Intelligent Agents and XML - A method for accessing webportals in both B2C and B2B E-Commerce

Mühlbacher, Jörg R., Reisinger, Susanne, Sonntag, Michael

Institute for Information Processing and Microprocessor Technology (FIM), Altenbergerstr. 69, A-4040 Linz, Austria
{muehlbacher, reisinger, sonntag}@fim.uni-linz.ac.at

Abstract. In E-Commerce today webportals are important and also intelligent agents grow in significance. Our approach is to combine them in designing webportals and interfaces for both users and agents. In this paper we discuss the problems in automatically accessing portals and possible solutions for them through using OOM methods. The solution selected by us, using an XML-based standard and dynamically reconfigurable protocols, is described afterwards and the methods used are shown. Afterwards we briefly present an example, a webportal for sports information.

Keywords: Agents, OOM, OOP, XML, E-Commerce, Webportals

1 Introduction

Both web portals and intelligent agents are important factors in the Internet and of growing importance especially in E-Commerce. However, combining these two is not that easy. The main problem is how agents retrieve information from a webpage, which is formatted for reading by people. Through the combination of using ebXML [2] and applying methods for object-oriented design on all parts involved, this gap can be at least ameliorated. We propose to use object-oriented modeling-techniques not only for the implementation of the software, but also for the design of the data structure to be exchanged including dynamic aspects of protocols.

2 Problems of automatically accessing webportals

Webportals provide a unified access to a large set of information on certain topics. However, different portals contain different content data and different methods of access. It is therefore hard for customers to locate and buy the information or goods they are interested in, as the methods and relative location change with each supplier. At the same time, a unified portal or organization is unrealistic (and probably not suited for all types of content). This is partly done by providers on purpose to avoid competition through complicating comparisons and therefore binding customers (if they can handle a portal, they will not move to another one, where they have to learn the use anew), both of which are especially important in B2C E-Commerce. In B2B

E-Commerce a difficulty is automation of procurement: Many goods can be bought cheaper or faster on the WWW, however the work has to be done completely manually every time (in contrast to this in conventional procurement pre-created forms and standardized procedures can be used). Therefore the need arises for an a) unified method for locating, accessing and buying information, which can be b) automated to a large degree, even including payment [11]. Using the combination of storing data in XML [4] for presentation on webpages and including the possibility for access by intelligent agents, both difficulties can be overcome, leading to an enhancement of E-Commerce.

Specific problems of the current design are:

? Webpages are designed for different user groups having distinct interests and varying habits, and according to their content. Also the behavior of the users desiring information differs. Shoppers often want detailed information on a specific product in a fast and easy way (B2B), while visitors of e. g. sport portals just want to browse or become generally informed (B2C).
? Because of diverging interests automatic access to webportals through programs is complicated, as there is no standardization how data is presented, where data is located on a webpage, or which categories define the organization of the website.
? The HTTP-protocol is not ideally suited for automatic access because the connection is closed after each request, passing of parameters is complicated, and negotiations are not possible.
? The content data of the communication introduces difficulties: Both the syntax and especially the semantics (more of a problem in automatic access than by humans) must be the same on both sides of the communication link to allow meaningful interaction.

Providers may not necessarily be delighted by this as they will probably face more competition [3]. However, they can also profit, e.g. through reaching more interested users or being able to present data in a special way not only for one specific group (agents) but also for different user groups as well (Personalization; different preferences or interests, people with disabilities, etc.).

3 Steps towards solutions

A possible solution would be specifying and using a custom protocol for accessing data. On the one hand the advantage of this solution is that all parties need only understand and implement one – the then common – protocol. On the other hand this solution has the disadvantage that a standard must be specified that is suitable for all areas and all users. In addition, providers must agree upon it. Moreover, a single standard coping with everything would be either very complicated, or useable for special tasks only with difficulties.

Using a binary protocol for the exchange of data is another possibility. This could be a protocol based on a certain method of serializing data, like Java object serialization or serializeable MFC objects or a specific program library. The advantage is gained speed in development and processing, and a relatively low required bandwidth

for communication. But the disadvantage is too serious: This works only on one type of system and one platform – it is not platform-independent.

As another possibility we consider mobile agents [1], [9] for searching and retrieving data [12]. They possess some advantages concerning protocols: Communication with other partners (either other agents or webportals) can be automatically adapted by the agents for a meaningful interaction with their surrounding. Therefore they are a good choice for retrieving data when negotiations are necessary. Also, the mobility of agents saves bandwidth because the bulk of the communication is handled locally and only the agent (with compressed and filtered results) needs to be transferred [10].

Agents are also relevant in connection with protocols for payment: Although they are standardized (e. g. SET), unique systems like vouchers, debiting or private E-Cash exist (see [15] for an overview of different payment systems in connection with agents). They usually have many things in common (like identification and transfer of some data), but the actual data-objects and the sequence of messages differ. Agents can adapt themselves to these protocols and allow in this way a wider diversity, also improving E-Commerce.

Using XML for the representation of data would be a good basis for retrieving data by the agents and also for the provider of it: An agent can easily extract information from XML as it includes the concept of an explicit definition of the data structure. So no additional transformation before extraction of information (see [5] for an overview of products employing this technique) is required. And using XSL (eXtensible Stylesheet Language) [6] allows different views on and presentations of the same data, a benefit for providers. Even when modeling the content data alone, an object-oriented representation is appropriate. The reuse of components allows agents to understand the data at least partially, while this is impossible when using unique and proprietary definitions.

4 Reconfigurable protocols for information retrieval

Protocols are an important factor in communicating with agents and always should be adapted to the business processes and not the other way round. Our way to provide adaptable state-based protocols is based on a two-fold object-oriented approach: We build a static implementation hierarchy, as well as a dynamic hierarchy of calling other protocols as elements of one. The latter can be stacked to any depth, but no interaction across different levels is possible (subprotocols must terminate before the parent protocol can resume).

This object-oriented approach also allows extensions of existing protocols in two ways: First the protocol is implemented object-oriented and can be extended through subclasses, which overwrite methods (i. e. state transitions) in superclasses. Secondly, providing it with different “plug-in protocols” as subprotocols at runtime to change its behavior in some details (user-defined or negotiated with the partner).

An advantage of this approach is that if an agent does not understand a certain subprotocol, in some cases another one could be substituted (e.g. a known superclass of the unknown one) or the subprotocol simply be left out. This allows an agent to do

transactions at least in a rudimentary way, e.g. without reliable identification of the partner or without using special discounts or options. Another advantage for the developer is that creating protocols dynamically in a hierarchy allows using an object-oriented modeling approach for the protocol itself and not only its implementation.

5 Modeling content data as a class hierarchy with OOM-techniques

For modeling the content data, OOM-techniques [14] are appropriate to use, too. With object-oriented analysis you can easily analyze the content data for possible classes and attributes. These classes and attributes can be transformed into XML representation without the necessity of an additional encoding.

Even though (pure) XML does not offer the concept of object-oriented programming (OOP), with XML-Schema [16] it is possible to work with inheritance and datatypes (like string, float, etc.). Also different namespaces are supported. Including data types further improves the reliability, as the agent can then (in some cases) retrieve information even from unknown data types (e.g. retrieving the price by searching for the only element consisting of the type "Currency").

The advantages of this object-oriented modeling are that agents can retrieve at least some information from the data, even though they cannot interpret the specification of the actual object: understanding one of the superclasses may often suffice. This means for the owner of the agents that he at least gets a feedback and can decide whether it makes sense to give the agent more specific information (or abilities) or not.

Another advantage of OOM [8] is that older classes can be used as components if new data types are required. The specification of classes is also written in XML and therefore can be distributed easily (e.g. in addition to content data, so the recipient can manually interpret them through comments or names, even though the agent cannot).

6 Sample implementation

This system was implemented in Java based on an agent system [13] developed at the institute, which has a special focus on security [7]. The communication is based on ebXML-messages (a set of specifications for using XML for a modular E-Commerce framework with a focus on business processes), but also includes local broadcasts, which are not provided for in the standard. The data transmitted is also modeled in XML. Currently, as a research project a web-portal for sports is under development, which will be also accessible by agents.

As an example, the definition of the content data specified for a member of a portal (called PortalMember) is presented in Fig. 1. First of all, a namespace is declared, where all XML-Schemas and XML-Files are included. Furthermore in this XML-Schema for the data of portal members, the schema for members in general (e.g. portals, clubs, etc.) called MemberData is included, which is used as a "base-class" for PortalMember. PortalMember inherits from MemberData and is extended with additional elements and attributes that are specifically needed for members of portals.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema targetNamespace="http://www.fim.uni-linz.ac.at"
                    xmlns="http://www.fim.uni-linz.ac.at"
                    xmlns:xsd="http://www.w3.org/2000/10/XMLSchema"
                    elementFormDefault="qualified">
<xsd:include schemaLocation="MemberDataDoc.xsd"/>
  <xsd:element name="PortalMember" type="PortalMember" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:complexType name="PortalMember">
     <xsd:complexContent>
       <xsd:extension base="MemberData">
        <xsd:sequence>
          <xsd:element name="Fee" minOccurs="0" maxOccurs="unbounded">
           <xsd:complexType>
             <xsd:sequence>
                <xsd:element name="Amount" type="xsd:double"/>
                <xsd:element name="Currency" type="xsd:string"/>
            </xsd:sequence>
           </xsd:complexType>
        </xsd:element>
        <xsd:element name="DEntrance"  type="xsd:date"/>
        <xsd:element name="DWithDrawal" minOccurs="0" maxOccurs="1" type="xsd:date"/>
        <xsd:element name="UserID"  type="xsd:string"/>
        <xsd:element name="Password"  type="xsd:string"/>
       </xsd:sequence>
       <xsd:attribute name="FeePaid" use="required" value="No">
        <xsd:simpleType >
          <xsd:restriction base="xsd:string" >
            <xsd:enumeration value="Yes"/>
           <xsd:enumeration value="No"/>
          </xsd:restriction>
        </xsd:simpleType>
       </xsd:attribute>
     </xsd:extension>
   </xsd:complexContent>
  </xsd:complexType>
</xsd:schema>
```

Figure 1: Schema for members of portals (Extension of general members)

The example above is used for retrieving information on a portal member (which is only allowed after identification). The agent can use the information provided e. g. for autonomously paying the recurring fee or verifying the personal data.

7 Conclusion

As explained, using XML and hierarchical object-oriented modeling of data is not sufficient, because the semantics also must be specified: Different implementers should not only create compatible programs, they must also adhere to the same semantics, which is of special importance in open systems like the Internet. ebXML with its specification of both syntax and semantics and the storage of them in a public repository is an important step in this direction.

However, even though ebXML is simpler than EDI, it is not trivial (and cannot be because of the complicated requirements it should fulfill). It is therefore sensible to use intelligent agents to support this, providing even more interoperability through automatic adaptation and flexibility through fast adjustment to new requirements and

platforms. But also agents benefit from using XML: Analyzing and interpreting the data gets much easier compared to HTML or text-files.

The combination of using XML for the data and intelligent agents for the processing seems therefore to be ideal, as both benefit from each other. To fully realize them, however, all aspects, static and dynamic ones, need to be modeled and implemented with a view on object orientation. This combination allows extending the user-groups of web-portals to include agents with little work. This includes the benefit of automatic information retrieval. It also addresses the concerns of information robbery by competitors through the possibility for identification and/or payment.

8 Acknowledgement

This paper is a result of a project sponsored by the Austrian National Bank (Project No. 7742) and the country of Upper-Austria (Wi/Ge 201.515/1-2000/Wwin).

References

1. Brenner, W., Zarnekow, R., Wittig, H.: Intelligente Softwareagenten. Grundlagen und Anwendungen. Springer, Berlin (1998)
2. ebXML, http://www.ebxml.org (March 2001)
3. Glushko, R. J., Tenenbaum, J. M., Meltzer, B.: An XML Framework for Agent-based E-Commerce. Communications of the ACM, Vol. 42, No. 3. acm Press, New York (1999)
4. Golfarb, C. F., Prescod, P.: The XML Handbook. Prentice-Hall, New York (1998)
5. Guttman, R., Moukas, A., Maes, P.: Agents as Mediators in Electronic Commerce. In: Klusch, M. (Ed.): Intelligent Information Agents. Agent-Based Information Discovery and Management on the Internet. Springer, Berlin (1999)
6. Holzner, S.: XML Complete. McGraw-Hill, New York (1998)
7. Hörmanseder, R., Sonntag, M.: Mobile agent security based on payment; ACM SIG Operating Systems, Vol. 34, No. 4, New York (2000)
8. Jacobson, I., Ericsson, M., Jacobson, A.: The object advantage - business process reengineering with object technology. Addison-Wesley, New York (1995)
9. Jennings, N., R.: An agent-based approach for building complex software-systems. Communication of the ACM, Vol. 44, No. 4, acm Press, New York (2001)
10. Kotz, D., Gray, R. S.: Mobile Agents and the Future of the Internet. ACM SIG Operating Systems, Vol. 33, No. 3, New York (1999)
11. Mühlbacher, J. R., Sonntag, M.: Teaching Software Engineering and Encouraging Entrepreneurship through E-Commerce. In: Proceedings 2nd International Conference on Innovation Through E-Commerce - IeC99. Manchester (1999)
12. Papazoglou, M., P.: Agent-oriented technology in support of e-business. Communication of the ACM, Vol. 44, No. 4, acm Press, New York (2001)
13. POND – Agent System: http://www.fim.uni-linz.ac.at/Research/Agenten/index.htm
14. Rumbaugh, J.: Object oriented modeling and design. Prentice Hall, New York (1991)
15. Vogler, H., Moschgath, M., Kunkelmann, T.: Enhancing Mobile Agents with Electronic Commerce Capabilities. In: Klusch, M., Weiß, G. (Ed.): Cooperative Information Agents II. Learning, Mobility and Electronic Commerce for Information Discovery on the Internet. Springer, Berlin (1998)
16. W3C: XML Schema: http://www.w3.org/XML/Schema (March 2001)