RDF DEMARCATED RESOURCES IN AN AGENT BASED TRAVEL SUPPORT SYSTEM

Maciej GAWINECKI, Minor GORDON, Ngoc T. NGUYEN, Marcin PAPRZYCKI , Michał SZYMCZAK

ABSTRACT: 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, the 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 illustrate how our ontology can be used not only to store data in an agent-based travel support system, but also to support delivery of personalized content to users.

1. Introduction

Software agents and ontologies are two essential elements of the next generation World Wide Web. Researchers like the author of [12] 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 [5].

With the tacit assumption that software agents are the future of Internet-computing and ontologies are the way to provide agents with data, we have developed an agent-based travel support system [2, 7]. All functions of the system that can naturally be decomposed into agents are implemented in this way [10]. However, in the system software agents do not search for responses only after queries are posed (an agent-only design), but also collect data and store in the form of ontologically demarcated tokens that constitute the local representation of the "world of travel." When a user issues a query, the response is prepared from these tokens, filtered (personalized) to match individual preferences and delivered back to the user.

The aim of this note is two-fold. First, we introduce our proposals for ontologies of *hotel* and *restaurant*. Second, we show how they can be used to represent user profiles and to deliver personalized information. We proceed as follows. In the next section we briefly describe the high-level structure of our system. We follow (in Section 3) with the description of *hotel* and *restaurant* ontologies. In Section 4 we show how the user profile can be instantiated and used once ontologies have been defined.

2. System overview

The general schema of the proposed system is depicted in Figure 1, and consists of content collection, management and delivery, with the central database storing

RDF demarcated travel-object tokens. Let us now briefly summarize each of the components presented in Fig. 1 (for more details see [7]).

Verified Content Providers *(VCP)*: One of the problems with data from the Internet is its dynamic nature and unreliability [13]. To answer the question: "how to provide user with trustworthy data?" we utilize the concept of *Verified Content Providers* – sites known to provide **reliable** and **consistently available** information (e.g. that do not appear, disappear and change their interface randomly) [1].

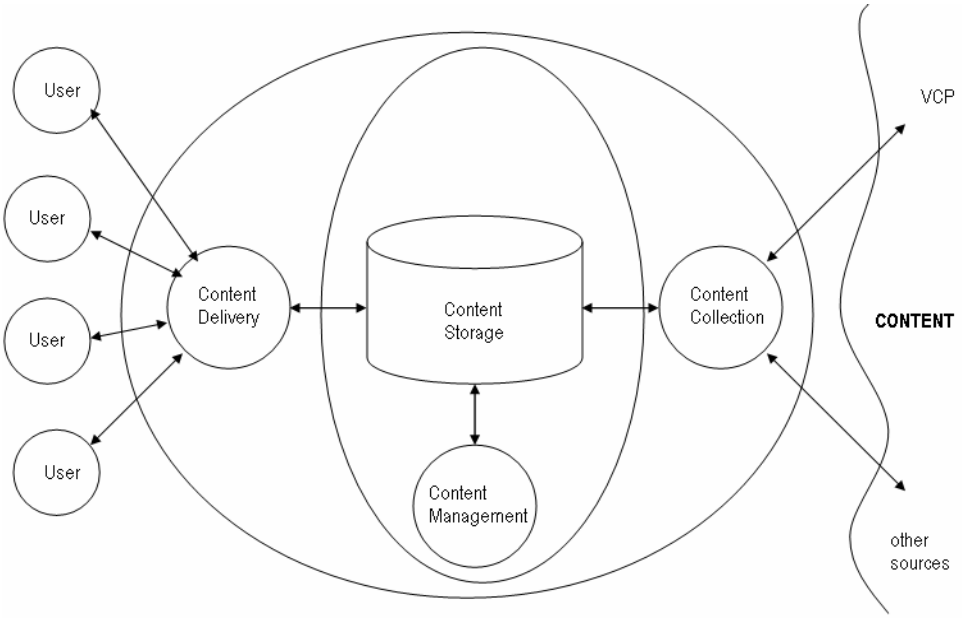

Figure 1. Infrastructure for the travel support system

Other Sources *(OS)*: *OS* refer to all non-*VCP* sources that, while not fully trusted, should not be omitted. Information collected from these sources will be even more important when the SemanticWeb [5] will result in small sites becoming easily consumable (due to usage of ontologies) content providers (as easily as the *VCP*s).

Content Collection Subsystem (***CCS***): In our system data is stored in a semantically demarcated form. Various search/wrapper agents deliver RDF tagged triples that are stored in a JENA based repository [9]. Sets of triples, defined by the ontology, constitute travel-object tokens and originate from both the *VCP*s and the *OS*.

Content Management Subsystem (***CMS***): *CMS* involves all functions related to management of data stored in the central repository. Here, among others, we deal with incomplete tokens and with explicitly and implicitly time sensitive information (e.g. changes of opening times of a ZOO, resulting from longer days in the Summer).

Content Delivery Subsystem (***CDS***): *CDS* is responsible for all data

manipulations resulting in delivery of personalized content. Here, agents receive a query from the user and perform necessary actions (e.g. extraction of tokens from the repository and various filtering techniques) to provide information matching her preferences.

Users of the system access it via Internet-enabled devices, such as PC-based browsers, palmtops and WAP-conversant phones etc. (for more details see [8]).

Since this note is focused on the central repository containing instances of objects defined in travel ontology, let us now present our ontologies of *hotel* and *restaurant.*

3. Hotel and Restaurant Ontology

There exist a number of general ontologies and travel ontologies. Among top-level ontologies one should mention: Cyc, WordNet,Suggested Upper Merged Ontology (SUMO) and the SENSUS project, while among travel-related ontologies the most important are: Open Travel Alliance, Mondeca, Travel Game in Agent Cities, Harmonize and DAML-based ontologies. Unfortunately, after thorough search we were not able to find a clean and complete ontology of basic travel entities such as a hotel, a pub, or a movie theater. Thus it was necessary to start by formally defining some 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 an application that the ontology is to serve. In our case the travel ontology is used to organize information collected form the Internet. Therefore, we have started from the Internet and analyzed web sites of the most popular (top Google ranking) travel sites and decided that the top-level of the proposed hotel ontology should consists of the following classes:

- *site* – contains common characteristics of places to visit,
- *hotel room* – combines types of rooms in particular hotel,
- *amenities* – specifies amenities available within the *site* and in the *room*.

Where, *site* represents the real-world characteristics of places such as hospitals, bars, arenas etc. These properties are inherited by all subclasses of *site* included in the ontology, which also 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 N3notation [14], namespaces omitted) defining the hotel room (suite):

```
:SampleHiltonSuiteXXX a hroom:HotelRoom;
    hroom:standardName                  "Suite";
    hroom:nbrOfRoomsOfThisType          "40";
```

```
    hroom:nbrOfRoomsOfThisType          "15";
    hroom:theAmenities
          <#RegularEconomyAmenities>,
          <#ExtraPayedAmenities>,
          <#SuiteAmenities>.
:RegularEconomyAmenities a amenities:Amenities;
    amenities:content
    "http://StandardHiltonFurniture.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 [3]; possibly the only existing large repository of RDF demarcated, travel-related data. Since there is no explicitly defined ontology provided within ChefMoz we have: made explicit the *restaurant* ontology found the data, and used it to produce a clean dataset [6]. Data cleaning illustrated possible pitfalls of dealing with data co-created by multiple (50,000+) "authors." What follows is a snippet of "code" defining a café, and illustrating representation of means of payment.

```
<#Poland_ZP_Radom_Capri_Kawiarnia1370880459>
a res:Restaurant
; res:accepts
       money:AmericanExpressCard,
       money:DebitCard,
       money:DinersClubCard,
       money:Cash,
       money:MasterCardEurocard,
       money:VisaCard
; res:cuisine res:CafeCoffeeShopCuisine.
```

Obviously, *hotel* and *restaurant* ontologies are closely related (e.g. a hotel has a restaurant on-site). This leads us to a concept of *near-by facility*. Its origin is in the analysis of the information available on the Web (let us recall that we develop ontology to organize existing information, and that **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. 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 hotel. This time we have provided "exact" values as they would be associated with particular fields in the ontology (for more details see [6]).

```
:NewSampleHiltonHotelXX a vSite:HotelALike;
      vSite:myVCard <#ThisHotelsVCard>;
      vSite:type                          "hotel";
      vSite:maxPeopleGetIn                "250";
      vSite:hasEntryFee                   "0";
      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            "101";
      vSite:numberOfStars                 "3";
      vSite:numberOfAllRooms              "115";
      vSite:nameOfTheFamily               "Hilton Hotels";
      vSite:staffLanguages                "German, Czech";
      vSite:hotelFacilities
            <#SampleSolarium>,
            <#SampleCasino>,
            <#SampleKiosk>,
            <#SampleSwimmingPool>,
            <#SampleRestaurant>;
      vSite:nearbyFacilities
            <#SampleGolfCourse>,
            <#SampleZoo>;
      vSite:typesOfRoomsAvailable
            <#SampleHiltonSuiteXX>,
            <#SampleHiltonStandardXX>.
```

4. User Profile, its Creation and Management

Travel-object tokens are utilized to provide user with personalized content. Therefore, user preferences should be represented in a form closely related to the domain ontology of the system and be incrementally adjustable, to keep up with users' changing interests and preferences. We have achieved this goal in a following way.

User Profile Representation is based on statements about concepts defined in our ontology ("represent opinion about concepts"). Here, concepts become the subject of the opinion statement obtained through the reification

feature of the RDF, e.g. through a sentence: *Italian cuisine is user's favorite with probability X*. We then build a directed acylic probability graph (user profile), where concepts become nodes described by probability of being favorites of an individual user. Here, conditional probabilities are based on relations derived from the domain ontology.

Initial Profile creation is one of the more difficult problems of any recommender system and our solution is to employ stereotyping. Here, users are classified into templates representing group-features [4, 11]. Initially, these templates are acquired from responses to a survey conducted among potential users. New users are then asked to fill-in a questionnaire and responses provided there are matched against the templates. As the system operates, user behavior data is gathered and mined not only to study interests of an individual, but also to adjust group-templates.

Relevance Feedback allows us to update the user profile. Since this feedback requires actual data, for each user we record a complete log of interactions with the system, containing both positive ("user selected restaurant *Y"*) and negative ("user never selected hotel *Z"*) behavior traces (implicit feedback [11]). Explicit feedback is obtained in terms of rating suggestions provided by the system. Information from implicit and explicit feedback is used to adjust probabilities in user profile graph.

Adjustment of User Profile consists of changing probabilities in the user profile graph. Let us look into "hotel interests." A high rate of visits to a given hotel indicates not only an opinion about that particular hotel, but also about its features, such as: exercise facility or price. Preference for a feature is derived from frequency of its occurrence in user's history (*favouriteProb*) and the domain interference (*fromDomainProb*). When history-based learning [4, 11] is applied, we proceed as follows: (1) compute frequency of occurrence of different concepts in user's history (relatively to the history of all users; separately for the *implicit* and the *explicit* feedback); (2) compute *favouriteProb* by combining results of (1) with importance given to the *explicit* feedback; (3) for each node (3a) compute *fromDomainProb* values by performing domain interference: this is done using upwards propagation (*favouriteProb* is propagated from leaves to super-concepts); e.g. if the user is interested in "hotels with smoking rooms," then the user is presumed to consider *amenities* as an important factor of hotel selection; (3b) combine *favouriteProb* and *fromDomainProb* into *interestingProb*; (4). Results derived in (3b) classify concepts in the user's profile as *significantly interesting*, *significantly uninteresting* or *unclassified*. This is done using univariate significance analysis [4], i.e. if a feature appears in the history less frequently than in a random sample, the user is not to interested in it.

Utilization of the Profile. Response preparation starts with a set of feasible response-tokens **expanded** (only) by a number of agents [7]. The "maximum" set of tokens has to be filtered according to user's preferences. Here, we compute the probability that a given hotel is preferred by the user. This is done for each object (*hotel*) in the following way: (1) we extract the sub-graph consisting of paths between the *hotel* and its features that are *significantly interesting* for the user, and (2) compute the total probability of that sub-graph. To consider the situation context, before (1) above, the value of the *interestingProb* is

increased for each feature appearing in the query. Finally, the *interestingProb* is increased if the *hotel* was rated as interesting. The resulting probabilities are used to filter and rank hotel tokens. Hotels that are ranked as *significantly uninteresting* have probability close to 0 and can be removed from the set. The remaining *hotels* have probabilities above the threshold and will be ranked accordingly (these with highest probability that will be displayed first).

Combining with other recommender techniques. Since our system is agent-based, it is very easy to combine various recommender techniques. At any given time the current response consists of a number of travel-object tokens. If we assume that different agents are responsible for different recommendation techniques, results of their work will consist of adding, removing or rearranging tokens in the response set. To fuse results of different recommending techniques (a group of agents performing collaborative-filtering, while another agent making feature-based recommendations) weighted average can be used to combine their recommendations [4].

5. Concluding remarks

In this not we have discussed how RDF demarcated data can be utilized in an agent-based travel support system. In the system sets of RDF triples defining travel-object tokens are stored in the JENA repository and utilized to deliver personalized content to the user. We are utilizing ontologies also as ways of defining user profiles. 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 subsequent notes.

References:

1. Abramowicz W., Kalczyński P. J.: 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 (2002)
2. Angryk R., Galant V., Gordon M., Paprzycki M.: 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 (2002) 719-725
3. ChefMoz Dining Guide, http://chefmoz.org
4. Fink J., Kobsa A.: User Modeling for Personalized City Tours. Artificial Intelligence Review, 18 (2002) 33–74
5. Fensel D.: Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce, Springer, Berlin (2001)
6. Gawinecki M., Gordon M., Paprzycki M., Szymczak M., Vetulani Z., Wright J.: 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

7. Gordon M., Paprzycki M.: Designing Agent Based Travel Support System. Submitted for publication
8. Gordon M., Paprzycki M., Gawinecki M., Kaczmarek P.: The Problem of Agent-Client Communication on the Internet. PDCP, to appear
9. JENA: http://www.hpl.hp.com/semweb/
10. Jennings N. R.: An agent-based approach for building complex software systems. CACM, 44, 4, (2001) 35-41
11. Kobsa A., Koenemann J., Pohl W.: Personalized Hypermedia Presentation Techniques for Improving Online Customer Relationships. The Knowledge Engineering Review, 16:2, (2001) 111-155
12. Maes P.: Agents that Reduce Work and Information Overload. Communications of the ACM, 37, 7, (1994) 31-40
13. Nwana H., Ndumu D.: A perspective on software agents research. The Knowledge Engineering Review, 14, 2, (1999) 1-18
14. N3 notation, http://www.w3.org/DesignIssues/Notation3.html

Maciej GAWINIECKI, Paweł KACZMAREK, Michał Szymczak
Department of Mathematics and Computer Science, Adam Mickiewicz University
ul. Umultowska 87, 61-614 Poznań, Poland

Ngoc Thanh NGUYEN
Department of Computer Science, Technical University of Wrocław
Wybrzeże Wyspiańskiego 27, 50-370 Wrocław, Poland

Marcin PAPRZYCKI
Computer Science Department, OSU, Tulsa, Oklahoma, USA
Computer Science, SWPS, ul. Chodakowska 18/31, 03-815 Warszawa, Poland
e-mail: Marcin.Paprzycki@swps.edu.pl