Surfing the market and making sense of the web:

Interfacing the web to an open agent-based market infrastructure *

Joakim Eriksson Niclas Finne
Telia Research AB, S-123 86 Farsta, Sweden
joakime@sics.se, nfi@sics.se

Sverker Janson
Swedish Institute of Computer Science
SICS, Box 1263, S-164 28 Kista, Sweden
sverker@sics.se

June 11, 1996

1 Introduction

At a first glance, the web may seem like a good infrastructure for electronic commerce, but with a closer look, and with a view to future *open agent-based market infrastructures*, the insufficiencies become apparent. There are

- no standards for describing the interests of participants in the market, such as having products for sale, and
- no standards for interacting with other participants in the market, such as vendors, buyers, and brokers.

These problems are not caused by the web. They exist already in ordinary "human-based" markets. But with such standards, the Internet-based marketplace could turn into a *global marketspace* [1] that would make possible any degree of automation of access to and matching of interests, and of interaction between participants.

We are currently exploring what types of formats and protocols are needed for these marketplace interactions. In this position paper we will discuss the relation between our work and the web. In summary, we see no competition between an agent-based marketplace and the web. They have quite complementary rôles.

2 Related work

Among the hundreds (thousands?) of approaches to doing commerce on the net, we pick a few that have some similarities to our own.

*Presented at the WWW5 workshop: "Programming the Web - a search for APIs", Paris, May 6, 1996

Buy & Sell Online [2] is one example of a *classified ad marketplace* that tries to collect classified ads into one place, and in which users can use keyword search to find interesting ads. Since the information is given as free format text, it is difficult (bordering to impossible) to interpret the information and to automate the market. Also, such marketplaces are both centralized and closed.

Webra by Polycon AB [3] has well-structured information to ease search. Webra can notify users via email when something interesting appears. Webra is centralized and closed.

Kasbah [4] is an agent-based marketplace developed by MIT Media Lab. The information is structured and all interests are handled by agents that negotiate with each other. The users are notified when an agreement has been reached. The current system is a simulator.

In general, the creators of most marketplaces have aimed for a centralized and closed services where web-users can go shopping, and where the operator of the marketplace can charge for the service, instead of an open infrastructure for commerce.

3 An open agent-based market infrastructure

We will not give a complete description of our market infrastructure in this paper, only discuss the aspects relevant to this paper: openness, structured information, interaction protocols

Openness

A marketplace should be open in much the same manner as the web. Nobody should own the marketplace. Anyone should be able to enter it and declare interests. Anyone should be able to offer services such as brokering. It should just be a matter of starting your own marketplace server and announcing its availability.

For a high degree of openness, it is necessary to have open standards for formats and protocols, that can evolve over time. To be open not only in theory, but also in practice, the formats and protocols must be very simple.

Structured information

The information on the marketplace should be well-defined to allow processing by computers. This precludes describing interests in free form natural language text, or, even worse, using graphics and audio. Of course, such information could/should be available for human consumption. Negroponte's phrase *bits about bits* summarizes what is needed succinctly [5].

As an illustration of the current situation with the web, assume that you are interested in a trip to London. You will probably look for some keywords, e.g., "london buy travel", using a search service such as Alta Vista. This will give you thousands of hits, of which at most a few are relevant for you. How would a program know? Also, the information contained in the few pages actually found is virtually impossible to process automatically. (If you find a better way, please let us know.)

We are investigating very simple knowledge representation techniques, with very simple object-oriented representations. But our system will be open to more sophisticated approaches to knowledge exchange, such as KIF [6].

Interaction protocols

A marketplace has to support more activities than just finding the desired product or customer. There should exist well-defined interaction protocols. In addition, these protocols should make sense for human-human, human-agent and agent-agent interaction, to make possible any mix of human and automated participants in the market.

The interaction protocols should support (at least)

- advertising interests
- searching for interests
- negotiation

A proper treatment of issues such as security, authentication, and payment is quite necessary in these protocols (but is not the focus of our work).

4 Surfing the market

The web is the major access route to Internet. Given this fact, new Internet-based infrastructures, such as the open electronic marketplace that we are developing, must have an open interface to the web to make any impact at all. The interface to the web must handle more than just letting users with a web browser use services like brokers, customer advisors, etc. It should at least offer web-based users the possibility to

- See all (public) interests
- Present an interest to the market
- Interact with agents and market servers

Information

Since information about objects can be stored in different formats, not all directly readable by web-browsers, it must be possible to convert object information into HTML-pages and get the information accessible from the web. This conversion must be done automatically at each market server which provides object information. One method of doing this conversion, which we have been testing in our prototype, is to let all objects have a URL and do a automatic method call when the object is accessed from the web. This method call generates a web-page which describes the object.

Freedom for market servers The requirement that all market servers must be able to convert their information to web-pages does not restrict the way this information is presented. Some servers may show the information as tables, others may have nice descriptions with pictures and animations.

Freedom for web surfers Users may not want to view the information the way the marketplace serves it. They may just want to see some basic information which would be sent when using a common interaction protocol. One way of giving user freedom to view the information as he wants it is to implement a plug-in to a web browser which converts from the common interaction protocol to HTML and back.

Interaction

Since web-based users must be able to interact with agents in the marketplace, market servers and other programmed participants should have an interface between the web and their interaction protocols. This can be done using HTML-forms in the communication between web and market server. A typical interaction starts when a user submits a proposal to a market server. The server responds by either accepting, rejecting or refining the proposal given. If the server refines the proposal then the user can accept, reject or refine it, using a form sent by the server. This procedure continues until one of the parts either accepts or rejects the others proposal. This HTML-form interaction could provide the web user with about the same expressiveness as the full interaction protocol.

5 Making sense of the web

The web should be available to the marketplace. We consider three possibilities: (1) extracting and integrating information, (2) integrating services, and (3) using the web for representing interests.

Extracting and integrating information

Our marketplace infrastructure should be able to integrate information from existing web-warehouses. This can be done using interpreters that understand the format used by the web-warehouse for describing products. The interpreters convert product descriptions into a format that is understood by all marketplace participants.

Integrating services

To fully integrate a web-warehouse with the marketplace, we also have to integrate the interaction (buying, selling). While integration of information could be done in a static way (the information is interpreted at a regular basis and stored as "marketplace ready" information), the integration of interaction must be done whenever the interaction takes place. This means that we have to create agents that translate the interaction from the common interaction protocol of the marketplace to the specific interaction protocol of the web-warehouse.

Figure 1 illustrates the possibility to interact with a web-warehouse that has been integrated with the marketplace. If many warehouses are integrated, a web-based customer can learn one of the interaction interfaces provided in the marketplace and then use the same interface to interact with many different web-warehouses.

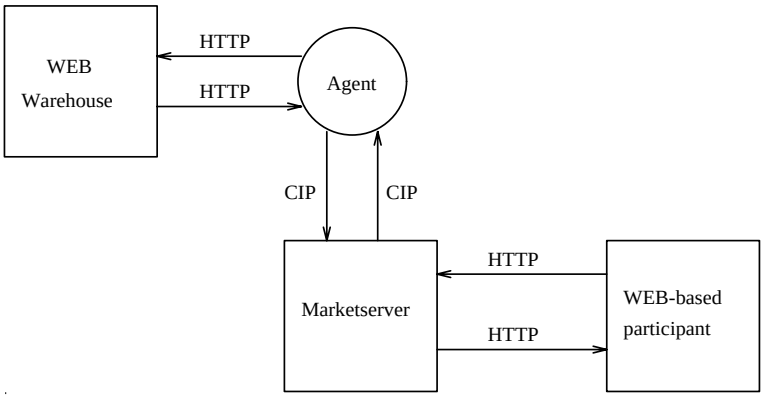

Figure 1: Integration of a web-warehouse into the market

Representing interests

We have developed a format called Uniform Object Descriptions (UOD) for writing web pages that can be directly mapped to objects in our object model. Since this format allows for quite readable pages, it offers a simple means for entering the marketplace using only a web server.

6 A marketplace prototype

We have developed a marketplace prototype, implementing several of the ideas described above.

The prototype features

- an object-oriented information abstraction
- distribution of objects between market servers
- a web interface to information and services
- a market server which reads structured-web pages
- a broker which matches interests and notifies users via mail

Figure 2 illustrates how several market servers announce their services to the others.

Information abstraction

We have in this prototype chosen an object-oriented information abstraction. A class in our abstraction describes the fields of an object with name and type. A field can be of the type method, which means that this field is a procedure which can be called. A class also has a parent class for inheritance.

Information distribution

We have defined a simple communication protocol, the Distributed Object Protocol (DOP), for distribution of classes and objects. The protocol supports searching for

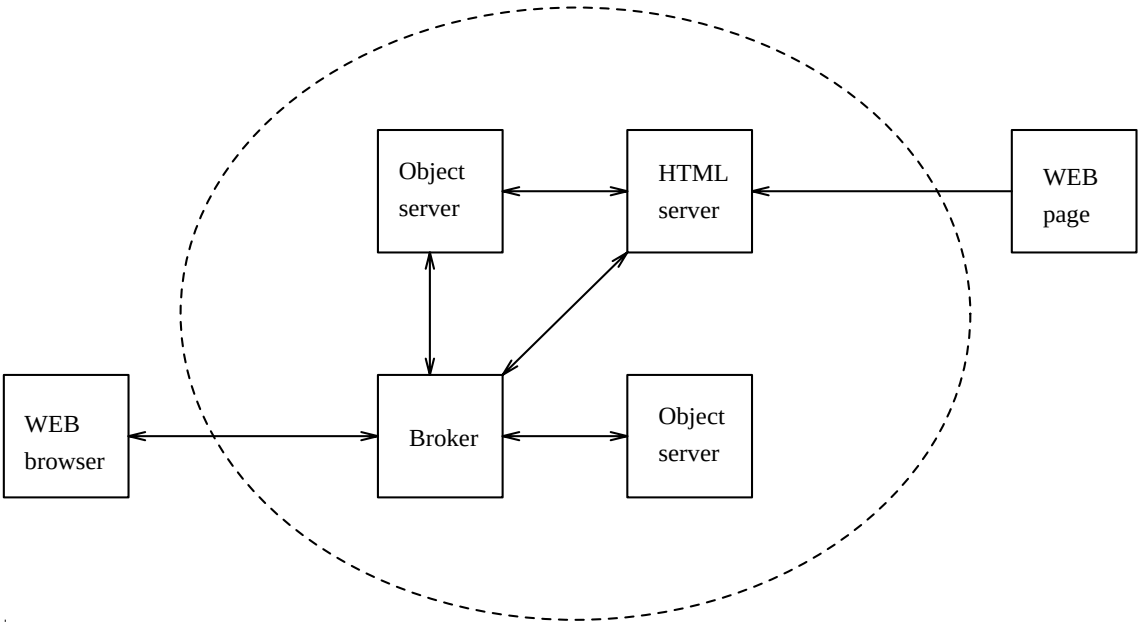

Figure 2: Marketplace with four market servers

classes not currently supported in that server. We allow servers to replicate classes to avoid unnecessary communication.

To be able to communicate, servers must know other servers addresses and the protocols they understand. For this we have an object type called service object which contains information about another server. When a server starts up, it must register to another server to be able to communicate with it (when registering to another server a service object is returned which describes the other server).

DOP messages are divided into four categories: server registering, class distribution, object distribution and object method calls.

Example of a DOP message

```
dop v0.9
msg(header(Id,Sender,Receiver),object(find,t_interest,[]).
```

This simple example illustrates the syntax of a DOP message, first there is a header to indicate that the message is a DOP version 0.9 message. The actual message contains two parts, one identification part (the header part) and one content part. The content part describes in the example that someone wants to find all objects of the class t_interest.

Interfacing the WWW

The market servers understand both DOP and HTTP. As a result, all objects can be reached from the web by just giving a URL to the object. When accessing objects from a web-browser the object's display method creates a web-page. This gives a direct way of viewing objects from the web.

We have also implemented a parser for UODs, Uniform Object Descriptions, to be able to read objects directly from web pages.

Example of an object described using UOD (also as web-page in figure 3)

```
01 <TITLE>Interest Games-page</TITLE>
02 <BODY BACKGROUND=http://www.sics.se/ nfi/html/background.gif
03 TEXT=#000000 BGCOLOR=#afafae LINK=#00f0ff VLINK=#800080> <BR>
04 <CENTER> <IMG SRC="http://www.sics.se/ joakime/dderby.jpg"> <BR>
05 <H2>For sale:  Destruction Derby</H2>
06 One of the most destructive racing games ever !  <BR>
07 For more information see <A HREF=
08 "http://www.vidgames.com/ps/software/dderby.html">
09 unofficial playstation homepage</A>
10 <BR><BR><BR><P>
11 <UOD CLASS = t_interest>
12 <TABLE BORDER=2 CELLSPACING=4>
13    <TR><TD>Agent = nfi@sics.se
14    <TR><TD>Class = game
15    <TR><TD>Type = sell
16    <TR><TD>Match = buy
17    <TR><TD><!- Description{>
18    <TR><TD>Title = Destruction Derby
19    <TR><TD>Category = Driving/Racing
20    <TR><TD>Producer = Sony Psygnosis
21    <TR><TD>Platform = Playstation <!- }>
22 </TABLE>
23 </UOD>
24 </CENTER>
```

The interesting section of the above HTML-page is the one between `<UOD CLASS = t_interest>` and `</UOD>` where the object is described. The first label indicates both where the object description begins and which class this object is an instance of (t_interest in the above case). Then there follows a couple of lines of the format `<FieldName = Value>` which says that the field called `FieldName` has the given value. Lines 17-21 is an example of a field which contains a list of other fields. Line 17 also illustrates the possibility to hide fields from being displayed by browsers.

Object servers

The object servers are the basic information suppliers in our current prototype but they can operate in different ways. One object server can simply work as a database where objects can be stored while another can take information from structured web-pages (structured using the UOD-format) and distribute this information.

In addition, there are other services, such as brokers, that allow customers to specify interests to be matched against other interest in the marketplace. When a matching interest is found, the customer is notified by email.

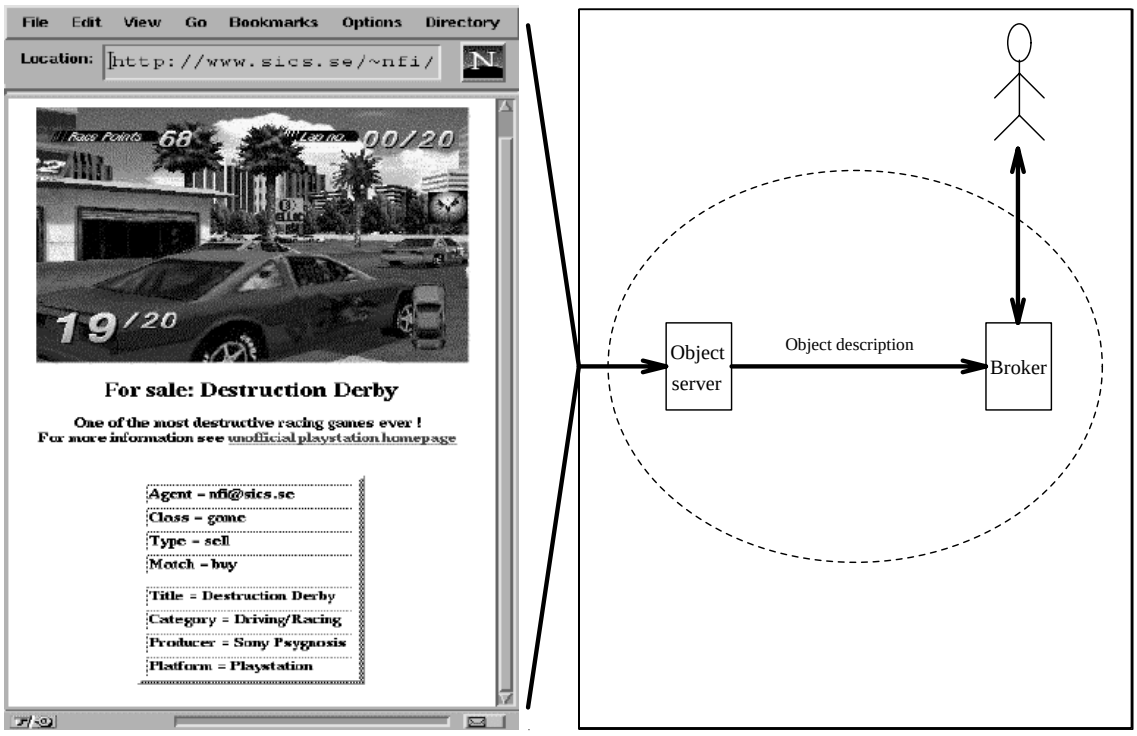

Figure 3: Example of a scenario

The following simple scenario illustrates a possible interaction in the marketplace.

- A user declares an interest to a broker.
- Another user declares an interest as a web-page (in UOD).
- The object server parses the web-page and makes it available as an interest to the rest of the "structured marketplace".
- The broker matches the two interests and notifies the user.

The above scenario suggests what type of structured objects can be described using UOD. Since our information model is object oriented, both the person (or program) writing the description and the program which reads it, must have information about its class. This means that classes must be distributed over the marketplace, in various formats, to be useful.

7 Conclusions and Future Work

We have outlined how the web as a marketplace could be complemented by an open marketplace infrastructure that would co-exist with the web in an almost symbiotic manner. We have presented some of our design criteria and described our current prototype system.

In our continued research, we will continue refining the interaction and information models, and their implementations in terms of existing and new standard Internet formats and protocols, such as HTTP, FTP, SMTP, WHOIS++, etc.

References

[1] Christopher Koch. Mutable markets. *CIO Magazine*, 1995. URL:http://www.cio.com/CIO/cio_12_1_95_markets.html

[2] Buy & Sell Press. Buy & Sell Online. URL: http://www.buysell.com.

[3] Polycon AB. Webra. URL: http://www.polycon.fi/webra.

[4] Anthony Chavez and Pattie Maes. Kasbah: An Agent Marketplace for Buying and Selling Goods, 1996. URL: http://jeeves.media.mit.edu/Papers/paam96.ps.

[5] Nicholas Negroponte. *Being Digital.* Vintage, 1996.

[6] Richard E. Fikes Michael R. Genesereth. Knowledge interchange format version 3.0 reference manual. Technical report, Computer Science Department, Stanford University, 1992.