About Time for News

by

Jonathan A. Sheena

Submitted to the Department of Electrical Engineering
and Computer Science in partial fulfillment of the
requirements for the degree of

Master of Engineering in Electrical Engineering and
Computer Science
and
Bachelor of Science in Electrical Engineering and
Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 1996

Author .....
Department of Electrical Engineering and Computer Science
May 28, 1996

Certified by .....
Walter R. Bender
Associate Director for Information Technology, MIT Media Laboratory
Thesis Supervisor

Accepted by .....
Fredrick R. Morgenthaler
Chairman, Department Committee on Graduate Theses

About Time for News

by

Jonathan A. Sheena

Submitted to the Department of Electrical Engineering and Computer Science on May 28, 1996, in partial fulfillment of the requirements for the degree of Master of Engineering in Electrical Engineering and Computer Science and Bachelor of Science in Electrical Engineering and Computer Science

Abstract

Today's personalized news systems relieve the reader of much of the burden of filtering through the immense amounts of information available at any given minute. However, few take into account changes in a reader's day-to-day life and how these changes may affect his or her interests. This thesis describes a personalized news system that uses observations of the reader's environment to augment the structure and content of a personalized newspaper. The system is implemented as an extension of the MIT Media Laboratory's Fishwrap Personalized News project.

Thesis Supervisor: Walter Bender
Title: Associate Director for Information Technology, MIT Media Laboratory
This work was supported in part by International Business Machines and the News in the Future Consortium

Table of Contents

Personalized Systems 9

Personalized systems as we know them 9
Scope 14

Content vs. Context 17

What streams do we watch? 18
What do we do with the data? 20

About Time for News: The Architecture 21

Architecture 21
Sensors 22
Sensor Manager 24
Observations 25
Making the inferences 26

Tying Environment Sensitivity into FishWrap 29

Why *FishWrap*? 29
Inference methods 30
Presentation in *FishWrap* 31

Results and Future Directions 35

Results 35
Future Directions 36

List of Figures

High level model of personalized systems page 10

Non-persistent models of users page 11

Persistent user models page 12

Expanded user model page 15

Weather Guy page 18

FishWrap Sensor Architecture page 22

Sensors "listen" to user data streams page 23

Schematic of Sensor Manager page 24

Two observations from Jonathan's observation database page 27

FishWrap's implementation of the Glue process page 30

Geographic location interest inferred from a calendar entry page 32

Interest in a FishWrap topic inferred from recently visited web pages page 33

Evolution of Glue's user modeling module page 36

An alternative inference representation page 39

CHAPTER 1

Personalized Systems

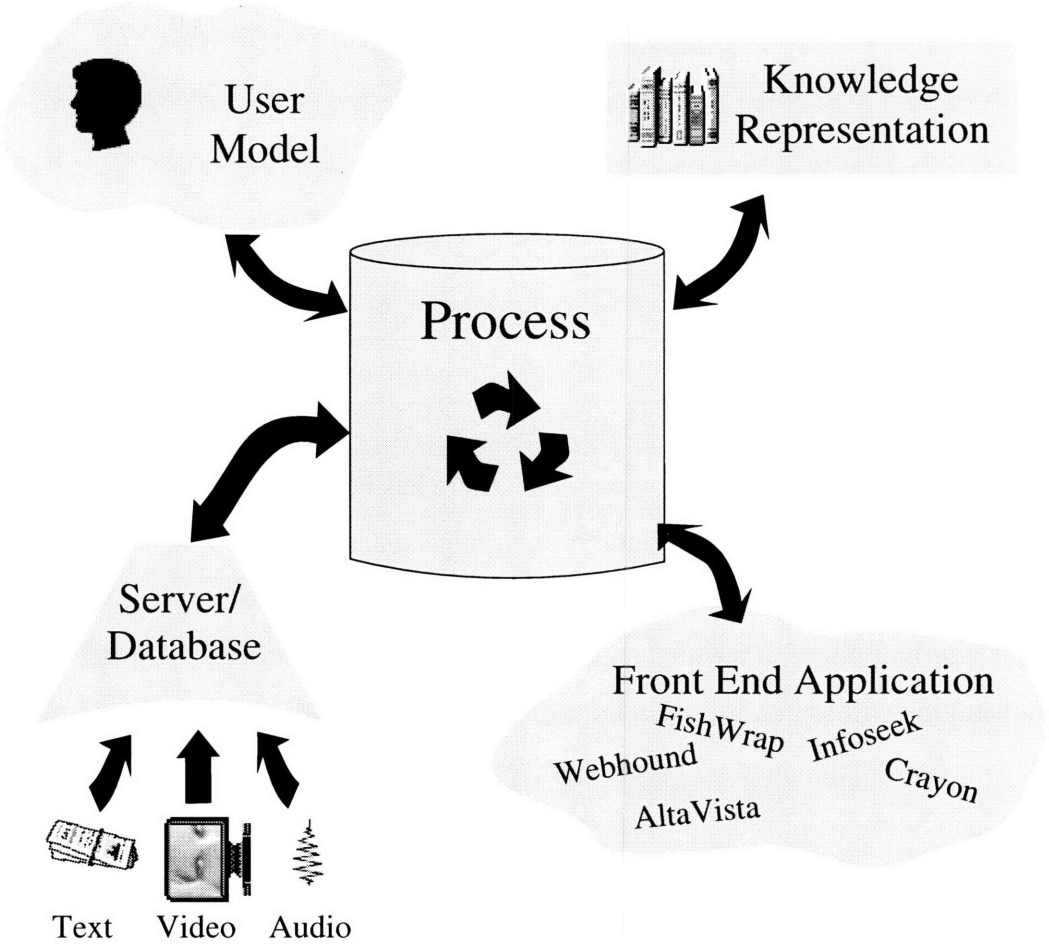
Personalized systems as we know them

In response to today's information overload, there has been a proliferation of filtering systems to help the casual user navigate through large spaces of information. Enormous indexes are commonplace on the internet, making it easy for a user to search through a week's worth of news, or the entire Internet with just a few keywords. Digital's *AltaVista*[1] search engine will search for given key words in 30 million documents in less that one second. National telephone directories, complete road atlases for the United States, movie, music, and book listings, are all indexed and searchable by anyone at any time.

These systems allow users to search a large index and return responses that best match a search. Some systems, in addition, let users customize their experience with the ability to preset specific searches. *Crayon*,[2](CReAte Your Own News) for example, gives users a template newspaper in which they fill in their favorite news sources, and *TV1*[3] lets users create customized television listings.

1. *AltaVista* is a trademark of Digital Equipment Corporation.
2. *Crayon* is a trademark of NETPressence, Inc.
3. *TV1* is a trademark of New Century Productions Inc.

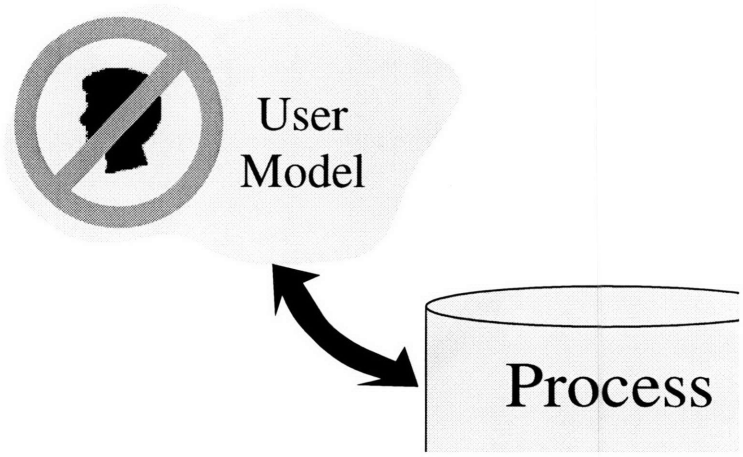
FIGURE 1. This high level model describes most state of the art personalized systems. The scope and implementation of each piece will differ as will the internal process, but the model remains the same.



More sophisticated systems learn from the users' interactions with the system to better personalize the navigation experience. The *FishWrap*[1] [5],[6] personalized news system, reorganizes a reader's newspaper sections and topics to reflect the reader's changing reading habits. And *Webhound* [12] correlates a user's tastes in Web documents to all other users in a large database to present documents most "similar" to those in which the user

1. *FishWrap* is a trademark of Massachusetts Institute of Technology

FIGURE 2. ***AltaVista*, and other index-based engines do not keep persistent models of users.**

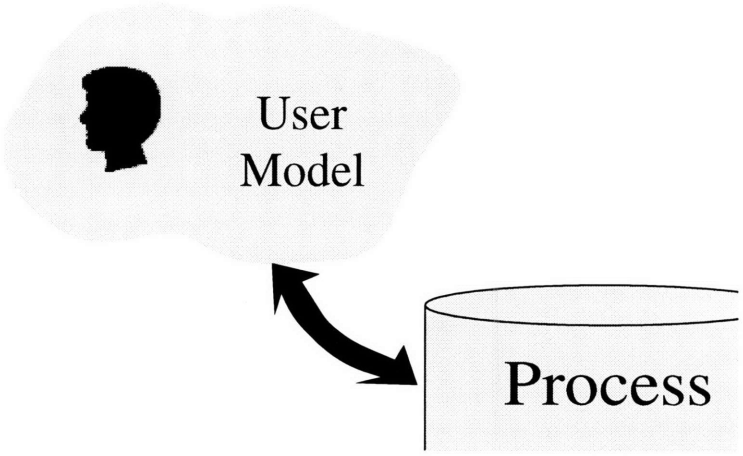


has expressed interest; over time *Webhound* learns what features of documents are most important to each user.

All the above personalized systems, as all state of the art personalized systems, fit in the general model shown in Figure 1. In each example there exists a central process that guides interaction between some set of user modeling, knowledge representation, and database modules with a specific user interface. Most systems have some way to describe, or model their users (even if for only one session). All use some language, or knowledge representation, to describe the mass of data they encompass, and each has a unique way of presenting data and interacting with the user. *Infoseek, Crayon, Webhound,* etc... all fit in this model, and just differ in implementation of each module, and in the process each uses to pull the pieces together. In *FishWrap*, this process is called the *Glue Process.* I will refer to the "Glue process", or the "Glue model", as a general model for personalized systems

Focussing on the user modeling portion of the Glue process, we can begin to some of the differences in these systems. *AltaVista*, and other index-based engines are very powerful but, as emphasized in Figure 2, do not maintain models of individual users. In these sys-

FIGURE 3. Systems like Crayon or Infoseek user persistent user models so users maintain state between sessions.



tems users have no identity between sessions, and thus cannot build upon prior experiences.

A system like *Crayon* or *Infoseek*, on the other hand, uses persistent user models to keep track of users and user interests between sessions, and has advantages that non modeling systems do not. These systems take advantage of the user modeling portion of the Glue model, as seen in Figure 3, and can "remember" what they learn from and about the user.

Still, a limitation to this model is it that it lacks an outlet whereby the process can learn about the user beyond the user's interaction with the system. Ideally, if a personalized news system could keep tabs on the comings and goings of its reader, it might better be able to infer some of the reader's changes in interests to produce a better newspaper.

For example, if a friend from Madagascar is visiting me tomorrow, I am likely to have an immediate interest in that part of the world. That interest might take precedence over my usual interests. A news system like *FishWrap* or *Crayon* will pick up on such changes only if explicitly told. In *FishWrap* I would have to add a new section with news about that region, then remove it when I am no longer interested. A personalized news system

should instead pick up on this new interest and inject into my base profile a sudden (and perhaps fleeting) interest in Madagascar.

In this way a good personalized news system, or any personalized system, can benefit greatly from observations both collected and inferred about its subject. A good personalized system will account for, and even predict changes in user's environment that affect her interests.

> "People exhibit change between sessions with a computer.... As we change, we want our computers' user models to change with us." [14]

Global and local changes affect people's interests in countless ways. On a global scale, for example, the assassination of Prime Minister Yitzhak Rabin shifted world focus to Israel. While, much closer to home, a cooking show I saw yesterday may spur my interest into the medical benefits of garlic tomorrow, or an upcoming trip to Singapore might prompt me to suddenly keep up with events of the Far East.

News editors have been doing the same for whole communities for hundreds of years. The role of a news editor is to tailor a publication to the interests of her community of readers, and those interests fluctuate and change over time.

> "The newspaper industry ... is unsurpassed in its ability to gather and organize vast quantities of time sensitive information" [3]

A good news publication is often said to have "a finger on the pulse" of the community when its articles are relevant and timely. Reporters and editors understand what is important at that moment, and pride themselves on being able to predict "the next big thing". A good editor can filter out from innumerable potential stories the ones that pertain to her community today. Harold Evans, author of *Newsman's English,* describes the role of the 'copy-taster' for a large publication which best fits this bill:

> On a paper of moderate size, one man can select news from three agencies and staff. On a multi-sectioned newspaper ... there may have to be specialist selectors. ... There is no bet-

> ter title for this work than the British one of 'copy-taster': the old name perfectly describes the work. He must be a man with a sensitive news palate. He savours all the news. On a big daily newspaper he will have to make a thousand snap rejections. Of course he cannot read every one of the hundreds of thousands of words that come at him. He skims the copy and because he is up to the minute with the news and in tune with the wants of his newspaper and its reader he can detect, at a taste, what is suitable. He is like the professional wine-taster. He does not have to drink the whole bottle; a tablespoonful or the mere bouquet will do to declare whether it is palatable for that particular paper at that particular moment. [9]

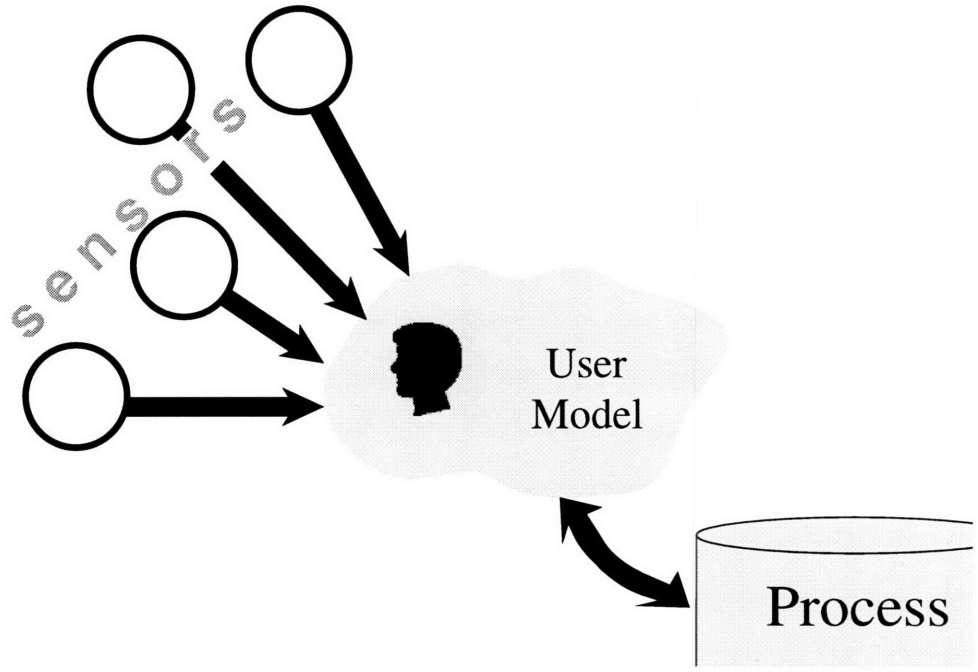
The Doppelgänger [13],[14] architecture developed by Jon Orwant at the MIT Media Laboratory was built to be a that 'finger on the pulse' of individuals. It is an architecture for modeling of users and communities. Doppelgänger takes advantage of data collected by external observers, or *sensors*, to make inferences and generalizations about users and about changes and patterns in a user's habits. It is by making observations about the user that we enhance the user model of the Glue process to make Glue applications better electronic 'copy-tasters'.

Scope

In this thesis, I intend to introduce some ideas present in Doppelgänger into the Glue model by giving the Glue user model sensors into the user's environment, as shown in Figure 4. Specifically, I will show how expanding Glue to include user observation outside the user's interaction with the system can enhance the user's overall experience.

To that effect, I first present a system that is sensitive to changes in the user's environment, and can monitor and record user activity. Then I show how that system can be used by a personalized news application, namely *FishWrap*, to augment the content and structure of a personalized news system.

FIGURE 4. The new expanded user model described here takes advantage of sensor input to enhance user models.



CHAPTER 2 *Content vs. Context*

Computational environments are not always as rich as we would like them to be. A run of the mill desktop computer cannot read its user's facial expressions to determine if she is happy or angry, or sad[1], and most desktop machines do not even know the *name* of the person in front of them. This makes the normal computer a context-poor environment. Compare, for a moment, the bland data of a weather report in Figure 5a to the context sensitive presentation in Figure 5b. *Weather Guy* [10] knows where in the world the reader is originally from, and presents the weather for that individual. Someone from Alaska would consider 70 degree weather in March unseasonable warm, while a Puerto Rican would not. While *Weather Guy* knows what climate the reader is accustomed to, most on-line weather reporters do not have that advantage.

So, given that most computational environments are contextually poor, how can we make the inferences about the user's changing interests mentioned in the previous chapter?

1. However, plenty of work points in that direction [20].

FIGURE 5. a) On the left is today's weather from the national weather service b) on the right are two "Weather Guy" reports, one for an MIT student from Juneau, and one for a student from San Juan

```
Last update was at: March 20
  15:10:01 EDT 1996
The weather observed at BOSTON
  (BOS) at 02:56 PM EDT was:
   The skies were mostly cloudy.
   Temperature: 53F (17C)   Dew-
  point: 36F (7C)
   Winds from the W (270 degs) at
  20 mph gusting to 26 mph.
   ...
```

What streams do we watch?

Since it is difficult to observe the person behind the screen directly, we have to rely on clues from the user's visible interactions to guess at the her interests. The underlying assumption is that the content she is presented with, and seeks out throughout the course of a day reflects at some level, her interests. Monitoring that content can give insight into those changing interests. For example, if a user reads twenty different stories about *Babylon 5*[1] then the computer should deduce her interest in *Babylon 5*, and should include any *Babylon 5* news in her newspaper.

1. *Babylon 5* is a trademark of Time Warner Entertainment Co.

The list of text-based, content-rich streams that are monitored on the user's behalf is shown in Table 1.

TABLE 1. List of monitored data streams.

web browsing	People often use the web as a resource for interesting documents, which makes web browsing one source of clues toward users' changing interests.[a]
changes in web bookmarks	These are points that the user has marked as important for one reason or another. Additions or subtractions are good clues to the user's changing interests
calendar	A user's calendar often includes very concise entries for the activities of the day, and provides important information about day to day affairs.
mail	Though private mail is often hard to analyze for content, some more formal mailing lists carry well formed content on a specific subject.

a. Though much of what we encounter on the web is "content-free", people specifically seeking content can often find it.

Web browsing

From a user's web browsing we try to extract *topical* information. Though unstructured, the web is rich in topical information, and from a large number of observations, patterns begin to emerge which point toward the user's interests. In this way, a single observation means little, but many observations supporting the same topical observation will count for a great deal.

Changes in web bookmarks

These are points that the user has explicitly marked as interesting. Therefore, we can assign a higher confidence to topical observations made from bookmark files. When the user makes changes to her bookmark file, those changes are important clues to the user's changing interests.

Calendar

While a calendar file may not be as rich in topical information as a web document, a user's calendar is a source of very *timely* information. Calendar files are also can be rich sources of *geographic* observations. Any traveller's calendar will be rich in geographic informa-

tion, from which we can infer shifting geographic interests. Since calendar entries are most often concise and to the point, we can give a high confidence to any observations made by this sensor.

Mail

Electronic mail represents the least structured stream since mail is used as often as a broadcast medium (mass mailing lists) as it is for personal communication. We take the same approach with email as with web browsing and try to extract many, low confidence *topical* observations in the hope that patterns of topical interest will emerge.

What do we do with the data?

At some point, the question must arise, "what relevant information do we extract from these observed items?" The answer is unclear. Obviously each sensor is an expert is some way about the items it extracts, but at the same time there is a great deal of general knowledge that all sensors should be able to harness.

To gain the best from both worlds, every observed item is passed through a general content understanding process, then handed back to the sensor to augment or modify those observations with sensor specific knowledge.

For each stream we will extract *topical, geographic, and timely* observations. Though many other kinds of observations are possible, we choose these types of observations since they map well onto the news domain.

CHAPTER 3

About Time for News: The Architecture

It is hard to talk about user modeling, or specifically, user *monitoring* without invoking insidious images of Big Brother, and without making people feel queasy about lack of privacy. In this system, all the observations are user initiated. Since none of the sensors have any special privileges, they can only operate when explicitly activated by the user. The user always has the ability to view observations made during the observation phase or after the fact, and can always clear the database of all accumulated knowledge about him. Further, the collected data is deposited in the user's private *FishWrap* account, and is only used for applications that serve that user. It is in this light that we continue.

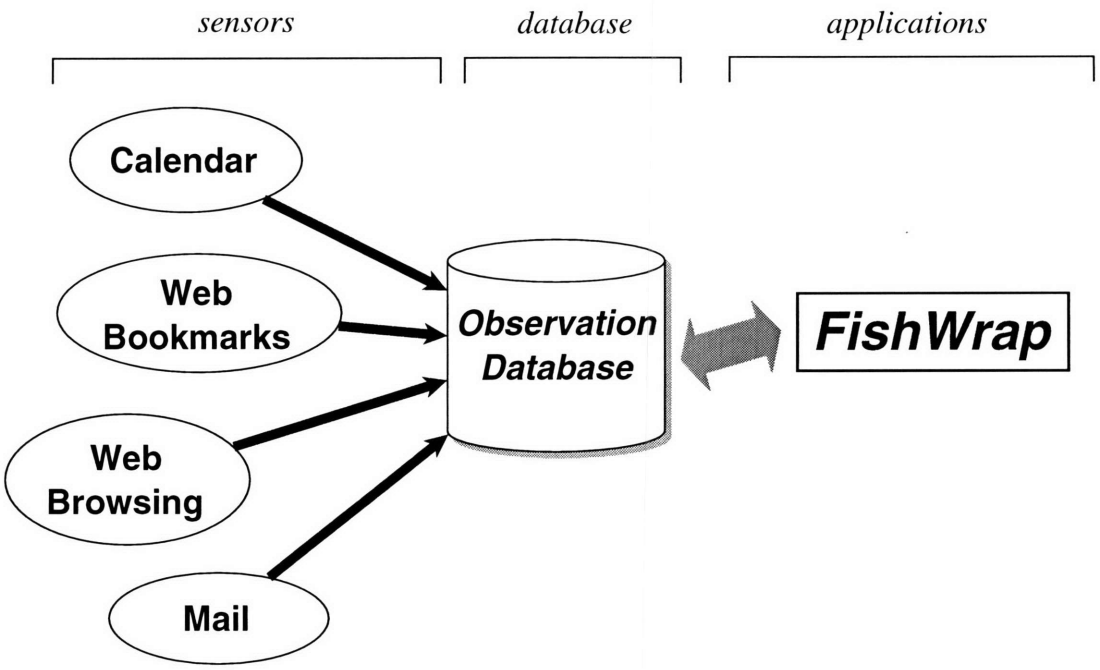
Architecture

This system consists of three parts, (see Figure 6) and in many ways resembles the Doppelgänger architecture. Those pieces are:

- sensors
- observation database
- applications

For each user, a group of sensors collect data about that user in the form of individual observations, and then stores those observations in the user's private observation database.

FIGURE 6. FishWrap Sensor Architecture

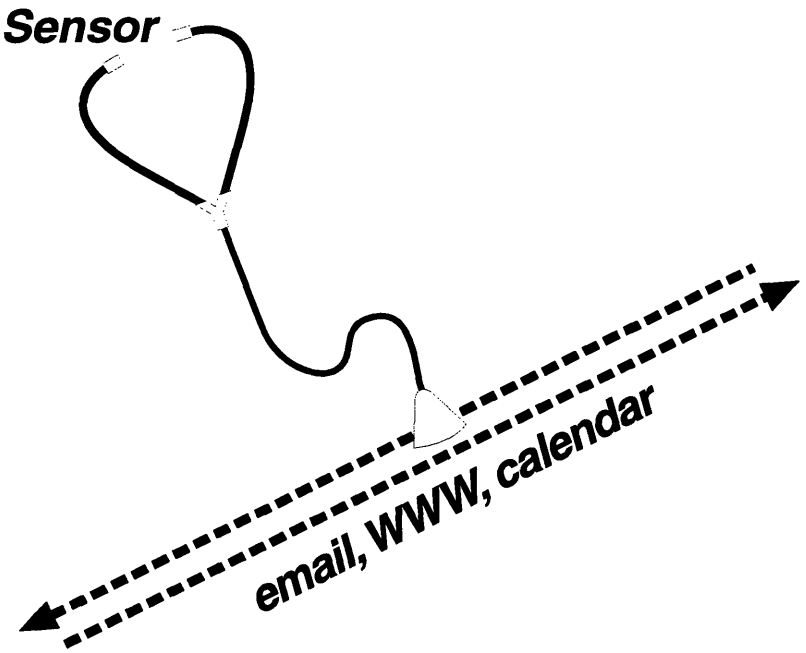


Applications (such as *FishWrap*) can at any time take advantage of those observations to gain insight into the user's changing interests.

Sensors

The sensors in this system are passive observers that siphon individual elements from a particular stream in the user's computational environment (see Figure 7). An "element" can be a web page, or a mail message, or a calendar entry, or a login, or any other interaction which a sensor is built to monitor. Each individual sensor is designed to watch a given data stream, and is designed to speak the language of that data stream. For example, the mail sensor is able to extract individual mail messages from a user's mail spool for analysis, but cannot read a calendar file. In this way, sensors are *specialized*. Each sensor

FIGURE 7. Sensors "listen" to user data streams



is like a translator; it speaks the language of the stream it is monitoring and translates items to an application independent language that all sensors, applications and databases share.

Sensors monitor small things about the user, individual web pages, or zephyr[1] conversations, or calendar entries. Orwant best describes the use of "big stuff" vs. "little stuff".

> Some types of information are often deemed uninteresting ... because they aren't "deep", i.e. related to some aspect of the user's cognitive state, such as his plans, beliefs, or knowledge. But shallow information -- the "little stuff" ... is useful [because] these minutiæ can provide profound insights into deeper states ... and little stuff is much easier to obtain than big stuff [14]

A sensor is broken down into four basic operations:

- initialize the sensor `(initializer)`
- are there new items to observe? `(newness_test)`

1. Zephyr is MIT's instant notification system. Zephyr is a trademark of MIT.

- extract new items `(extractor)`
- translate items to application independent format `(converter)`
- make any expert inferences about the items `(signature)`

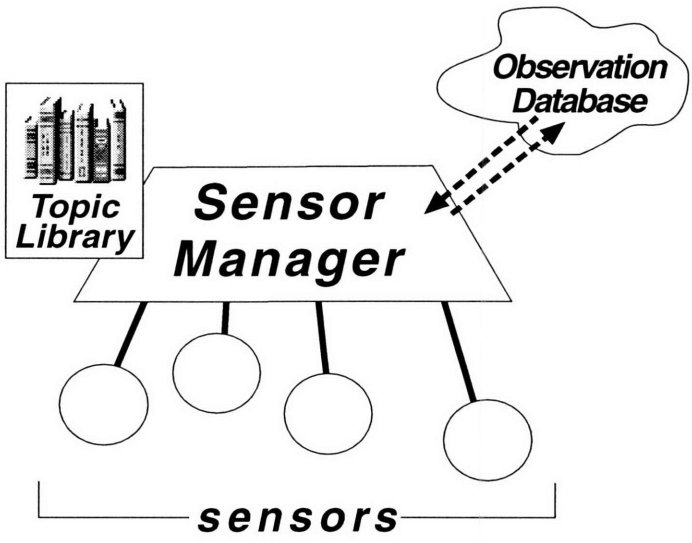
All sensors implement these same four operations for their data stream of specialization.

Sensor Manager

A collection of sensors is managed by a *Sensor Manager* (see Figure 8). This manager decides which sensors should be active at any given time, and centrally processes all sensor responses. Since it is computationally expensive to have each sensor perform content analysis on each item, and expensive for each sensor to maintain its own connection to the observation database, this functionality is centralized in the sensor manager. Moreover, to give the user more control over sensors, the manager provides one point of contact.

The sensor manager also allows for a relative weighting of each sensor's data. Observations from the user's calendar will be more sparse, but likely to be much more meaningful

FIGURE 8. Schematic of Sensor Manager



than observations made from user's web browsing. For this reason, the manager assigns a confidence to each sensor. Confidence values are {`low`, `medium`, `high`, or `very_high`[1]}.

The Sensor Manager is a simple shell that performs these tasks when run:

- choose active sensors
- checks last observation time for each sensor
- initializes each active sensor `(sensor:initialize)`

for each sensor:

- check for new items to observe `(sensor:newness_test)`
- extract new items `(sensor:extractor)`
- translate new items to application independent format `(sensor:converter)`
- apply Glue's content understanding (from the knowledge representation module) to new items
- apply the sensor's own content understanding on each new item `(sensor:signature)`
- send new observations to observation database, with assigned confidences

The sensor manager puts sensors to sleep when they produce no new data over a period of time, and polls sleeping sensors less often than those that are awake.

Observations

Sensors collect and analyze different elements from different sources, but all must translate their findings into "observations". Since the goal is to be able to glean inferences from a set of observations, observations must adhere to a common format. At the same time, that format should not preclude the development of different kinds of observations. For that reason, the observation data can take any form, but is tagged with an observation type that determines how to interpret that observation.

1. `very_high` is reserved for user feedback.

Each observation is broken up into a number of pieces:

TABLE 2. Basic elements of an observation.

`item`	pointer to a stored representation of the original object (if it exists)
`sensor type`	type of sensor used (webhistory sensor, email sensor etc ...)
`observation type`	describes how to interpret the observation, either as "fishwrap topics", or "geographic observations", etc ...
`observations`	list of observations, each observation is interpreted using according to the above type.
`confidence`	sensor specific confidence. "how much do we trust this observation vs. another"
`time stamp`	time at which the observation was made, or the time to which the observation pertains[a]

a. If, for example, a sensor observes a new entry in a web history file for a document visited yesterday, the observation will be stamped with the time the document was viewed rather than the time of the observation.

Two sample observations are shown in Figure 9.

Making the inferences

An architecture for making and storing observations has been presented, but nothing has been said about making the inferences from the observations. In some sense the actual mechanism for making observations is outside the scope of this architecture. It is an application specific process that decides what to do with the observations. One could imagine an application that just looks for observations made from the users calendar to augment reminder messages, or another that uses the whole database to try to find differences in observation patterns between weekdays and weekends. In terms of the Glue model, (recall Figure 1) the Glue's user modeling module is extended to provide an outlet for communication with the observation database, but uses an *application callback*[1] for an application

1. In the Glue model the process includes hooks for applications to modify default behavior. For example, the part of the process that sorts articles in each topic allows the application to determine how to sort articles, be it by recency, source, author, or by any arbitrary function.

FIGURE 9. Two observations from Jonathan's observation database:
a) a "place" observation from Jonathan's calendar whose entry mentions "Paris", France.
b) an item observation looking for "fishwrap topics" which matches the "morbid" and "babylon5" topics.

a)

```
(("item" "Rscrapook#jsheena:calendar_832824000.0")
  ("observation source" "calendar" )
  ("observations"
    (("PARIS" "" "FRANCE" ) ) )
  ("observation type" "places" )
  ("confidence" 3 )
  ("unixtime" 832889657 )
  ("uniqueid" "reel.media.mit.edu:832889556" ) )
```

b)

```
(("item" "proxy#Rscrapbook#jsheena:http://www.hyper
      ion.com/lurk/universe/five-year-overview.html" )
  ("observation source" "webhistory" )
  ("observations"
    ("morbid" "babylon5" ) )
  ("observation type" "fishwrap topics" )
  ("confidence" 1 )
  ("unixtime" 832941077 )
  ("uniqueid" "reel.media.mit.edu:832941081" ) )
```

specific process to make inferences from the observations. The next chapter describes specific extensions to the *FishWrap* personalized new system as one example of an application that takes advantage of the user model's new capability.

CHAPTER 4

Tying Environment Sensitivity into FishWrap

Why FishWrap?

FishWrap[1] is the Media Laboratory's personalized electronic newspaper that has been available for use by the entire MIT community since 1993. *FishWrap* was originally called the *"Freshman FishWrap"* since its original intent was to provide a publication for the incoming freshman class that could bridge the gap between local news of Boston and local news of whatever their hometown. To that end, *FishWrap* draws on local sources like MIT students, administration, as well as local, national and international sources like the *Associated Press, Reuters, Knight-Ridder Tribune, The Boston Globe, The New York Times, BPI Entertainment*, etc ... just to name a few.

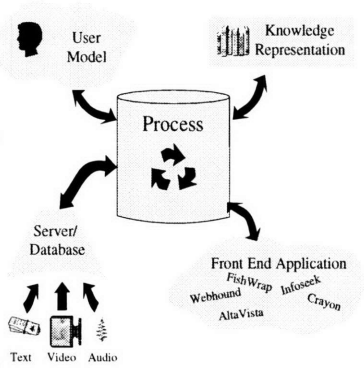
FishWrap is an excellent platform for other experiments since its architecture is very modular and has been used as a basis for a number of experiments including investigations into automated restructuring [11], a community wide front page called "PageOne", and context sensitive comics like "Weather Guy" (see Figure 5).

that illustrates the modules that make up *FishWrap*'s underlying architecture called Glue. Each module is built separately and communicates with the supervising process through a

1. *FishWrap* is so named after the old journalist's adage "Yesterday's news wraps today's fish"

FIGURE 10. FishWrap's implementation of the Glue model first presented in Figure 1. The shaded region shows the specific point in the glue process which we expand to include sensory input

load environment variables
`glue2up` reads the user's profile
for each section in the profile
load the section information, including the list of topics included in that section
for each topic in the current section
`glue2requests` translates the interest in a particular topic (found in the profile) into query for given news server type
`appSearchModify` modifies search with application specific parameters
`glue2server` submits the query to the server
`appPostFilter` filters the articles returned from the server using application specific parameters
`glue2bookie` records those articles as "presented" in the users personal database
`appReorder` reorders articles using application specific parameters
`appRender` renders that topic using application specific parameters.
done with topic
`appPostSectionFilter`
`appRender` finishes rendering that topic.
done with section
when we're done with the paper, we can relinquish control back to appMain, the specific applic tion's main loop.



prespecified protocol. By defining the protocol and not the implementation of each module, the details of the modules are abstracted out of the model. That abstraction allows developers to easily replace modules such that 1) neither the implementation of the process, nor that of any other module is affected and 2) all applications built upon the Glue architecture can suddenly take advantage of the new module without having to be modified.

For *FishWrap* to take advantage of the new capabilities of the Glue user modeling module, *FishWrap* must include a method, or *callback*, for gleaning inferences from the observed data.

Inference methods

When a user visits *FishWrap*, the user modeling module in the underlying Glue process consults the observation database and calls the *FishWrap* specific hook to generate inferences from the data. As described at the end of the previous chapter, Glue provides the mechanism for communicating with the observation database, and uses an *application callback* to make the inferences.

In its current implementation, *FishWrap*'s inference callback separates the observations by "observation type" (recall Table 2) and uses a two step process to choose the most representative observations for each type.

Merge. The first step is to merge the observation set. *FishWrap* merges all identical observations into one observation with multiple hits. (Each hit is marks one instance of that observation).

Sort. The resulting list is then sorted by a calculated weighting for each observation (now composite observations). The weighting is determined by EQ 1.

$$\frac{(\text{average confidence})(\text{number of hits})}{(\text{average age})} = \text{weighting} \qquad \textbf{(EQ 1)}$$

This weighting prefers recent items with higher confidence that were observed more often.

After each set is merged and sorted, *FishWrap* chooses the top 10 inferences to incorporate in the new edition.

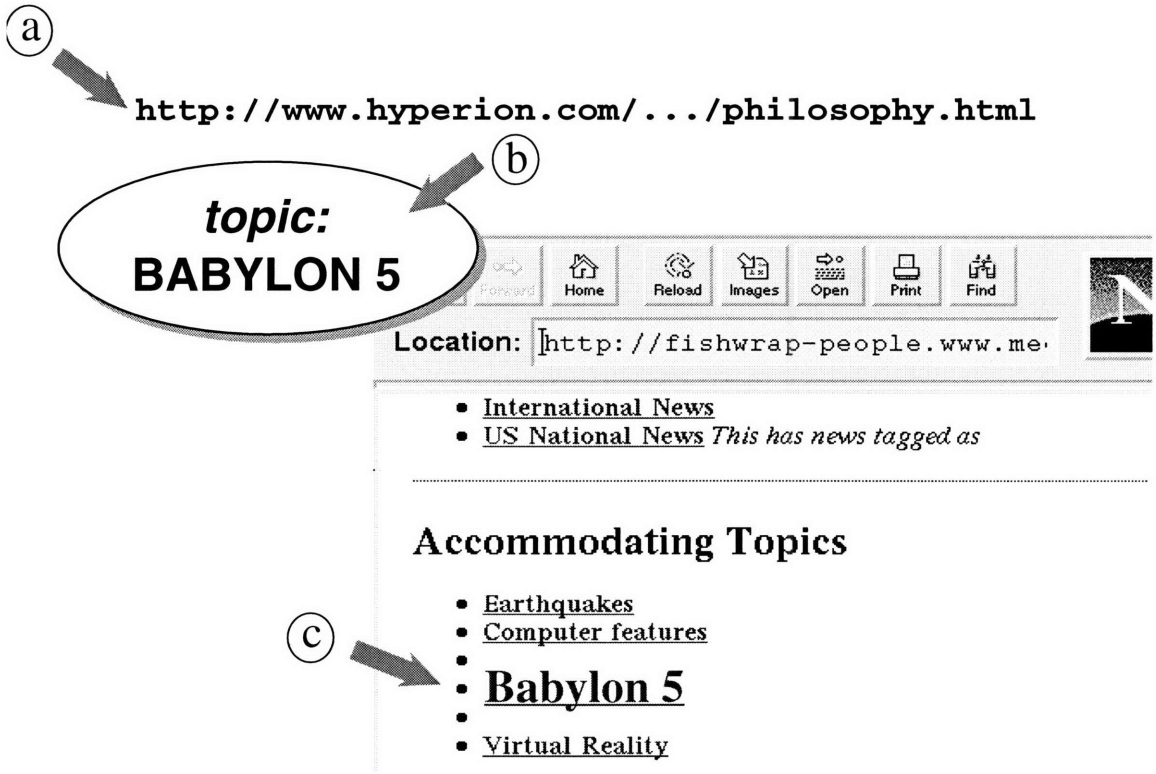
Presentation in FishWrap

To demonstrate different types of observations, content augmentation in *FishWrap* was presented in two sections. The first was labeled "Accommodating Topics", made of infer-

FIGURE 11. An example of a geographic location interest inferred from a calendar entry. The new topic is added to the *Accommodating Places* section.
a) a calendar entry mentions a conference in Oslo
b) the observation is recorded and stored in the observation database
c) a new *FishWrap* topic is created from generated inferences

ences from topical observations, and the second "Accommodating Places", made of inferences from geographic location observations. Each section included new topics gleaned from each set of observations. This division is arbitrary, and one could imagine many such divisions. Figure 11 shows the life of an observation, in this case the mentioning of Oslo, Norway in a calendar file. That observation, along with many others is stored in the observation database. When that user generates a new edition of *FishWrap*, that observation is considered by *FishWrap*'s inference generation callback, and is included in the user's new edition as a place of interest. In the same way, Figure 12 demonstrates inference of an interest in a topic from observations made of a user's web browsing.

FIGURE 12. An example of interest in a *FishWrap* topic inferred from recently visited web pages. The new topic is added to the *Accommodating Topics* section.
a) user visits web sites that discuss *Babylon 5*
b) the observation is recorded and stored in the observation database
c) a new *FishWrap* topic is created from generated inferences



CHAPTER 5

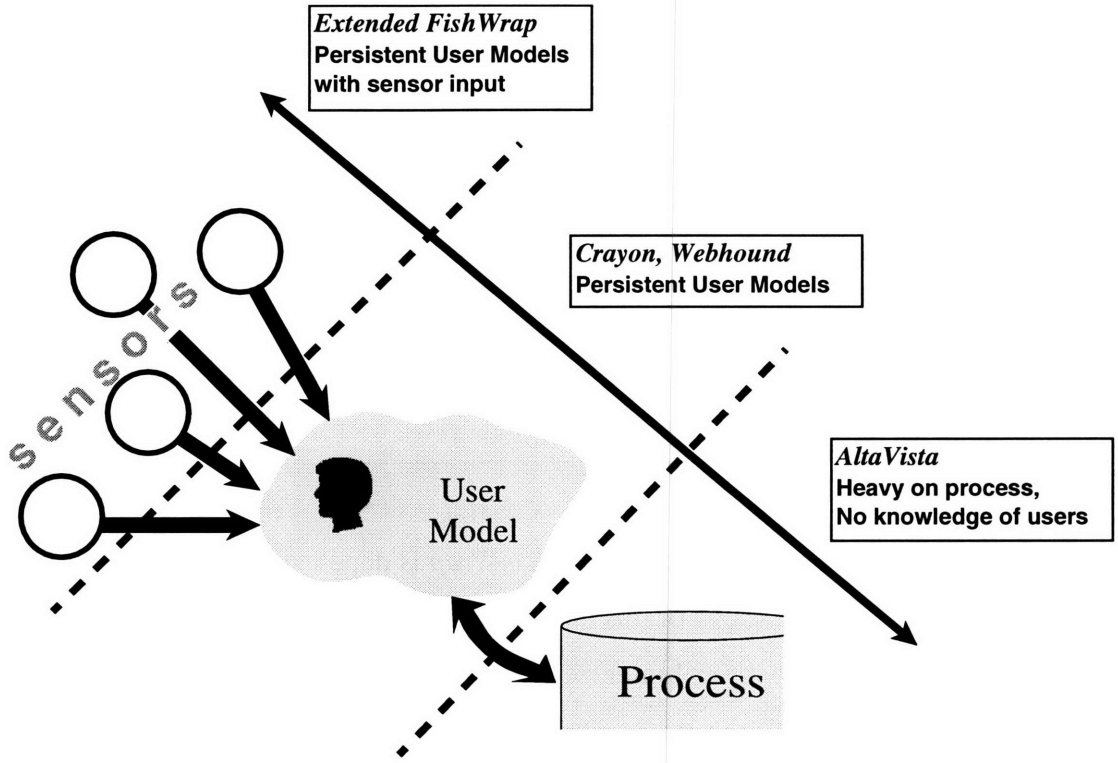
Results and Future Directions

Results

The success or failure of these additions to *FishWrap* is dependent on many pieces of the system that are outside the scope of this thesis. Most notably, the general content analysis package used to determine topical inferences from observed items sometimes reaches unexpected conclusions. For example, after one user browsed through dozens of web pages of different bed and breakfasts in Cape Cod, one of the most representative topics in her new edition was the "Pet Features" topic since each page talked about whether or not pets were allowed. In this way it was difficult to evaluate the success of topical inferencing independent of the content understanding software. In an informal user survey more people appreciated the "Accommodating Places" section that introduced inferences of geographic location more than the "Accommodating Topics" section that introduced topical inferences. This preference reflects, at least in part, the limitations of the content understanding module used in this example.

One subject commented that he would have rather seen a different division of inferences in his paper. He would have rather seen new sections divided along the boundary of observation confidence rather than observation type such that inferences from high confidence

FIGURE 12. Evolution of Glue's user modeling module



observations appeared in one section, and inferences from low confidence observations appeared in another.

Future Directions

The evolution of the Glue model (recapped in Figure 12) does not confine development to the example outlined in the previous chapter. A number of stages in the process are still open ended and deserve exploration.

More sensors

Any system based on observations is fundamentally limited by what it can observe. Web browsing and calendar entries only tell so much about a user. For example, if the system knows who a user enjoys communicating with, it might be reasonable to infer that that user might be interested in some of the same things as his friends. Or if the system knew what the user actually reads rather than just what a user is presented with, it might be able to better assign confidences in observations.[1] One obvious sensor that was not incorporated is one to monitor the *FishWrap* articles the user reads. It might be feasible to infer interest in article subtopics. For instance, if a user subscribes (explicitly) to the "armed forces" topic, and reads a number of articles matching both "armed forces" and "gay rights", then it might be reasonable to infer an interest in articles matching the "gay rights" topic, and augment the paper with that topic. Alternatively, the system could use that knowledge to try to hone in on a user's interests by adding a new topic "gay rights and armed forces" that only included articles at the intersection of both topics.

Feature confidence

Feedback from end applications could be incorporated to recalculate feature confidence. Different users might consider different features important. For example, it should be possible to glean from feedback, whether a user considers mentioned geographic location more important than matched topics. Or perhaps mail from a particular person always peaks the user's interests. In *FishWrap*, feedback can be of the form of what articles are read and what articles are not, or else another application could ask the reader for explicit feedback at the end of each article[2]. Monitoring user responses to inferences, then correlating those responses back to the space of observation types might give insight into what

1. The addition of more sensors will add more depth to the collected observations, however, more sensors increase the problem's dimensionality, giving the system more variables to consider.
2. FeedbackPaper included a "thumbs up" and a "thumbs down" at the end of each article to allow readers to give feedback about their interests in that article [5].

types of observations are useful as clues into user interests and what are not. *Webhound* analyzed feature weighting for features of web documents.

Inference methods

The method of inference presented here was one simple example, and presented some problems. For example, one observed item that matched "morbid" and "babylon 5" topics is treated independently of an observation that just matched "babylon 5" topic. Clearly these two observations are in some way connected. Deerwester et al. [7] demonstrate a system that indexes documents by mapping them into a space defined by all keywords occurring in a set of documents. Singular value decomposition of that space provides a means of comparing a query to all documents in this keyword space. It seems feasible to be able to apply this same idea to observation space. But instead of using SVD as a mechanism for measuring item similarity, use it to find most representative observations. A method such as this one would account for overlap between observations.

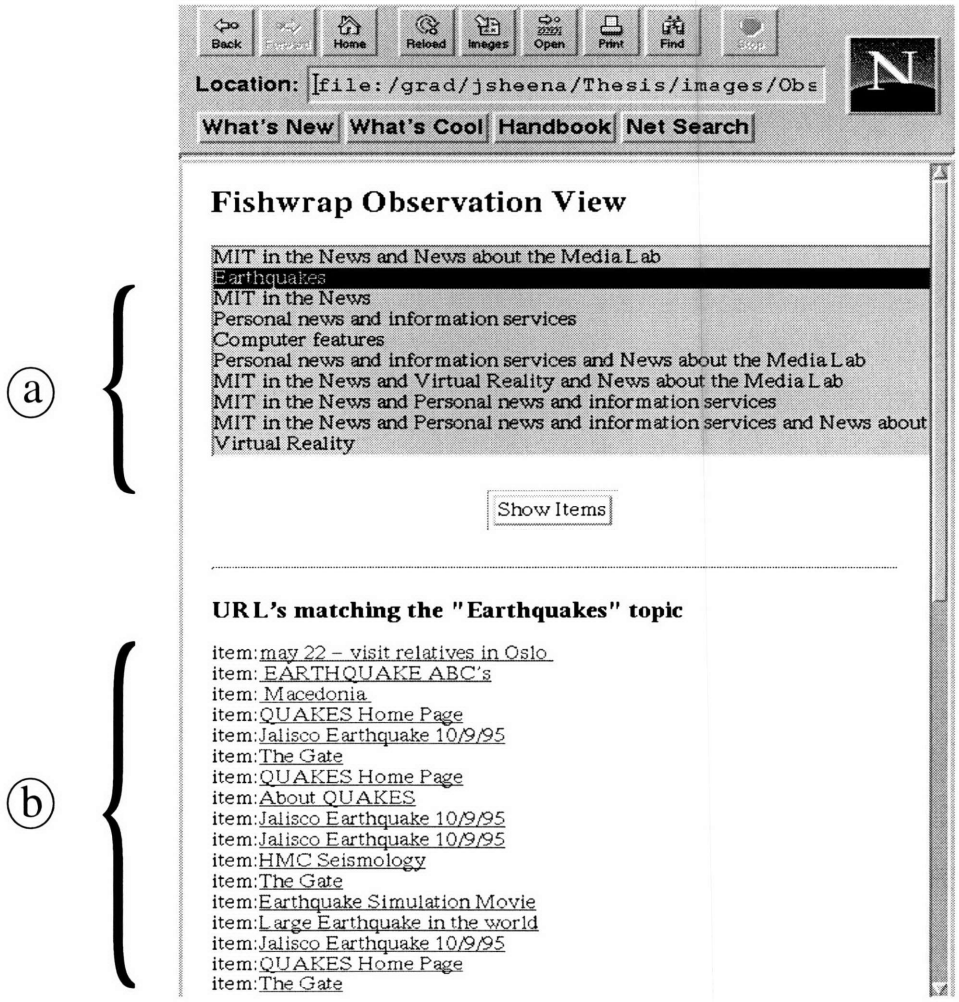
Presentation

Different presentations can add different flavors to the generated inferences. Instead of a using the inferences to augment a personalized newspaper, one could instead use the inferences to re-present the data itself. One such example was implemented. Here observed web pages are listed by observation, and sorted using the inference method described in chapter 4 (see Figure 13). Many other different presentation forms are possible. Rhodes explores use of similar technology in his *Remembrance Agent* [15] as a desktop reminder tool for stored documents that are similar to current ones. Other presentation paths might augment email or calendar reminders with relevant news.

Tying together a community

The future work presented above all relates sensor observations of a user to a large space of data. What we have not yet explored is the idea of relating users to each other to build communities based on inferred interests. [2]

FIGURE 13. An alternative inference representation: web pages arranged by topic.
a) Listing of topics sorted by inference method described in chapter 4.
b) Web pages from web history matching selected topic, in this case, "Earthquakes".



Tailoring content to the individual

In this system, content of individual articles has been left alone. However, it is now the case that we can augment content itself given the context of a set of observations. For example, it would be interesting to take the route of Sara Elo's PLUM [8] which augments disaster articles by putting the scale of damage and destruction in a context with which the

reader is familiar. *FishWrap* could use sensory information the same way, to augment key concepts found in articles to those found in the observed elements, be they web documents, or mail messages, or any other relevant sensory input.

Acknowledgments

I'd like to thank my advisor, Walter Bender. Walter is an incredible source of inspiration and guidance and I owe him thanks for the opportunity and environment in which i could grow.

To Pascal Chesnais, I owe a special thanks. It is he who kept me on the path toward an attainable goal, but never let me lose sight of the big picture. I would still be floundering had it not been for him.

Doug Koen, Brad Bartley, Michelle McDonald, Karrie Karahalios, Matthew Mucklo, Jon Orwant, Dan Gruhl, Klee Deines and all the UROPers and graduate students of the Garden made the environment livable, even at 3am, and kept the framework going on which this thesis is built.

To my friend, Steven Telio, I owe even more than my facial hair. Without his constant taunting, this thesis might never have been written.

Thanks to all my friends at Agents Inc., for all their encouragement.

To Sandhya Deo, it was her presence and her charm that made this time in my life sweet. I feel blessed to have such a friend.

To my parents and family, I give my immeasurable thanks for their unending support and patience. If you only knew what I put them through.

References

[1] Nathan Abramson, The Dtype Library or, How to Write a Server in Less Time than it Takes to Read This Manual. Technical report, Electronic Publishing Group, MIT Media Laboratory, December 1992.

[2] Walter Bender, Pascal Chesnais, Sara Elo, Alan Shaw, Michelle Shaw. Enriching communities: Harbingers of news in the future. IBM Systems Journal, Vol. 35, Nos. 3&4, 1996.

[3] Walter Bender, Hakon Lie, Jonathan Orwant, Laura Teodosio, and Nathan Abramson. Newspace: Mass Media and Personal Computing. *USENIX Conference Proceedings*, Nashville, TN, 1991.

[4] Alan W. Blount, Self-Organizing News. Master's Thesis, MIT Media Laboratory, 1991.

[5] Pascal R. Chesnais, and Douglas Koen. Strategies for personal dynamic systems: News in the future. In *NextWorld Expo*, 1993.

[6] Pascal R. Chesnais, Matthew J. Mucklo, and Jonathan A. Sheena. The Fishwrap Personalized News System. *IEEE Second International Workshop on Community Uncorking Integrating Multimedia Services to the Home*, 1995.

[7] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, September, 1990.

[8] Sara Elo. PLUM: Contextualizing News for Communities Through Augmentation. S.M. thesis, MIT Media Laboratory, 1995.

[9] Harold Evans. Newsman's English. Holt, Rinehart and Winston, Great Britain, 1972.

[10] Mark Hurst. iComix. Description of interactive comics: http://fishwrap-comics.www.media.mit.edu/iComix/files/manifesto.html, 1995.

[11] Douglas B. Koen. Automated Restructuring of an Electronic Newspaper. S.B. thesis, MIT, 1994.

[12] Yezdi Lashkari. Feature Guided Automated Collaborated Filtering. S.M. thesis, MIT Media Laboratory, 1995.

[13] Jonathan L. Orwant. Doppelgänger: A User Modeling System. S.B. thesis, MIT, 1991.

[14] Jonathan L. Orwant. Doppelgänger Goes to School. S.M. thesis, MIT Media Laboratory, 1993.

[15] Bradley J. Rhodes. Remembrance Agent: A continuously running automated information retrieval system. *AAAI-Spring Symposium '96: Acquisition, Learning maladministration: Automating Tasks for Users,* 1996.

[16] Benjamin Schoon. Fishpaper: Automatic Personalized Newspaper Layout. S.B. thesis, MIT Media Laboratory, 1994.

[17] Upendra Shardinand. Social Information Filtering for Music Recommendation. M.Eng thesis, MIT Media Laboratory, 1994.

[18] Ryan A. Smith. The Timeliness Server: Context-Dependent Management of Distributed News. S.B. thesis, MIT, 1993.

[19] Charles W. Therrien. *Decision, Estimation, and Classification.* John Wiley & Sons, New York, 1989.

[20] Matthew Turk and Alex Pentland. Eigenfaces for Recognition. *Journal of Cognitive Neuroscience*, Volume 3, No. 1, 1991.

[21] WWW Reference Library Documentation. Available at *http://www.w3.org/pub/WWW/Library/*