An XML-Based Adaptive Multi-agent System for Handling E-commerce Activities

Pasquale De Meo[1], Domenico Rosaci[1], Giuseppe M.L. Sarnè[1], Giorgio Terracina[2], and Domenico Ursino[1]

[1] DIMET
Università Mediterranea di Reggio Calabria
Via Graziella, Località Feo di Vito, 89060 Reggio
Calabria, Italy
[2] Dipartimento di Matematica
Università della Calabria, Via Pietro Bucci
87036 Rende (CS), Italy
{demeo,domenico,sarne}@ing.unirc.it, terracina@mat.unical.it,
ursino@unirc.it

Abstract. In this paper we propose an XML-based adaptive multi-agent system for handling e-commerce activities. More specifically, our system aims at supporting a customer, visiting an e-commerce site, in the search of products and/or services present therein and appearing to be appealing according to her/his past interests and behaviour. The system is adaptive w.r.t. the profile of both the customer and the device she/he is exploiting for visiting the site. Finally, the system is XML-based since XML is exploited for both storing the agent ontologies and handling the agent communication.

1 Introduction

1.1 Motivations

In the last decade e-commerce has emerged as a significant, both social and cultural, phenomenon; indeed, many vendors provide their customers with the possibility to buy their products also by using the Internet; moreover, several trading companies exist carrying out their activity only via Web (e.g., Amazon).

Operations executed by a customer, whenever she/he buys a product or a service from an e-commerce site, can be usually grouped into two main activities, namely: *(i)* the exploitation of a Web search engine (such as Altavista or Google) for finding potentially interesting sites; *(ii)* the use of a Web browser for navigating through the pages of a site she/he judged interesting. While a large variety of tools already exists over the Internet for helping a customer in searching potentially interesting sites, only few applications are available for supporting the navigation through an interesting site.

Some of the most common of these applications are based on *recommender systems* [14,15]. These have been conceived for helping the customer of an e-

M. Jeckle and L.-J. Zhang (Eds.): ICWS-Europe 2003, LNCS 2853, pp. 152–166, 2003.
© Springer-Verlag Berlin Heidelberg 2003

commerce site in her/his purchase activity. In order to perform its tasks, a recommender system must process available data; this can be done by exploiting various computation tools such as cross-sell lists and information filters.

Another important tool available in a large number of e-commerce sites is a (usually rough) manager of customer profiles, exploited for proposing personalized offers. However, the profile construction methodology causes some problems. Indeed, it requires a customer to spend a certain amount of time for constructing and/or updating her/his profile; in addition, it stores only information about the issues which the customer claims to be interested in, without considering other issues somehow related to those just provided, possibly interesting her/him in the future and that she/he disregarded to take into account in the past.

In spite of recommender systems and customer profile managers, generally, when accessing an e-commerce site, a customer must personally search the issues of her/his interest through the site. We argue that, for improving the effectiveness of the e-commerce service, it is necessary to increase the interaction between the site and the customer, on the one hand, and to construct a rich profile of the customer, taking into account her/his desires, interests and behaviour, on the other hand.

In addition to this, it is necessary to take into account a further, important factor. Presently, many of the e-commerce sites, as well as many of the e-service providers, assume the visitor operates with a large color display and fast network connections. However, nowadays, electronic and telecommunications technology is rapidly evolving in such a way to allow cell phones, palmtops and wireless Personal Digital Assistants to navigate on the Web. These mobile devices do not have the same display or bandwidth capabilities as their desktop counterparts; nonetheless present e-service providers deliver the same content to all device typologies [1].

In order to overcome the problems outlined previously, some challenges must be tackled. First, a customer can access *many* e-commerce sites; a faithful and complete profile of her/him can be constructed only taking into account her/his behaviour on accessing all these sites. In other words, it should be possible to construct a *unique structure on the customer side*, storing her/his profile and, therefore, representing her/his behaviour on accessing *all these sites*.

Second, for a given customer and e-commerce site, it should be possible to compare the profile of the customer with the offers of the site for extracting those issues that probably will interest the customer. Existing techniques for satisfying such a requirement are mainly based on the exploitation of either log files [5, 17] or cookies [12]. Techniques based on *log files* are capable to register only some information about the actions carried out by the customer on accessing the site; however, they are not capable of matching customer preferences and site contents. Vice versa, techniques based on *cookies* are able to carry out a certain, even if primitive, matching; however they need to know and exploit some personal information that a customer might consider private.

Third, it should be necessary to overcome the typical "one-size-fits-all" philosophy of present e-service providers by developing systems capable of adapting their behaviour to both the profile of the user and the characteristics of the device

she/he is exploiting for accessing them [2]. In the past, various approaches have been proposed for handling e-commerce activities. Some of them are based on traditional recommender systems [1]; other ones (e.g., *Firefly* [16], *Tete-a-Tete* [9], *WEBS* [10], *SETA* [4], *e-CoUSAL* [7]) have been designed in the context of the agent technology. These approaches construct, maintain and exploit a customer profile; therefore, we can consider them adaptive w.r.t. the customer; however, to the best of our knowledge, none of them is adaptive w.r.t. the device.

On the other side, in various computer science research fields, a large variety of approaches adapting their behaviour to the device that the user is exploiting, has been proposed [3,6]. These are particularly general and interesting; however, to the best of our knowledge, none of them has been conceived for handling e-commerce activities.

1.2 General Characteristics of the Approach

This paper aims at providing a contribution in this setting; indeed, it is an attempt to solve all the three problems mentioned previously. More specifically, it presents *ec-XAMAS* (e-commerce oriented XML-based Adaptive Multi-Agent System), a multi-agent system for handling e-commerce activities capable of adapting its behaviour to both user and device profiles. In *ec-XAMAS* an agent is present in each e-commerce site, handling the information stored therein, as well as the interaction with the customers. In addition, an agent is associated with each customer, adapting its behaviour to the profiles of both the customer and the device she/he is exploiting for visiting the sites.

Actually, since a customer can access an e-commerce site by different devices, her/his profile cannot be stored in only one of them; as a matter of fact, it is necessary to have a unique copy of the customer profile which registers her/his behaviour in visiting e-commerce sites during the various sessions, possibly carried out by means of different devices. For this reason the customer profile must be handled and stored in a support different from the devices generally exploited for accessing the sites. As a consequence, on the customer side, it appears compulsory the exploitation of a client-server multi-agent system consisting of: *(i)* a *server agent*, which stores the profiles of both involved customers and devices; *(ii)* a *client agent*, associated with a specific customer operating with a specific device, which supports the customer in her/his activities and is deactivated at the end of the session. We call *XAMAS* such a multi-agent, customer oriented, component of *ec-XAMAS*.

The previous description points out that three typologies of agents operate in *ec-XAMAS* for each navigation session, namely: *(i)* a *Site Agent*, associated with an e-commerce site; *(ii)* a *XAMAS Client* associated with a specific customer and a specific device, supporting the customer activities during a specific e-commerce session; *(iii)* a *XAMAS Server*, storing and handling both customer and device profiles.

Our approach constructs and maintains a profile for each customer; it stores information about the visits she/he carried out on e-commerce sites in the various sessions. Whenever a customer visits an e-commerce site, the *XAMAS*

Client and Server cooperate for updating her/his profile. As a consequence, for each customer, a *unique profile* is maintained, storing information about her/his behaviour in visiting *all e-commerce sites*. In this way, our approach solves the first problem mentioned above.

Whenever a customer accesses an e-commerce site, the corresponding agent sends its ontology to the *XAMAS* Client associated with the customer and the device she/he is exploiting. As it will be clear in the following, the site agent ontology stores information about available products and/or services. The *XAMAS* Client determines similarities between the products/services offered by the site and the interests of the customer. Each of these similarities represents an issue of interest for the customer offered by the e-commerce site. For each similarity, both the site agent and the *XAMAS* Client agent cooperate for presenting to the customer a group of Web pages illustrating the offers of the site appearing of interest for her/him.

We argue that this behaviour provides our model with the capability of supporting the customer in searching interesting products in the e-commerce sites. Moreover, as it will be clear in the following, the technique our approach uses for constructing the Web pages presented to the customer allows to determine not only information probably interesting for the customer but also other issues possibly interesting for her/him in the future and that she/he disregarded to take into account in the past. As previously pointed out, this is a particularly interesting feature for an approach devoted to deal with e-commerce services.

Last, but not the least, it is worth observing that, since *the customer profile management is carried out at the customer side*, no information about the customer profile is sent to the e-commerce site. In this way, *ec-XAMAS* solves privacy problems left open by cookies. All this reasoning allows us to claim that also the second problem introduced previously is solved by our approach.

As previously pointed out, in *ec-XAMAS*, the device profile plays a central role. Indeed, the Web pages of an e-commerce site presented to a customer, as well as their formats, depend on the characteristics of the device she/he is presently exploiting. However, the capability of *ec-XAMAS* of adapting its behaviour to the device the customer is exploiting is not restricted to the Web page visualization; indeed, the exploited device influences also the computation of the interest degree of a customer for the products and/or services available at each site. More specifically, as it will be clear in the following, one of the parameters, which the interest degree associated with a product/service is based on, is the time the customer spends in visiting the corresponding Web pages. This time is not to be considered as an absolute measure, but it must be normalized w.r.t. both the characteristics of the exploited device and the navigation costs. The following example allows to clarify this intuition. Assume that a user visits a Web page for two times and that each visit takes n seconds. Suppose, also, that during the first access she/he exploits a mobile phone having a low processor clock and supporting a connection characterized by a low bandwidth and a high cost. During the second occurrence, she/he uses a personal computer having a high processor clock and supporting a connection characterized by a high bandwidth and a low cost. It is possible to argue that the interest the user

exhibited for the page in the former access is greater than that she/he exhibited in the latter one. Also other device parameters influence the behaviour of *ec-XAMAS*; they are illustrated below. All this reasoning allows us to argue that *ec-XAMAS* solves also the third problem mentioned above.

The previous description points out that, in *ec-XAMAS*, many agents are simultaneously active; they strongly interact each other and continuously exchange information. In this scenario, an efficient management of information exchange appears crucial. Actually, the efficiency in the information exchange management has become a challenging issue in most areas of computer science research, particularly for Web-based information systems. In this area, the most promising solution to this problem was the definition of XML, a novel language for representing and exchanging data over the Internet. XML embodies both representation capabilities, typical of HTML, and data management features, typical of DBMS's; it is presently considered as a standard for the Web. XML capabilities make it particularly suited to be exploited in the agent research; as a matter of fact, agents using XML for both representing and handling their own ontology have been already proposed in the literature [8]. In *ec-XAMAS* the role of XML is even more central since it is crucial for guaranteeing the management of an information base (e.g., the agent ontology) in devices characterized by limited bandwidth and processor clock (where traditional, generally resource-expensive, information source handlers cannot operate).

2 The ec-XAMAS Component Managing the Customer Activities

As previously pointed out, in the customer side of *ec-XAMAS* a client-server multi-agent system, called *XAMAS*, operates. In *XAMAS*, a *Client Agent* CA_{ij} is associated with a user U_j exploiting a device D_i; it supports U_j during her/his navigation activities carried out by means of D_i. Observe that more than one Client Agent can be associated with the same user, if she/he exploits different devices; analogously, more than one Client Agent can be associated with the same device, if it is exploited by different users.

A *Server Agent* SA stores and manages information about both user and device profiles; it supplies also a set of services to the clients.

2.1 The XAMAS Client Agent

A *XAMAS* Client Agent CA_{ij} is associated with a device D_i and a user U_j. It is activated at the beginning of a navigation session which U_j carries out by means of D_i; it supports U_j in the activities she/he performs during the session and is deactivated at the end of the session.

After its activation, CA_{ij} requires to the Server Agent SA the current profile UP_j of U_j and retrieves from D_i the device profile DP_i. It stores UP_j and DP_i in its own ontology and exploits them for supporting U_j in her/his navigation activity. In addition, it registers information about the behaviour of U_j during the session.

At the end of the session, CA_{ij} contacts SA and provides it with the collected information about the behaviour of U_j; such an information is exploited by SA for updating the profile of U_j in its own ontology.

Ontology
The ontology of CA_{ij} consists of a tuple $\langle DP_i, UP_j, VisitedC_{ij}, VisitedR_{ij}\rangle$, where DP_i indicates a Device Profile, UP_j is relative to a User Profile, $VisitedC_{ij}$ stores information about the concepts which U_j visited during the current session and $VisitedR_{ij}$ contains information about the relationships existing among concepts belonging to $VisitedC_{ij}$.

The Device Profile DP_i.
DP_i stores the characteristics of the device D_i. More specifically, a tuple $\langle DId_i, B_i, PPB_i, MIS_i, TE_i, AE_i, VE_i, PE_i, SE_i, PC_i\rangle$ is associated with D_i, where: *(i)* DId_i is the *Identifier* of D_i. *(ii)* B_i is the *Bandwidth* of D_i; it denotes the maximum bit rate guaranteed by D_i during the data transmission activities. *(iii)* PPB_i is the *Price Per Byte* of D_i; it indicates the cost for sending or receiving a byte of data by means of D_i. *(iv)* MIS_i is the *Max Item Size* of D_i; it denotes the maximum size that a Web page could have for being accessed by means of D_i. *(v)* TE_i (resp., AE_i, VE_i) is the *Text Enabled* (resp., *Audio Enabled*, *Video Enabled*) field of D_i; it indicates if D_i can handle a text (resp., an audio, a video). *(vi)* PE_i (resp., SE_i) is the *Printing Enabled* (resp., *Storage Enabled*) field; it denotes if D_i is capable of printing (resp., storing) data. *(vii)* PC_i is the *Processor Clock* of D_i; it denotes the CPU clock of D_i.

The User Profile UP_j.
UP_j can be described by a tuple $\langle UId_j, SCounter_j, CSet_j, RSet_j\rangle$. More specifically:

- UId_j is a code identifying U_j.
- $SCounter_j$ counts the number of sessions carried out by U_j.
- $CSet_j = \{C_1, C_2, \ldots, C_p\}$ is the set of concepts interesting for U_j. A tuple $\langle CId_l, Name_l, FVTS_l, LVTS_l, VC_l, ATI_l, PI_l, SI_l, IDegree_l\rangle$ is associated with each concept $C_l \in CSet_j$. Here:
  - CId_l is the *Identifier* of the concept C_l;
  - $Name_l$ is the *Name* of C_l;
  - $FVTS_l$ is the *First Visit Time Stamp* associated with C_l; it represents the first time instant in which some instance of C_l has been accessed;
  - $LVTS_l$ is the *Last Visit Time Stamp* relative to C_l; it indicates the time instant in which U_j concluded to visit the lastly accessed instance of C_l;
  - VC_l is the *Visit Counter* of C_l; it represents the total number of accesses to instances of C_l carried out by U_j;
  - ATI_l is the *Access Time Indicator* associated with C_l. As pointed out in the Introduction, our model takes into account that the user navigation is highly influenced by both the characteristics of the exploited device and the navigation costs. In particular, the time spent by the user in

accessing a concept is not to be considered as an absolute measure, but it is necessary to "normalize" it w.r.t. both the characteristics of the device she/he is exploiting and the navigation costs; ATI_l has been conceived for handling this aspect. More specifically, ATI_l is defined as: $ATI_l = \sum_{k=1}^{SCounter_j} \left(ATI_l^k\right)$, where ATI_l^k is the Access Time Indicator relative to the concept C_l in Session k. It is defined as $ATI_l^k = \frac{T_l^k \times PPB_i^k}{B_i^k \times PC_i^k}$, where T_l^k represents the time U_j spent for visiting instances of C_l during Session k whereas PPB_i^k, B_i^k and PC_i^k denote, respectively, the Price Per Bite, the Bandwidth and the Processor Clock relative to the device D_i exploited by U_j for visiting C_l during the k^{th} Session.

- PI_l is the *Printing Indicator* associated with C_l. It is an indicator of the interest of U_j for C_l. Indeed, it assumes that if, during a session, U_j has a device capable of printing and does not print any instance of C_l, then U_j shows a low interest for C_l. PI_l can be computed as: $PI_l = \frac{\sum_{k=1}^{SCounter_j} PI_l^k}{SCounter_j}$, where PI_l^k is the Printing Indicator relative to the concept C_l in Session k. If PE_i^k indicates the *Printing Enabled* field of the device D_i exploited by U_j in Session k, PI_l^k can be defined as:

$$PI_l^k = \begin{cases} 0.5 \text{ if } PE_i^k = 0 \\ 1 \quad \text{if } PE_i^k = 1 \text{ and at least one instance of } C_l \text{ has been printed in Session } k \\ 0 \quad \text{if } PE_i^k = 1 \text{ and no instance of } C_l \text{ has been printed in Session } k \end{cases}$$

  Observe that, if $PE_i^k = 0$, it is not possible to derive any information about the desire of U_j of printing an instance of C_l during Session k. This uncertainty is expressed by setting PI_l^k to 0.5.
- SI_l is the *Storage Indicator* associated with C_l. It is analogous to the *Printing Indicator* and can be computed as: $SI_l = \frac{\sum_{k=1}^{SCounter_j} SI_l^k}{SCounter_j}$, where:

$$SI_l^k = \begin{cases} 0.5 \text{ if } SE_i^k = 0 \\ 1 \quad \text{if } SE_i^k = 1 \text{ and at least one instance of } C_l \text{ has been stored in Session } k \\ 0 \quad \text{if } SE_i^k = 1 \text{ and no instance of } C_l \text{ has been stored in Session } k \end{cases}$$

  Here, SE_i^k is the *Storage Enabled* field of the device D_i exploited by U_j during Session k.
- $IDegree_l$ denotes the *Degree of Interest* of U_j for C_l. In order to define a formula for computing $IDegree_l$ the following observations have been taken into account [6]: *(i)* The interest that a user shows for a concept C_l is strictly related to the time she/he spent for visiting the various instances of C_l. As a consequence, $IDegree_l$ is directly proportional to ATI_l. *(ii)* The interest of a user for a concept C_l grows with the growing of the number of accesses to the instances of C_l. Therefore, $IDegree_l$ is directly proportional to VC_l. *(iii)* The more remote (resp., recent) the First Visit Time Stamp (resp., the Last Visit Time Stamp) is, the

higher the Interest Degree is. A good indicator of this intuition is given by the ratio $\frac{LVTS_l - FVTS_l}{TS - FVTS_l}$, where $FVTS_l$ (resp., $LVTS_l$) indicates the First Visit Time Stamp (resp., Last Visit Time Stamp) of C_l and TS denotes the current Time Stamp. *(iv)* The interest that a user shows for a concept C_l is directly proportional to both the Printing Indicator PI_l of C_l and the Storage Indicator SI_l of C_l.
From the considerations outlined above, we deduce that $IDegree_l$ can be computed as:

$$IDegree_l = ATI_l \times VC_l \times \left(\frac{LVTS_l - FVTS_l}{TS - FVTS_l}\right) \times PI_l \times SI_l$$

– $RSet_j$ represents the set of relationships existing among concepts belonging to $CSet_j$. In our model, a relationship R_{st} between two concepts C_s and C_t indicates that, in the past, U_j accessed an instance of C_t (*Target Concept*) via an hyperlink present in an instance of C_s (*Source Concept*). A tuple $\langle RId_{st}, C_s, C_t, TC_{st}\rangle$ is associated with each relationship $R_{st} \in RSet_j$. Here: *(i)* RId_{st} is the *Identifier* of the relationship R_{st}; *(ii)* C_s is the *Source Concept* of R_{st}; *(iii)* C_t is the *Target Concept* of R_{st}; *(iv)* TC_{st} is the *Transition Counter* of R_{st}; it specifies how many times an instance of C_t has been accessed via an instance of C_s.

The Set $VisitedC_{ij}$.
The set $VisitedC_{ij}$ stores information about the concepts which U_j visited during the current session. A tuple $\langle CId_l, Name_l, FVTS'_l, LVTS'_l, VC'_l, ATI'_l, PI'_l, SI'_l\rangle$ is associated with each concept $C_l \in VisitedC_{ij}$. The parameters $CId_l, Name_l$, $FVTS'_l, LVTS'_l$ and VC'_l are the same as those we have seen previously for concepts of $CSet_j$, whereas the parameters ATI'_l, PI'_l and SI'_l are analogous to the corresponding ones we have seen for concepts of $CSet_j$ except that they refer to one single session. In particular $ATI'_l = \frac{T_l \times PPB_i}{B_i \times PC_i}$, and

$$PI'_l = \begin{cases} 0.5 \text{ if } PE_i = 0 \\ 1 \quad \text{if } PE_i = 1 \text{ and at least one instance of } C_l \text{ has been printed in the current session} \\ 0 \quad \text{if } PE_i = 1 \text{ and no instance of } C_l \text{ has been printed in the current session} \end{cases} \qquad SI'_l = \begin{cases} 0.5 \text{ if } SE_i = 0 \\ 1 \quad \text{if } SE_i = 1 \text{ and at least one instance of } C_l \text{ has been stored in the current session} \\ 0 \quad \text{if } SE_i = 1 \text{ and no instance of } C_l \text{ has been stored in the current session} \end{cases}$$

In the previous formulas, PPB_i, B_i, PC_i, PE_i and SE_i are the characteristics of the device D_i exploited by U_j in the current session, whereas T_l is the total time U_j spent in visiting instances of C_l in the current session.

The Set $VisitedR_{ij}$.
$VisitedR_{ij}$ contains information about the relationships existing among concepts belonging to $VisitedC_{ij}$. For each relationship $R_{st} \in VisitedR_{ij}$, a tuple $\langle RId_{st}, C_s, C_t, TC_{st}\rangle$ is defined; the parameters relative to the tuple are the same as those described previously for relationships of $RSet_j$.

In order to simplify the description of the ontology of CA_{ij}, we have represented it by means of abstract data structures, such as sets; however, as previ-

ously pointed out, it is actually stored as an XML document; its XML Schema is shown in Figure 1.

```
<?xml version="1.0" >
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:simpleType name=''YesNoType''>
       <xs:restriction base=''xs:NMTOKEN''>
          <xs:enumeration value=''YES''/>
          <xs:enumeration value=''NO''/>
       </xs:restriction>
    </xs:simpleType>
    <xs:complexType name=''Device_ProfileType''>
       <xs:attribute name=''DeviceIdentifier''
          type=''xs:ID'' use=''required''/>
       <xs:attribute name=''Bandwidth''
          type=''xs:nonNegativeInteger'' use=''required''/>
       <xs:attribute name=''PricePerByte'' type=''xs:float''
          use=''required''/>
       <xs:attribute name=''MaxItemSize''
          type=''xs:nonNegativeInteger'' use=''required''/>
       <xs:attribute name=''TextEnabled'' type=''YesNoType''
          use=''required''/>
       <xs:attribute name=''AudioEnabled'' type=''YesNoType''
          use=''required''/>
       <xs:attribute name=''VideoEnabled'' type=''YesNoType''
          use=''required''/>
       <xs:attribute name=''PrintingEnabled''
          type=''YesNoType'' use=''required''/>
       <xs:attribute name=''StorageEnabled''
          type=''YesNoType'' use=''required''/>
       <xs:attribute name=''ProcessorClock''
          type=''xs:nonNegativeInteger'' use=''required''/>
    </xs:complexType>
    <xs:complexType name=''ConceptType''>
       <xs:attribute name=''ConceptIdentifier''
          type=''xs:ID'' use=''required''/>
       <xs:attribute name=''Name'' type=''xs:string''
          use=''required''/>
       <xs:attribute name=''FirstVisitTimeStamp''
          type=''xs:dateTime'' use=''required''/>
       <xs:attribute name=''LastVisitTimeStamp''
          type=''xs:dateTime'' use=''required''/>
       <xs:attribute name=''VisitCounter''
          type=''xs:nonNegativeInteger'' use=''required''/>
       <xs:attribute name=''AccessTimeIndicator''
          type=''xs:float'' use=''required''/>
       <xs:attribute name=''PrintingIndicator''
          type=''xs:float'' use=''required''/>
       <xs:attribute name=''StorageIndicator''
          type=''xs:float'' use=''required''/>
       <xs:attribute name=''InterestDegree''
          type=''xs:float''/>
    </xs:complexType>
    <xs:complexType name=''RelationshipType''>
       <xs:simpleContent>
          <xs:extension base=''xs:string''>
             <xs:attribute name=''RelationshipIdentifier''
                type=''xs:ID'' use=''required''/>
             <xs:attribute name=''Source'' type=''xs:IDREF''
                use=''required''/>
             <xs:attribute name=''Target'' type=''xs:IDREF''
                use=''required''/>
             <xs:attribute name=''TransitionCounter''
             type=''xs:nonNegativeInteger'' use=''required''/>
          </xs:extension>
       </xs:simpleContent>
    </xs:complexType>
    <xs:complexType name=''User_ProfileType''>
       <xs:sequence>
          <xs:element name=''Concept'' type=''ConceptType''
             minOccurs=''0'' maxOccurs=''unbounded''/>
          <xs:element name=''Relationship''
             type=''RelationshipType'' minOccurs=''0''
             maxOccurs=''unbounded''/>
       </xs:sequence>
       <xs:attribute name=''UserIdentifier'' type=''xs:ID''
          use=''required''/>
       <xs:attribute name=''SessionCounter''
          type=''xs:nonNegativeInteger'' use=''required''/>
    </xs:complexType>
    <xs:complexType name=''Visited_CType''>
       <xs:sequence>
          <xs:element name=''Concept'' type=''ConceptType''
             minOccurs=''0'' maxOccurs=''unbounded''/>
       </xs:sequence>
    </xs:complexType>
    <xs:complexType name=''Visited_RType''>
       <xs:sequence>
          <xs:element name=''Relationship''
             type=''RelationshipType'' minOccurs=''0''
             maxOccurs=''unbounded''/>
       </xs:sequence>
    </xs:complexType>
    <xs:element name=''XAMAS-Client-Ontology''>
       <xs:complexType>
          <xs:sequence>
             <xs:element name=''Device_Profile''
                type=''Device_ProfileType''/>
             <xs:element name=''User_Profile''
                type=''User_ProfileType''/>
             <xs:element name=''Visited_C''
                type=''Visited_CType''/>
             <xs:element name=''Visited_R''
                type=''Visited_RType''/>
          </xs:sequence>
       </xs:complexType>
    </xs:element>
</xs:schema>
```

Fig. 1. The XML Schema of a *XAMAS* Client

Behaviour

The behaviour of CA_{ij} consists of five steps:

Step 1. It derives the profile of D_i from D_i itself and stores it in its ontology.

Step 2. It contacts the Server Agent SA for obtaining the profile of U_j. If U_j is already registered in its current ontology, SA returns the current UP_j. Otherwise, SA registers U_j in its ontology, creates an empty UP_j and returns it to CA_{ij}. CA_{ij} stores in its ontology the profile of U_j, as returned by SA.

Step 3. It exploits information stored in its ontology for supporting U_j in her/his activities; in other words, it must be capable of recommending U_j, acting as a Content Based Recommender System [1]; this task is presented in Section 3. It is worth pointing out again that, differently from most of the Recommender Systems already proposed in the literature, CA_{ij} takes into account the device exploited by the user for the navigation and, therefore, it adapts its suggestions also to the characteristics of the device.

Step 4. It monitors U_j in her/his activities during the whole session and stores in its ontology information about the sets of concepts and relationships she/he visited. In particular, assume that U_j accesses a concept C_t via a concept C_s. Assume, also, that the *First Visit Time Stamp*, the *Last Visit Time Stamp*, the *Access Time Indicator*, the *Printing Indicator* and the *Storage Indicator* associated with C_t are $FVTS_t$, $LVTS_t$, ATI_t, PI_t and SI_t. First CA_{ij} must verify if C_t is already present in $VisitedC_{ij}$. Assume that C_t is already present in $VisitedC_{ij}$ where it is associated with the tuple $\langle CId_t, Name_t, FVTS^*, LVTS^*, VC^*, ATI^*, PI^*, SI^* \rangle$, then, CA_{ij} must set $LVTS^* = LVTS_t$, $VC^* = VC^* + 1$, $ATI^* = ATI^* + ATI_t$, $PI^* = max(PI^*, PI_t)$, $SI^* = max(SI^*, SI_t)$. If C_t is not already present in $VisitedC_{ij}$, CA_{ij} must insert C_t, along with the tuple $\langle CId_t, Name_t, FVTS_t, LVTS_t, 1, ATI_t, PI_t, SI_t \rangle$, into $VisitedC_{ij}$. If a relationship R_{st} connecting C_s to C_t, characterized by the tuple $\langle RId_{st}, C_s, C_t, TC_{st} \rangle$, already exists in $VisitedR_{ij}$, CA_{ij} must increment TC_{st} of one unit. Vice versa, if no relationship from C_s to C_t exists in $VisitedR_{ij}$, then the tuple $\langle RId_{st}, C_s, C_t, 1 \rangle$ must be inserted into $VisitedR_{ij}$.

Step 5. At the end of the session, CA_{ij} contacts SA and provides it with UId_j, DP_i, $VisitedC_{ij}$ and $VisitedR_{ij}$.

2.2 The XAMAS Server Agent

The *XAMAS Server Agent SA* is always active. It stores the profiles of users exploiting the system as well as the profiles of available devices. It is capable of performing the following tasks: *(i)* insertion, deletion and update of a user profile; *(ii)* insertion, deletion and update of a device profile; *(iii)* information exchange with the *XAMAS* Client Agents.

Ontology

The ontology of SA consists of a pair $\langle DPSet, UPSet \rangle$, where $DPSet$ is a set of Device Profiles and $UPSet$ is a set of user profiles. More specifically:

- $DPSet = \{DP_1, DP_2, \ldots, DP_n\}$ where DP_i is the profile of the device D_i. DP_i stores information describing the characteristics of the device D_i. A tuple $\langle DId_i, B_i, PPB_i, MIS_i, TE_i, AE_i, VE_i, PE_i, SE_i, PC_i \rangle$ is associated with a Device Profile DP_i. The components of this tuple have the same semantics as those introduced in Section 2.1.
- $UPSet = \{UP_1, UP_2, \ldots, UP_n\}$, where UP_j is the profile of the user U_j. UP_j can be described by a tuple $\langle UId_j, SCounter_j, CSet_j, RSet_j \rangle$ where UId_j, $SCounter_j$, $CSet_j$ and $RSet_j$ have the same structure and semantics as the corresponding ones described in Section 2.1.

As for the Client Agent ontology, also the Server Agent ontology is stored as an XML document; the corresponding XML Schema is similar to that shown in Figure 1; we do not show it for space limitations.

Behaviour
The Server Agent SA can be contacted by a Client Agent CA_{ij} at the beginning of a navigation session for obtaining the profile of the associated user U_j. If U_j is already registered in the ontology of SA, it returns the current profile of U_j to CA_{ij}. Otherwise, SA first registers U_j in its ontology, associates an identifier UId_j and an empty profile UP_j with her/him, and returns both UId_j and UP_j to CA_{ij}.

SA is contacted also by CA_{ij} at the end of a navigation session; in this case, CA_{ij} provides SA with $DP_i, UId_j, VisitedC_{ij}$ and $VisitedR_{ij}$; SA must update its ontology accordingly. More specifically, first SA examines DP_i. If this profile is already stored in its ontology and some of its parameters have been changed, SA must carry out the suitable updates. Otherwise, if DP_i is not already present in its ontology, SA must insert it into $DPSet$.

After this, SA must update information about UP_j. First it examines all concepts in $VisitedC_{ij}$. Let C_t be one of them and let $\langle CId_t, Name_t, FVTS_t, LVTS_t, VC_t, ATI_t, PI_t, SI_t\rangle$ be the corresponding tuple. Assume that also C_t belongs to $CSet_j$; let $\langle CId_t, Name_t, FVTS^*, LVTS^*, VC^*, ATI^*, PI^*, SI^*, IDegree^*\rangle$ be the tuple associated with C_t in $CSet_j$ and let $SCounter_j$ be the current number of sessions carried out by U_j. Then, SA sets $LSVT^* = LSVT_t$, $VC^* = VC^* + VC_t$, $ATI^* = ATI^* + ATI_t$, $PI^* = \frac{PI^* \times SCounter_j + PI_t}{SCounter_j + 1}$, $SI^* = \frac{SI^* \times SCounter_j + SI_t}{SCounter_j + 1}$, $IDegree^* = ATI^* \times VC^* \times \left(\frac{LVTS^* - FVTS^*}{TS - FVTS^*}\right) \times PI^* \times SI^*$.

Vice versa, if C_t does not belong to $CSet_j$ then it, and the corresponding tuple in $VisitedC_{ij}$, must be inserted into $CSet_j$.

After all concepts of $VisitedC_{ij}$ have been examined, SA processes $VisitedR_{ij}$. Let R_{st} be a relationship of $VisitedR_{ij}$ and let $\langle RId_{st}, C_s, C_t, TC_{st}\rangle$ be its corresponding tuple. Assume that a relationship R^*_{st} from C_s to C_t exists in $RSet_j$ and let $\langle RId^*_{st}, C_s, C_t, TC^*_{st}\rangle$ be the corresponding tuple. In this case, SA must set $TC^*_{st} = TC^*_{st} + TC_{st}$. Vice versa, if no relationship exists in $RSet_j$ from C_s to C_t then SA inserts R^*_{st}, along with the corresponding tuple, into $RSet_j$. Finally, the *Session Counter* of U_j is incremented of one unit.

After SA has examined both $VisitedC_{ij}$ and $VisitedR_{ij}$, it must carry out a pruning activity in order to avoid UP_j to undefinitely increase. The pruning policy implemented by SA is the following. A concept C_l can be pruned from UP_j if it has been not particularly interesting for U_j in the nearest past and appears to be not particularly interesting for her/him in the future.

A concept C_l is considered not particularly interesting for the user in the nearest past if, in UP_j, $IDegree_l$ is smaller than a certain threshold th_{Int}.

A concept C_l is considered not particularly interesting for the user in the future if the "distance" between C_l and each of the interesting concepts present in UP_j is greater than th_{Int}. The notion of "distance" between two concepts C' and C'' is defined as follows. Consider a graph G_{UP_j} such that: *(i)* each concept in UP_j is represented by a node in G_{UP_j}; *(ii)* each relationship R_{st} from C_s to C_t is represented by an arc from the node corresponding to C_s to the node corresponding to C_t in G_{UP_j}. The distance between two concepts C' and C'' is

the length of the minimum path in G_{UP_j} from the node corresponding to C' to the node corresponding to C''.

Clearly, if a concept is pruned from UP_j, all relationships involving it are pruned from UP_j too.

3 The ec-XAMAS Component Managing the E-commerce Site Activities

As previously pointed out, in the *ec-XAMAS* platform, an agent is present in each e-commerce site, handling the information stored therein, as well as the interaction with the customers. In the next sub-sections we examine its ontology and behaviour.

3.1 Ontology

The ontology of the site agent stores the content of the corresponding site and can be represented as a set of concepts and a set of relationships among concepts. It is worth pointing out that such a kind of representation is general and valid for each typology of sites (e.g., e-commerce, e-learning, e-government) and representation formats (e.g., relational databases interfaced with HTML, XML). In the e-commerce context we are considering in this paper, a concept can be specialized into "product/service typology" and, therefore, relationships among concepts represent relationships existing among product/service typologies[1]. Analogously to the other agents we have introduced previously, also the ontology of the Site Agent is stored as an XML document. However, due to space constraints we do not show it.

In particular, each concept C_l is associated with a tuple $\langle CId_l, Name_l, MaxS_l, DT_l \rangle$, where CId_l is the identifier of the concept C_l, $Name_l$ is its name, $MaxS_l$ is the maximum size (in bytes) that a Web page describing the product/service typology corresponding to the concept might have, and DT_l indicates the Data Type of C_l, i.e., it denotes whether C_l is represented as a text, an audio or a video in the site. Analogously, each relationship R_{st} is associated with a tuple $\langle RId_{st}, C_s, C_t \rangle$, where RId_{st} is the identifier of R_{st}, C_s is the source concept and C_t is the target concept. There is a relationship from C_s to C_t if, in the Web page describing C_s, there is at least one link to the Web page describing C_t.

The ontology of the Site Agent is created when the corresponding site is made available over the Internet; such a task is carried out by a software module which examines the site and constructs the catalogue of both the available product typologies and their relationships. The same software module is activated also for suitably modifying the ontology whenever a concept is either created or removed or updated in the corresponding site.

[1] Clearly, in an e-commerce context, also concepts stored in the customer profile of *XAMAS* Clients and server represent product typologies and, consequently, the consistency of our model is guaranteed.

3.2 Behaviour

We are now able to examine the bahaviour of the Site Agent and, more in general, of the whole *ec-XAMAS* platform.

Let $CustSet = \{c_1, c_2, \ldots, c_n\}$ be a set of customers and let $SiteSet = \{s_1, s_2, \ldots, s_m\}$ be a set of e-commerce sites; assume that each customer of $CustSet$ and each site of $SiteSet$ have an associated agent and let $CustAgSet = \{CustAg_1, \ldots, CustAg_n\}$ and $SiteAgSet = \{SiteAg_1, \ldots, SiteAg_m\}$ be the sets of these agents.

Each $CustAg_p$ is a *XAMAS* Client; in particular, if, during a session, a customer c_j is exploiting a device D_i for carrying out her/his e-commerce activities, $CustAg_p$ is a *XAMAS* Client CA_{ij}. Such an architecture leads to the following consequences: *(i)* if, in different sessions, a customer exploits different devices for navigation, then different *XAMAS* Clients are associated with her/him; *(ii)* all involved customers and devices are registered in a *XAMAS* Server SA.

Let c_p be a customer and let s_q be an e-commerce site. We assume that a Customer Agent $CustAg_p$ (resp., a Site Agent $SiteAg_q$) is associated with c_p (resp., s_q) and that the ontology of $CustAg_p$ (resp., $SiteAg_q$) is stored in the XML document $XOnt_p$ (resp., $XOnt_q$).

For each visit of c_p to s_q, the following four steps are carried out:

Step 1. $SiteAg_q$ sends to $CustAg_p$ a copy of $XOnt_q$. Note that the choice to exploit XML for handling both agent ontologies and agent communications makes sending an ontology from an agent to another one extremely simple and direct.

Step 2. When $CustAg_p$ receives $XOnt_q$, it selects the subset CS of the concepts stored therein appearing to be interesting for c_p, according to her/his profile. More specifically, each concept C_l, such that it, or one of its synonyms, is also in $XOnt_p$, is inserted into CS[2]. For each relationship $\langle RId_{st}, C_s, C_t\rangle$ of $XOnt_q$ such that C_s, or one of its synonyms, belongs to $XOnt_p$, C_t is added to CS. In this way, CS captures not only the concepts that could be appealing to c_p, according to her/his past interests, but also related concepts that could interest c_p in the future and whose existence she/he possibly ignored.

After this, each concept C_l in CS is ranked, i.e., a coefficient $Rank_l$ is associated with it. $Rank_l$ indicates how much C_l could be of interest for c_p; it takes into account both the previous interest degree c_p has shown for C_l and the device D she/he is exploiting in the current session. Assume that the tuple $\langle DId, B, PPB, MIS, TE, AE, VE, PE, SE, PC\rangle$ describes D in $XOnt_p$. Assume, moreover, that the tuple $\langle CId_l, C_l, MaxS_l, DT_l\rangle$ is associated with C_l in $XOnt_q$ and that $IDegree_l$ is the Interest Degree of the user for C_l, stored in the profile of c_p (if neither C_l nor one of its synonyms are present in $XOnt_p$, $IDegree_l$ is set to 0). $Rank_l$ is computed as follows:

$$Rank_l = DTCoeff(DT_l, TE, AE, VE) \times (1 + IDegree_l) \times \frac{B \times PC}{MaxS_l \times PPB}$$

where,

[2] In order to obtain concept synonymies $CustAg_p$ exploits existing tools such as *WordNet* [11] or *MindNet* [13].

$$DTCoeff(DT_l, TE, AE, VE) = \begin{cases} 0 \text{ if } DT_l = \texttt{TEXT} \text{ (resp., } \texttt{AUDIO, VIDEO}) \\ \quad \text{and } TE = 0 \text{ (resp., } AE = 0, VE = 0) \\ 1 \text{ otherwise} \end{cases}$$

Here, $DTCoeff$ is a coefficient indicating if the data type associated with the concept C_l can be processed by the device D which c_p is exploiting. The term $\frac{B \times PC}{MaxS_l \times PPB}$ is a coefficient which measures the capability of D in downloading and handling instances of C_l.

After a rank has been associated with each concept of CS, $CustAg_p$ sends the ranked CS to $SiteAg_q$.

Step 3. $SiteAg_q$ selects only the portions of the associated e-commerce site containing instances of the concepts stored in CS and sends them to $CustAg_p$.

Step 4. $CustAg_p$ visualizes them to c_p in order to let her/him to carry out her/his choices; the visualization policy takes into account the ranking of the concepts in CS. Whenever c_p makes a choice, and therefore accesses a concept stored in s_q, her/his profile is updated, according to the technique illustrated in Section 2.1, for registering this event.

4 Conclusions

In this paper we have proposed *ec-XAMAS*, an XML-based and adaptive multi-agent system for supporting a customer, visiting an e-commerce site, in the search of products and/or services present therein and appearing to be appealing according to her/his past interests and behaviour.

We have seen that, in *ec-XAMAS*, three typologies of agents are present, namely: *(i)* a *Client Agent*, associated with a specific customer, operating on a specific device, which supports her/him during a navigation session; *(ii)* a *Server Agent*, which stores the profiles of both the involved customers and devices; *(iii)* a *Site Agent*, associated with an e-commerce site, which handles both the information stored therein and the interaction with the customers.

We have shown that *ec-XAMAS* is adaptive w.r.t. the profile of both the customer and the device she/he is exploiting for visiting the site. Finally, we have seen that it is XML-based since XML is exploited for both storing the agent ontologies and handling the agent communication.

As for future work, we plan to study the possibility to enrich the proposed multi-agent model with other features capable of improving its effectiveness and completeness in supporting the various activities related to e-commerce. As an example, it may be interesting to categorize involved customers on the basis of their profiles, as well as involved e-commerce sites on the basis of their contents.

We plan also to make the proposed model even more adaptive by considering the possibility to adapt its behaviour on the basis not only of the device a customer is exploiting during a navigation session but also of the context (e.g., job, holidays) in which she/he is operating.

Acknowledgment. This work was partially funded by the Information Society Technologies programme of the European Commission, Future and Emerging Technologies under the IST-2001-33570 INFOMIX project and the IST-2001-37004 WASP project.

References

1. Communications of ACM,"Recommender Systems". volume 40(3). ACM, 1997.
2. Communications of ACM,"Adaptive Web". volume 45(5). ACM, 2002.
3. C. R. Anderson, P. Domingos, and D. S. Weld. Adaptive web navigation for wireless devices. In *Proc. of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI 2001)*, pages 879–884, Seattle, Washington, USA, 2001. Morgan Kaufmann.
4. L. Ardissono, A. Goy, G. Petrone, M. Segnan, L. Console, L. Lesmo, C. Simone, and P. Torasso. Agent technologies for the development of adaptive web stores. In *Agent Mediated Electronic Commerce, The European AgentLink Perspective*, pages 194–213. Lecture Notes in Computer Science, Springer-Verlag, 2001.
5. A.G. Buchner and M.D. Mulvenna. Discovering internet marketing intelligence through online analytical web usage mining. *SIGMOD Record*, 27(4): 54–61, 1998.
6. P.K. Chan. A non-invasive learning approach to building web user profiles. In *Proc. of KDD-99 Workshop on Web Usage Analysis and User Profiling (WebKDD'99)*, pages 7–12, San Diego, California, USA, 1999. Lecture Notes in Computer Science, Springer.
7. F.J. Garcia, F. Paternò, and A.B. Gil. An adaptive e-commerce system definition. In *Proc. of International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH'02)*, pages 505–509, Malaga, Spain, 2002. Lecture Notes in Computer Science, Springer-Verlag.
8. R.J. Glushko, J.M. Tenenbaum, and B. Meltzer. An xml framework for agent-based e-commerce. *Communications of the ACM*, 42(3): 106–114, 1999.
9. R.H. Guttman, A.G. Moukas, and P.Maes. Agent-mediated electronic commerce: A survey. *Knowledge Engineering Review*, 13(2): 147–159, 1998.
10. R. Lau, A. Hofstede, and P. Bruza. Adaptive profiling agents for electronic commerce. In *Proc. of the CollECTeR Conference on Electronic Commerce (CollECTeR 2000)*, Breckenridge, Colorado, USA, 2000.
11. A.G. Miller. WordNet: A lexical database for English. *Communications of the ACM*, 38(11): 39–41, 1995.
12. J.S. Park and R.S. Sandhu. Secure cookies on the web. *IEEE Internet Computing*, 4(4): 36–44, 2000.
13. S.D. Richardson, W.B. Dolan, and L. Vanderwende. MindNet: acquiring and structuring semantic information from text. In *Proc. of International Conference on Computational Linguistics (COLING-ACL'98)*, pages 1098–1102, Montreal, Quebec, Canada, 1998. Morgan Kaufmann.
14. B.M. Sarwar, G. Karypis, J.A. Konstan, and J. Riedl. Analysis of recommendation algorithms for e-commerce. In *Proc. of ACM Conference on Electronic Commerce (EC-00)*, pages 158–167, Minneapolis, Minnesota, USA, 2000. ACM Press.
15. J.B. Schafer, J.A. Konstan, and J. Riedl. E-commerce recommendation applications. *Data Mining and Knowledge Discovery*, 5(1-2): 115–153, 2001.
16. U. Shardanand and P. Maes. Social information filtering: Algorithms for automating "word of mouth". In *Proc. of the Computer-Human Interaction Conference (CHI'95)*, pages 210–217, Denver, Colorado, 1995. Addison-Wesley.
17. O.R. Zaiane, M. Xin, and J. Han. Discovering web access patterns and trends by applying OLAP and data mining technology on web logs. In *Proc. of the IEEE Forum on Reasearch and Technology Advances in Digital Libraries (ADL'98)*, pages 19–29, Santa Barbara, California, USA, 1998. IEEE Computer Society.