Chapter I

Developing Dynamic Packaging Applications using Semantic Web-Based Integration

Jorge Cardoso, Universidade da Madeira, Portugal

Abstract

Dynamic packaging has been introduced as an innovative technology allowing for the automated online configuration and assembling of packaged travel products for individual customers. Dynamic packaging applications require a suitable integration of heterogeneous, autonomous, and distributed tourism information systems. This integration is a complex and difficult issue. The Semantic Web, a relatively new concept, brings a set of emerging technologies and models that need to be explored and evaluated to assert its use for the implementation of more integrated dynamic packaging applications. In this chapter, we analyze dynamic packaging application requirements and present an architecture that enables the integration of tourism data sources and creation of dynamic packages using semantic annotation, semantic rules, ontologies, Web services, and Web processes. We will recognize that the Semantic Web is a good candidate able to supply a solution for overcoming the interoperability problems that (current) dynamic packaging applications face.

Introduction

Tourism has become one of the world's largest industry players, and its growth shows a consistent year-to-year increase. The World Tourism Organization (http://www.world-tourism.org/) predicts that by 2020 tourist arrivals around the world will increase over 200%. Tourism has become a highly competitive business for tourism destinations all over the world. Competitive advantage is no longer natural, but increasingly driven by science, information technology, and innovation.

The continuing growth in the use of the Internet has transformed the world into a global village. For example, e-tourism-related Web sites provide a vast amount of rich information, maps, pictures, sounds, and services on destinations throughout the world. A study by Forrester (Forrester, 2005) estimates that business-to-business (B2B) revenues will reach $8.8 trillion in 2005 and business-to-customer (B2C) revenues in the U.S. will reach $229.9 billion by 2008.

The Internet is already the primary source of tourist destination information for travelers. About 95% of Web users use the Internet to gather travel-related information and about 93% indicate that they visited tourism Web sites when planning for vacations (Lake, 2001). The number of people turning to the Internet for vacation and travel planning has increased more than 300% over the past 5 years. It has outpaced traditional sources of information on tourist destinations within a short period of time. One major cause for the growth of e-tourism is that it extends existing business models, reduces costs, and expands and introduces new distribution channels.

Evidence indicates that the effective use of information technology is crucial for tourism businesses' competitiveness and prosperity, as it influences their ability to differentiate their offerings as well as their production and delivery costs. Tourism is an information-based industry and one of the leading industries on the Internet. For example, it is anticipated that most sectors in the travel industry throughout the world will have Web sites on the Internet. Thus, it is vital for every tourism destination and travel business to embrace the use of information technology and exploit its potential.

Barnett and Standing (2001) argue that the rapidly changing business environment brought on by the Internet requires organizations to quickly implement new business models, develop new networks and alliances, and be creative in their marketing. In order to compete in the electronic era, businesses must be prepared to use technology-mediated channels, create internal and external value, formulate technology convergent strategies, and organize resources around knowledge and relationships (Rayport & Jaworski, 2001).

Tourism information systems (TIS) are a new type of business system that serve and support e-tourism and e-travel, such as airlines, hoteliers, car rental companies, leisure suppliers, and travel agencies. These systems rely on travel-related information sources to create tourism products and services. The information present on these sources can serve as the springboard for the development of a variety of systems, including dynamic packaging applications, travel planning engines, and price comparison applications.

In this chapter we are particularly interested in studying the development and implementation of dynamic packaging applications. Dynamic packaging can be defined as the combination of different travel products, bundled and priced in real time, in response to the requests of the consumer or booking agent. In dynamic packaging applications, consumer requirements shape the response of the packaging system, the final price, and the products of travel packages. Our approach to the development of dynamic packaging applications encompasses the use of the latest information technologies such as the Semantic Web, Web services, Web processes, and semantic packaging rules.

E-tourism is a perfect application area for Semantic Web technologies since information integration, dissemination, and exchange are the key backbones of the travel industry. Therefore, the Semantic Web can considerably improve e-tourism applications (DERI International, 2005). Dynamic packaging application solutions deal with B2B integration and B2C transactions. While organizations have sought to apply semantics to manage and exploit data or content to support integration, Web processes are the means to exploit its application, increasingly made interoperable with Web services.

Web services and Web processes are defined as loosely coupled, reusable components that encapsulate functionality and are distributed and programmatically accessible over standard Internet protocols. They constitute one of the “hot” areas of the Web technology supporting the remote invocation of business functionality over the Internet through message exchange. They provide an “information” layer that allows integrating different data standards to exchange information seamlessly without having to change the proprietary data schemas of tourism organizations.

Semantics can also be used to formally specify the packaging rules that influence which products will be part of dynamic packages. The use of semantic packaging rules has several benefits for dynamic packaging applications since travel managers or travel agents, without programming experience, can manage and change packaging rules to reflect market conditions; packaging policies can be easily communicated and understood by all employees; and rules can be managed in isolation from the application code.

Dynamic Packaging Applications

Currently, with most tourism information systems, travelers need to visit multiple independent Web sites to plan their trip, register their personal information multiple times, spend hours or days waiting for response or confirmation, and make multiple payments by credit card. Consumers are discouraged by the lack of functionalities. Dynamic packaging applications are emerging in response to these limitations and have caught the attention of major worldwide online travel agencies.

The Dynamic Packaging Model

A dynamic packaging application allows consumers or travel agents to customize trips by bundling trip components. Customers can specify a set of preferences for a vacation, for example, a 5-day stay on Madeira Island, then the dynamic packaging application dynamically accesses and queries a set of tourism information sources to find products such as air fairs, hotel rates, car rental companies, and leisure activity suppliers in real time. In the off-line world, such packages used to be put together by tour operators in brochures. This new dynamic packaging technology includes the ability to combine multiple travel components on demand, in creating a reservation. The package that is created is handled seamlessly as one transaction and requires only one payment from the consumer, hiding the pricing of individual components.

Main Players: Expedia, Travelocity, and Orbitz

The travel industry's three most dominant online agencies—Expedia, Travelocity, and Orbitz—are leading the development of dynamic packaging technology, and they continue to put significant investment into providing an efficient and sophisticated booking experience. Travelers are given the opportunity to construct customized packages by choosing the airline carrier, their flight, the hotel location, the car rental company, their insurance, other travel products such as theme park passes, and even tours.

Expedia is the largest online travel agency. Expedia follows the merchant model, that is, it consigns hotel rooms at a wholesale rate and resells them to consumers. The key in the merchant model is to negotiate satisfactory agreements with providers. Expedia has stated that the popular durations requested by consumers are not the traditional 7/14 night model, but holidays of 3, 5, and 8 nights, a level of flexibility that is outside the costing model of most charter-based, mass-market tour operators. This is one of the strategies having lead to its top market position. From the customers' view point, the Expedia business model has two major drawbacks. When Expedia sells all of its allocated hotel rooms, it informs customers that no rooms are available for sale. This is misleading because there might be rooms available outside of Expedia's allocated share. Moreover, Expedia does not fully disclose the taxes and fees that will be added to the sale price. In some cases additional tax and service fees mean that consumers might actually pay more than if they had booked the room directly from the hotel.

Expedia's use of dynamic packaging is one of the best among the competition: Using Expedia's Web site, consumers can book airline tickets and hotel rooms, and also book a shuttle to pick them up at the airport and set up prepaid restaurant meals. In this way Expedia focuses on the total journey of consumers. Expedia pioneered dynamic packaging in 2002 and now gets almost 30% of revenue from package buyers (Mullaney, 2004).

Travelocity provides Internet and wireless reservation information for more than 700 airlines, more than 55,000 hotels, and more than 50 car rental companies (PRNewswire, 2002). In addition, Travelocity offers more than 6,500 vacation packages, tour and cruise departures, and a vast database of destination and interest information. It is now the second largest online travel agency. Travelocity launched a new merchant model hotel program offering advantages so compelling that more than 2,000 hotels signed agreements to participate. Travelocity can pull rates and availability directly from the hotel's central reservation system (CRS). This eliminates the time and costs associated with manually allocating blocks of rooms to a separate system for discounted sales. Travelocity can provide a "single view" of room inventory. This is an advantage compared to the merchant model of competitors. Also, Travelocity pays the hotels immediately upon checkout, eliminating the waiting period for payment that hotels experience with other merchant model distributors.

Travelocity made a strategic acquisition of Site59.com, whose dynamic packaging technology allows Travelocity to respond to the growing popularity of Expedia's dynamic packages. Travelocity dynamic vacation technology will be the first to allow users to book specific airline seats and hotel rooms themselves, in real time. Travelocity has included taxes and fees in its products and strives to only list flights and rooms still available.

Since launching its Web site to the general public in June 2001, **Orbitz** has become the third largest online travel site in the world. It was founded by five major airlines, American, Continental, Delta, Northwest, and United. The main objective was to compete with Expedia and online ticketing sales, hoping to take advantage of increase in ticket sales online. The launch of Orbitz, a $100 million joint venture (Hospitality, 2005), demonstrates the high cost of entry into the travel space. It is a costly undertaking that requires cooperation with existing industry players. Therefore, new entrants face enormous challenges.

Orbitz had a perceived advantage over Travelocity and Expedia because it had a deeper inventory of "Web fares," the heavily discounted tickets promoted on the carriers' own Internet sites (CBS NEWS, 2003). This advantage has drawn wide-ranging criticism from Expedia and Travelocity with the claim that the airline-backed ticketing operation is antithetical to competition in the industry and hurts consumers. Orbitz has lowered distribution costs for its suppliers by sharing a portion of the fees that global distribution systems (GDSs) pay to Orbitz as an incentive for booking travel on their systems. Orbitz further reduced distribution costs for several airlines through their participation in the Orbitz Supplier Link technology program, which allows Orbitz to sell some tickets without using a GDS.

Orbitz's Web site has already completed the implementation of its dynamic packaging engine. One major characteristic of Orbitz strategy is that the customer relationship does not end when a customer buys a travel product. Orbitz is the only travel site with a customer care team that monitors nationwide travel conditions for travelers. The care team gathers and interprets Federal Aviation Administration (FAA), National Weather Service, and other data providing the latest information on flight delays, weather conditions, gate changes, airport congestion, or any other event that might impact travel via mobile phone, pager, personal digital assistant (PDA), or e-mail.

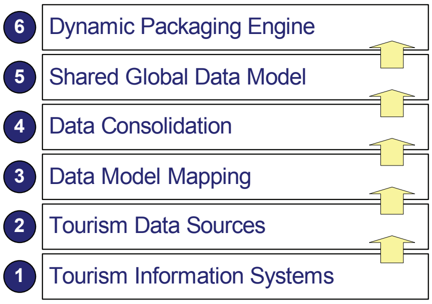
Dynamic Packaging Application Architecture

The development of dynamic packaging applications is a complex issue since it requires the integration of distributed systems with infrastructures that are not frequently encountered in more traditional centralized systems. For dynamic packaging applications to be successful it is indispensable to studying their architecture. The study of architectural strategies has a critical impact on early decisions in system development; it is both cost effective and efficient to conduct analyses at the architecture level, before substantial resources have been committed to development (Bass, Clements, & Kazman, 1998). Therefore, we will undertake a study of our approach to dynamic packaging application development by presenting its architecture.

We propose an architecture for dynamic packaging applications composed of six layers: (1) tourism information systems, (2) tourism data sources, (3) data model mapping, (4) data consolidation, (5) shared global data model, and (6) dynamic packaging engine. The relationships between these layers are illustrated in Figure 1.

To better understand the purpose of each architectural layer, we will briefly describe them in this section and give a detailed presentation in the following sections.

Figure 1. Architecture of semantically enabled dynamic packaging applications



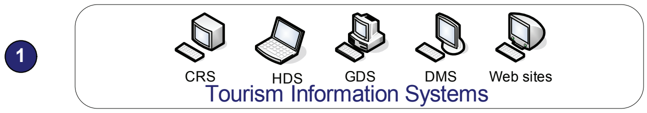
- **Tourism information systems.** The information needed to build dynamic packages is stored in tourism information systems, such as CRS, GDS, HDS, DMS, and Web sites.
- **Tourism data sources.** Each tourism information system makes travel data available through data sources in one or more formats, such as HTML, XML, RDF, flat files, relational model, and so forth.
- **Data model mapping.** In our architecture, data on data sources is mapped to the concepts of a common ontology to facilitate the integration of information.
- **Data consolidation.** The various segments of the common ontology constructed from individual data sources are consolidated using procedures described using an abstract business process model.
- **Shared global data model.** With the data consolidated in the previous level, we populated the shared global data model, represented with an e-tourism ontology, by creating instances.
- **Dynamic packaging engine.** Based on the information present in the e-tourism ontology, we extract knowledge to build dynamic packages.

Tourism Information System Integration

Tourism information systems provide travel agencies and customers with crucial information such as flight details, accommodations, prices, and the availability of services. Dedicated and specialized information systems are providing real time tourism data to travel agents, customers, and other organizations.

A few years ago, e-tourism applications were mainly focused on handling transactions and managing catalogs. Applications automated only a small portion of the

Figure 2. The various tourism information systems that need to be integrated



electronic transaction process, for example, taking orders, scheduling shipments, and providing customer service. E-tourism was held back by closed markets that could not use each other's services due to the use of incompatible protocols.

Business requirements of dynamic applications, however, are evolving beyond transaction support and include requirements for the interoperability and integration of heterogeneous, autonomous, and distributed tourism information systems. The objective is to provide a global and homogeneous logical view of travel products that are physically distributed over tourism data sources. However, in general, tourism information systems are not designed for integration. A considerable number of tourism information systems were developed in the 1960s when the integration of information systems was not a major concern.

One of the challenges that dynamic packaging applications face is the integration of the five tourism information systems most widespread in the tourism industry that are a fundamental infrastructure for providing access to tourism information, namely, computerized reservation systems (CRS), global distribution systems (GDS), hotel distribution systems (HDS), destination management systems (DMS), and Web sites (Figure 2).

Computerized Reservation System

A CRS is a travel supplier's own central reservation system (Inkpen, 1998). A CRS enables travel agencies to find what a customer is looking for and makes customer data storage and retrieval relatively simple. These systems contain information about airline schedules, availability, fares, and related services. Some systems provide services to make reservations and issue tickets. CRS were introduced in the 1950s as internal systems within individual organizations. With time and with the development of communication technologies they became available to travel agencies and other organizations. CRS are extremely popular and widespread, especially among airlines. It is estimated that 70% of all bookings are made through this channel (European Travel Agents' and Tour Operators' Associations, 2004).

Global Distribution System

A GDS is a super switch connecting several CRSs. A GDS integrates tourism information about airlines, hotels, car rentals, cruises, and other travel products. It is used almost exclusively by travel agents. The airline industry created the GDS concept in the 1960s. As with CRSs, the goal was to keep track of airline schedules, availability, fares, and related services. Prior to the introduction of GDSs, travel agents spent a considerable amount of time manually entering reservations. Since GDSs allowed automating the reservation process for travel agents, they were able to be productive and turn into an extension of the airline's sales force (HotelOnline, 2002). The use of these systems is expensive since they charge a fee for every segment of travel sold through the system. There are currently four major GDSs (Inkpen, 1998): Amadeus, Galileo, Sabre, and Worldspan. Today, 90% of all U.S. tickets are sold through these four global distribution systems (Riebeek, 2003).

Hotel Distribution System

An HDS works closely with GDSs to provide the hotel industry with automated sales and booking services. An HDS is tied into a GDS, allowing hotel bookings to be made in the same way as an airline reservation (Inkpen, 1998). HDSs may be categorized into two main types: (1) the HDS is linked directly to the hotel's own booking system and in turn linked with a GDS that can be accessed by booking agents, and (2) dedicated companies provide a reservation system linked to airline GDSs.

Destination Management Systems

DMSs supply interactively accessible information about a destination, enabling tourist destinations to disseminate information about products and services as well as to facilitate the planning, management, and marketing of regions as tourism entities or brands (Buhalis, 2002). These systems offer a guide to tourist attractions, festivals, and cultural events, coupled with online bookings for accommodation providers. They also feature weather reports, Web movies, and feed from Web cams positioned in popular tourist areas. One of the goals of DMS is to develop flexible, tailor-made, specialized, and integrated tourism products. Two of the most well known DMSs include Tiscover[1] (Austria) and Gulliver[2] (Ireland).

Direct Distribution using Web Sites

The Internet is revolutionizing the distribution of tourism information and sales. Small and large companies can have Web sites with "equal Internet access" to international tourism markets. Previously, many companies had to use their booking systems as platforms from which to distribute their products via existing channels, such as GDSs. Recently, companies such as the airlines, have chosen the strategy to sell tickets on their own Web sites to avoid using a GDS (Dombey, 1998). This is the simplest and cheapest strategy to sell tickets since they do not have to pay a fee to the GDS. Small providers, such as local hotels, can use the Internet to supply information about their products and allow the automatic booking of rooms and other services. A recent survey (O'Connor, 2003) revealed that over 95% of hotel chains had a Web site, with almost 90% of these providing technology to allow customers to book directly.

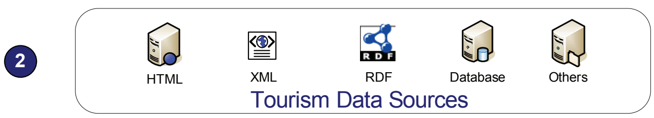
Tourism Data Source Integration

Given the rapid growth and success of tourism data sources, it becomes increasingly attractive to extract data from these sources and make it available for dynamic packaging applications. Manually integrating multiple heterogeneous data sources into applications is a time-consuming, costly, and error-prone engineering task. According to industry estimates, as much as 70% of information technology spending may be allocated for integration-related activities. Consequently, many organizations are looking for solutions that can make the integration of information systems an easier task (Gorton, Almquist, Dorow, Gong, & Thurman, 2005).

Data source integration is a research topic of enormous practical importance for dynamic packaging. Integrating distributed, heterogeneous and autonomous tourism information systems, with different organizational levels, functions, and business processes to freely exchange information can be technologically difficult and costly.

Dynamic packaging applications need to access tourism data sources to query information about flights, car rentals, hotels, and leisure activities. Data sources can be accessed using the Internet as a communication medium. The sources can

Figure 3. The various tourism data sources to be integrated



contain hypertext markup language (HTML) pages present in Web sites, databases, or specific formatted files, such as extensible markup language (XML), resource description framework (RDF), or flat files. To develop a robust dynamic packaging application it is important to classify each data source according to its type of data since the type of data will influence our selection of a solution to achieve data integration. For dynamic packaging applications, tourism data sources can host three major types of data: (1) unstructured data, (2) semi-structured data, and (3) structured data.

Types of Data

Data can be broken down into three broad categories (Figure 4): (1) unstructured, (2) semi-structured, and (3) structured. Highly unstructured data comprises free-form documents or objects of arbitrary sizes and types. At the other end of the spectrum, structured data are what is typically found in databases. Every element of data has an assigned format and significance.

Unstructured Data

Unstructured data is what we find in text, files, video, e-mails, reports, PowerPoint presentations, voice mail, office memos, and images. Data can be of any type and do not necessarily follow any format, rules, or sequence. For example, the data present on HTML Web pages are unstructured and irregular.

Unstructured data does not readily fit into structured databases except as binary large objects (BLOBs). Although unstructured data can have some structure—for example, e-mails have addressees, subjects, bodies, and so forth, and HTML Web

Figure 4. Unstructured, semi-structured, and structured data

Unstructured data

The university has 5600 students.
John's ID is number 1, he is 18 years old and already holds a B.Sc. degree.
David's ID is number 2, he is 31 years old and holds a Ph.D. degree. Robert's ID is number 3, he is 51 years old and also holds the same degree as David, a Ph.D. degree.

Semi-structured data

```
<University>
 <Student ID="1">
  <Name>John</Name>
  <Age>18</Age>
  <Degree>B.Sc.</Degree>
 </Student>
 <Student ID="2">
  <Name>David</Name>
  <Age>31</Age>
  <Degree>Ph.D. </Degree>
</Student>
….
</University>
```

Structured data

ID	Name	Age	Degree
1	John	18	B.Sc.
2	David	31	Ph.D.
3	Robert	51	Ph.D.
4	Rick	26	M.Sc.
5	Michael	19	B.Sc.

pages have a set of predefined tags—the information is not stored in a way that allows for easy manipulation by applications and computers.

Semi-Structured Data

Semi-structured data lie in between unstructured and structured data. *Semi-structured data are data that has some structure, but are not rigidly structured.* This type of data include unstructured components arranged according to some predetermined structure that can be queried using general-purpose mechanisms.

Semi-structured data is organized into entities. Similar entities are grouped together, but entities in the same group may not have the same attributes. The order of attributes is not necessarily important and not all attributes may be required. The size and type of same attributes in a group may differ. An example of semi-structured data is a curriculum vita (CV). One person may have a section of previous employment, another person may have a section on research experience, and another may have a section on teaching experience. We can also find a CV that contains two or more of these sections.

A very good example of a semi-structured formalism is XML which is a de facto standard for describing documents that is becoming the universal data exchange model on the Web and for B2B transactions. XML supports the development of semi-structured documents that contain both metadata and formatted text. Metadata is specified using XML tags and defines the structure of documents. Without metadata, applications would not be able to understand and parse the content of XML documents. Compared to HTML, XML provides explicit data structuring using Document Type Declaration (DTD) (XML, 2005) or XML Schema Definition (XSD) (World Wide Web Consortium, 2005b) as schema definitions. Figure 4 shows the semi-structure of an XML document containing students' records of a university.

Structured Data

In contrast, structured data *is very rigid* and uses strongly typed attributes. Data is organized in entities and similar entities are grouped together using relations or classes. Entities (records or tuples) in the same group have the same attributes. Structured data have been very popular since the early days of computing, and many organizations rely on relational databases to maintain very large structured repositories. Recent systems, such as customer relationship management (CRM), enterprise resource planning (ERP), and content management systems (CMS) use structured data for their underlying data model.

What Tourism Data Sources Need to be Integrated?

Data sources contain tourism information which is fundamental for dynamic packaging applications. A data source includes both the source of data itself and the connection information necessary for accessing the data. Data sources are uniquely identifiable collections of stored data called data sets for which there exists programmatic access and for which it is possible to retrieve or infer a description of the structure of the data, that is, its schema. We have identified various tourism data sources that need to be considered when integrating tourism information systems: flat files; HTML Web pages; XML and RDF data sources; and relational databases.

Flat Files

A tourism data source can be a flat file that is accessible via the file system application program interface (API). A flat file is a generic term for text file formats such as comma separated value (CSV), tab delimited, fixed width, and so forth. Flat file formats are supported by a wide collection of tourism information systems because they can be used as an interoperable format for exchanging information between different applications. In practice, flat files have proven to be very useful for allowing users to share information.

However, though they are supported by many applications, flat files generally require additional processing to be integrated seamlessly with common data formats. Since tourism information can often be stored in flat files, dynamic packaging applications need to include methods to integrate these data into a common data model. This requires the development of specific software application modules to access and extract the necessary data.

Hyper Text Markup Language

With the growth of the Web, many tourism information providers already have Web sites for storing and advertising the description of tourism services and products. Almost all Web sites support static HTML pages accessible through a Web server via the HTTP protocol. Dynamic packaging applications require integrating Web-based data sources in an automated way for querying, in a uniform way, across multiple heterogeneous Web sites, containing tourism-related information.

Extensible Markup Language

XML (XML, 2005) is a semi-structured data model that promises to accelerate the construction of systems that integrate distributed and heterogeneous data. XML provides a common format for data across the network and is being supported by a vast number of data management tools. Unlike HTML, which controls how data is represented, XML allow organizations to define data schemas that relate XML tags with data content.

The travel industry has been adopting XML as a common format for data exchanged across travel partners. For example, the Open Travel Alliance (OTA)[3] provides a vocabulary and grammar for communicating travel-related information as tags implemented using XML across all travel industry segments. XML is well suited in this context since schema for defining the XML tags can differ among industries, and even within organizations. Furthermore, the three major worldwide online travel agencies—Expedia, Travelocity, and Orbitz—have also adopted the XML standard to enable the exchange of supplier information using XML-based exchange formats.

Resource Description Framework

The RDF (World Wide Web Consortium, 2005a) provides a standard way of referring to metadata elements and metadata content. RDF builds standards for XML applications so that they can interoperate and intercommunicate more easily, facilitating data and system integration and interoperability. RDF is a simple general-purpose, metadata language for representing information on the Web and provides a model for describing and creating relationships between resources. A resource can be a thing, such as a person, a song, or a Web page. With RDF it is possible to add predefined modeling primitives for expressing semantics of data to a document without making any assumptions about the structure of the document. In a first approach, it may seem that RDF is very similar to XML, but a closer analysis reveals that they are conceptually different. If we model the information present in an RDF model using XML, human readers would probably be able to infer the underlying semantic structure, but general purpose applications would not.

While XML is being widely used across all travel industry segments, RDF is a recent data model and its adoption is just starting in areas such as digital libraries, Web services, and bioinformatics. Nevertheless, as the number of organizations adhering to this standard starts growing, it is expected that the travel industry will also adopt it.

Databases

In modern tourism organizations, it is almost unavoidable to use databases to produce, store, and search for critical data. Yet, it is only by combining the information from various database systems that dynamic packaging applications can take a competitive advantage from the value of data. Different travel industry segments use distinct data sources. This diversity is caused by many factors including lack of coordination among organization units; different rates of adopting new technology; mergers and acquisitions; and geographic separation of collaborating groups.

To develop dynamic packaging applications, the most common form of data integration is achieved using special-purpose applications that access data sources of interest directly and combine the data retrieved with the application itself. While this approach always works, it is expensive in terms of both time and skills, fragile due to the changes to the underlying sources, and hard to extend since new data sources require new fragments of code to be written. In our architecture, the use of semantics and ontologies to construct a global view will make the integration process automatic, and there will be no requirement for a human integrator.

Tourism Data Source Integration

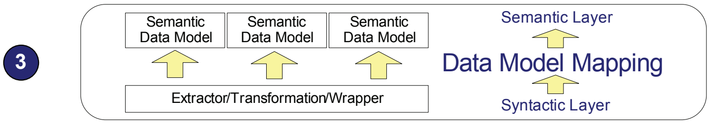
The technologies and infrastructures supporting the travel industry are complex and heterogeneous. The vision of a comprehensive solution to interconnect many applications and data sources based entirely on standards, such as the one provided by OTA (2004), that are universally supported on every computing platform, is not achieved in practice and far from reality.

Data integration is a challenge for dynamic packaging applications since they need to query across multiple heterogeneous, autonomous, and distributed (HAD) tourism data sources produced independently by multiple organizations in the travel industry. Integrating HAD data sources involves combining the concepts and knowledge in the individual tourism data sources into an integrated view of the data. The construction of an integrated view is complicated because organizations store different types of data, in varying formats, with different meanings, and reference them using different names (Lawrence & Barker, 2001).

To allow the seamless integration of HAD tourism data sources rely on the use of semantics. Semantic integration requires knowledge of the meaning of data within the tourism data sources, including integrity rules and the relationships across sources. Semantic technologies are designed to extend the capabilities of data sources allowing to unbind the representation of data and the data itself and to give context to data. The integration of tourism data sources requires thinking not of the data itself but rather the structure of those data: schemas, data types, relational database constructs, file formats, and so forth. Figure 5 illustrates the component in layer 3 of our architecture which carries out the mappings between different data models.

This layer can be seen as a middleware level that implements the interfaces to the data sources to be integrated. These interfaces must overcome the heterogeneities of communication protocols as well as the heterogeneities regarding programming languages. Since the results are typically returned in different formats, the interfaces should translate them into the reference data model which is used inside the middleware.

Figure 5. Mapping between different data models



The *syntactic* data present in the tourism data source, such as databases, flat files, HTML and XML files, are extracted and transformed using extractors and wrappers. An important aspect of tourism data sources is that there is no single generic method to retrieve data source data. Additionally, the schema of the tourism data sources may or may not be available. In some data sources, such as XML documents, the data sets may be self-described and schema information may be embedded inside the data sets. In other cases, such as with databases, the system may store and provide the schema as part of the data source itself but separately for the actual data. Finally, some sources may not provide any schema. This is the case of HTML Web pages. For this situation, methods need to be developed to analyze the data and extract its underlying structure.

Once the data has been extracted and transformed, we use metadata to link the data with tourism ontologies. Tourism ontologies are the backbone of semantic dynamic packaging applications and explicitly define a set of shared tourism concepts and their interconnections. They make explicit all concepts in a taxonomical structure, their attributes, and relations. Wrappers, information extraction, and text analysis combine information with ontologies and thereby create metadata. These tasks can be done automatically.

Putting a *semantic layer* on a syntactical architecture creates an environment where integration issues can be upgraded to an abstract level where graphical modeling allows a higher degree of flexibility when developing and maintaining semantic integration.

Data Integration using a Global Data Model

One simple approach to data integration is to implement each interface to data sources as part of individual development projects by hand coding the necessary data conversions. This approach is time consuming and error prone. It is necessary to implement N*(N–1) different translation interfaces to integrate N data sources. For dynamic packaging applications—where more than 100 tourism data sources may need to be integrated—this approach is not feasible.

A more advanced approach uses hubs or brokers to achieve data and process integration. With this approach it is necessary to have two translation interfaces per data source, one interface in and one out of the hub or broker. The number of required interfaces between systems is 2*N. The data is not translated directly from a source system to a destination system, but it is translated using a global data model present in the hubs or brokers.

Another solution is to map all data sources onto an expressive global data model and automatically deploy all the translation interfaces from these mappings. This approach requires N mappings and the use of ontologies to develop expressive global data models. In our architecture for dynamic packaging applications, we use this last approach.

Data Extraction and Transformation

To achieve tourism data source integration, extractors and wrappers can be used to extract the data that will be reconsolidated later. The extractors attempt to identify simple patterns in data sources and then export this information to be mapped through a wrapper. Since dynamic packaging applications use information stored in various HAD data sources, an extractor has to be implemented for each kind of data source to import. Therefore, a database extractor, an HTML extractor, an XML extractor, and an RDF extractor have to be implemented.

As an example, let us describe the structure of an HTML extractor. Dynamic packaging applications should be able to extract relevant information from an unstructured set of HTML Web pages describing tourism products and services. The role of the *HTML extractor* is to convert the information implicitly stored as an HTML document, which consists of plain text with some tags, into information explicitly stored as part of a data structure. This information is processed in order to provide meaning to it, so that dynamic packaging applications can "understand" the texts, extract, and infer knowledge from it. As will be shown later, this process of providing meaning to the unstructured texts is achieved using e-tourism ontologies. In the case of the Web, the extractor has to deal with the retrieving of data, via the HTTP protocol (through a GET or a POST method). An extractor is split into two separate layers:

1. retrieval layer
2. extraction layer

The retrieval layer deals with accessing the source through a GET or a POST method. This layer is in charge of building the correct URL to access a given resource and to pass the correct parameters. It should also handle redirections, failures, and authorizations.

The extraction layer is specific to the resource and deals with the actual extraction, taking advantage of the HTML grammar as well as regular expression patterns. Each extraction layer consists of a set of extraction rules and the code required to apply those rules. The extraction language should be expressive enough to capture the structure expressed by the resource or document. At this level the extracted information should be regarded as a string.

To program our extractors we have selected Compaq's Web language (formerly known as WebL) (Compaq Web Language, 2005). WebL is an imperative, interpreted scripting language for automating tasks on the Web that has built-in support for common Web protocols like HTTP and FTP, and popular formats such as HTML and XML.

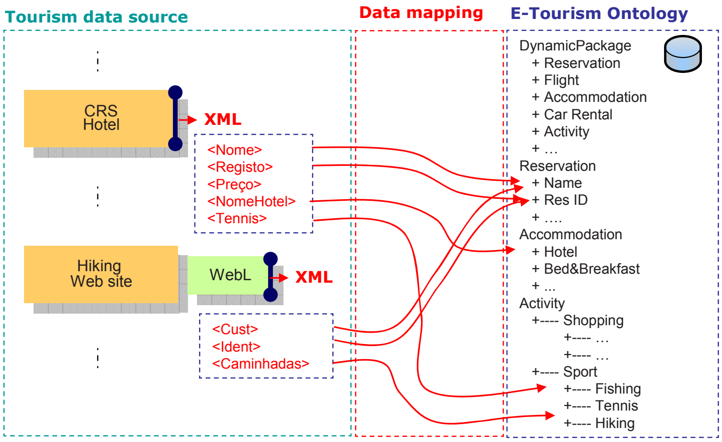
A critical problem in developing dynamic packaging applications involves accessing information formatted for human use and transforming it into a structured data format (Werthner & Ricci, 2004). *Wrappers* are one of the most commonly used solutions to access information from data sources being in charge of transforming the extracted information into the target structure that has been specified according to the user's needs. Wrappers have to implement interfaces to data source and should take advantage of generic conversion tools that can directly map extracted strings into say dates, zip codes, or phone numbers. These interfaces must overcome the heterogeneities of communication protocols as well as the heterogeneities regarding programming languages.

Data Model Mapping

There are many factors that make data integration for dynamic packaging applications a difficult problem. However, the most notable challenge is the reconciliation of the semantic heterogeneity of the tourism data sources being integrated. For dynamic packaging applications one of the best solutions toward reconciling semantic heterogeneity is the use of languages for describing semantic *mappings*, expressions that relate the semantics of data expressed in different structures (Lenzerini, 2002). Figure 6 illustrates the mappings established between XML data sources and the semantic data model used by our dynamic packaging application. Our common data model is defined using an e-tourism ontology specified using the Web Ontology Language (OWL) (World Wide Web Consortium, 2004).

OWL offers a common open standard format capable of representing both structured data, semi-structured, and unstructured data. Thus, OWL can be used as a common interchange format. We will discuss the details of this approach in section 3.5. For each tourism data source type, that is, flat files, relational models, XML, HTML, or RDF, mappings need to be defined to reference concepts present in our e-tourism ontology.

Figure 6. Mapping between different data models



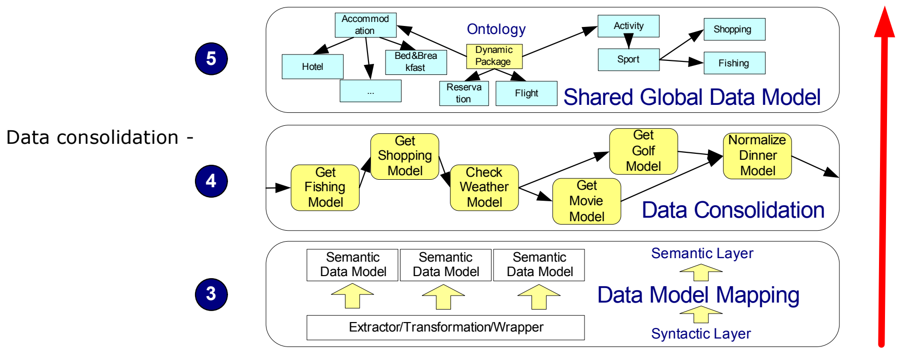
Data Consolidation

Data consolidation focuses on the orchestration of interactions between multiple local data models, as illustrated in Figure 7. Local data models (layer 3) are combined to create a global data model (layer 5) using the data consolidation layer (layer 4). As explained previously, to facilitate the integration of data source and construct local and global data models, we have adopted OWL as the standard format for information exchange.

One of the key principles of our approach is the separation of the process being implemented from the data being manipulated. We consolidate the semantic data models using processes to subsequently create a shared global data model. To achieve this incorporation, we define processes using workflow management systems and technology. We use two main software components to consolidate data: process designer and workflow engine. The process designer permits graphically designing processes that will consolidate the semantic data models. This tool permits defining business rules representing the integration logic. The workflow engine is a state machine that executes the workflow activities that are part of a process. It supports the execution of decision nodes; subprocesses; exception handling; forks and joins; and loops.

The processes describing the activities that are necessary to construct our shared global data model, based on the semantic data models, are formally specified using the business process execution language for Web services (BPEL4WS) (BPEL4WS, 2003) and semantic data models are interfaced with Web services (Chinnici, Gudgin, Moreau, & Weerawarana, 2003). BPEL4WS provides a language for the formal

Figure 7. Integration with business processes



specification of (business) processes by defining an integration model that facilitates the development of automated process integration in both intra-organization and B2B settings.

At runtime, as the processes are executed, their Web services are invoked. Web services present an efficient solution to reduce integration efforts and to quicken the creation of interfaces that allow for communication with semantic data models. In dynamic packaging applications, Web service-based solutions have the following advantages:

- loosely coupled integration of tourism information systems leading to reduced development costs and more flexibility, and
- reduced dynamic packaging applications' complexity due to the use of standardized interfaces.

Web services are easier to design, implement, and deploy than any other traditional distributed technology, such as RPC and CORBA. At the foundation of Web services architecture are software standards and communication protocols such as XML; Simple Object Access Protocol (SOAP) (World Wide Web Consortium, 2002); Hyper Text Transfer Protocol (HTTP); Universal Description, Discovery, and Integration (UDDI) (UDDI, 2002); and Web Services Description Language (WSDL) (Christensen, Curbera, Meredith, & Weerawarana, 2001), which allow information to be accessed and exchanged easily among different programs. These technologies allow applications to communicate with each other regardless of the programming languages they were written in or the platform they were developed for. Web services are not used to build monolithic systems; they are a set of technologies with the objective of putting together existing applications to create newly distributed systems.

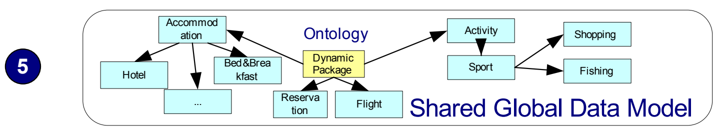
Shared Global Data Model

In order to develop efficient dynamic packaging applications, we believe that it is not required to adopt a common hardware platform or common database vendor. What is needed is a "shared global data model across participating tourism information systems." Requiring the organizations of the tourism industry to have a common hardware platform or database is not realistic. Figure 9 shows the various approaches to data integration.

The use of a shared global data model is a cornerstone of the design of many applications that require data integration. It brings integration costs and efforts down to a minimum. With a shared global data model a dynamic packaging application can merge all the information made available by CRS, GDS, HDS, DMS, and travel agents' Web sites, thus allowing cross-departmental and cross-organizational integration. Our shared global data model is represented with an ontology providing a common understanding of tourism data and information (Figure 9).

Figure 8. Tight and loose coupling approaches to data integration (Robbins, 1996)

Figure 9. Shared global data model defined using the e-tourism ontology



In the following sections we discuss the semantic model and semantic language selected to represent our shared global data model, the problems that semantic data sources face when integrated, and the steps involved in the development of our e-tourism ontology.

Shared Common Vocabulary

A shared global data model is not useful for data integration unless the sources being integrated share common vocabulary elements representing some shared conceptual model. Depending on the approach, different data models can be used to add semantics to terms—such as controlled vocabularies, *taxonomies,* thesaurus, and ontologies—and different degrees of semantics can be achieved.

Controlled vocabularies are at the weaker end of the semantic spectrum. A controlled vocabulary is a list of terms that have been enumerated explicitly with an unambiguous and non-redundant definition. A *taxonomy* is a subject-based classification that arranges the terms of a controlled vocabulary into a hierarchy without doing anything further. A thesaurus is a networked collection of controlled vocabulary terms with conceptual relationships between terms. It is an extension of a taxonomy by allowing terms to be arranged in a hierarchy and also allowing other statements and relationships to be established between terms, such as equivalence, homographic, hierarchical, and associative (National Information and Standards Organization, 2005).

Ontologies are similar to taxonomies but use richer semantic relationships among terms and attributes, as well as strict rules about how to specify terms and relationships. Compared to the other approaches, ontologies provide a higher degree of expressiveness. Furthermore, expressive standards have already been developed (for example, OWL [World Wide Web Consortium, 2004]) to construct ontologies and are being used in practical applications. For these reasons, we have selected ontologies for our dynamic packaging architecture to explicitly connect data from tourism information systems and to allow machine-processable interpretation of data.

Semantic Integration

To provide a dynamic packaging application for integrating disparate heterogeneous data sources, a common modeling language is needed to describe data, information, and knowledge. Since computers have no built-in mechanism for associating semantics to words and symbols, an ontology is required to allow dynamic packaging applications to determine semantically equivalent expressions and concepts residing in HAD tourism data sources. Agreeing on the terminology and sharing the same ontology for each tourism domain is a pre-condition for data sharing and integration (Wiederhold, 1994).

After studying several online travel, leisure, and transportation sites, we concluded that there is a lack of agreement on conventions in the tourism industry. The following are some of the differences found among several data sources:

- Web sites written in English use syntactically different words than Web sites written in Portuguese, but with the same semantics. For example, tennis/tenis, walking/caminhadas, and time/hora.
- The price of tourism products and services are expressed in many different currencies (euros, dollars, British pounds, etc.)
- The time specifications do not follow a standard format. Some Web sites state time in hours, others in minutes, others in hours and minutes, and so forth.
- The way of expressing time also varies. For example, 1 hour and 30 minutes, 1h and 30 min, 1:30 h, 90 min, one hour and thirty minutes, ninety minutes, 1:30 pm, and so forth.
- The keywords used to specify a date are not expressed in a normalized way. Some Web sites express a day of the week using the words Monday, Tuesday,…, Sunday, while other use the abbreviations M, T, …, Su.
- The temperature unit scale is not standard. It can be expressed either in degrees centigrade or in degrees Celsius.
- Numerical values are not expressed in a normalized way. They can be expressed with numbers: 1, 2, and 3 or with words such as one, two, and three.

One big challenge for dynamic packaging applications is to find a solution to cope with the nonstandardized way of describing tourism products and services. There are no conventions or common criteria to express transportation vehicles, leisure activities, and weather conditions when planning for a vacation; several ways were found among all the tourism data sources consulted. Our objective is to find a solution to surpass this lack of standardization by automatically understanding the different ways of expressing tourism products and services. We argue that semantics and ontologies are good candidates for dynamic packaging information systems since they allow us to associate metadata to data sources making the data machine understandable and processable.

E-Tourism Ontologies

Ontologies are the key elements enabling the shift from a purely syntactic to a semantic integration and interoperability. An ontology can be defined as the explicit and formal descriptions of concepts and their relationships that exist in a certain universe of discourse. When a particular user group commits to an ontology, it has been proven to be a solution for data integration because it offers a shared, organized, and common understanding of data which allows for a better integration, communication, and interoperability of inter- and intra-organizational tourism information systems.

Ontologies describe the things that exist in a domain. This includes properties, concepts and rules, and how they relate to each other. For dynamic packaging applications, an ontology with the appropriate tourism concepts needs to be built for identifying destinations, activities, weather forecasts, places, dates, and relationships. We identify the need for two distinct types of ontologies: local ontologies and shared global ontologies. Local ontologies define the semantics of specific tourism data source domains, such as hotels, car rentals, and airlines. In addition, we also consider the notion of shared global ontologies, which are common semantics shared between all the tourism domains and tourism information systems, that is, these ontologies model the information that resides in many separate domains.

Our initial tasks were to select a semantic language to model our ontologies (local and shared global ontologies), select an ontology editor to construct, browse, and manage the ontologies under development, and adopt a methodology to develop the ontologies. These tasks are described in the following sections.

Ontology Language Selection

Several languages have been developed to support the Semantic Web. These structured languages can carry meaning besides giving structure to data. Some languages are more directed to providing meaning to data, while others go further and can make assertions and infer knowledge.

In this area, the major developments are being made by an international Semantic Web research activity, spearheaded by the World Wide Web Consortium (W3C) (www.w3.org) and the Defense Advanced Projects Research Agency (DARPA) Agent Markup Language (DAML, 2005) program. The newest languages are developed based on the progress from previous ones, evolving and improving their characteristics. The most relevant semantic languages that need to be considered for developing ontologies for e-tourism are the following:

- **RDF.** RDF (W3C, 2005) and RDF schema (RDFS) became a W3C recommendation in 1999. It is a general framework to describe the contents of Internet resources. RDFs can be used directly to describe an ontology by making objects, classes, and properties available to programmers.
- **DAML+OIL.** The DARPA agent markup language + ontology inference layer (DAML+OIL) (DAML, 2005) is an extension of XML and RDF. DAML+OIL aims at complete support for defining ontologies. It provides rich constructors for forming complex class expressions and axioms for enabling reasoning and inference on ontology data.
- **OWL.** OWL (W3C, 2004) is a semantic markup language for publishing and sharing ontologies on the Web. It is the newest Semantic Web standard and became a W3C recommendation in February 2004.

From the different Semantic Web languages available (e.g., RDF, RDFS, DAML+OIL, and OWL) we have selected OWL to develop our e-tourism ontologies. This decision was based on two reasons. Firstly, OWL is a standard developed as a vocabulary extension of RDF, RDFS, and is derived from DAML+OIL. The standardization of OWL by the W3C allows semantics to move out of the research and development community and into broad-based, commercial-grade platforms for building highly distributed and cross-enterprise applications. Secondly, OWL provides a sound theory of meaning from which to build highly expressive data models. It expresses and includes a large set of primitives that are indispensable to building expressive ontologies. Primitives include cardinality constraints, class expressions, data types, enumerations, equivalence, and inheritance. OWL language is particularly well suited to formalize ontologies for the tourism industry by defining classes and properties of those classes and defining individuals and asserting properties about them. Furthermore, it is possible to conduct advanced knowledge inference, compared to other approaches.

Editor Selection

Ontology editors are tools that enable viewing, browsing, codifying, and modifying ontologies. Choosing the right editor for our project can become a daunting task since many choices exist and an appropriate tool selection depends on the level of user experience, the languages supported, the architecture, and the scalability.

Examples of popular editors include OilEd, OntoEdit (n.d.), WebODE, and Protégé (n.d.). OntoEdit is an ontology engineering environment supporting the development and maintenance of ontologies using graphical means. The editor supports representations of F-Logic, RDF Schema, and OIL. OilEd (Bechhofer, Horrocks, Goble, & Stevens, 2001) is an ontology editor allowing the user to build ontologies using DAML+OIL. Unfortunately, the current version of OilEd does not provide a full ontology development environment. It does not support the development of large-scale ontologies, versioning, argumentation, and many other activities that are involved in ontology construction. WebODE (Arpírez, Corcho, Fernández-López, & Gómez-Pérez, 2003) is a scalable workbench for ontological engineering that provides services for editing, browsing, importing, and exporting ontologies to classical and Semantic Web languages. Protégé (n.d.) is an extensible, platform-independent environment for creating and editing ontologies and knowledge bases. It is a tool which allows users to construct domain ontologies, having various storage formats such as OWL, RDF, and XML.

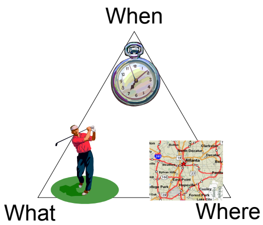
After conducting an analysis of ontology editors, we have selected Protégé (n.d.) to construct our ontologies for the tourism industry for four main reasons: (1) it includes implementations for the major computing platforms (such as Mac OS X, AIS, Solaris, Linux, and Windows), (2) it allows the construction of ontologies using OWL, (3) it is supported by a strong community of developers, such as academic, government, and biomedicine, and (4) it is a free and open source tool.

Ontology Development Methodology

Tourism is a data rich domain. This data is stored in many hundreds of data sources and many of these sources need to be used in concert during the development of the dynamic package and its applications. Our e-tourism ontologies provide a way of viewing the world of tourism. They organize tourism-related information and concepts. It will become clear later how the ontologies will allow us to achieve integration and interoperability through the use of a shared vocabulary and meanings for terms with respect to other terms. It should be noted that this is a work in progress; our tourism ontologies are not complete yet. We are still gathering new concepts for the taxonomies and developing new axioms.

Our ontologies were built to answer three main questions (Figure 10) that can be asked when developing dynamic packages for a tourist: what, where, and when.

Figure 10. What, where, and when



- **What.** What can a tourist see, visit, and what can he do while staying at a tourism destination?
- **Where.** Where are the interesting places to see and visit located?
- **When.** When can the tourist visit a particular place? This includes not only the day of the week and the hours of the day, but also the atmospheric conditions of the weather. For example, some activities cannot be undertaken if it is raining.

There are several ways of building ontologies (Fernández López, 1999; Jones, Bench-Capon, & Visser, 1998; Uschold & Gruninger, 1996). Our approach has involved the following steps:

- We have devised a unique and explicit definition for concepts from the tourism domain. Examples of concepts include *nightlife, sightseeing, relaxation,* and *shopping.* These definitions were precise enough to discriminate the various concepts in the ontologies.
- A root node concept has then been selected to embrace the variety of tourism domain-relevant concepts.
- Concepts were arranged and structured using classes and subclasses. The resulting ontology was transformed into a hierarchical tree. For example, *nightlife, sightseeing, relaxation,* and *shopping* are subclasses of the concept *activity.*
- Information concerning the disjointedness of classes was made explicit. Relations, such as inverse and transitivity, were also identified. For example, the *sightseeing* concept is disjoint from the *shopping* concept.
- Background knowledge for each concept was added to express domain-relevant properties. For example, a tourism organization has a telephone number, a fax number, an address, and an e-mail.

Ontology Creation

In an early stage of our project, the ontologies were implemented using Protégé (n.d.) editor. This was a very time-consuming task since it was necessary to find out information about real tourism activities and infrastructures on the Web and feed them into the knowledge base.

The main components of the tourism ontologies are concepts, relations, instances, and axioms. A concept represents a set or class of entities within the tourism domain. Activity, Organization, Weather, and Time are examples of concepts used. These concepts were represented in OWL in the following way:

```
…
<owl:Class rdf:ID="Activity"/>
<owl:Class rdf:ID="Organization"/>
<owl:Class rdf:ID="Weather"/>
<owl:Class rdf:ID="Time"/>
<owl:Class rdf:ID="Directions"/>
<owl:Class rdf:ID="Transportation"/>
…
```

The class **Activity** (which answers to the question "What") refers to sports, such as skiing, sightseeing or any other activity, such as shopping or visiting a theatre. The class **Organization** (which answers to the question "Where") refers to the places where the tourist can carry out an activity. Examples of infrastructure that provide the means for exerting an activity include restaurants, cinemas, or museums. The classes **Time** and **Weather** (which answers to the question "When") refers to the time and weather conditions which allow a tourist to carry out an activity at a certain place. The ontologies also include relations which describe the interactions between concepts or the concept's properties. For example, the concepts Fishing and Hiking are subconcepts of the concept Sport.

```
…
<owl:Class rdf:ID="Fishing">
   <rdfs:subClassOf>
     <owl:Class rdf:about="#Sport"/>
   </rdfs:subClassOf>
 </owl:Class>
 <owl:Class rdf:ID="Hiking">
   <rdfs:subClassOf>
     <owl:Class rdf:about="#Sport"/>
   </rdfs:subClassOf>
</owl:Class>
…
```

The ontologies also include associative relationships. Relationships relate concepts across a taxonomy of concepts. For example, the relationship hasActivity related the class **Organization** with the class **Activity**. This means that an organization in the tourism industry may supply a kind of activity to its customer, such as Hiking and Surfing.

```
…
  <owl:ObjectProperty rdf:ID="hasActivity">
    <rdfs:range rdf:resource="#Activity"/>
    <rdfs:domain rdf:resource="#Organization"/>
</owl:ObjectProperty>
…
```

While classes describe concepts in the domain, specific elements of a class are instances. For example, a class of **WeatherConditon** represents all the weather conditions that can be verified. Specific weather conditions are instances of this class, such as Cloudy, Showers, and Scattered Showers. However, deciding whether something is a concept of an instance is difficult and often depends on the application (Brachman, McGuinness, Patel-Schneider, Resnick, & Borgida, 1991).

Finally, axioms are used to constrain the values for classes and instances. Axioms are used to associate classes and properties with either partial or complete specifications of their characteristics and to give other logical information about classes and properties (W3C, 2004). For example:

```
…
  <owl:Class rdf:about="#Surfing">
    <rdfs:subClassOf rdf:resource="#Sport"/>
    <owl:disjointWith rdf:resource="#Hiking"/>
</owl:Class>
…
```

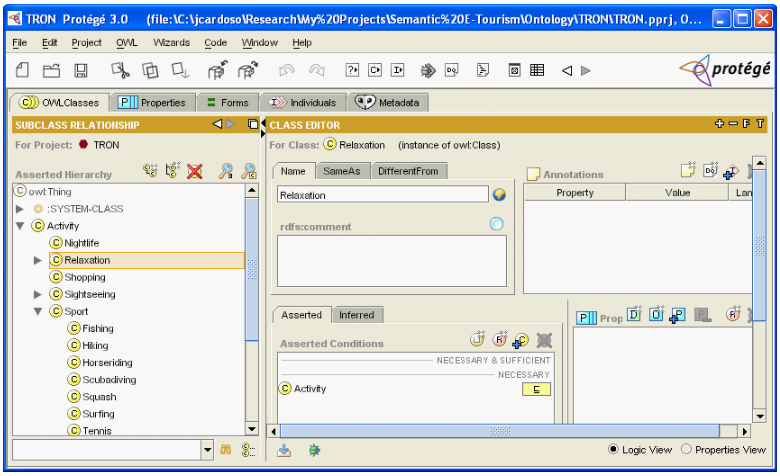
This example expresses that instances belonging to one subclass, for example, Surfing, cannot belong to another subclass, for example, Hiking. A partial view of one of the e-tourism ontologies developed using Protégé (n.d.) is illustrated in Figure 11.

Creating Dynamic Packages

Dynamic packages are automatically created by the dynamic packaging engine. Our architecture includes not only the dynamic packaging engine, but also the rule editor, rule repository, and the rule engine (Figure 12).

The configuration of the dynamic packaging engine involves the following activities. During the rule development phase, the rule designer defines packaging rules using the rule editor application. The rule editor, a component that provides an interface to the rule repository, supports the creation and modification of packaging rules through a graphical user interface. Packaging rules are codified and stored in

Figure 11. Using Protégé to develop e-tourism ontologies

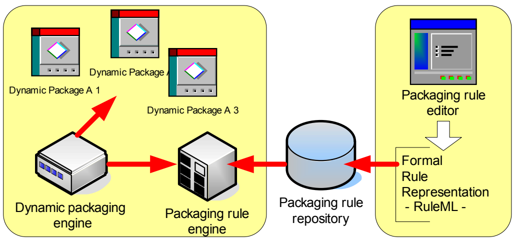


an integrated repository, providing a central point for definition and change, which can later drive dynamic package construction.

Packaging rules are logic statements that describe the policies and procedures to create dynamic packages for travel consumers. When traditional rule programming approaches are used, packaging rules are hard coded into the applications themselves, making rules difficult to develop and expensive to modify. By contrast, using a rule repository, packaging rules are stored externally and are separated from the dynamic packaging application, making the creation and modification of rules easier.

The packaging rules engine and repository architecture provide a structure for separating dynamic packaging logic from dynamic packaging applications. This separation is one of the main advantages of using a rule engine to implement packaging rules since it allows changes to be made to the created packages to reflect new business

Figure 12. Dynamic packaging engine architecture



policies. Packaging rules can be used to define new travel products and services, offer new promotions, or define high and low travel seasons.

Packaging Rules

A packaging rule is a statement that influences which tourism products will be part of a dynamic package. Dynamic packages are constructed in real time based on a set of constraints that are specified using packaging rules. For example, a travel agency may define that a dynamic package, which includes a trip and 5-day stay in New York, should cost less $3,000. In this example, three packaging rules define dynamic packages to create: Rule1 (duration, 5 days), Rule2 (local, New York), and Rule3 (less than, $3,000).

Since these rules are business oriented, they are defined and managed by business people. This makes the packaging rule approach attractive to dynamic packaging applications since users become an integral part of any package construction. The use and management of explicit packaging rules has several benefits for dynamic packaging applications:

- shorter time needed for changing packaging rules and making this change effective in dynamic package construction decreases;
- increased profit on travel products by a faster reaction to changing market demands and taking into account current market conditions; and
- improved customer satisfaction due to a better customization of travel products and services according to customer preferences.

In the travel industry, most organizations do not formally identify or store rules. Instead, although travel managers and travel agents use rules periodically, they exist only in the software code that runs packaging applications. Rules are "lost" in application code. As a result, the people that are directly in contact with the rules that dictate what sort of travel packages should be created, at a given time of year under specific market conditions, are not travel managers or travel agents but rather the information systems (IS) staff who convert packaging requirements into lines of code. Moreover, when rules are embedded in application code it becomes difficult to locate and change business logic and each alteration requires recompiling application code. Separating packaging rules from application code allows packaging policies to be easily communicated and understood by all employees, and rules can be managed in isolation from application code.

Formal Specification of Packaging Rules

Packaging rules can be expressed using formal languages. Examples of languages include UML[4], the ILOG rules language,[5] the Business Rules Markup Language (BRML[6]), and RuleML[7]. RuleML is an XML markup language for rules based on declarative logic and allows rules to be expressed as modular components using standard XML tags. Facilities are offered to specify different types of rules such as derivation, transformation, and reaction rules. One very attractive capability is the ability to specify queries and inferences on ontologies and mappings between ontologies. This last feature was the main reason why we have selected RuleML to model packaging rules. Since our shared global data model is expressed using an ontology, it makes sense to use a rule modeling language that can express packaging rules using the concepts present in our e-tourism ontologies. For example, the sentence, "renting an AVIS car, class **B**, costs thirty euros per day," is a packaging rule which is modeled using RuleML in the following way,

```
<Atom>
  <Rel>renting</Rel>
  <Ind>AVIS</Ind>
  <Ind>car</Ind>
  <Ind>class B</Ind>
  <Ind>per day</Ind>
  <Ind>30 euros</Ind>
</Atom>
```

A more complex example would be to model the sentence, "a customer is premium if he has spent a minimum of 5,000 euros on a travel package."

```
<Implies>
  <head>
    <Atom>
      <Rel>premium</Rel>
      <Var>customer</Var>
    </Atom>
  </head>
  <body>
    <Atom>
      <Rel>spending</Rel>
      <Var>customer</Var>
      <Ind>minimum of 5000 euro</Ind>
    </Atom>
```

```
  </body>
</Implies>
```

Packaging rules expressed in RuleML and semantics go hand in hand since they are independent of the inference engine used to implement an application. This allows exchange of rules between different engines. Before executing packaging rules, the rules are translated to an inference engine language, such as Java Expert System Shell (Jess)[8], LISP, or Prolog.

Types of Packaging Rules

We can identify several categories of rules for packaging applications. Namely, fact rules, computation rules, inferred knowledge rules, action enabling rules, and constraint rules (Von Halle, 2001). The following examples describe the several categories of rules used in dynamic package applications.

Fact rules. A simple fact rule is "renting an AVIS car, class **B**, costs thirty euros per day." This example of a rule has already been given previously with the illustration of the corresponding modeling in RuleML.

Computation rules. Calculate the price of a dynamic package for a customer by subtracting any discounts (hotel, car, and flight discounts) from the base fee:

PackagePrice =
BasePrice – HotelDiscount – CarDiscount – FlightDiscount

Inferred knowledge rules. The following rules state than if a customer buys a travel package which costs more than 5,000 euros, then he/she will have a discount of 15% on the price of the package minus 3,000 euros.

If PackagePrice > 5000
THEN Discount = 0.15*(PackagePrice–3000)
ELSE Discount = 0.

Action enabling rules. These rules, also called triggers, force dynamic packaging applications to take some predefined actions on the occurrence of an event. For example, a customer should be notified if his/her credit card is rejected.

IF rejection of credit card payment
THEN notify customer

Constraint rules. An example of this type of rule designates that the final price of a dynamic package should always be inferior to the sum of its individual products.

(PackagePrice – Discount) <= sum(Individual travel products)

Dynamic Packaging Engine

The dynamic packaging engine is responsible for reading the rules specifications and generating valid packages, that is, travel packages that comply with the packaging rules. The engine relies on linear programming (LP) to generate packages dynamically. LP is a subset of mathematical programming that allows packaging rules to be represented as linear equations. The LP module makes constrained decisions to maximize (or minimize) a linear objective function associated with package requirements. An example of the decisions the LP module could make is how many days a traveler can spend in a five star hotel in Lisbon, Portugal, near the ocean, with a rented car, for 1,500 euros.

Once packaging rules are represented using linear equations, as either the objective or constraint, an LP can be used to solve very large problems generating an optimum dynamic package. This means that given an objective function used in the LP, one can be certain that the best possible dynamic package will be constructed. The speed and quality of the dynamic package configurations produced by the LP module allows planners to explore different scenarios with different packaging prices, number of days a customer wishes to stay at a location, or levels of comfort provided by a hotel.

Future Trends

Travel agents are faced with changes in the tourism industry that have led to reduced commission revenues. For example, in 1997, the major United States airlines reduced the commission rate payable to traditional travel agencies and online travel agencies from 10% to 8%, and from 8% to 5%, respectively. In addition, as of 1998, many airlines have implemented a zero commission (Joystar, 2005). Additionally, vacation providers are expected to follow the airlines and eventually apply zero commissions (Forrester, 2005). As a result, travel agents have to look for new ways to increase their profit margins. One way is to acquire tools to offer their own services to dynamically package their client's holiday requirements. This added value will allow travel agents to earn their margins through a combination of reduced commission and booking fees. Therefore, dynamic packaging is critical to today's travel industry;

airlines, hotels, tour operators, and travel agencies need to create custom packages for consumers. The development and implementation of modern dynamic packaging application is therefore a major concern for the travel industry.

The Semantic Web promises to provide applications for Internet users through the use of metadata (e.g., RDF and OWL) attached to various information resources on the Web. In the future, these new technologies associated with the Semantic Web will be the foundation of "killer apps" providing a higher level of service to overcome the serious limitations of current Web technology in finding, integrating, understanding, interpreting, and processing information. This Semantic Web is based on machine processable semantics of data, enabling information processing via a computer improving the mechanization for many information processing tasks. Ontologies are necessary to link formal semantics with real world semantics and applications are needed to demonstrate how the Semantic Web can become a reality. Due to the requirements of dynamic packaging applications (e.g., interoperability, integration, knowledge inferring, and rule management), this type of application represents a good subject to develop a new breed of systems based on the Semantic Web.

Conclusion

With the growth of the demand for customized tourism itineraries, (online) agencies seek technology that provides their personnel and clients with the flexibility to put together unique dynamic packages from a range of alternatives, without having to be aware of the intricacy of contract rules and pricing issues. The concept of dynamic packaging is to bundle all the components selected by a traveler to produce one reservation. Despite where the inventory originates, the package that is created is treated as one operation and entails only one payment from the customer.

Even though the idea of dynamic packing has to some extent already been implemented by major online travel agencies (for example, Expedia, Travelocity, and Orbitz), we believe that current dynamic packing applications need to be enhanced with emerging technologies to facilitate the interoperability and integration of tourism data sources to speed up the time to market response. Previous studies have shown that the strategic potential of dynamic packaging technologies is currently limited due to interoperability and integration problems of existing travel information systems. In this chapter, we have described a systematic approach to deal with the lack of travel standards and to enable the data integration of travel data sources using the latest development on the Semantic Web: semantics and ontologies. We conclude that for the travel industry the Semantic Web can considerably improve dynamic packing applications.

Another limitation of current dynamic packing applications is the lack of solution to adequately manage the rules that govern the dynamic creation of travel packages. Traditional approaches hard code rules into dynamic packing applications. This solution does not permit the separation of packaging rules from application code. As a result, rules cannot be easily changed, managed, shared, or reused to reflect new business policies since they are hidden in the code. With the use of packaging rules, business users (i.e., nonprogrammers) can add and modify rules in an implementation-independent, business rule language. The use of packaging rules defined using semantic standards, such a RuleML, allows rules to be executed by different rule engines and therefore shared across the travel industry.

References

Arpírez, J. C., Corcho, O., Fernández-López, M., & Gómez-Pérez, A. (2003). WebODE in a nutshell. *AI Magazine, 24,* 37-47.

Barnett, M., & Standing, C. (2001). Repositioning travel agencies on the Internet. *Journal of Vacation Marketing, 7*(2), 143-152.

Bass, L., Clements, P., & Kazman, R. (1998). *Software architecture in practice.* Addison Wesley.

Bechhofer, S., Horrocks, I., Goble, C., & Stevens, R. (2001). *OilEd: A reason-able ontology editor for the Semantic Web.* Paper presented at the Joint German/Austrian conference on Artificial Intelligence (KI2001), Vienna.

Business Process Execution Language for Web Services (BPEL4WS). (2003). *Specification: Business process execution language for Web services version 1.1.* IBM. Retrieved February from http://www-106.ibm.com/developerworks/webservices/

Brachman, R. J., McGuinness, D. L., Patel-Schneider, P. F., Resnick, L. A., & Borgida, A. (1991). Living with classic: When and how to use a KL-ONE-like language. In J. Sowa (Ed.), *Principles of semantic networks: Explorations in the representation of knowledge* (pp. 401-456). Morgan Kaufmann.

Buhalis, D. (2002). *E-tourism: Information technology for strategic tourism management.* Longman.

CBS NEWS. (2003, July 28). *Online travel sites' new take off.* Retrieved from http://www.cbsnews.com/stories/2003/07/28/tech/main565494.shtml

Chinnici, R., Gudgin, M., Moreau, J., & Weerawarana, S. (Eds.). (2003, January 24). *Web services description language (WSDL) Version 1.2, W3C Working Draft 24*. Retrieved from http://www.w3.org/TR/2003/WD-wsdl12-20030124/

Christensen, E., Curbera, F., Meredith, G., & Weerawarana, S. (Eds.). (2001, March 15). *W3C Web services description language (WSDL)*. Retrieved from http://www.w3c.org/TR/wsdl

Compaq Web Language (CWL). (2005). Retrieved 2005 from http://research.compaq.com/SRC/WebL/

Darpa Agent Markup Language (DAML). (2005).

DERI International. (2005). *E-tourism working group*. . Retrieved 2005 from http://www.deri.org/research/groups/

Dombey, A. (1998). *Separating the emotion from the fact—The effects of new intermediaries on electronic travel distribution.* Paper presented at the ENTER Information and Communications Technologies in Tourism Conference, Istanbul, Turkey.

European Travel Agent's and Tour Operators' Association (ECTAA). (2004). Retrieved from http://www.ectaa.org/ECTAA%20English/Areas_dealt_with/Air_transport.htm#crs

Fernández López, M. (1999). *Overview of methodologies for building ontologies.* Paper presented at the Proceedings of the IJCAI-99 workshop on Ontologies and Problem-Solving Methods (KRR5), Stockholm.

Forrester. (2005). Forrester Research. Retrieved from http://www.forrester.com/

Gorton, I., Almquist, J., Dorow, K., Gong, P., & Thurman, D. (2005). *An architecture for dynamic data source integration.* Paper presented at the 38th Hawaii International Conference on System Sciences (HICSS-38).

Hospitality. (2005). *Trends in the online intermediary market: Travel agency/intermediary market*. *Hospitality eBusiness strategies.* Retrieved from http://www.hospitalityebusiness.com/hr081903142951.html

HotelOnline. (2002, October). *Global distribution systems in present times*. Retrieved from http://www.hotel-online.com/News/PR2002_4th/Oct02_GDS.html)

Inkpen, G. (1998). *Information technology for travel and tourism* (2nd ed.). Essex, England: Addison Wesley Longman Ltd.

Jones, D. M., Bench-Capon, T. J. M., & Visser, P. R. S. (1998). *Methodologies for ontology development.* Paper presented at the Proceedings IT&KNOWS Conference of the 15th IFIP World Computer Congress, Budapest.

Joystar. (2005). *JYSR annual report*. Joystar Inc. Retrieved from the World Wide Web.

Lake, D. (2001). *American go online for travel information.*

Lawrence, R., & Barker, K. (2001). Integrating data sources using a standardized global dictionary. In W. Abramowicz & J. M. Zurada (Eds.), *Knowledge discovery for business information systems* (pp. 153-172). Kluwer Academic Publishers.

Lenzerini, M. (2002). *Data integration: A theoretical perspective.* Paper presented at the Twenty-First ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS 2002), Madison, Wisconsin.

Mullaney, T. J. (2004). Design your own discount getaway. *Business Week.* Retrieved from http://www.adinfo.businessweek.com/magazine/content/04_10/b3873101_mz070.htm

National Information Standards Organization (NISO). (2005). *Guidelines for the construction, format, and management of monolingual thesauri*. Retrieved from http://www.niso.org/standards/resources/z39-19a.pdf

O'Connor, P. (2003). Online pricing—An analysis of hotel company practices. *Cornell Hotel & Restaurant Administration Quarterly, 44*(1), 10-19.

OntoEdit. (n.d.). (Version 0.6) [Computer Software]. AIFB, University of Karlsruhe. Retrieved 2005, from http://www.ontoknowledge.org/tools/ontoedit.shtml

Open Travel Alliance. (2004). Retrieved from www.opentravel.org

PRNewswire. (2002, April 9). *Travelocity.com ranks as top travel planning site for business travelers*. Retrieved from http://www.corporate-ir.net/ireye/ir_site.zhtml?ticker=TVLY&script=413&layout=-6&item_id=277266

Protégé. (n.d.). (Version 3.1.1) [Computer software]. Stanford Medical Informatics. Retrieved 2005, from http://protege.stanford.edu/

Rayport, J. F., & Jaworski, B. J. (2001). *E-commerce*. Boston: McGraw-Hill.

Riebeek, H. (2003). The ticket chase. *IEEE Spectrum.* Retrieved from http://www.spectrum.ieee.org/WEBONLY/publicfeature/jan03/tair.html

Robbins, R. J. (1996). Bioinformatics: Essential infrastructure for global biology. *Journal of Computational Biology, 3*(3), 465-478.

Universal Description Discovery, and Integration (UDDI). (2002). Retrieved from http://www.uddi.org/

Uschold, M., & Gruninger, M. (1996). Ontologies: Principles, methods and applications. *Knowledge Engineering Review, 11*(2).

Von Halle, B. (2001, April). Building a business rules system, part 4. *DM Review Magazine.* Retrieved from http://www.dmreview.com/article_sub.cfm?articleId=3201

Werthner, H., & Ricci, F. (2004). E-commerce and tourism. *Communications of the ACM, 47,* 101-105.

Wiederhold, G. (1994). *Interoperation, mediation and ontologies.* Paper presented at the International Symposium on the Fifth Generation Computer Systems, Workshop on Heterogeneous Cooperative Knowledge-Bases, Tokyo.

World Wide Web Consortium. (2005a). *Resource description framework*. Retrieved from http://www.w3.org/RDF/

World Wide Web Consortium (W3C). (2005b). *XML schema*. Retrieved from http://www.w3.org/XML/Schema

World Wide Web Consortium (W3C). (2002). *Simple object access protocol 1.1*. Retrieved from http://www.w3.org/TR/SOAP/

World Wide Web Consortium (W3C). (2004). *Web ontology language (OWL)*. Retrieved 2004, from http://www.w3.org/2004/OWL/

XML. (2005). *Extensible Markup Language (XML) 1.0 (Third Edition), W3C Recommendation 04 February 2004*. Retrieved from http://www.w3.org/TR/REC-xml/

Endnotes

1 http://www.tiscover.com/

2 http://www.gulliver.ie/

3 http://www.opentravel.org/

4 http://www.uml.org/

5 http://www.ilog.com/products/jrules/lifecycle/language.cfm

6 IBM Business Rules for E-Commerce: http://www.research.ibm.com/rules/home.html

7 The Rule Markup Initiative: http://www.ruleml.org/

8 http://herzberg.ca.sandia.gov/jess/