Chapter XXX

Ontologically Demarcated Resources in an Agent Based Travel Support System

Maciej Gawinecki, Minor Gordon, Ngoc Thanh Nguyen,
Marcin Paprzycki and Zygmunt Vetulani

Further development of the World Wide Web depends on the existence of data stored in a machine-consumable format. This in turn requires the design of domain ontologies, availability of information described with these ontologies, and agents for exploiting this information. In this paper we describe our attempt at designing travel ontology and utilizing it to store data and deliver personalized content to users of an agent-based travel support system.

1 Introduction

Software agents and ontologies are two essential elements of the next generation World Wide Web. Researchers like the author of (Maes 1994) have asserted that software agents can tame information overload by delivering personalized content. For this vision to materialize, information on the Internet must be available in a machine-consumable format, for instance in the form of ontologically-demarcated data (ChefMoz).

In the last 15 years there have been numerous attempts to apply the software agent paradigm to travel services and the pervasive problem of information overload (Maes 1994, Nwana 1999, Paprzycki 2003). The analogy of software agents to travel agents makes the analogy between real world travel agencies and online agent platforms seem trivial, yet the majority of the proposed systems never left the drawing board. The few active experiments in travel-related agent architectures we have discovered have either been limited in scope (Ndumu 1998, Poslad 2001, Suarez 1999) or abandoned. The CRUMPET project is a typical example of what happens with projects aiming at applying agents in travel support systems. CRUMPET was funded by the EU FP5 umbrella between 1999 and 2003, but as of April 2005 it is almost impossible to assess its

achievements, because its WWW site no longer exists and information about the project itself is spread across a few auxiliary sites and conference papers that resulted from it (Poslad 2001). Furthermore, over the last 5 years, a large number of researchers that were originally interested in agent-based systems have moved on to more fashionable ventures such as the Semantic Web or Ambient Computing, leaving behind a trail of papers and unfinished, only partially implemented designs. We have also been trying to develop such a system since 2001, so we can easily understand why these groups have moved on rather then complete a working prototype. Working with existing agent systems is a struggle with cutting edge, constantly evolving and highly unreliable technologies. The design of a system that seems to be quite reasonable one day may become infeasible and/or obsolete sometime before the initial implementation is completed. For instance, we have run into this problem when experimenting with data indexing and the ebXML Registry Repository (ebXML R/R). After many futile attempts we were forced to acknowledge that the existing at that time implementation of the repository was unreliable and incapable of handling "real size" load and had to abandon it, after nine moths of work invested in making the repository a centerpiece of our design (Gilbert 2004, Harrington 2003, Wright 2003).

In its most current design, our system stores semantically demarcated data in a central repository (data gathering rather than the indexing we originally envisioned (Gilbert 2004)) and is expected to deliver an extended travel itinerary, including the standard transportation and accommodation choices as well as restaurants, movie theaters, national parks, historical sites and other points of interest, any of which may be selected by the user from an array of options composed specifically for him/her (content personalization). The system is to be accessible via Internet-enabled devices, ranging from standard PC-based browsers to palmtops and WAP-conversant phones etc., and even non-human entities (such as other agents) (Galant 2002c).

In this paper we devote our attention to the usage of RDF demarcated data in the system. Therefore, we omit and assume to be "successfully addressed" a number of important questions (these questions are however very important and will have to be addressed in the future):

(a) economic model – how such a system will generate revenue for the company that implements it (see e.g. (Brady 2002, Chaudhury 2001, de Kare-Silver 2001, Grover 2001, Mohr 2001, Strass 2001)),

(b) user profiling and clustering (in the context of RFM analysis and cluster analysis) to discover and modify customer segments (see e.g. Galant 2002a, Rud 2001, Simon 2001, Sołtysiak 1998, Zhang 2002)),
(c) methodologies for data mining and modeling in the context of content delivery personalization (see e.g. (Rud 2001, Simon 2001)),
(d) personalized advertisement targeting (see e.g. (Brady2002, Chaudhury 2001, Galant 2002a)),
(e) dealing with conflicting information and, more generally, validating information from unverified Internet sources (see e.g. (Petry 2002)).

To show how the RDF demarcated data became the centerpiece of our system, we proceed as follows. In the next section, we present a high level overview of our system. We follow with the description of hotel and restaurant ontologies that have been developed thus far. In sections 4-6 we illustrate how the RDF demarcated data functions in the content collection, content management and content delivery subsystem. Subsequently we focus on content personalization, and more precisely on user profile representation and utilization. This paper is based on (Gordon 2005, Gawinecki 2005a, Gawinecki 2005b, Gawinecki 2005c)

2 System Overview

The overall architecture of the proposed system is depicted in Figure 1. Before proceeding, let us make two comments: (1) The proposed system belongs to the class of infomediaries and therefore its development will proceed within the framework presented in (Grover 2001); (2) The general system structure is a modification of the skeleton of the general e-commerce system presented in (Galant 2002d). Here, instead of dividing the complete system into two subsystem-spheres: supply and delivery, connected through a communication channel, we are introducing three sub-areas: content collection, content management and content delivery, where the content management subsystem, together with the central data repository are the centerpiece of the system. Let us now briefly summarize each of the components presented in Figure 1.

Verified Content Providers (*VCP*)
Today, a very large number of web sites provide some form of travel-related information. However, as pointed out in (Nwana 1999), there exist a few serious problems arising from the dynamic nature of the Internet-content. Information relevant to travel exists within small independently operated Web sites (consider all restaurants or non-chain hotels that have “private” websites). Since many of these sites are hosted by small local ISP’s, that may not be able to provide sufficient quality of service, these sites may migrate and possibly change their

URL's. In this case information that was once available (its location was known) cannot be found easily. Moreover, even if the information remains at the same location, the site may disappear periodically due to lack of quality of service. The second situation involves progress / changes in Web design. Due to technological changes (e.g. switching from plain HTML to PHP) or a general change in what is perceived to be state of the art in web-design, the layout of a given site may periodically undergo substantial changes, such that agents that were collecting information from that site are no longer able to access it without errors. To deal with this problem and the more general issues of accuracy and relevancy we utilize the concept of *Verified Content Providers* (Abramowicz 2002).

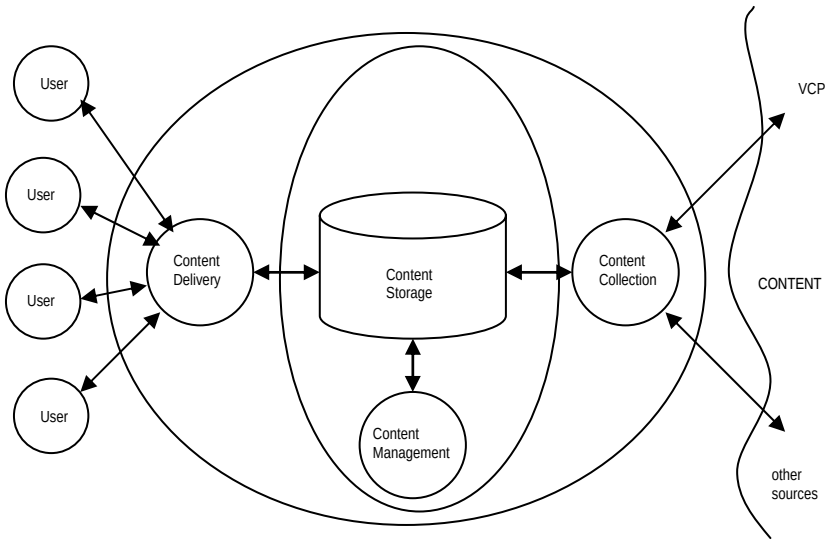

Figure 1. Top-level view of the travel support system

Conceptually, a *Verified Content Provider* (*VCP*) is a site that provides **reliable** and **consistently available** information. We assume that *VCP*s do not randomly appear and disappear from the Internet. Furthermore, *VCP*'s are expected to maintain the same site interface for extended periods of time. It can be even assumed that in the case of a commercial travel support system, contract signed with *VCP*s will require that in case of interface changes owners of travel support system will be informed beforehand. Finally, *VCP*s constantly provide reliable information and thus can be trusted. While the category of *VCP*s is used to mark "the best of breed" information sources, the remaining sources available on the

Internet can be assigned varying levels of trust and that level of trust can dynamically change over time. (e.g. a *VCP* may cease being *verified* if it no longer meets the criteria of trustworthiness while a different source may become a *VCP* by systematically delivering high quality information). Delving into details of designing such an adaptive trust system is outside scope of this paper.

*VCP*s can be divided into two groups: (i) *pushing* – those that "feed" content in a standard form (such as RSS feeds), and (ii) *static* – that require agents to collect pages periodically and extract the necessary information. It is important to stress that currently, existence of *VCP*s containing travel-related semantically demarcated content (except of academic demonstrator systems of minimal breadth and depth of available data; and the ChefMoz (ChefMoz) site containing RDF demarcated data of reasonable, but **not** fully machine consumable quality (Gawinecki 2005a), is just a wish that we hope one day will come true.

Other Sources

There exists a number of problems related to *unverified*, unstructured Internet-based information: (a) its amount, which makes an exhaustive search practically impossible, (b) the unreliability of data, and (c) contradicting sources that require application of sophisticated data deconfliction techniques. While we hope that the approach of relying primarily on trusted data providers (*VCP*s) will alleviate most of these problems (see also (Abramowicz 2000)), we still should not discard additional information available on the Internet and we utilize it whenever possible. This approach will become even more important when the idea of the Semantic Web will start to take hold and semantically demarcated information provided by small-independent sites will be much easier to process automatically.

Content Collection Subsystem (*CCS*)

In the proposed approach, information provided by or collected from the *VCP*s and from other sources is represented in form of RDF triples (RDF) and stored in the Jena system (Jena). Our *Content Collection System* (*CCS*) stores sets of RDF triples that in aggregation represent travel objects (hereafter referred to as *tokens*). Travel tokens originate either from a direct feed from *VCP*s or from an agent subsystem that collects data from the Internet and are stored in the central repository. While the size and the centralized nature of the repository will have important consequences to the scalability of the system, dealing with this problem is outside of the scope of this paper and will have to be addressed only when the system reaches such size that the performance of the repository will become the true performance bottleneck for the whole system. In Section 4 we present details of the functions and structures that comprise the *CCS* subsystem.

Content Management Subsystem (*CMS*)
This subsystem includes all functions related to the data stored in the Jena database. Observe that at least two cases of possibility of data inadequacy have to be taken into account. First, incomplete tokens – for instance, an object to be stored in the repository may contain the name, address and web site of a restaurant, but not the phone number (or some other information defined by our restaurant ontology (Gawinecki 2005a).) In this case such an object is incomplete and we must attempt to find the missing information. The second case involves data that is stored in the system and that is in some way time sensitive, e.g. cinema programs change on Fridays, and thus they have to be updated. Obviously, even information that is not explicitly time-sensitive (e.g. Museum opening times) has to be re-checked in some unspecified time intervals. In Section 5 we discuss this subsystem in more detail.

Content Delivery Subsystem (*CDS*)
Here the data stored in the central repository is manipulated for delivery to end users. The agents in this subsystem obtain a query from the user and work with the CMS to find information matching the user's personal preferences. Data presented to the user may be acquired from the repository, or result from additional Internet searches. We devote Sections 6 to describing this subsystem.

Users
The system will be accessed via Internet-enabled devices, ranging from standard PC-based browsers to palmtops and WAP-conversant phones etc., and even non-human entities (such as other agents) (Kaczmarek 2002, Gordon 2002). It is also the users that are the "reason" for the existence of our system. Therefore in Section 7 we discuss how to create and utilize user profiles when system data is stored as RDF demarcated triples.

As can be seen, the centerpiece of our system is the central repository containing instances of objects defined in travel ontology. Therefore, let us now briefly describe our ontologies of a *hotel* and a *restaurant* (more details can be found in (Gawinecki 2005a)).

3 Hotel and Restaurant Ontology

There exist a number of general ontologies and travel ontologies. Among top-level ontologies one should mention: Cyc, WordNet, SUMO and the SENSUS project, while among travel-related ontologies the most important are: OTA, Mondeca, Travel Game in Agent Cities, Harmonize and DAML-based ontologies. Unfortunately, after an extensive search we were not able to find a

clean and complete ontology of basic travel entities such as a hotel, or a pub. Thus we started by formally defining two basic concepts: *hotel* and *restaurant.*

There are two possible approaches to define ontologies: (1) to start with theoretical considerations, lexical analysis etc., or (2) to start from the application that the ontology is to serve. In our case, travel ontology is to be used to organize information collected from the Internet. Therefore, we have started from the Internet and analyzed web sites of the most popular (top Google ranking) travel sites and as a result of this process decided that the top-level of the proposed *hotel ontology* should consists of the following general classes:

- *site* – containing common characteristics of places to visit,
- *hotel room* – combining types of rooms in particular hotel,
- *amenities* – specifying amenities available within the *site* and in the *room*.

Here, *site* is a very general concept that represents real-world characteristics of places such as hospitals, bars, arenas etc. These properties are inherited by all its subclasses included in the ontology. These subclasses may then have their own specific characteristics, e.g. a *camping site* can have support for campers, while a *hotel site* includes availability of a health center. *Hotel room* defines the concept of the hotel (*hotel = place with rooms for rent*). On the basis of information found on the web we have specified properties defining this class (e.g. room standard or number of rooms in a given hotel). Additionally, we have created a subclass *room*, containing "room information." Finally, *amenities* are based on these available in a given hotel and/or room. Let us now present a “code” snippet (in N3 notation, namespaces omitted) defining the hotel room (suite):

```
:SampleHolidayInnSuiteXXX a hroom:HotelRoom;
       hroom:standardName              "Suite";
       hroom:nbrOfRoomsOfThisType      "30";
       hroom:nbrOfRoomsOfThisType      "25";
       hroom:theAmenities
              <#RegularEconomyAmenities>,
              <#ExtraPayedAmenities>,
              <#SuiteAmenities>.
:RegularEconomyAmenities a amenities:Amenities;
       amenities:content
       "http://EccentricHolidayInnFurniture.html";
       amenities:isStandard                  "1".
:SuiteAmenities a amenities:Amenities;
       amenities:content   "MSAccessDataBaseTable";
       amenities:isSuite               "1".
```

In the next step we have re-engineered the *restaurant* ontology underlying the ChefMoz project (ChefMoz); possibly the only existing large repository of RDF

demarcated, travel-related data. Since there is no explicitly defined ontology provided within ChefMoz we have: brought up the *restaurant* ontology found in the data, and used it to produce a clean dataset (Gawinecki 2005a). What follows is a snippet of "code" defining a café.

```
:Poland_WP_Poznan_13_Kotow__Pub_1065073893 a res:Restaurant;
  loc:streetAddress "ul. Wozna 2/3" ;
  loc:city          "Poznan" ;
  loc:country       "Poland" ;
  res:cuisine       res:BarPubBreweryCuisine ;
  res:hours         "Mon-Fri 11-2; Sat 15-2; Sun 17-2 (or
                    last customer)" ;
  res:locationPath  "Poland/WP/Poznan" ;
  loc:neighbourhood "Stare Miasto : Rynek i Starowka" ;
  res:parsedHours   "0-2,17-24|0-2,11-24|0-2,11-24|0-2,
                    11-24|0-2,11-24|0-2,11-24|0-2,15-24" ;
  loc:phone         "(+48) 604 20 42 85" ;
  loc:state         "WP" ;
  loc:zip           "61-776" ;
  res:title         "13 Kotow, Pub" .
```

Consider the case when a hotel has a restaurant on-site. This leads us to a concept of *near-by facility*. Its origin is, again, in the information available on the Web (let us recall that we develop ontology to organize existing information, and thus **this information "shapes" our ontology**). Typically, hotel sites include information about near-by facilities. Thus we have decided to add it to our *site* class. For sites that do not provide such information, a GIS subsystem (a part of the *CMS*) will be used to localize near-by facilities and augment token information. Thus, ontologies of *hotel* and *restaurant* interact at least: (1) when a *restaurant* is a part of *hotel amenities*, and (2) when they are each-others *near-by facilities*.

Let us now illustrate how the data is **stored** in the system through a part of a *site* class of an imaginary Hilton. Here "exact" values are associated with particular fields in the ontology (see also (Gawinecki 2005a)).

```
:NewSampleHolidayInnHotelXX a vSite:HotelALike;
      vSite:myVCard <#ThisHotelsVCard>;
      vSite:type                              "hotel";
      vSite:maxPeopleGetIn                    "3000";
      vSite:hasEntryFee                       "1";
      vSite:hasParking                        "1";
      vSite:isSeasonal                        "0";
      vSite:petsAllowed                       "1";
      vSite:maxPetSize                        "15",
            (rdfs:label  "inKilograms");
      vSite:facilitiesForDisabled             "1";
```

```
       vSite:meansOfPayment
              "visaCredit, visaDebit, Cheque, Cash";
       vSite:isAllDayOpen                "1";
       vSite:hasAmenities                "1";
       vSite:theAmenities  <#ThisHotelsAmenities>;
       vSite:wayOfReachingThePlace
                           "Motorway 75,
                     from Mimo exit 12 then turn left,
                     from Tito exit 11 turn right";
       vSite:specialOffer  "Pay 2 week-nights stay 3";
       vSite:numberOfFreePlaces          "79";
       vSite:numberOfStars               "3";
       vSite:numberOfAllRooms            "95";
       vSite:nameOfTheFamily             "Holiday          Inn
Hotels";
       vSite:staffLanguages       "Russian, English, Czech";
       vSite:hotelFacilities
              <#SampleFitnessClub>,
              <#SampleCasino>,
              <#SampleKiosk>,
              <#SampleSwimmingPool>,
              <#SampleRestaurant>;
       vSite:nearbyFacilities
              <#SampleThemePark>,
              <#SampleZoo>;
       vSite:typesOfRoomsAvailable
              <#SampleHolidayInnSuiteXX>,
              <#SampleHolidayInnStandardXX>.
```

Let us now look in more details into content collection, management and delivery subsystems, where RDF demarcated data is actually manipulated.

4 Content Collection Subsystem

The structure of the content collection subsystem is presented in Figure 2. Recall that the central repository of our system stores sets of RDF triples – semantically demarcated data tokens. Since currently there exist very few sources of RDF-demarcated data natively available on the Internet, we must extract HTML content from existing web sites and "manually" transform it into RDF triples. Our *Content Collection Subsystem* uses *Wrapper Agents* (*WA*) that interface with various WWW sites, mapping XML- or HTML-demarcated data into RDF triples describing travel objects. Sets of triples are gathered from a Web-site by *WA*s, and assembled into travel tokens, are sent to a *Coordinator Agent* (*CA*), which schedules *WA*s and relays the resulting triples to the Jena database. Note that some of the *VCP*s may, sooner or later, contain and/or start delivering RDF demarcated content directly to the system. In this situation all we have to do is to adjust some of the *WA*s to be able to handle these particular inputs, while the

remaining parts of the *CCS* remain the same. An interesting question will arise when ontologies available on the Internet will be different from our ontology. In this case, ontology matching and resolving techniques have to be applied, but this process is outside of the scope of this paper. For the time being we can assume that in this case selected *WA*s will have to take role of the ontology resolvers. In general, we assume a collection of *WA*s capable of interfacing with an assortment of data sources and delivering to the *CA* RDF demarcated tokens describing various travel resources. Communication between *WA*s and the *CA* occurs through exchange of ACL messages (RDF triples are serialized and send as a content of ACL Inform messages).

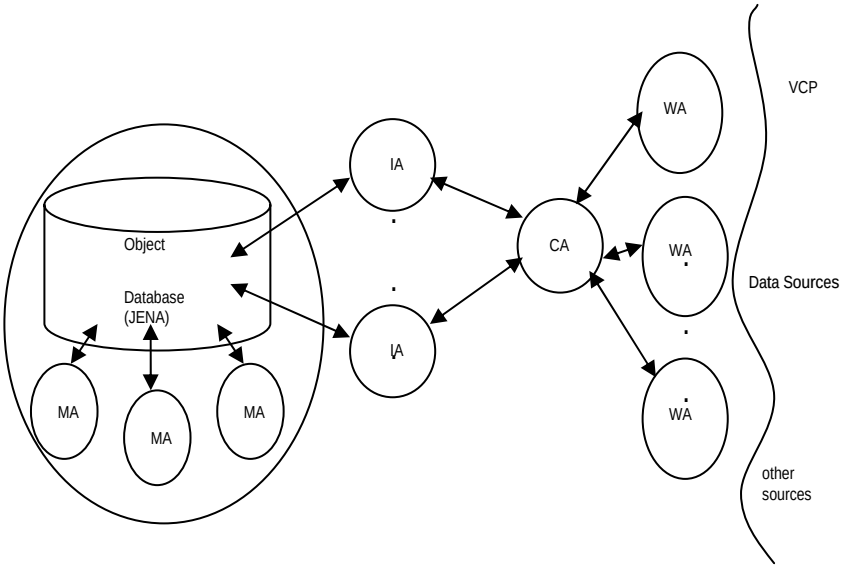

Figure 2. Content management subsystem: WA – wrapper agents, CA – coordinator agent, IA – indexing agents, MA – management agents

The primary role f the *CA* is to act as a large priority queue, where all data objects will be temporarily stored (since in an agent system data objects, such as a queue, have to be encapsulated in an agent). Obviously, having only one *CA* may become a bottleneck and in this case adding *CA*s may alleviate the problem (however, this purely technical problem is outside of the scope of this paper). The *CA* prioritizes the data in the queue. This is done by answering two questions: why is given data taken brought to the system? and, when it will be needed? If the data token is a result of web crawling, it will be assigned “basic” priority. If the token is a result of time-oriented trigger event (for instance that

the theater program is changing) then it will be assigned an "elevated" priority (it is possible that someone will soon request this information and thus it should be acted on expeditiously). Finally, tokens resulting from the "user queries" will be assigned "highest" priority as it is assumed that user is still online and necessary information has to be delivered as soon as possible.

Information is inserted into the repository by a pool of *Indexing Agents* (*IA*). These agents request from the *CA* (via an ACL message) the next data token to work on and obtain it wrapped in an ACL message. *IA*s check the completeness of data and incomplete tokens are marked accordingly before insertion into the repository. Note that such tokens may still be used in a response to a user query (especially when these tokens are widely queried, or they are the only available content pertaining to a given query). Finally, in the case of a tokens containing conflicting information, they are left in the system and marked accordingly for deconfliction to take place.

5 Content Management Subsystem

In our earlier work (Angryk 2002, Galant 2002d), the content management subsystem included both its current functions and the content collection functions described above. As a result of gathered experiences we decided to separate collection and management functions. Content management involves all the agents that operate on data stored **within** the repository. In this way we separate these agents and their functions form the rest of the system.

Thus far we have indicated three possible functions to be performed by the *CMS* agents. The first is related to the completeness of tokens. Incomplete tokens will be marked as such by the *IA*. *CMS* agents will traverse the repository to find incomplete tokens. They will then formulate queries to be answered and request (via an ACL message) from the *CA* that appropriate *WA* be released to search for the missing information. The second situation involves tokens containing conflicting information. They are marked as such by the *IA* and left for the deconflicting agents to deal with. Deconfliction may involve additional queries to the Internet as well as consideration of factors such as, the freshness of data, reliability of sources etc. The third situation deals with time-sensitive data. There is a large amount of travel-related information on the Internet that changes in regular intervals (e.g. programs of operas, theaters, cinemas, etc.). It is possible, for each of these situations, to establish proper time to find updated information. We assume here that the database will generate triggers that will result in agents involved in management of time-sensitive data to communicate with the *CA* to request an update of a given token. However, we must recognize that all data

available in the system is time sensitive. Even content that seems to be relatively "stable," like the restaurant menu or ZOO opening times change periodically. Therefore each data item will have a "time stamp" describing when it was created. After a specified time (different for different travel objects), the database will generate a trigger that will start the update function. Here we observe one of the adaptation mechanisms available in the system. When the update request does not result in changes, the time length before the next update will be increased, while the request resulting in change will cause the update time to be shortened.

Obviously, there may exist other management functions involving data in the central repository. Note, that since our data is stored in an RDF demarcated form, it can be easily extended to incorporate "management oriented" items to be utilized by data management agents. Here, the advantage of an agent system is that in such a case the only required function is to add new agent type that will perform the required functions (Jennings 2001).

6 Content Delivery Subsystem

The content delivery subsystem is responsible for answering user queries. Here the primary challenge is communication between clients and agents in our system. While in theory this should be relatively straightforward, in our earlier work (Galant 2002b) we discovered that this is not the case. Specifically, the absence of agent platforms to host agents on client devices limits the reach of the agent platform to protocols supported by the client, and those only to the extent that the agent system may be adapted to them. Thus we make only a minimal set of assumptions about expected capabilities of the device (like that it will be able to communicate with the Internet using the HTTP protocol). While the details of communication are provided in (Kaczmarek 2005), here, we will proceed with the assumption that the user has submitted a query through an Internet enabled device to her *Personal Agent* (*PA*) (see Figure 3). This message has been translated from its original form by the *Proxy Agent* (*PrA*) residing on the gateway HTTP server. Regardless of the form of the query the "query content" is extracted and wrapped into an ACL message and send to the *PA*. The *PA* forwards the message to two locations. First, it is stored in the user behavior database, where all user queries sent to the system and all system responses are logged for future mining (Galant 2002a). Based on the queries and responses and user responses to the queries it is possible to update user profile that can be used by the *PA* to filter and personalize content delivered to the user. Second, the query is send to the *DB Agent* (*DBA*). The *DBA* is the interface of the system to the Jena database. The *DBA* translates the user query into the *RDQL* language

used by Jena to query the entire set of RDF triples. The *DBA* executes the query and as a result obtains a set of tokens describing travel objects. These RDF demarcated tokens are then sent to the personalization infrastructure.

The personalization infrastructure consists of a number of *RDF Agents*. These are simplistic agents and each represents one of more of rules of the type "Szechuan food is also Chinese food" or "Romantic Comedy is also a Comedy." These rules are applied to the set of RDF-answers. Rule applications may involve querying the repository and expand the result set. The personalization infrastructure agents operate as a team passing the result set, wrapped in an ACL message from one to the next and their role is to maximize the set of responses to be potentially delivered to the user (no potential response is removed form the set).

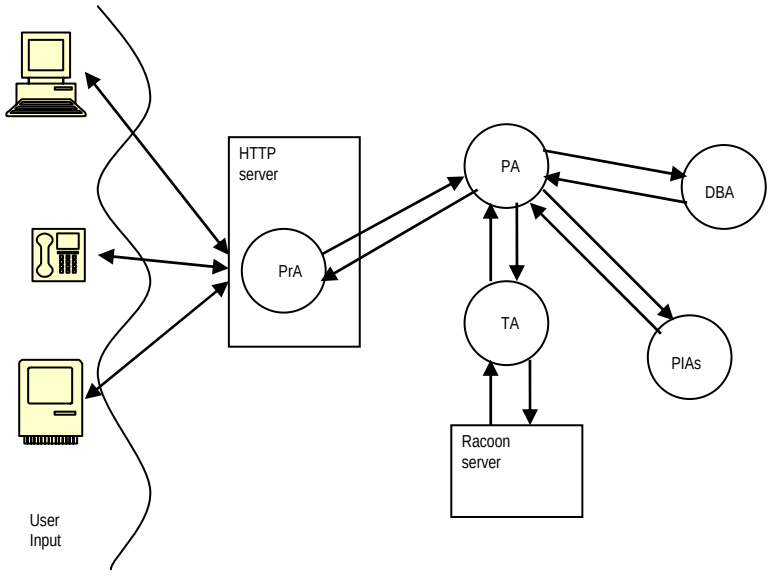

Figure 3. Content delivery subsystem: PA – personal agent, DBA – database agent, PIA – personalization infrastructure agents, PrA – proxy agent, TA – transformation agent

The maximal set of responses is sent back to the *PA*. The *PA* utilizes the user profile filter the answer set (see the next section). For instance, the *DBA* and the *PIA* agents may not know that the user never stays in Days Inn hotels and never visits Bennigans restaurants. Thus RDF triples representing these two chains will

be included in the expanded answer set. However, the *PA* will remove them. The final answer set is send to the *Transformation agent* (*TA*) that utilizes a Racoon (Racoon) server to render the response in a specific form, dependent on the user device (as well as back to the personalization infrastructure to be logged as a response to the given query and used in user behavior analysis (Galant 2002a)).

7 User Profile, its Creation and Management

User profile construction used in our project is based mainly on (Galant 2002a, Fink 2002, Montaner 2003, Kobsa 2001). We assume that the user profile must utilize in its representation the domain ontology (in our case the travel ontology underlying our system) and be adjusted incrementally, to keep up with changing interests and preferences. Let us look now at various aspects of user profile creation and management.

User Profile Representation
Since the RDF demarcated data defining travel object is the centerpiece of our system, it was obvious that the user profile has to be represented by statements that express user's opinion about concepts within the ontology. Here, the concept becomes the subject of an opinion statement through the reification feature of the RDF, e.g. through a sentence: *Chinese cuisine is user's favorite with probability X*. In this way we superimposed on top of our ontology a probability graph, where concepts become nodes described by probability of being favorites of a user. In case of lack of specific opinion (e.g. when user had no chance to form it) we calculate it on the basis of other concepts, where the association is described by ontological relations (*subclassOf*, *propertyOf*, *instanceOf* and *rangeOf)*. For instance, to decide what the user would choose: an Italian restaurant or a Pub with a lively atmosphere, we would take into consideration the fact that in the past she placed more emphasis on the cuisine concept rather than on the atmosphere concept. Therefore she will be suggested to visit an Italian restaurant.

Initial Profile
Getting to know user preferences is time-consuming and involves a natural problem of new user in the system (Galant 2002a). Our proposed solution is to employ stereotyping, which is based on the fact that creating an initial model is, in a sense, a classification problem, aimed at generating initial predictions about the user (Montaner 2003). The user model is initiated by classifying users in stereotypical descriptions (Rich 1979), representing the features of classes of users. Initially, these stereotypes will be acquired by classifying responses from the survey conducted among potential user. Then, the new user of the system

will be requested to fill-in the questionnaire (concerning demographic and domain-based data) and responses will be matched against the existing stereotypes. As the system operates we will be able to adapt the stereotypes. As the user behavior data is gathered it can be mined not only to study interests of an individual, but also clustered to obtain information about group-behavior. This information can be used to verify and update the stereotypes applied to new users as well as in other parts of the system (see also (Nistor 2002) for the role and applicability of information collected on various levels of social stratification).

Relevance Feedback

The recommender agent (*PA*) needs actual data to update user profile. To achieve this goal, for each user, we will collect a complete log of interactions with the system. This log will contains both positive (she selected restaurant *Y*) and negative (he never selected hotel *Z*) behavior based information, that constitute an implicit feedback (Kobsa 2001). Explicit feedback needs user support in form of rating suggestions provided by the system. Information gathered through both implicit and explicit feedback will be used to update user profiles (adjust probabilities in the probability graph representing user profile).

Adjustments to the User Profile

To focus our attention, let us look into the case of restaurants; however the technique applied here can be applied to other travel objects (Fink 2002). First, let us note that repeated visits to a given restaurant express not only opinion about that particular restaurant, but also about its features (concepts in our ontology), like: Thai cuisine or a smoking section. For the purpose of our system we adapt history-based learning proposed in (Fink 2002), which consists of: (1) history analysis, (2) computing normalized opinions and (3) utilizing domain inferences. In history analysis we find frequencies of occurrences of behaviors connected with particular concepts to compute *individual probabilities*. We analyze not only selection of a restaurant, but also its properties, which are defined by the domain ontology. Furthermore, we process keywords appearing in the user's query, as they represent user's interest in particular attributes of the restaurant. Obtained probabilities have to be normalized to the general population (resulting in *normalized probability*). This is done by comparing the *individual probability* against the normal distribution and allows us to divide concepts into interesting, uninteresting (tails of the distribution) and such that we cannot clearly classify. Eventually we infer probability of liking a certain concept by appropriately weighting and adding normalized probabilities of all its sub-concepts. The result is called the *inferred probability* and is used to adjust user profile.

Utilization of the Profile

Let us now illustrate the usage of the user profile by the *Personal Agent* (we will, again, without loss of generality, use restaurants as an example). After the *RDF* agents complete their work, a "maximum" set of travel objects is delivered to the *PA* that is asked to filter and order them according to user's preferences (feature-based recommending). For each restaurant in the set we compute probability that it is a favorite of the user. This is done in the following way: (1) from the ontology graph of a restaurant we crop a sub-graph consisting of concepts occurring in the *restaurant token* being rated; (2) then we compute the *total probability* of the sub-graph, by taking into account *normalized probability* when the concept was already successfully classified and – *inferred probability*, otherwise. To consider the *context* in which the ranking takes place, before step (1) above, the value of appropriate *normalized/inferred probability* is increased for each concept appearing in the current query. Finally, the *total probability* is increased if the restaurant was previously rated by the user (explicit feedback) as interesting (results of such process are presented in Figure 4). Obtained probabilities are used to filter and rank restaurants, and remove ones which are rated below certain threshold level.

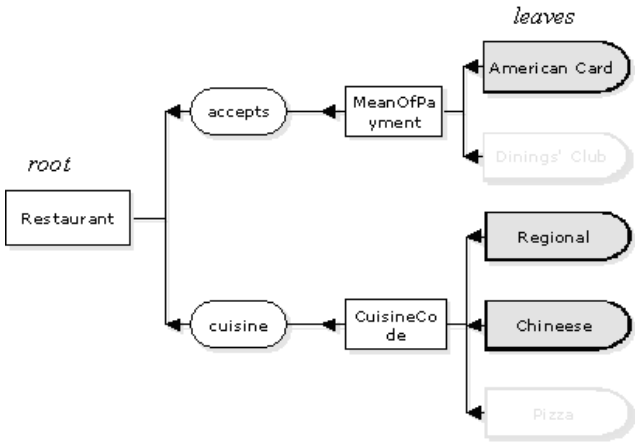

Figure 4. Sub-graph of probability graph based on the restaurant serving: Regional and Chinese cuisines and accepting American Express. Transparent leaves were found do not exist in a processed restaurant.

Combining with other recommending techniques
Personalization infrastructure proposed here is capable of applying different recommending techniques in parallel and combine results together. For example, it is easy to imagine that multiple agents representing individual users can be interacting with each-other within a collaborative-filtering setup, while other agents simply make feature-based recommendations. Weighted average is one of the possible ways of combining recommendations obtained in such multi-recommendation environment (Burke 2002).

8 Concluding Remarks

In this paper we have discussed how RDF demarcated data can be utilized in an agent-based travel support system. First, in the center of the system, sets of RDF triples defining travel-object tokens are stored in the JENA repository and become the way of semantically organizing the "world of travel." Second, the ontology of travel is used to represent user profiles and utilized to deliver personalized content to the user. Here, user profile becomes a reification of ontology and forms a probability graph that can be then used to filter and order information delivered in response to user queries. We will be reporting on our progress in implementing the system in the near future.

References

Abramowicz W., Kalczyński P. J. (2000): Building and Taking Advantages of the Digital Library for the Organizational Data Warehouse. In: Cobb M. et. al. (eds.), Proceedings of the Second Southern Conference on Computing, Hattiesburg, Mississippi, USA, October 26-28, 2000, CD

Angryk R., Galant V., Gordon M., Paprzycki M. (2002): Travel Support System - an Agent Based Framework. In: H. R. Arabnia, Y. Mun (eds.), Proceedings of the International Conference on Internet Computing (IC'02), CSREA Press, Las Vegas, NV 719-725, 2002

Brady R., Forrest E., Mizerski R. (2002): Marketing w Internecie. PWE, Warszawa, Poland

Burke R. (2002): Hybrid Recommender Systems: Survey and Experiments. User Modeling and User-Adapted Interaction, 12 , 4 , 331 - 370.

Chaudhury A., Malik D. N., Rao H. R. (2001): Web Channels in E-commerce. CACM, 44 (1), pp. 99-104

ChefMoz Dining Guide, http://chefdmoz.org

de Kare-Silver, M. (2001): E-shock: the New Rules. E-Strategies for Retailers and Manufacturers.z Palgrave, Houndmills, UK

ebXML R/R: http://ebxmlrr.sourceforge.net/

Fink J., Kobsa A. (2002): User Modeling for Personalized City Tours. Artificial Intelligence Review, 18 (2002) 33-74

Galant V. and Paprzycki M. (2002a): Information Personalization in an Internet Based Travel Support System. Proceedings of the BIS'2002 Conference, Poznań, Poland, April, 2002, pp. 191-202

Galant V., Gordon M., Paprzycki M. (2002b): Agent-Client Interaction in a Web-based E-commerce System. D. Grigoras (ed.), Proceedings of the International Symposium on Parallel and Distributed Computing, University of Iaşi Press, Iaşi, Romania, 2002, 1-10

Galant V., Gordon M., Paprzycki M. (2002c): Knowledge Management in an Internet Travel Support System. B. Wiszniewski (ed.), Proceedings of ECON2002, ACTEN, Wejcherowo, 97-104

Galant V., Jakubczyc J., Paprzycki M. (2002d): Infrastructure for E-Commerce. Proceedings of the 10th Conference on Extraction of Knowledge from Databases, Karpacz, Poland, May, 2002 (to appear)

Gawinecki M., Gordon M., Paprzycki M., Szymczak M., Vetulani Z., Wright J. (2005a): Enabling Semantic Referencing of Selected Travel Related Resources. In: W. Abramowicz, Proceedings of the BIS'2005 Conference, Poznań University of Economics Press, Poznań, Poland, (2005) 271-290

Gawinecki M., Gordon M., Nguyen N., Paprzycki M., Szymczak M. (2005b): RDF Demarcated Resources in an Agent Based Travel Support System. In: Proceedings of the 17th Mountain Conference of the Polish Information Society, (to appear)

Gawinecki M., Vetulani Z., Gordon M., Paprzycki M. (2005c): Representing Users in a Travel Support System, Proceedings of the ISDA 2005 Conference, to appear

Gilbert A., Gordon M., Nauli A., Paprzycki M., Williams S., Wright J., (2004): Indexing Agent for Data Gathering in an e-Travel System. Informatica, Vol. 28, No. 1, 69-78

Gordon M., Paprzycki M. (2005): Designing Agent Based Travel Support System, Proceedings of the ISPDC 2005 Conference, to appear

Grover V., Teng J. C. T. (2001): E-commerce and the Information Market, CACM, 44(4), pp.79-86

Harrington P., Gordon M., Nauli A., Paprzycki M., Williams S., Wright J. (2003): Using Software Agents to Index Data in an E-Travel System. N. Callaos (ed.), Electronic Proceedings of the 7th SCI Conference, Orlando, 2003, CD, file: 001428.pdf, 6 pages

Jena 2 - A Semantic Web Framework, Hewlett Packard, http://www.hpl.hp.com/semweb/jena2.htm

Jennings N. R. (2001): An agent-based approach for building complex software systems. CACM 44, 4, 2001, 35-41

Kaczmarek P., Gordon M., Paprzycki M., Gawinecki M. (2005): The Problem of Agent-Client Communication on the Internet. PDCP, to appear

Kobsa A., Koenemann J., Pohl W. (2001): Personalized Hypermedia Presentation Techniques for Improving Online Customer Relationships. The Knowledge Engineering Review, 16:2, 2001, 111-155

Maes P. (1994): Agents that Reduce Work and Information Overload. Communications of the ACM, 37, 7, 1994, 31-40

Mohr J. (2001): Marketing of High-Technology Products and Innovations. Prentice-Hall, Upper Saddle River, NY.

Montaner M., López B., De La Rosa J. L. (2003) : A Taxonomy of Recommender Agents on the Internet. Artificial Intelligence Review, 19, 2003, 285-330

Ndumu, D., Collins, J., Nwana, H. (1998): Towards Desktop Personal Travel Agents. BT Technological Journal, 16 (3), pp. 69-78

Nistor C. E., Oprea R., Paprzycki M., Parakh G. (2002): The Role of a Psychologist in E-commerce Personalization. Proceedings of the 3rd European E-COMM-LINE 2002 Conference, Bucharest, Romania, 2002, 227-231

Nwana H., Ndumu D. (1999): A perspective on software agents research. The Knowledge Engineering Review, 14, 2, (1999) 1-18

Paprzycki M., Galant V., Gordon M., (2002): Agent-Client Interaction in a Web-based E-commerce System," in: D. Grigoras (ed.), Proceedings of the International Symposium on Parallel and Distributed Computing, University of Iaşi Press, Iaşi, Romania, 2002, 1-10

Paprzycki M., Abraham A. (2003): Agent Systems Today; Methodological Considerations. Proceedings of 2003 International Conference on Management of e Commerce and e-Government, Jangxi Science and Technology Press, Nanchang, China, 416-421

Petry F., Cobb M., Ali D., Angryk R., Paprzycki M., Rahimi S., Wen L., Yang H. (2002): Fuzzy Spatial Relationships and Mobile Agent Technology in Geospatial Information Systems. Sztandera L., Matsakis P. (eds.) Soft Computing in Defining Spatial Relations, volume in series: Soft Computing, Physica-Verlag, to appear

Poslad S., Laamanen H., Malaka R., Nick A., Buckle P., Zipf A. (2001): CRUMPET: Creation of User-friendly Mobile Services Personalised for Tourism. Proceedings of: 3G 2001 - Second International Conference on 3G Mobile Communication Technologies. 26-29 March 2001. London, UK. http://conferences.iee.org.uk/3G2001/

Racoon: http://rx4rdf.liminalzone.org/Racoon

RDF Primer, http://www.w3.org/TR/rdf-primer

Rich E. (1979): User Modeling via Stereotypes. Cognitive Science 3. pp. 329-354

Rud O. P. (2001): Data Mining Cookbook. Modeling Data for Marketing, Risk, and Customer Relationship Management. Wiley, New York, NY

Simon A. R., Shaffer S. L. (2001) "Data Warehousing and Business Intelligence for E-commerce," Morgan Kaufman, New York, NY.

Sołtysiak S., Crabtree B. (1998): Automatic learning of user profiles - towards the personalization of agent service. BT Technological Journal, 16 (3), pp. 110-117

Strauss J., Frost R. (2001) : E-Marketing. Prentice-Hall, Upper Saddle River, NY

Suarez J. N., O'Sullivan D., Brouchoud H., Cros P. (1999): Personal Travel Market: Real-Life Application of the FIPA Standards. Technical Report, BT, Project AC317

Wright J., Gordon M., Paprzycki, M., Williams S., Harrington P. (2003):Using the ebXML Registry Repository to Manage Information in an Internet Travel Support System. W. Abramowicz and G. Klein (eds.), Proceedings of the BIS'2003 Conference, Poznań University of Economics Press, Poznań, Poland, 2003, 81-89

Zhang B., Li W., Xu Z. (2002): Personalized Tour Planning System Based on User Interest Analysis. Proceedings of the BIS'2002 Conference, Poznań, Poland, April, 2002, pp. 184-190