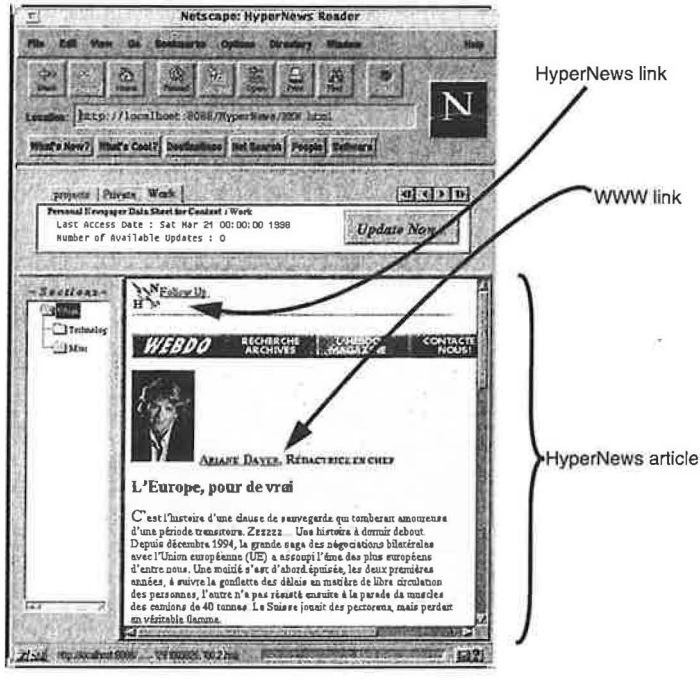

Figure 17 A HyperNews article example

7.2.1 The Interactive HyperNews Article Packager

This tool allows any user to package interactively a HyperNews article. There are four steps to achieve this:

1. First, the article content must be acquired through a URL of where the data can be retrieved from. An example of this step is shown in Figure 18. The top group is used to set the URL of the source, retrieve it and parse it's content in order to acquire the embedded elements as described in the resulting group at the bottom showing a summary of the article's content. In this example, four embedded images were identified and retrieved. The content of the article itself is stored in the file named "index.html" with a total of 5 files and 32 Kilobytes.

2. The second step consists of defining the article's properties such as title, author, abstract, price, primary and alternate classification, publish date, expiry date, volume and issue number, keywords, etc. At this point, a unique article identification number will be assigned to the article based on a subset of these attributes and a sequence number thus guaranteeing uniqueness of the article ID.

3. The third step provides a way to define whether the article is to be encrypted or not. In case encryption is requested, the article publisher will be able to define which credit in-

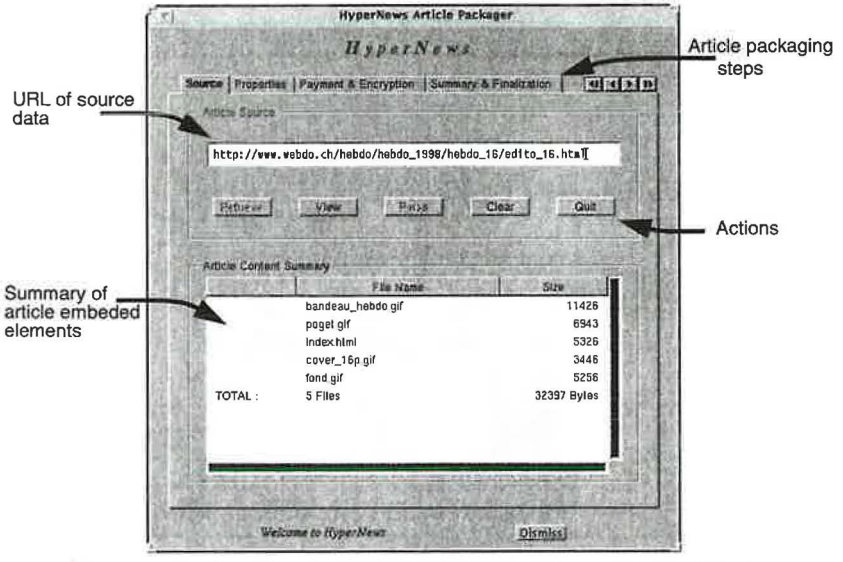

Figure 18 The HyperNews Interactive Article Packager (source data acquisition)

stitutions are to be supported for processing a payment for this article and whether the information provider himself is to be included as potential payment processor. This last option can be very useful for allowing internal processing of article access inside the providers domain for example or if a user has a special agreement with the provider and does not require anonymity.

4. Finally, the last step towards publishing consists of verifying the summary and finalizing the publication of the article. At this point the article will be checked for consistency and published thus becoming available for retrieval and dissemination.

7.2.2 The Batch HyperNews Article Packager

The previous interactive tool is specially useful for information consumers and sporadic publishing of individual articles or also when articles are published on irregular time intervals directly by the authors themselves. However it is most likely that such an editorial process will be discussed prior to committing the publishing process of a set of articles. In this case it would be unrealistic to require such an interactive process for each and very article but rather have them all published in batch mode. For this reason an other tool was included in the set of information provider tools: the Batch HyperNews Article Packager. This tool, shown in Figure 19, allows an information provider to do exactly this by providing a file in a specific format holding for each article the required information allowing it to be packaged into a HyperNews article agent. This is done in two steps: first, the file is read and checked for consistency and integrity (left image of Figure 19). Article agent names are allocated in advance thus allowing cross-referencing of HyperNews articles within the file. Second, the user after checking the results of the first step in

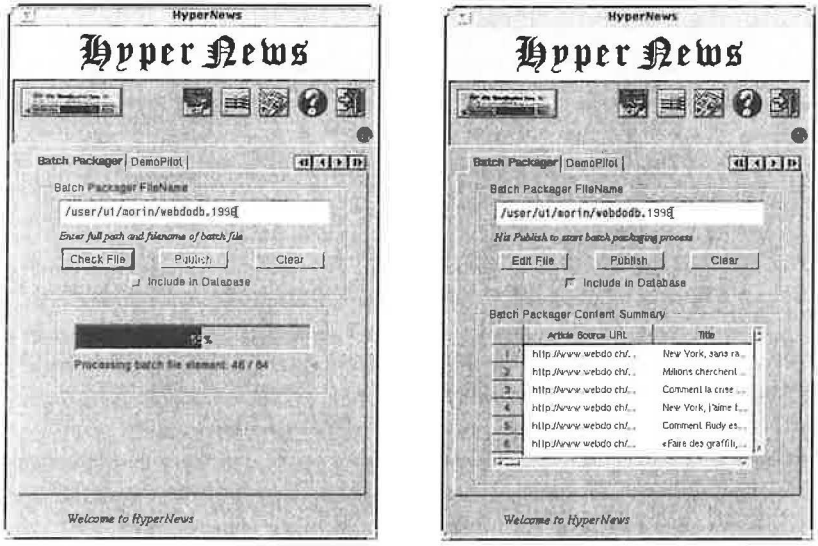

Figure 19 The HyperNews Batch Article Packager

a table can commit the publishing of all the articles and also include them in the database if desired (right image of Figure 19).

8 Security issues

Security is of paramount importance in an infrastructure such as HyperNews specially when it comes to issues related to copyright, intellectual property rights and mobile code migrating over open networks between a priori untrusted locations. Reason for which such an infrastructure must enforce authoritativeness, trust among communicating parties, untampering of content during migration and prevent copyright infringements through encryption.

The RSA [28][29] public key algorithm (512-bit) is used for asymmetric cryptography. It is used for both encryption and digital signatures. The MD5 [30] one-way hash function was used to compute message digests of transferred objects.

Like for the article content, we choose to use the IDEA [31] symmetric block cipher for the session keys. The keys are 128 bits long and their validity in time can range from unique usage (i.e., use once and throw away) to a given time interval based on the providers policy with respect to this matter.

In the current implementation the deciphering of the article content is achieved by the article agent itself upon receiving the article key from the credit institution when successfully processed. The key is kept in memory only for the time needed to complete the decryption and immediately garbage collected afterwards. This way, the key is neither stored in the users environment nor revealed to other agents but the one it concerns. The arguments such as a memory

dump and other similar attacks stay valid however scrambled memory techniques and hardware could be used to address such issues. Moreover, the article key being unique to a single article even in case of a compromised key this would only give access to one specific article. And such article having properties of low value and short life time makes it even less attractive to break.

An other issue of critical importance is the acquisition of the public keys over open networks. Reason for which in future implementations the trust chain could be enforced through certification authorities such as VeriSign's digital IDs or the ISO authentication framework X509 protocols [32]. From this point on, security is enforced at all levels of the architecture: agent environment, core environment and HyperNews application through techniques mentioned above.

9 Assessment and validation

In order to assess and evaluate the achievements of the HyperNews project, this section briefly presents some quantitative results and summarizes a public demonstration held at Computer'98 in Lausanne in cooperation with our industrial partner L'Hebdo.

9.1 Quantitative results

For the demonstration, we have used a sample of articles taken from all the issues of L'Hebdo between January and April 1998 (i.e., 16 issues with a total of 228 articles). The overhead of the agent encapsulation of an article is roughly averaged to a constant value of 1.5 Kilobytes. As shown in table 1, examples are provided for the smallest and the biggest article. The average values of the whole sample are also provided.

	Article Object Size (bytes)	*Article Agent Size (bytes)*	*Overhead (bytes)*	*Overhead (%)*
Smallest Article	677	2'106	1429	211.08
Biggest Article	180'399	181'801	1402	0.78
Average of the sample (228 art.)	42'262	43'723	1460	3.46

Table 1: HyperNews Article Agent Overhead

There is also the question of the cost of accessing the content which for the time being requires one to two network accesses depending on the life time of the session key and the decryption of this content. Measurements have not been made yet but will be thoroughly addressed during the trial phase of the project. In the scope of the demonstration, these issues seamed reasonable but are inherently bound to the network load for data transfer and to the processing power of the host for the decryption.

9.2 Public demonstration

The Computer'98 exhibition was held for it's 20th edition at Palais de Beaulieu, Lausanne, from April 28 to May 1, 1998. It is the major computer and information technology event and fair of the french part of Switzerland.

L'Hebdo magazine, member of the Ringier publishing group, has been very active in the field of Web based content publishing since 1995 both as an on-line supplement to the paper based magazine as well as selected excerpts of the magazine published on their Web server. They have undertaken not less than five different versions of their Web site (Webdo), reaching now over one million hits for the first quarter of 1998. Webdo has now acquired it's autonomy as a virtual full edition starting this fall. Early 1996, contacts were established between the Object Systems Group of the University of Geneva - CUI, and L'Hebdo to study the opportunities of content commercialization over open networks such as the Internet while enforcing copyright control. These initial discussion led to the HyperNews project supported by the Swiss Priority Program for Information and Communication Structures (SPP-ICS 5003-45333), launched in June 1996, where L'Hebdo is our industrial partner.

In the scope of this collaboration, L'Hebdo has hosted the HyperNews team on their stand at Computer'98 to demonstrate the HyperNews prototype. Announcement of this event was published in L'Hebdo magazine of April 16, 1998 and an article presenting the project [33] appeared in the Webdo Mag, issue number 4, April 1998 distributed freely throughout the exhibition (also distributed as a supplement of L'Hebdo magazine of April 23, 1998). Moreover for the occasion a project description flyer was kindly printed by L'Hebdo for distribution on the stand during the exhibition.

9.2.1 Demonstration scenario

For the purpose of the demonstration and to illustrate the "cross information-source" aspect of HyperNews, three different information sources were used. Namely: L'Hebdo for which all the articles of 1998 were available as HyperNews articles, L'illustre, another magazine published by the Ringier publishing group for which several articles were also available, and a third "fake" information provider, named D.P. was used for the sake of the demonstration. All three information providers were running remotely on different hosts at the University of Geneva, respectively on a Sparc Ultra 2, a Sparc Ultra 1 and a Sparc station 4.

From the point of view of the information consumers, three different users were available. The first one (User 1), a Sparc Ultra 1, running on the site of the demonstration. A second one (User 2) was running remotely on a host in GM D (Bonne) and a third one (User 3) was hosted by the University of Geneva. This configuration is summarized in Figure 20.

The demonstrator allowed for full customization of the environment. Namely, to define new contexts together with their corresponding information and presentation profiles.

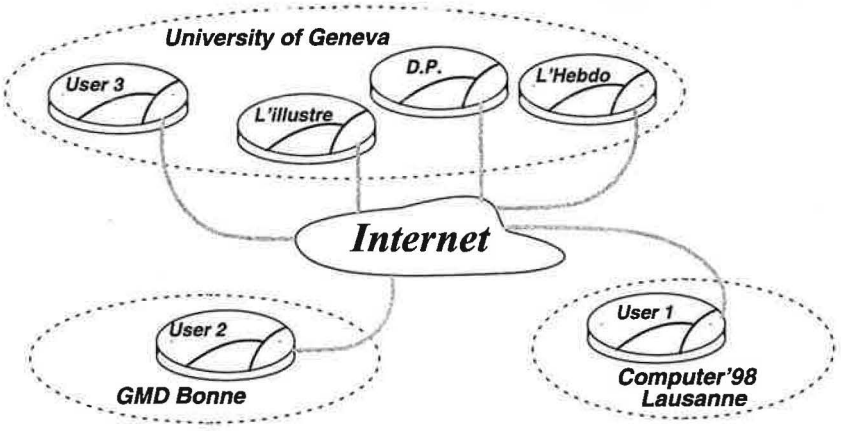

Figure 20 HyperNews demonstration configuration map

9.2.2 Feedback and Reactions

These four days of demonstration have been very successful. Many demonstrations of the HyperNews prototype were given throughout the Computer'98 event. Generally speaking, the most recurring comments and feedback received were the following:

- Very strong interest regarding the issues addressed by the HyperNews project
- Very high relevance of the project towards commercially viable value added services
- Strong potential of the approach (standard) bound however to wide spread, general acceptance (i.e., it is unrealistic to use such a service only for one or two information providers).
- Very good understanding and acceptance of the fact that our approach is not in competition with "traditional Web/Internet content and services" but rather in the field of trusted and copyrighted value added services for which a fee could be required depending on the commercial policy of their providers.
- From potential users point of view, this is exactly the type of service they have been waiting for ever since they started "surfing" on the Internet in order to cope with all this overwhelming information as well as the burden of navigating and managing as many passwords and accounts as the number of information sources they are interested in. However, here again it is very unlikely that a user would download and install such an environment only to access a couple of information sources.
- Finally, from people who were familiar with the project, either through technical reports or discussions, seeing the resulting prototype made everything clear about the project and it's objectives.

In summary, this opportunity we have been given to demonstrate the results of our work through a demonstrator within a public event was very useful both to collect feedback, reactions and to gain visibility for the project and the addressed issues.

9.3 Other dissemination means

Following the Computer'98 exhibition, a radio interview was made about the HyperNews project in the "Village Global" [34] program of the Radio Suisse Romande La Premiere on May 16, 1998. The roughly 15 minutes interview was a very general description of the HyperNews project presenting the issues and goals of the project targeted to a non technical audience.

Similarly, a newspaper article about the project was published in the June 10, 1998 issue of the Tribune de Genève [35], in a column dedicated to research issues and achievements of Ph.D. candidats at the University of Geneva.

10 Conclusion

All together, the implementation of the HyperNews prototype took about six month at a high pace. It represents over thirty thousand lines of source code in Java produced by a single programmer.

The next major step is to upgrade HyperNews to JDK 1.2 and to test thoroughly the prototype under Windows 95/98/NT. Then, a self instal distribution of the prototype environment needs to be built before going any further. Testing of HyperNews in real world conditions is planed for this year and will be done in cooperation with L'Hebdo through a selected group of users of Webdo. This will provide the necessary feedback from both information consumers and providers.

Our work within the HyperNews project will continue and address two major issues. Namely, off-line operation, via smart card technology, in order to offer the flexibility of not being tied to a network for content access and the generalization of the approach to other types of documents being in different formats (i.e., postscript, pdf, etc.), having different characteristics with respect to issues such as lifetime, price, security, size, etc. This work also represents the basis of a framework for the commercial distribution of documents over open networks. This should offer the necessary abstraction to classes of electronic publishing applications sharing common requirements.

References

[1] T. Berners-Lee, R. Cailliau, A. Luotonen, H. Frystyk Nielsen and A. Secret, "The World Wide Web", *Communications of ACM*, Vol. 37, No 8, August 1994, pp. 76-82

[2] Jean-HenryMorin and Dimitri Konstantas, "Towards Hypermedia Electronic Publishing", Proceedings of *second IASTED/ISMM International Conference on Distributed Multimedia Systems and Applications*, Stanford, California, August 7-9 1995.

[3] Dimitri Konstantas, Jean-Henry Morin and Jan Vitek, "MEDIA : A Platform for the Commercialization of Electronic Documents", in *Object Applications*, Ed. D. Tsichritzis, CUI, University of Geneva, August 1996.

[4] Prevelakis V., Konstantas D. and Morin J-H., 1997, "Issues for the Commercial Distribution of Electronic Documents", CMS 97, in Communications and Multimedia Security, Volume 3, S. Katsikas (Ed.), Joint Working Conference IFIP TC-6 and IFIP TC-11, 22-23 September, 1997, Athens, Greece, pp. 265-276.

[5] Jean-Henry Morin , ``Requirements for a Hypermedia Electronic-Newspaper Environment Based on Agents", *Objects at Large*, D. Tsichritzis (Ed.), Centre Universitaire d'Informatique, University of Geneva, July 1997, pp. 177-193.

[6] Jean-Henry Morin , ``HyperNews: a Hypermedia Electronic-Newspaper Environment Based on Agents", in *Proceedings of HICSS-31, Hawaii International Conference On System Sciences*, IEEE 1998, January 6-9, 1998, Kona, Hawaii, Volume II, pp 58-67.

[7] Jean-Henry Morin and Dimitri Konstantas, "HyperNews: A MEDIA Application for the Commercialization of an Electronic Newspaper", in *Proceedings of SAC'98, 1998 ACM Symposium on Applied Computing*, Atlanta, Georgia, February 27 - March 1, 1998, pp. 696-705.

[8] Ken Arnold and James Gosling, "The Java Programming Language", The Java Series, Addison-Wesley, 1996.

[9] James Gosling and Henry McGilton, "The Java Language Environment, A White Paper", Sun Microsystems, Inc., 1996

[10] James Gosling, Bill Joy and Guy Steel, "The Java Language Specification", The Java Series, Addison-Wesley, May 1996, http://www.javasoft.com/docs/white/langenv/

[11] Sun Microsystems, "Java Technology Home Page", Sun Microsystems, Inc, http://java.sun.com/

[12] Sun Microsystems, "The JDBC Database Access API", Sun Microsystems, Inc, http://java.sun.com/products/jdbc/index.html

[13] J. Steven Fritzinger and Marianne Mueller, "Java Security", Sun Microsystems, Inc, 1996.

[14] Sun Microsystems, "Secure computing with Java: now and the future", White Paper, Sun Microsystems, Inc, http://www.javasoft.com/marketing/collateral/security.html

[15] Li Gong, Marianne Mueller, Hemma Prafullchandra and Roland Schemers, "Going Beyond the Sandbox: An Overview of the New Security Architecture in the Java(TM)", *USENIX Symposium on Internet Technologies and Systems (USITS '97)*, Sun Microsystems, Inc., December 8-11, 1997, Monterey, California.

[16] Sun Microsystems, "Java Beans", Sun Microsystems, Inc, http://java.sun.com/beans/

[17] Sun Microsystems, "Java Foundation Classes (JFC)", Sun Microsystems, Inc, http://java.sun.com/products/jfc/index.html

[18] Sun Microsystems, "Java Foundation Classes: now and the future", White Paper, Sun Microsystems, Inc, http://java.sun.com/marketing/collateral/foundation_classes.html

[19] Sun Microsystems, "The Swing connection", Sun Microsystems, Inc, http://java.sun.com/products/jfc/swingdoc-current/index.html

[20] Markus Strasser, Joachim Baumann and Fritz Hohl, "*Mole - A Java Based Mobile Agent System*", *Second ECOOP Workshop on Mobile Object Systems*, University of Linz, July 8-9, 1996.

[21] J. Baumann, F. Hohl, K. Rothermel, M. Schwehm, M. Straßer, "Mole 3.0: A Middleware for Java-Based Mobile Software Agents", to appear in Proceedings Middleware'98, Chapman & Hall, 1998.

[22] ObjectSpace, "Voyager", ObjectSpace Inc, http://www.objectspace.com/voyager/index.html

[23] Systemics, "The Cryptix cryptographic package", Systemics Ltd., http://www.systemics.com/software/cryptix-java/

[24] Stingray Software, "Objective Blend Whitepaper", Stingray Software, http://www.stingsoft.com/obj/whitepaper.asp

[25] Stingray Software, "Objective Grid / Java Whitepaper", Stingray Software, http://www.stingsoft.com/ogj/whitepaper.asp

[26] Gianpaolo Cugola, Carlo Ghezzi, Gian Pietro Picco and Giovanni Vigna , "Analyzing Mobile Code Languages", in Mobile Object Systems, J. Vitek and C. Tschudin (Eds.), Springer-Verlag, LNCS 1222, 1997, p101.

[27] R. Mori and M. Kawahara, "Superdistribution: The Concept and the Architecture", *Transaction of the IEICE*, Vol. E 73, no. 7, 7, July 1990, pp. 1133-1146.

[28] R.L. Rivest, A. Shamir, and L.M. Adelman, "A Method for Obtaining Digital Signatures and Public Key Cryptosystems", in *Communications of the ACM*, Vol. 21, n. 2, Feb 1978, pp. 120-126.

[29] R.L. Rivest, A. Shamir, and L.M. Adelman, "On Digital Signatures and Public Key Cryptosystems", MIT Laboratory for Computer Science, Technical Report 212, 1979.

[30] R.L. Rivest, "The MD5 Message Digest Algorithm", RFC 1321, April 1992.

[31] Bruce Schneier, "Applied Cryptography Second Edition: Protocols, Algorithms and Source Code in C", Wiley, 1996.

[32] CCITT, Recommendation X.509, "The Directory-Authentication Framework", CCITT, ITU, Geneva, 1989

[33] Pascal Montjovent, "HyperNews: L'info transparente", Webdo Mag No 4, avril 1998, p 25, http://www.webdo.ch/webdo_mag/webdo_mag_04/hypernews_04.html

[34] Pierre-Philippe Cadert, Jean-Marc Sandoz et Noël Tortajada, "Village Global", *Radio Suisse Romande La Première*, May 16, 1998, http://www.rsr.ch/LaPremiere/pages/villageglobal.htm

[35] Adrien Bron, "Un Genevois bouleverse les médias électroniques", *Tribune de Genève*, June 10, 1998, p 38.