Implementing message exchange between airlines' GDSs and travel systems with ontologicaly demarcated data

Mladenka Vukmirović
Montenegro Airlines, Podgorica, Montenegro
Maciej Gawinecki, Paweł Kobzdej
Systems Research Institute, Polish Academy of Science
Maria Ganzha,
Elbląg University of Humanities and Economics, Elbląg, Poland
Marcin Paprzycki
SWPS, Warsaw, Poland

Abstract. *The development of airlines' global distribution systems is in an interesting stage of evolution. Currently airline industry is defining new protocols that are to profit from utilization of the state-of-the-art technology such as agent systems. In the meantime we are developing an agent-based Travel Support System (TSS), with a goal of providing passengers with more personalized information in regard to the desired journey. Obviously, the TSS, to realize its goals must be able to communicate with various (airline) global distribution systems (GDS)s. In this paper we are presenting our initial attempt of utilizing new trends emerging in the airline industry while developing a communication channel between the TSS and GDSs.*

Keywords. software agents, air-travel ontology, travel support system, message exchange.

1. Introduction

Our current work is focused on providing comprehensive support for informational needs of a traveler. This work followed two concurrent paths. In one of them we proposed a novel approach to selling airline tickets. Here, we started from a model agent based e-commerce system (see [7, 8] and references collected there). We have utilized its modified infrastructure to incorporate airline ticket auctioning capabilities. Obviously such a service requires interfacing it with global distribution systems (GDS)s used by the airline industry, among others, to manage air travel schedules, reservations and ticket sales [4, 10, 17, 25, 28]. The aim of the second project was development and initial implementation of an agent based travel support system (TSS). In the TSS, travel-related data is represented as instances of an RDF demarcated travel ontology (of hotel and restaurant) and used to deliver personalized information to the users (see [15,16] and references collected there). The next necessary step is to merge these two projects and when conceptualizing the merger we have realized the need of resolving different data representations existing in both systems. Since the TSS utilizes ontologically demarcated data, a natural solution was to create air travel ontology to be fully integrated with hotel and restaurant ontologies developed for the TSS [9, 10, 16]. We have designed such an ontology by utilizing industry standards that are in detail given in International Air Travel Association (IATA) Manuals [1, 6, 21, 22], and other industry indicatives, among which our focus was set on the Open Travel Alliance (OTA) messaging [19, 20, 22], together with reuse of best practices of existing ontologies. Results of this work have been reported in [17, 18, 25, 28]. Obviously, development of a unified travel ontology that includes air travel ontology solved only a part of the problem – the internal representation of air travel data within the TSS. What still needs to be addressed is communication of the TSS with other entities servicing needs of travelers and in this paper we discuss how this goal can be achieved. Due to space limitation we focus our attention only on communication between airlines' GDSs and the TSS.

Let us start from a bit of history. Contemporary global distribution systems were developed in 1970s and were based on host-terminal technology that involves extensive message exchange (such as TypeB messages, EDIFACT etc.) between GDSs and inventory systems of various airlines as well as different GDSs among themselves. These exchanges of messages were strictly defined by a set of detailed industry rules governed by IATA. Currently, a new trend can be observed, which will result, over time, in message exchanges being re-deployed utilizing modern technologies, which are currently being widely explored in the industry (see [23, 24, 26], for more details). This move is necessary since advances of computer technology that took place since 1970s have to be effectively utilized with a goal of simplifying business practices.

In this context, for the airline industry, IATA has recently taken a lead in defining a new set of standards that are to employ modern technologies such as XML, SOAP, and even ontologies and agent systems [2, 3, 23, 24, 26]. ARINC and SITA (which contributed a great deal to the existing message exchange solution in the aviation business) followed the lead and joined forces with IATA in an on-going project with the goal of implementation of a standard for messaging using XML, called TypeX messaging. This standard is to be an evolution of the IATA TypeB messaging [26]. However our attention has been focused on a concurrent stream of development – the OTA messaging system [20]. This is because IATA intends to include OTA messages into TypeX message definitions. In other words, OTA messages already exist as a standard and will be utilized by IATA, while TypeX messages are still being worked on. Summarizing, in our work we are following initiatives undertaken by the travel industry and try to reuse as much as possible from these emerging standards.

This being the case we can make our goal more specific: we are interested in facilitating message exchange between our system and OTA-utilizing entities. Furthermore, in this paper, our focus is on air-travel-related OTA messages. We start with a brief description of the TSS and the air-travel ontology used in it. We follow with a description of scenarios where our system could be contacted or had to communicate with OTA understanding entities. We complete the paper by describing how the translation process between the TSS understood content and the OTA messaging looks like.

2. Travel Support System travel ontology

Our travel ontology was created by merging two ontologies – one created for the TSS (and consisting of hotel and restaurant ontologies [9, 10, 15, 16]) and the second created for agent-based airline ticket auction system [17, 18, 25, 27, 28]. While developing the latter one, we have established that existing air-travel ontologies are "academic" in nature – and this explains the lack of important features when it comes to dealing with actual air travel data [28]. Therefore, we decided to create a new ontology that would: (1) utilize IATA mandated data; (2) utilize as much as possible from the existing travel ontologies – provided that applicable IATA resolutions and recommended practices are followed, (3) match features included in the OTA specification, and (4) be ready to be merged with our existing travel ontology. Hence, we applied a bottom-up approach and our initial goal was to model reservations occurring in the AMADEUS global distribution system. Let us stress once more that at the beginning of our work we have decided that to establish communication between our system and other travel-related entities we will use OTA messaging (which fast becomes an industry standard [23, 26]). Thus, integration with the OTA messaging system was one of important goals of our endeavor and heavily influenced the resulting shape of air-travel ontology [18].

To illustrate the structure of the air-travel ontology we present our *AvailabilityDisplay class* (and later we will use it when discussing exchanges of messages). This class presents availability of seats on plains for a selected route and carriers offering flights. What follows is a fragment of its N3 described definition (complete definition may be found in [11]).

```
@prefix flt: <AirTravel/Flight#>.
@prefix cls:
    <AirTravelCodes/IATAClasses#>.
@prefix :
    <AirTravel/AvailabilityDisplay#>.

:AvailabilityDisplay a  rdfs:Class.

:details  a   rdf:Property;
  rdfs:domain :AvailabilityDisplay;
  rdfs:range  :AvailableFlights.

:AvailableFlightElement a rdfs:Class.

:flight  a  rdf:Property;
  rdfs:domain :AvailableFlightElement;
  rdfs:range  flt:Flight.

:classAvail a   rdf:Property;
  rdfs:domain :AvailableFlightElement;
  rdfs:range :AvailableClasses.

:AvailableClasses a   rdfs:Class.

:classavailable  a  rdf:Property;
  rdfs:domain :AvailableClasses;
  rdfs:range  :AvailableClassElement.

:AvailableClassElement  a rdfs:Class.

:class  a  rdf:Property;
  rdfs:domain :AvailableClassElement;
```

```
  rdfs:range cls:BookingClass.

:noAvailableSeats a rdf:Property;
  rdfs:domain :AvailableClassElement;
  rdfs:range xsd:integer.
```

3. Message exchange scenarios

Let us now discuss sample scenarios that involve message exchanges between our system and other travel entities.

3.1. Scenario 1

In the first scenario let us consider a tourist who is looking for a complete package consisting of air ticket + hotel + restaurant (we selected these three entities as they currently "exist ontologically" in our system, however this request could also involve golf + opera + archeological museum). This request from the user has a form of a query-string and is transferred from the user-device into the TSS through a somewhat involved mechanism described in [29, 30, 31]. The *Personal Agent* would transform this request into SPARQL query [29, 30, 31] that would then be executed by the *Database Agent* on the Jena [13, 14] persisted central repository. This translation would work easily for the restaurant and hotel parts of the query as information about these entities was assumed to be stored in the system (though as soon as an actual reservation process is to be involved this assumption has to be relaxed and an OTA exchange like the following one has to take place). However, air travel information cannot be stored in the system. Specifically, it is possible to store selected "static" parts of the information (e.g. airport codes and addresses and their amenities) and the parts that changes only periodically (e.g. carriers that fly between given airports), but one cannot store actual flight schedules and seat availability information. Therefore the incoming user-query has to be split into two parts: (1) part to be executed internally (involves only query-string into SPARQL translation as described in [29, 30, 31]), and (b) part of the query sent to the GDS. It is the latter part that is of interest to us. Here the query-string has to be translated into an appropriate OTA RQ message ([19]). This message is then to be send to the GDS. The GDS will respond with an OTA RS message that contains the requested information. Content of this message has then to be translated into instance(s) of our travel ontology. Let us stop here for a moment and argue why the latter translation is needed. Why not simply send the response back to the user (the way that travel agents often do). The reasons are the following: (1) the main goal of our system is to deliver personalized information. This is achieved through representing user preferences as special instances of our travel ontology using an overlay model [32-35]. Therefore, to be able to apply user preferences to data obtained form the GDS (even in the simple case when only travel Warsaw to Podgorica is to be facilitated), we need to have travel data as instances of air travel ontology. Only then we will be able to apply weight representing user preferences to find out that she loves to travel on airlines representing "SkyTeam" alliance and definitely hates flying "BA" and (2) in the case of a more complicated query that involves several entities, the TSS needs to be able to apply reasoning involving their (multiple) instances. Therefore it is absolutely necessary to transform the incoming OTA carried information into instances of our TSS travel ontology and further process them in this form.

3.2. Scenario 2

In the TSS we expect to develop a number of data management agents [15]. One of important roles of these agents will be to keep data stored in the system correct and up to date. While it will not be possible to keep the actual schedule of airlines available, we can easily assume that TSS will store information about airlines that fly between two airports; as it usually changes only a few times a year. This being the case, it has to be considered that data management agents will occasionally request an update on the list of the airlines that fly between given two airports, or information how one can reach certain destination flying out of given airport (e.g. Hattiesburg, Mississippi) with certain carrier on a given day of the week. It will be the GDS that will be the best source of such information. Therefore the data management agent has to be able to formulate an appropriate OTA query and then translate the OTA response into instances of our travel ontology to update the information in the system.

3.3. Scenario 3

This scenario is most far fetched, but also should be considered. Let us assume that our TSS becomes a service that is available to other travel related entities. In this case the question has to be asked, what language can these entities use to "converse" with our system. One of obvious answers that should be clear by now is: by utilization of OTA messaging. Let us note that OTA messages, being a standard for travel related communication, actually separate the internal representation of data from the way that the conversation about this data takes place. Let us assume that a “Worldwide Travel Agency” (WTA) decided to utilize our TSS as their way of supporting clients, but also opened its services to other (smaller) travel agencies (this would be an example of service oriented architecture). We can now assume that the other travel agencies will send requests to the WTA’s TSS system using OTA messaging, without any knowledge as to how the actual data is represented inside of the TSS that the WTA uses.

To service these requests, the TSS has to be able to accept OTA messages and to translate them into an appropriate form to retrieve information from ontologicaly demarcated data. Hence, the translation between incoming OTA messages and queries that have to be executed in our system has to be provided. Following *Scenario 2* let us assume that an OTA message arrives to the TSS, requesting a list of carriers that fly from Tulsa, Oklahoma to Baltimore, Maryland. Since this data is stored internally, the TSS does not have to forward this query to the GDS, but can translate the received OTA message into a SPARQL query. This query will then be executed on the internal Jena database and response packed into an OTA message and send back to the requestor.

Summarizing, we have argued that in the proposed TSS at least the following translations between the internal functions and representations used in the system and the OTA messages are required:

(a) HTTP query-string into an OTA request,

(b) OTA response (OTA RS) into an instance of an ontology,

(c) instance of an ontology into an OTA response message (OTA RS),

(d) OTA request message (OTA RQ) into a SPARQL query,

(e) response to a SPARQL query into an OTA response message (OTA RS)

We are following with an example that illustrates one of these translations.

4. Translation example

Due to limited space we will focus our attention on description of equivalency between OTA response messages and instances of our ontology. Following examples provided in previous sections, let us assume that the WTA agency has received request for all direct flights and their availability for the route from Paris (PAR) to Warsaw (WAW) operated by Air France (AF). Hence, in Figure 1 we depict the received OTA RQ message for requested route and specified date and carrier as well as the number of passengers traveling together (with their preferences). This message is then translated into a following SPARQL query:

```
PREFIX flt: <AirTravel/Flight#>.
PREFIX trv:
  <AirTravel/AvailabilityDisplay#>.
PREFIX arc:
  <AirInfrastructureCodes/AirportCode#>.
PREFIX  arl:
  <AirInfrastructure/Airport#>.
PREFIX alc:
  <AirInfrastructureCodes/AirlineCode#>.
SELECT ?display
WHERE
{
  ?display a trv:AvailabilityDisplay;
           trv:details ?flight.
  ?flight  flt:origin ?origin;
           flt:destination ?dest;
           flt:departureTime;
           flt:smokingAllowed false;
           flt:flightDate 2007-03-20;
           flt:mcarrier alc:AF.
  ?origin  arl:airportCode arc:PAR.
  ?dest    arl:airportCode arc:WAW.
}
```

When executed against data in our system the following instances are returned as a result; due to space limitation instances are given in N3 notation and with only partial information (complete code may be found in [12]).

```
@prefix:
<http://www.ibspan.waw.pl/travel/tmp>.
@prefix
trv: <AirTravel/AvailabilityDisplay#>.
@prefix flt: <AirTravel/Flight#>.
@prefix
cls: <AirTravelCodes/IATAClasses#>.

:avlDispPARWAW
   a  trv:AvailabilityDisplay;
   :details [
       a  trv:AvailableFlightElement;
       trv:flight  flt:AF2046;
       trv:classAvail [
         a  trv:AvailableClasses;
         trv:classavailable [
           a  trv:AvailableClassElement;
           trv:class  cls:C;
           trv:noAvailableSeats  "9".
         ] ;
       ] ;
     ] ;
   trv:agtndata trv:agtnsign.
```

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<OTA_AirRulesRQ xmlns="http://www.opentravel.org/OTA/2003/05" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
        instance" xsi:schemaLocation="http://www.opentravel.org/OTA/2003/05 OTA_AirRulesRQ.xsd" EchoToken="36732"
        TimeStamp="2007-02-15T11:00:00" Target="Production" Version="2.001" SequenceNmbr="293" PrimaryLangID="en"
        DirectFlightsOnly="true">
    <POS>
        <Source AgentSine="102" PseudoCityCode="TGD" ISOCountry="ME" ISOCurrency="EUR" AirlineVendorID="1A">
              <RequestorID Type="5" ID="35896241" />
              <BookingChannel Type="1" />
        </Source>
    </POS>
    <RuleReqInfo NegotiatedFare="false">
          <FareReference>QPROYM</FareReference>
          <RuleInfo />
          <FilingAirline Code="YM" />
          <DepartureAirport LocationCode="TGD" />
          <ArrivalAirport LocationCode="VIE" />
     </RuleReqInfo>
  </OTA_AirRulesRQ>
```


Figure 1. OTA RQ message requesting availability with passenger preferences.


```xml
<?xml version="1.0" encoding="UTF-8" ?>
<OTA_AirAvailRS xmlns="http://www.opentravel.org/OTA/2003/05"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opentravel.org/OTA/2003/05
        OTA_AirAvailRS.xsd" EchoToken="12354" TimeStamp="2007-02-
        24T09:30:47-05:00" Target="Production" Version="1.002"
        SequenceNmbr="1">
      <Success />
<OriginDestinationOptions>
   <OriginDestinationOption>
     <FlightSegment DepartureDateTime="2007-03-20T09:35:00"
        ArrivalDateTime="2007-03-20T11:50:00" StopQuantity="0"
        FlightNumber="2046" JourneyDuration="P0Y0M0DT02H15M"
        SmokingAllowed="false" OnTimeRate="90" Ticket="ET">
         <DepartureAirport LocationCode="PAR" />
         <ArrivalAirport LocationCode="WAW" />
         <MarketingAirline CompanyShortName="AF" />
         <MarketingCabin CabinType="Business" RPH="2" />
         <MarketingCabin CabinType="Economy" RPH="3" />
         <BookingClassAvail ResBookDesigCode="C"
                   ResBookDesigQuantity="9" RPH="2" />
         <BookingClassAvail ResBookDesigCode="D"
                   ResBookDesigQuantity="5" RPH="2" />
         <BookingClassAvail ResBookDesigCode="F"
                   ResBookDesigQuantity="9" RPH="2" />
         <BookingClassAvail ResBookDesigCode="Y"
                   ResBookDesigQuantity="7" RPH="3" />
         <BookingClassAvail ResBookDesigCode="W"
                   ResBookDesigQuantity="7" RPH="3" />
         <BookingClassAvail ResBookDesigCode="E"
                   ResBookDesigQuantity="3" RPH="3" />
         <BookingClassAvail ResBookDesigCode="G"
                   ResBookDesigQuantity="R" RPH="3" />
       </FlightSegment>
   </OriginDestinationOption>
   <OriginDestinationOption>
     <FlightSegment DepartureDateTime="2007-03-20T12:25:00"
         ArrivalDateTime="2007-03-20T14:45:00" StopQuantity="0"
         FlightNumber="2346" JourneyDuration="P0Y0M0DT02H15M"
         SmokingAllowed="false" OnTimeRate="90" Ticket="Paper">
         <DepartureAirport LocationCode="PAR" />
         <ArrivalAirport LocationCode="WAW" />
         <MarketingAirline CompanyShortName="AF" />
         <MarketingCabin CabinType="Business" RPH="2" />
         <MarketingCabin CabinType="Economy" RPH="3" />
        <-- booking classes for this flight -->
     </FlightSegment>
   </OriginDestinationOption>
   <OriginDestinationOption>
     <FlightSegment DepartureDateTime="2007-03-20T16:00:00"
      ArrivalDateTime="2007-03-20T18:15:00" StopQuantity="0"
      FlightNumber="1046" JourneyDuration="P0Y0M0DT6H30M"
      SmokingAllowed="false" OnTimeRate="90" Ticket="Paper">
   <-- availability details for this flight -->
     </FlightSegment>
   </OriginDestinationOption>
   <OriginDestinationOption>
     <FlightSegment DepartureDateTime="2007-03-20T18:35:00"
        ArrivalDateTime="2003-08-13T20:50:00" StopQuantity="0"
        FlightNumber="1246" JourneyDuration="P0Y0M0DT4H30M"
        SmokingAllowed="false" OnTimeRate="90" Ticket="Paper">
   <-- availability details for this flight -->
     </FlightSegment>
   </OriginDestinationOptions>
</OTA_AirAvailRS>
```


Figure 2. OTA RS message containing the response to the request.

Based on the instance of *AvailabilityDisplay class* shown above and related classes from our ontology the OTA RS message, presented in Figure 2, is created. This OTA RS message may be forwarded to the agency that originally requested the information.

5. Concluding remarks

Expanding on results of our earlier research, here we have described our initial attempt at creating an interface between the Travel Support System and the AMADEUS GDS. Trying to keep pace with changes taking place in the airline industry, we have based our work on new technologies being currently considered for communication in airlines' systems. This resulted on definition of initial framework for message exchange between two noted systems. Here, we have proposed several message exchange scenarios and presented an example of messages exchanged to facilitate one of them. Results presented constitute a basis on which we are developing a complete parser that is going to support the six necessary translations specified at the end of Section 3. We will report on our progress in subsequent publications.

References

[1] Airline Coding Directory, Effective 1December 2006 – 31 May 2007, 71st Edition – Airline Destignators, IATA

[2] ARINC, http://www.arinc.com/

[3] ARINC-SITA Industry Technical Work Group On AIRLINE XML STANDARDS, ATI's TypeX DXF Specifications Document Solution Presentation "TXSpecs"

[4] Badica, C., Ganzha, M., Paprzycki, M.: UML Models of Agents in a Multi-Agent E-Commerce System. In: Proceedings of the IEEE Conference of E-Business Engineering. IEEE CS Press, Los Alamitos, CA, 56-61, 2005.

[5] Badica, C., Ganzha, M., Paprzycki, M., Pirvanescu, A.: Experimenting With a Multi-Agent E-Commerce Environment. In: V. Malyshkin (Ed.): Proceedings of PaCT'2005, LNCS 3606, Springer-Verlag, 393-402, 2005.

[6] City Code Directory, 43th Edition, Effective 9 December 2005 – 31 December 2006

[7] C. Bădică, A. Bădiţă, M. Ganzha, M. Paprzycki, Implementing Rule-Based Automated Price Negotiation in an Agent System. Journal of Universal Computer Science (to appear)

[8] C. Bădică, A. Bădită, M. Ganzha, M. Paprzycki, Developing a Model Agent-based E-commerce System. IJie Lu et. al. (eds.) E-Service Intelligence-Methodologies, Technologies and Applications, Springer, Berlin, 2007, 555-578

[9] Gawinecki M., Gordon M., Paprzycki M., Szymczak M., Vetulani Z., Wright J., Enabling Semantic Referencing of Selected Travel Related Resources. In: W. Abramowicz (ed.) Proceedings of the 8 th International Conference on Business Information Systems (BIS 2005), Poznań University of Economics Press, Poznań, Poland, 2005, 271-288

[10] Gordon M., Kowalski A., Paprzycki M., Pełech T., Szymczak M., Wąsowicz T., Ontologies in a Travel Support System. In: D. J. Bem et. al. (eds.) Internet 2005, Technical University of Wroclaw Press, 2005, 285-300

[11] http://e-travel.sourceforge.net/

[12] IATA SSIM, Mar 1, 2006 until Sep 30, 2006

[13] Jena 2 Ontology API – General concepts, http://jena.sourceforge.net/ontology/index.html#generalConcepts

[14] Jena, http://jena.sourceforge.net/

[15] M. Ganzha, M. Gawinecki, M. Paprzycki, R. Gasiorowski, S. Pisarek, W. Hyska (2006) Utilizing Semantic Web and Software Agents in a Travel Support System. In: A. F. Salam and Jason Stevens (eds.) Semantic Web Technologies and eBusiness: Virtual Organization and Business Process Automation, Idea (to appear)

[16] Minor Gordon, M. Paprzycki (2005) Designing Agent Based Travel Support System. In: Proceedings of the ISPDC 2005 Conference , IEEE Computer Society Press, Los Alamitos, CA, 2005, 207-214

[17] M. Vukmirović, M. Paprzycki, M. Szymczak (2006) Designing ontology for the Open Travel Alliance Airline Messaging Specification, In: M. Bohanec et. al. (eds.), Proceedings of the 2006 Information Society Multiconference, Josef Stefan Institute Press, 101-105

[18] M. Vukmirović, M. Szymczak, M. Gawinecki, M. Ganzha, M. Paprzycki, Designing new ways for selling airline tickets, Informatica (to appear)

[19] OpenTravel™ Alliance, Message Users Guide. 2006B Version 1.0, December 2006

[20] OTA, http://www.opentravel.org/

[21] PSC Resolutions Manual, Effective 1 June 2006 – 31 May 2007, 26th Edition

[22] Passenger Tariff Coordination Conferences Manual, Composite, Dec 9, 2005 until Dec 31, 2006

[23] Passenger XML Working Group (XMLWG), http://www.iata.org/workgroups/xmlwg.htm/

[24] SITA, http://www.sita.aero/default.htm/

[25] Szymczak M., Gawinecki M., Vukmirović M., Paprzycki M., Ontological reusability in state-of-the-art semantic languages, Proceedings of the XVIII Summer School of PIPS (to appear)

[26] TypeX, http://www.typex.aero/YACS/

[27] Vukmirović M., Ganzha M., Paprzycki M.: Developing a Model Agent-based Airline Ticket Auctioning System. In: Proceedings for the IIPWM Conference, LNAI

[28] Vukmirović M., Szymczak M., Ganzha M., Paprzycki M.: Utilizing Ontologies in an Agent-based Airline Ticket Auctioning System. In: Proceedings of the 28th ITI Conference, IEEE Computer Society Press, Cavtat, Dubrovnik, Croatia, 385-390

[29] Gawinecki M., Gordon M., Paweł Kaczmarek, Paprzycki M. The Problem of Agent-Client Communication on the Internet. Scalable Computing: Practice and Experience , 6(1), 2005, 111-123

[30] Paprzycki M., Kaczmarek P., Gawinecki M., Vetulani Z. (2005) Interakcja Użytkownik - Agentowy System Wspomagania Podróży. In: J. Kisienicki et. al. (eds.) Informatics and Modern Management , PTI Press, Katowice, 2005, 189-198

[31] Kaczmarek P. (AMU, 2005) Multimodal Communication Between Users and Software Agents

[32] Gawinecki M., Kruszyk M., Paprzycki M. (2005) Ontology-based Stereotyping in a Travel Support System. In: Proceedings of the XXI Fall Meeting of Polish Information Processing Society, PTI Press, 73-85

[33] Gawinecki M., Vetulani Z., Gordon M., Paprzycki M. (2005) Representing Users in a Travel Support System. In: Proceedings of the ISDA 2005 Conference, IEEE Computer Society Press, Los Alamitos, CA, 393-398

[34] Fink J., Kobsa A., "User Modeling for Personalized City Tours," Artificial Intelligence Review, 18, 2002, 33–74

[35] Greer, J.McCalla, G. 1993. Student modeling: the key to individualized knowledge based instruction. NATO ASI Series F, 125.