eTourPlan: A Knowledge-Based Tourist Route and Activity Planner

by

Tshering Dema

Bachelor of Computer Science,
Kanglung College, University of Delhi, 2004

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Computer Science

In the Graduate Academic Unit of Computer Science

Supervisor(s):	Harold Boley, Ph.D., Computer Science Przemyslaw Rafal Pochec, Ph.D., Computer Science
Examining Board:	Gerhard Dueck, Ph.D, Computer Science, Chair Bruce Spencer, Ph.D, Computer Science Yevgen Biletskiy, Ph. D, Electrical and Computer Engineering

This thesis is accepted

Dean of Graduate Studies

THE UNIVERSITY OF NEW BRUNSWICK

September 2008

Dedication

This thesis is dedicated to my loving parents.

Abstract

Tourism is the world's largest and fastest growing industry. There are conventional tourism service providers which are competitively trying to provide the best travel services to customers based on their interests. The Semantic Web is a major endeavour to enhance the Web by enriching its content with semantic (meta)data that can be processed by inference-enabled Web applications. eTourism is a good candidate for such enrichment, since it is an information-based business. As with any such business, providing the relevant information for the consumer means a better end product. Thus, providing a well-structured and comprehensive Knowledge Base (KB) for consulting will bolster the eTourism business. In this thesis, we have designed and implemented a KB consisting of tourism domain-specific information. Our KB stores facts about Bhutan, which are structured by a light-weight ontology (adapted from the Harmonise eTourism ontology) and used by partonomy rules that encode the geographical partitioning of regions and provide a basis for activity search capabilities. On top of these, planning rules are applied to deduce recommendations of routes, activities (attractions and events), and accommodations. This thesis also discusses transferring Friend Of A Friend (FOAF) concepts for semantically describing persons or organizations, to tourist-entity profiles. The FOAF-like Harmonise relation "relatedTo" between tourist entities is used to chain through provinces and attraction profiles, hence to provide attraction-centric recommendations. This prototype, eTourPlan, an eTourism planner using Semantic Web techniques has been implemented in RuleML/POSL. Results of running eTourPlan in the prototype RuleML engine OO jDREW are reported.

Acknowledgements

My graduate study experience in UNB has been fruitful with the motivation and guidance from many people. Firstly, I would like to extend my heartiest gratitude to my supervisors, Dr. Harold Boley and Dr. Przemyslaw Rafal Pochec, for their invaluable insights and continuous support throughout the graduate program.

My special thanks go to the OO jDREW Open Source Project, mainly Ben Craig, for his continued technical support. I would like to take this opportunity to thank Dr. Bruce Spencer, Dr. Gerhard Dueck, and Dr. Yevgen Biletskiy, for taking time from their busy schedules to read my thesis.

I would like to thank Canadian International Development Agency (CIDA), for the funding and the scholarship offered through the Strengthening Support for Education in Bhutan (SSEB) Project and the Royal Government of Bhutan for selecting me as a candidate. My immense gratitude also go to the Bhutan Project Staff for their kindness and support, which made my whole stay in Canada a wonderful one. Lastly, I would like to thank all my friends who have helped me a lot and my parents for their love and prayers.

Table of Contents

List of Tables

List of Figures

Acronyms

BU	-	Bottom Up
CBR	-	Case-Based Reasoning
DAML	-	DARPA Agent Markup Language
FOAF	-	the Friend Of A Friend project
FOL	-	First Order Logic
HTML	-	Hypertext Markup Language
IFITT	-	International Federation for IT, Travel and Tourism
IMHO	-	Interoperable Minimum Harmonise Ontology
jDREW	-	java Deductive Reasoning Engine for the Web
KB	-	Knowledge Base
MathML	-	Mathematical Markup Language
Naf	-	Negation as failure
OO jDREW	-	Object Oriented java Deductive Reasoning Engine for the Web
OO RuleML	-	Object-Oriented Rule Markup Language
OTA	-	Open Travel Alliance
POSL	-	Positional-Slotted Language
Prolog	-	PROgramming in LOGic
RDF	-	Resource Description Framework
RDFS	-	Resource Description Framework Schemas
SWRL	-	Semantic Web Rule Language
TD	-	Top Down
TTI	-	Travel Technology Initiative
WRL	-	Web Rule Language
WTO	-	the World Tourism Organisation
W3C	-	World Wide Web Consortium
XML	-	Extensible Markup Language
XSLT	-	Extensible Stylesheet Language Transformations

Chapter 1

Introduction

Tourism is the world's largest and fastest growing industry [56]. The World Tourism Organization predicts that one billion international tourists will travel by the year 2010 [53]. Tourism has become a highly competitive business for tourism destinations all over the world. There are many conventional tourism service providers which are competitively trying to provide the best travel plans and recommendations to their customers based on each customer's interests. eTourPlan is a knowledge-based route and activity planner for eTourism. To accommodate a tourist's requests based on his or her preferences, it offers various options for retrieving accurate information about tourist destinations. The interoperability and integration of the available information on the Web is enhanced by using Semantic Web techniques [47][12].

In Section 1.1, we give an overview of eTourism and how Semantic Web technology can be used for improving information retrieval and interpretation for users by semantically connecting the various tourism subdomains. Next, we present the thesis objectives, followed by the thesis organization in Section 1.2.

1.1 Overview

Every trip starts with a plan. Most of the prevalent travel recommenders [52][6][42] are location-centric and therefore do not function as complete trip planners. This is a problem since, e.g., the time necessary to visit a number of places at a destination can be more than a traveller's available time. Automated planners should consider the preferences and requirements of tourists for activities, accommodations, and route planning.

Currently, tourist consultants and travellers must visit multiple independent websites for various information such as accommodation and activity facilities to plan a trip tailored to given preferences. Outside the realm of travel packages in mass tourism, it is difficult and time-consuming to find the right products or services for more 'holistic' travel planning. This lack of standards in the tourism domain brings up the necessity of integration of heterogenous information sources. (Individualized) eTourism is a good application area for Semantic Web technologies, since meaningful information gathering, integration, distribution, and exchange are the backbone of the travel industry. In this thesis, we aim to bring semantics and structure to bear on tourist information on the Web. The two main characteristics of the tourism domain that make it suitable for Semantic Web technologies are the heterogeneity of the market and the distributed nature of the high volume of information on the Web [7]. A knowledge-based system should start with a tourist's preference specification, search for implicit facts, ignore irrelevant information, combine several Web resources of the tourism subdomains, and generate a coherent travel plan.

An ontology is a formal conceptualization of a particular domain that is shared by a group of people [38]. The Harmonise ontology [48] is a standard ontology for eTourism. Ontologies are used in knowledge-based systems as conceptual frameworks for providing, accessing, and structuring information in a comprehensive manner [47]. Rules can then be applied on the ontology-structured facts to derive more knowledge

from the KB. The implementation of an integrated system of search, recommendation and planning would be tedious to implement if it was solely ontology-based.

The main focus of this thesis, the eTourPlan prototype, is a travel planner which, according to user preferences and constraints, aids in the selection and scheduling of various aspects of a tour such as events, attractions, and routes. The planner can also function independently as a location-centric recommender of the aforementioned tourist entities. In addition to these two primary operations, it can function as a search engine for relevant tourist information.

Another focus of this thesis is the use of the FOAF [21] concept in the tourism domain. The FOAF vocabulary has been used mainly to create semantic profiles of persons and organizations. One of the applications of FOAF vocabularies is Expert Finding [18]. In this thesis, we extend FOAF profiles for persons or organisations to FOAF-like profiles for tourist entities, covering part of the country of Bhutan as a case study. A similar application of FOAF to eTourism has been explored for the Tyrol region [7]. We created FOAF-like profiles for selected provinces of Bhutan and their touristic information about attractions and events.

Our eTourPlan prototype explores the reasoning potentials of rules on structured facts stored in a KB. The key functionalities of the eTourPlan prototype tested on the Bhutan KB are discussed later in this thesis. In this work, we have represented the profiles of tourist entities in RuleML/POSL syntax. The fact base could also be stored in a database. The task of exploring reasoning potentials of rules on a complete KB of a multi-dimensioned domain (i.e. the tourism domain) is pursued in this thesis. Our executable specification of the knowledge-based tourist route and activity planner eTourPlan can later be integrated with a (relational) database or translated to a self-contained database application.

1.2 Thesis Objectives and Methodology

The objectives of this thesis are to design, implement, and evaluate an eTourism prototype for Bhutan:

- To design a light-weight ontology using the Resource Description Framework Schema (RDFS) to capture all the touristic subdomains [aligned with the Harmonise standard].
- To build a Bhutan fact base, structured by this ontology, using the Object-Oriented Rule Markup Language (OO RuleML) in its presentation syntax of the Positional-Slotted Language (POSL). Tourist entities are to be described using FOAF-like profile facts.
- To implement rule subsystems needed for generating a travel plan containing tour recommendations:
  - Partonomy rules for the subdivision of regions
  - Derivation rules to deduce transitive closure facts about distances etc
  - Inference rules for various planning and recommendation modes
  - Query rules to perform semantic searches
- To evaluate the KB by systematically testing each of the rule subsystems with varying user preferences.
- To evaluate the overall operation of the eTourPlan prototype as run in the OO jDREW reasoning engine by generating results for varied travel scenarios.

1.3 Thesis Organization

The organization of the thesis is as follows. A background study of travel planner and recommender systems for tourism, along with brief introductions to other basic

concepts, is presented in Chapter 2. In Chapter 3, we describe the rule languages and rule engines used for our work. Then, in Chapter 4, we present the design of an ontology using RDFS, aligned with the Harmonise ontology, and the descriptions of province-centric FOAF-like profiles of tourist entities for the Bhutan tourist information in OO RuleML/POSL syntax. In Chapter 5, we discuss the implementation details of the rule subsystems for each of the tourism subdomains defined in our fact base. We start with partonomy rules for the administrative structuring of a country such as Bhutan in Section 5.1.1, followed by distance and route computation in Section 5.1.2. The rule system for route planning is discussed in Section 5.2.1, followed by discussions on the search rule system and the rule system for location-centric travel recommendation in Sections 5.2.2 and 5.2.3. In Section 5.2.4, the main eTourPlan rule system is discussed. Chapter 6 shows the application of the various rule subsystems for the eTourPlan prototype on the Bhutan tourism KB. It also demonstrates the experimental results of the key operations. Finally, in Chapter 7, we conclude the thesis with a summary of our contributions and a discussion of possible future work.

Chapter 2

Background

This chapter describes the general concepts of travel planning and Semantic Web techniques. In Section 2.1, we introduce tour planning and recommendation methodologies. This is followed, in Section 2.2, by a description of the applicability of the Semantic Web in the tourism domain. The FOAF vocabulary is discussed in Section 2.3, and the Harmonise ontology is presented in Section 2.4. Finally, Section 2.5 concludes this chapter with a description of a semantic eTourism prototype.

2.1 Tour Planner and Recommender Systems

Travel planning is a complex and dynamic process because there are multiple factors that influence the destination choice. Destination choice is determined by the availability of travel facilities and by the user's preferences such as length of travel, mode of transportation, accommodation type, and activity theme. Therefore, it is necessary to combine planning and recommendation in order to give users a complete vacation package with maximum options.

We have developed travel planning strategies in the domain of touristic travel. Planning a travel for a tourist can be done in any one of the cells of the 3*3 matrix

shown in Table 2.1. In this thesis, we have implemented rule systems for all three columns (aspects) of the complete planning row (strategy), described in detail in Section 5.3 of Chapter 5. The partial and sequence planning strategies are only principally covered in route planning in Section 5.2.1 and are not entirely implemented in this thesis.

Table 2.1: Tourist travel planning matrix

	Attraction-Only Planning	Event-Only Planning	Event-Centric Planning
Complete Planning	Planning attractions based on related locations	Planning events based on their dates and locations	Planning events with additional attraction recommendation
Sequence Planning	System orders the user-specified attractions	System orders the user-specified events	System orders both events and attractions
Partial Planning	System orders and adds to user-specified attractions	System orders and adds to user's specified events	System orders and adds to user's specified events and attractions

The second and third aspects of complete planning can be integrated, and the system allows users to check the option of keeping it as "event-only" or "event-centric" planning. The traveller wants to attend events while touring the country. This planning includes optional attraction recommendations along the routes between event locations.

The literature shows that recommendation is a common service in the tourism subdomains of travel and accommodation. Recommender systems are now used on increasing numbers of e-Commerce sites. They make the tourism services more attractive for users. The two most successful recommender system technologies are Triplehop's TripMatcher, used by www.ski-europe.com and VacationCoach's expert advice platform, Me-Print, used by Travelocity.com [52]. However, neither of the two supports users in planning a user-defined trip package that includes a scheduled list of preferred locations along with accommodation and sightseeing recommendations according to their preferences.

Manuela and Tobias [10] discussed the four basic filtering approaches for building a recommender system:

- Collaborative Filtering: Recommendation based on the behaviour of users from their profiles including their preferences. Also referred to as social filtering, where recommendations are made based on other users with similar preferences.

- Content-based Filtering: Filtering that is based either on information retrieval or attribute-based filtering systems. This approach focuses on the semantics and structure of the content and might lack interaction with the user.

- Knowledge-based Filtering: Filtering that encompass the explicit structured representation of knowledge using ontologies and rules. With its concise knowledge representation, deriving implicit facts from ontology-structured facts using rules is a good way of recommendation. It can make the recommendation as wide-ranging as its knowledge base [22][23].

- Hybrid Systems: Hybrid systems can combine any of the above methods. NRC-IIT's RACOFI technology combines rules and collaborative filtering[5]

Our literature research on designing recommender systems [10][23], which focused on comparisons of the various approaches, suggests that the collaborative and content-based approaches are not perfectly suited for decision making in tourism. For instance, a user who has travelled for the purpose of hiking might also like to go to a restaurant or theatre after a tiring day. So, making recommendations only with a content-based or collaborative recommender system in the tourism and travel domain is not flexible and intelligent enough to incorporate the various preferences of users. On the other hand, knowledge-based recommenders are not restricted to extracting preferences from the interaction history or ratings but can respond to the user's stated need. Researchers have pointed out the importance of recommender systems to support multiple decision styles [52].

Most of the existing recommender systems are location-centric recommenders, which are concerned with users' activity schedules once users get to a destination rather than planning entire trips within timeframes according to users' preferences and constraints.

The DieToRecs [35] and Entre Restaurant Recommender [23] systems use the Case-based reasoning (CBR) approach in their planning. Case-based reasoning treats the object to be recommended as a case and employs CBR techniques to locate a similar case from the KB. From a survey and experiment described in [23], Burke concluded that collaborative filtering does not improve the performance over the knowledge-based component acting alone.

The literature study supports the need of providing more planning options to our users along with recommendations in the tourism domain. This would, along with time, distance, and event schedules, employ users' preferences in order to provide the best possible plan to fulfill end users' needs. The aim of our eTourPlan prototype is to function as an integrated system of search engine, recommender, and planner.

2.2 The Semantic Web

Search engines support human users with information available on the Web. The World Wide Web is a vast and growing source of information and services. Currently, the Web, which is read and understood by humans, rather than machines, is mostly designed in Hypertext Markup Language (HTML). This "Visual Web" does not allow computers to automate information processing, integration, and interoperability [25]. The Semantic Web aims at making information understandable by machines, so that they can perform tedious work involving knowledge representation and data integration automatically. Such a representation would be desirable for structuring the vast information about tourist destinations.

Hendler, Lassila, and Berners-Lee, the inventor of WWW, URIs, HTTP, and HTML, define the Semantic Web (as quoted in [43]) as,

> "an extension of the current Web that enables knowledge representation by using extensible markup language (XML), RDF (Resource Description Framework), and ontologies including rules to make inferences."

The Semantic Web is an endeavour to advance the Web by enriching its content with semantic metadata that can be processed by inference-enabled Web applications. It provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries. It is a collaborative effort led by the W3C with participation from a large number of researchers and industrial partners. It focuses not only on display, but also on machine-understandable metadata. It is largely based on the Resource Description Framework, creating descriptions of information available on the World Wide Web. Ontologies and automated reasoning are key techniques in the Semantic Web initiative. Rules can be used to draw inferences, to express constraints, to specify policies, to react to events and changes, to transform data, etc. We introduce two important notions in the Semantic Web: metadata and the Resource Description Framework.

2.2.1 Metadata

Metadata is structured information that describes, explains, locates, or otherwise makes it easier to retrieve, use, or manage an information resource [49]. In simple words, metadata is data about data. It ensures that resources will survive and continue to be accessible into the future.

According to the W3C, metadata is machine-readable information for the Web[1]. Metadata can describe resources at any level of aggregation, and it enriches resource discovery, interoperability, archiving, and preservation.

[1] `www.w3.org/Metadata`

2.2.2 Resource Description Framework

The Resource Description Framework (RDF), a W3C Recommendation, is a framework for modelling information at a metadata level on the Web [51]: RDF is designed for widespread and decentralised use [20]. RDF is therefore the formal data model for machine-understandable metadata used to provide standard descriptions of Web resources for facilitating data and system integration and interoperability.

RDF/XML is its XML syntax, and is at the base of the Semantic Web, most other languages corresponding to higher layers are built on top of it. RDF uses a triple representation of metadata, consisting of subject, predicate, and object. The subject represents the resource, the predicate expresses a relationship between the subject and the object, while the object is the object (another resource or a literal) of this relationship. RDFS is a language for describing RDF vocabularies in RDF. It has mechanisms to describe RDF classes and properties, such as attributes of resources and relationships between them. RDFS provides a mechanism one of whose purposes is integrating multiple metadata schemas extracted from distributed information [49].

RDF can be used for resource discovery in search engines, for cataloguing and describing content and content relationships by intelligent software agents to facilitate knowledge sharing and exchange[2].

The Semantic Web data model is connected with the model of relational databases. An RDF triple can be directly mapped to a record in a table in a relational database. Semantic Web has the ability to express inferences on the vast amount of relational database information on the Web. Another advantage of the Semantic Web is that it allows to add information integrating to different databases on the Web and to perform advanced operations across them[3]. Using database technology would require to 'normalize' knowledge objects. In [29], Debenham describes how 'objects' are used to represent knowledge (i.e., complex structures and rules) and how it is stored as

[2] www.w3.org/TR/1999

[3] www.w3.org/DesignIssues/RDFnot.html

information (i.e., relations) after normalization. Furthermore in [30], he stated a single principle of normalization that applies to objects irrespective of whether they represent data, information, or knowledge items.

2.3 Friend Of A Friend

The FOAF project was founded by Dan Brickley and Libby Miller. FOAF emerged in 1998, as an RDF description of Dan Brickley in his homepage. FOAF is an open community-led initiative which is tackling the wider Semantic Web goal of creating a machine processable web of data[4]. FOAF enables Semantic Web methods to be applied for personal homepages, linking together FOAF profiles of different persons who publish data with well-defined semantics.

FOAF is an application of the RDF Web and it provides the basic vocabulary for the RDF Web by defining useful RDF properties. The original FOAF vocabulary is specified via the namespace URI 'http://xmlns.com/foaf/0.1/', and consists of two components: vocabulary about classes and vocabulary about properties. Vocabulary about classes is designed to express the type of an object (e.g., foaf:Person), while vocabulary about properties is used to express the type of a relationship or an attribute (e.g., foaf:knows and foaf:name, respectively). A similar approach can be used for linking together a Web of FOAF-like profiles for tourist entities [7].

2.4 The Harmonise Ontology

Since it is important to have a common view of the tourism business domain that can be used as a reference point for application interoperability, we must at least have a common understanding of the relevant concepts and relationships, which can be represented by sharing a standard ontology.

[4] `www.xml.com/pub/a/2004/02/04/foaf.html`

Harmonise[5] is a network of cooperating actors working together in the tourism domain to achieve information interoperability as well as to provide tools for the tourism marketplace. The Harmonise ontology is defined as an overall solution for information exchange in travel and tourism [37]. Some key players in the Tourism Harmonisation Network are the Open Travel Alliance (OTA), the World Tourism Organisation (WTO), the Travel Technology Initiative (TTI), and the International Federation for IT, Travel and Tourism (IFITT) [42].

The Interoperable Minimum Harmonise Ontology (IMHO) defines the building of an ontology as a social phenomenon to represent a common, sharable view of the application domain. IMHO has a set of concepts of the travel and tourism domain, which are used within different data formats, and in this way enables mapping between those formats. The Harmonise ontology is currently comprised of classifications of data items for events, attractions, accommodations, and restaurants. Harmonise considers events and attractions as two primary entities because they are highly relevant to the tourism domain. Second to the above two entities is accommodations as they are one of the main business domains for tourism on the net. Harmonise has undergone a market validation by 12 pilot organizations based across Europe through a project called Harmo-TEN [2004-2005] and the final phase of implementation called Harmo-NET has started in 2006 and is still ongoing [37].

The Harmonise service allows travel and tourism organisations using different message standards or data formats to exchange data in a cost-effective way by offering fully supported open software with assured reliability.

2.5 Semantic eTourism Prototype

Our eTourism prototype uses a locally stored Knowledge Base (KB) to generate a travel plan. The KB consists of object-centric facts, which are structured by ontologies.

[5] `www.harmonet.org`

Ontologies provide a good basis for reasoning and classifying the various information in the tourism domain. They provide uniform definitions and therefore increase comprehension and knowledge sharing, remove semantic ambiguity, and are fundamental to automated knowledge extraction on the Web [38]. Based on a machine-readable representation of information in the form of ontologies, facts, and rules, eTourPlan is a knowledge-based prototype in which semantic rules are implemented on the ontology-structured fact base to deduce intelligent services such as travel planning and recommendations, and precise search facilities. We can proceed from the conventional method of syntactic search (i.e., keyword search) of Web resources to semantic search of knowledge by using Semantic Web techniques such as ontologies and rules. Having such a well-structured KB would also enable automated information extraction and processing in the Semantic Web [13].

Ontologies are mainly used where there are multiple subdomains needed to structure a domain. We consider the definition of five main tourism subdomains [41]:

- **Regions**: The concept of region is fundamental to geography, and is of particular value in gaining an understanding of the special nature of different places and areas. Having an organized manner of structuring regions helps us in locating tourist entities in the tourism domain.

- **Transportation**: For many destinations within regions, transportation plays a vital role in the development of a tourism infrastructure.

- **Accommodations**: Accommodation is a term used to encompass the provision of bedroom facilities on a commercial basis within the hospitality or tourism industry. Primarily, it is associated with the hotel, resort, guest house, and similar sectors.

- **Events**: Planned events are either one-time events, such as a special soccer match, or periodic events, such as yearly cultural celebrations. Art and enter-

tainment, business and trade, as well as sports and education are event categories of particular interest in the context of the tourism domain.

- **Attractions**: Tourist attractions are places of interest open to the public, offering recreation, education, or historical interest. (e.g., theme parks, historic houses, museums, art galleries, zoos, temples, and leisure complexes).

In our prototype, the vocabularies and concepts of accommodations, events, and attractions are the three tourism subdomains collected from the Harmonise ontology. We use the basic idea of FOAF profiles to create FOAF-like semantic profiles for all relevant tourist entities. We implemented rule subsystems for each of the subdomains, which are later integrated into the main travel planning rule system. A knowledge-based travel planner can infer specific plans and make location-centric recommendations for each of the destinations by applying decision rules to the KB. Our eTourPlan prototype can provide solutions to a wide range of queries for tourist information according to the users' need.

eTourPlan uses a light-weight ontology that enriches the FOAF-like semantic profiles of provinces, events, attractions, and accommodations. We have based our KB on Bhutan tourism information. The administrative subdivision of a country is governed by partonomic rules, which are mainly used to either locate (and get the full address of) any specific tourist entity. All province-to-province routes and distance times, measured in hours, are precomputed and stored in our KB along with other facts. This precomputation frees the planner from computing these route at run time. We show the various operations of our eTourPlan prototype on the Bhutan KB in Chapters 5 and 6. The key operations selected to be implemented in our prototype are as follows:

1. Parametric search of tourist information: Users might want to get detailed information about any particular province, event, attraction, accommodation, or route. eTourPlan allows information search that fits a number of simultaneous

criteria (the parameters to searches). It allows searches by various options such as name, type, theme, or location. Providing key information such as event dates and their URLs along with other, finer details, helps our users to choose the most suitable travel date. It is also time saving and more fun for our users to get the precise information on the first click instead of having to wander over many websites.

2. Recommend touristic route: This is for users who have very little idea about their preferences, but wish to make a good travel with the system's recommendation. The recommender builds a touristic route along the attractive provinces linked together in our KB, providing relevant details of attractions, events, and accommodation facilities in each of the provinces.

3. Recommend activities for user-preferred provinces: Assuming the user has fixed travel destinations, the system provides the recommendation of activities and route details for each of the provinces. This is called location-centric recommendation.

4. Plan an attraction or event-centric travel: Based on operations 1-3, eTourPlan can generate complete travel plans either for attraction-only or for event-centric with attraction recommendation. Event-centric planning is based on a temporal-geographic search criterion, and attraction-only planning is based on a purely geographic search of related attractions in our KB.

Chapter 3

Rule Languages and Tools

This chapter presents the rule languages and semantic tools that are relevant to this thesis. In Section 3.1, we give a brief introduction to rule languages, followed by rule engines in Section 3.2. Finally, we discuss the architecture of OO jDREW in Section 3.3.

3.1 Rule Languages

Besides its focus on Web ontologies, the Semantic Web has a focus on Web rules because they can express policies, regulations, etc for e-Commerce applications, which increased the need for rule languages over the last few years[1]. Some of the rule languages are Semantic Web Rule Language (SWRL), Web Rule Language (WRL), Rule Interchange Format (RIF), and Rule Markup Language (RuleML). RuleML (XML syntax) and its presentation syntax POSL [16] are the declarative languages that we have used in this thesis.

[1] `www.ruleml.org`

3.1.1 Rule Markup Language

The Rule Markup Initiative has defined a shared Rule Markup Language (RuleML), permitting both forward (bottom-up) and backward (top-down) rules in XML for deduction, rewriting, and further inferential-transformational tasks as well as XML-based production and reaction rules for (event-)condition-action tasks. Over the years, rules, as exemplified by RuleML, PRR [55], and RIF [19] have shown their impact on e-Commerce as well as the Semantic Web. Moreover, rule interchange is playing a significant role in Knowledge Representation (KR) [14], as exemplified by Common Logic[2].

RuleML is built using input from other standards work, i.e., Mathematical Markup Language (MathML), DARPA Agent Markup Language (DAML), and Extensible Stylesheet Language Transformations (XSLT) as well as co-evolving with the above-mentioned efforts that allows to publish and share rulebases on the World Wide Web. The goal of RuleML is put forward in a Coverpages technology report [1].

> "Our main goal is to provide a basis for an integrated rule-markup approach that will be beneficial to all involved and to the rule community at large ... This RuleML kernel language can serve as a specification for immediate rule interchange and can be gradually extended ... "

RuleML 0.91 is the current version sustaining a family of sublanguages, such as First Order Logic (FOL) RuleML, SCLP RuleML (Situated Courteous Logic Programs RuleML), Fuzzy RuleML, and Reaction RuleML. In this thesis, we focus only on (Object Oriented)RuleML, an extension of RuleML, now part of RuleML 0.91, that permits "slotted" (attribute-value) typed descriptions in RuleML.

[2] `http://cl.tamu.edu`

3.1.2 Positional-Slotted Language

Prolog, which stands for PROgramming in LOGic [26], is the most widely available language in the logic programming paradigm. It is based on the mathematical notions of relations and logical inference. Prolog is a logic programming language that uses the Horn subset of first order logic.

Prolog is a declarative language whose Horn clauses can be facts as well as rules used to describe problems rather than coding algorithms as in the imperative programming. Prolog allows the user to issue a query, and its engine searches and applies clauses in order to provide the user with an answer.

POSL is derived from Prolog. It is a human-readable (presentation) syntax for Semantic Web knowledge. It combines Prolog's positional and F-logic's (standing for Frame Logic) [44] slotted syntaxes for representing knowledge (facts and rules) in the Semantic Web. In [16], the author describes that POSL not only accommodates many kinds of assertional-logical and object-centered modeling styles, but also achieves conciseness and orthogonality at large. The compactness of the language makes it easier for humans to write and read in POSL than in XML syntax. Since it is interconvertible with RuleML, the advantage of XML, being more machine-readable, is thus preserved. The bidirectional translator for POSL to RuleML and vice-versa is available online[3].

3.2 Adaptation of the Rule Engine OO jDREW

In this section, we describe the rule engine OO jDREW, which serves as a tool for RuleML. Mandarax [2], SweetRules [3] and jDREW [54] have also been developed as rule engines supporting various subsets of RuleML.

[3] `www.jdrew.org/oojdrew`

3.2.1 Basic OO jDREW

OO jDREW is an object oriented extension of jDREW (java Deductive Reasoning Engine for the Web) [54]. It is a reasoning engine for executing RuleML rule markup and the Object Oriented extension for RuleML as well. OO jDREW is one of the engines used by the distributed Rule Responder architecture[4].

OO jDREW has two modes of operations: OO jDREW BU (Bottom Up) and OO jDREW TD (Top Down). Bottom-up execution (forward reasoning) is used to infer all derivable knowledge from a set of clauses. Such bottom-up reasoning uses as input facts at the bottom of a proof tree while all derivable ones are added as output. Top-down execution (backward reasoning), in contrast, is used to reduce queries from the top-level queries to more specific queries, etc until it may finally succeed with facts unifying with all subqueries.

OO jDREW BU: In OO jDREW Bottom Up, rules are used to derive new facts from given facts until a fix point is reached. Three features are designed in OO jDREW BU:

- Type Definition, realized in RDFS and POSL syntax, which comprises the subClassOf taxonomy (type hierarchy for the user's KB
- Assertional KB, permitting both RuleML and POSL syntax, stores the facts and rules used for Bottom up deduction
- Running the Forward Reasoner, either in RuleML or POSL syntax, provides users with all derived facts and after executing the forward reasoning.

OO jDREW TD: In OO jDREW Top Down, rules are used to answer queries by reducing them to subqueries until facts are reached. Three features are designed in OO jDREW TD:

- Type Definition, which is same as in OO jDREW BU

[4] `http://responder.ruleml.org`

- KB, which stores the fact and rules, as in OO jDREW BU, for query on demand
- Querying the KB, either in RuleML or POSL syntax

3.2.2 Various Architectures of OO jDREW

We have experimented with the various models of using the two OO jDREW reasoners (Top-down and Bottom-up). In this section, we will discuss our approaches and issues. We chose one architecture after several rounds of testing of a sample KB for the different architecture modes.

3.2.2.1 Integration of OO jDREW BU and TD

We attempted to integrate the two approaches such that our system would work with two levels of computation. We first execute the Bottom-up reasoner and generate derived facts from the original set of clauses. The new KB is then queried as the KB for Top-down execution of OO jDREW.

The need for an integrated archtecture comprising Top-down and Bottom-up arose for the shortest path distance computation. We wanted to get precomputed facts from a set of clauses using Bottom-up and then apply some optimization rules on the generated KB to derive the solutions to a user's query. From our experiments we discovered that same set of clauses cannot be always used for both Top-down and Bottom-up OO jDREW.

The current differences between OO jDREW Top-down and Bottom-up are:

- Naf (Negation as failure) of OO jDREW TD cannot be used in OO jDREW BU
- There are variable binding issues in BU with recursive predicates that work well in OO jDREW TD

With these differences, we could not run the KB unchanged in Bottom-up that works well in Top-down execution mode.

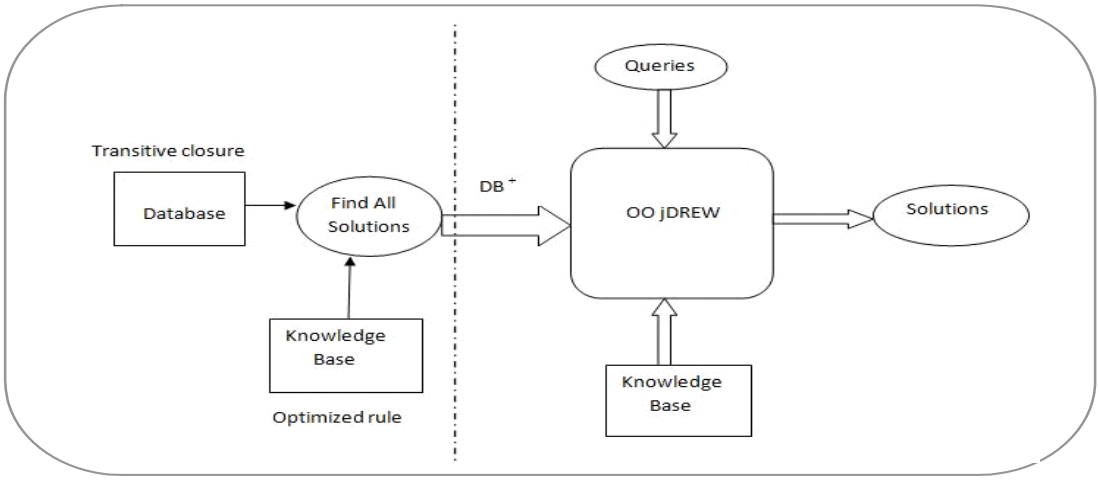

Figure 3.1: Top-Down FindAll Solutions

3.2.2.2 Top-Down FindAll Solutions

After testing a sample KB in both modes of execution, we decided to primarily use Top-down execution for almost all rulebases in our KB. To permit this, the backward reasoning engine has been internally enhanced with a primitive to find all solutions to a query collecting all solutions that would normally be delivered one at a time. The Top-Down FindAll Solutions primitive allows the Top-down engine to first find all (of finitely many) solutions to a selected query. The answer is then returned as a fact base of the primitive parameterized by the query. The architecture of Top-Down FindAll Solutions in OO jDREW is as shown in Figure 3.1. We use this architecture to solve the problem of optimal distance computation in terms of shortest route time. The distanceTime facts and the transitive closure dTR predicates will be stored in the database. Our Findall Solutions class generates all possible routes between every pair of provinces. We then apply the optimization rule to find the shortest route between every pair from the generated dTR facts. Our new database of precomputed route facts will be added to the main KB and used in the TD backward reasoner to answer various queries.

Chapter 4

Knowledge Base Design: Ontology and Facts

In Section 4.1, we give a brief introduction to ontologies and how light-weight RDFS ontologies are used to structure the facts in our eTourPlan KB. The descriptions of eTourPlan's FOAF-like profiles for the main tourist entities incorporating Bhutan tourist information used in eTourPlan are described in section 4.2.

4.1 Ontology Design

Ontologies represent the real world in a systematically structured way. By consistently defining deep terms for the same real-world entities, ontologies provide a 'reference model' for their domains, a strong set of terms which can be used to simplify communications between domain experts and therefore increase comprehension and knowledge sharing.

We represent the concepts (or 'classes') of the tourism subdomains in RDF Schema (RDFS) light-weight ontologies, adapted from the Harmonise ontology, discussed earlier (Section 2.3). These ontologies are used to structure the FOAF-like profiles of the

Bhutan tourism subdomains represented in POSL.

RDFS provides modeling primitives for organizing (Web) objects into hierarchies. It is viewed as a basic language for writing light-weight ontologies that are subClassOf taxonomy. In RDFS, objects sharing similar characteristics can be typed with classes. Examples of eTourism classes are events, attractions, accommodations, etc. Individuals belonging to a class are often referred to as instances of that class. For example, the "2008_Film_Festival" can be an instance of the events class.

Classes can be grouped into hierarchies through the subClassOf relationship: a class C is a subclass of a class D if every instance of C is also an instance of D. For example, every instance of a cultural festival is an event instance since cultural festival is a subclass of the event class. Our ontology is adapted from the Harmonise ontology for the following reasons: 1) Harmonise is a mature and standard ontology used by many renowned tourism organizations, 2) It supports interoperability among many agents and applications in the tourism domain, and 3) It is expressed in RDFS, whose SubclassOf hierarchies are supported by our rule engine OO jDREW.

4.2 Ontology and FOAF-like Profile Descriptions of eTourPlan

We now look at each of the tourism subdomains considered in our eTourPlan prototype and their corresponding RDFS light-weight ontologies. The accommodation, attraction, and event subclasses are represented using RDFS subClassOf properties adapted from Harmonise ontology definitions. The administrative subdivision of a country such as Bhutan is implemented with partonomy rules, which are discussed later (Chapter 5).

These ontologies are designed using Protege, which is an ontology-development tool used in such a way that it can provide the type definitions. The ontology is kept light-weight because currently OO jDREW supports only the subClassOf property. We

also implemented certain RDFS domain and range restrictions of properties by using "slots" in OO RuleML[1]. Slotted facts and rules are represented in POSL where a typed slot becomes slot name -> slot filler:filler type, where "->" is used for associating a slot name to a slot filler and ":" is used for appending a filler type to a slot filler. This representation is used to describe profiles for each of the tourist entities. The classes (RDFS light-weight ontologies) provide the type definitions for each of the subdomains, and the instances of tourist entities for these classes are described with FOAF-like profile descriptions. The knowledge about any of the tourist entities can then queried by using the typed rule system OO jDREW.

4.2.1 Province Profile

Every tourist entity can be located in some administrative part of a country. Since provinces are the official top-level governmental administrative division of various countries, we have designed FOAF-like profiles for provinces. Each of the province profiles includes information such as the URL, capital city, geographic details such as elevation and area. In addition, it also provides tourist information such as number of attractions, number of events and number of accommodations found in the province. The vocabularies that are borrowed from the Harmonise naming conventions are prefixed with an 'hs', while those with 'et' prefixes are the user-defined vocabulary for the eTourPlan KB. Namespaces are represented as prefixes before a '.' symbol[2] for the purpose of implementation. It also has an "hs.relatedTo" slot, used to provide the next adjoining province to visit. This FOAF-like relation among the province profiles forms a touristic route across the country. This aspect of connecting the 10 selected provinces of Bhutan, with respect to their neighbouring relations is shown in Figure 4.1. The FOAF relation among these related provinces is symmetric.

We use the name of the province as the object identifier of its profile, and in POSL

[1] `www.ruleml.org/indoo/indoo.html`

[2] This is because the symbol ':', commonly used to express namespaces, as in many other languages is reserved in POSL as a type infix for separating terms from their order-sorted types.

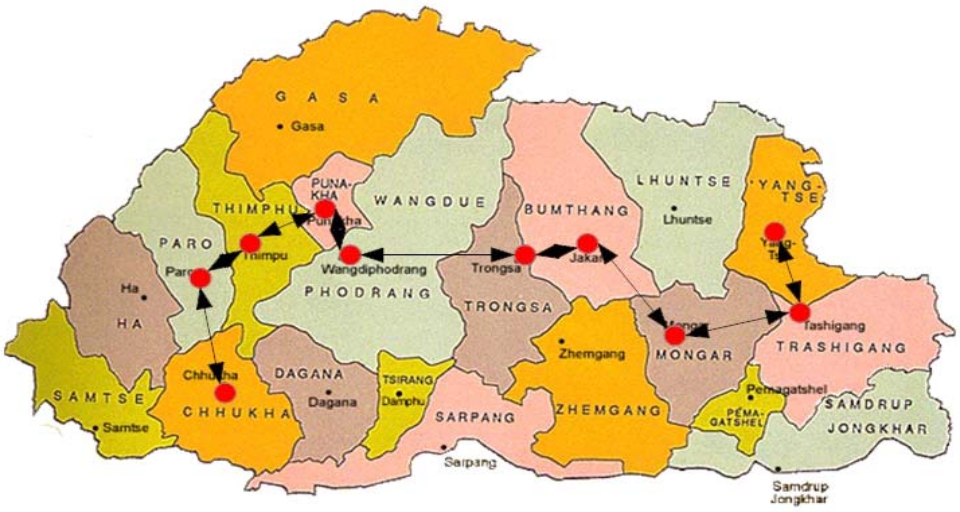

Figure 4.1: The relation between provinces in the KB

is represented before all the regular slots of our object-centric, slotted profile facts. The type of the object is attached to the object name with a colon infix (':'). A hat symbol '^' is used to seperate the identifier from the rest of the slots (properties) in the profile.

Profile of Thimphu Province

```
province(Thimphu:Province^
                hs.url->"http://www.thimphu.gov.bt";
                et.capital->Thimphu_City:City;
                et.area->"1,819 sq.km";
                et.elevation-> "1,300 to 7300 meters";
                et.numBlocks->10:Integer;
                et.numAttractions->3:Integer;
                et.numEvents->2:Integer;
                et.numAccommodations->0:Integer;
                hs.languagesSpoken->"Dzongkha";
                hs.description->"Thimphu is the capital of Bhutan";
                hs.relatedTo->Punakha:Province).
```

We want to note that these profile descriptions in POSL syntax are interconvertible to OO RuleML, and they can be tested in either of the syntax in the reasoning engine, OO jDREW. The detailed information about these province profiles (cf. Appendix B) can be extracted through queries in the OO jDREW TD reasoning engine. Such rules

and queries are discussed later in chapters 5 and 6.

4.2.2 Events Class

The Events subdomain provides a model of a real-world event, characterized in terms of locations, timescale, theme, and other properties. Events are of two main types, periodic events and single occurrence events. An example of periodic event is 'Sunday_flea_Market', which occurs once a week periodically, or an annual festival that occurs once a year on a particular date, whereas a single occurrence event occurs only once, like the "Beijing_2008_Sum- mer_Olympics". However, for the purpose of this thesis, we have considered both kinds of events under one category. The subclassOf hierarchy of the events subdomain is implemented by a light-weight ontology and the domain and range restrictions of other properties are represented in the FOAF-like fact instances of events in POSL format. The events class has nine subclasses, each of which is further subclassified into smaller classes as shown in Figure 4.2. This diagram was produced by Protege. An excerpt from the RDFS taxonomy describing a portion of the subClassOf hierarchy for the events class is shown below:

Events
- Festival
  - Annual_festival
  - Dance_festival
  - Film_festival
  - Gastronomic_festival
  - International_festival
  - Music_festival
  - National_festival
  - Seasonal_festival
  - Traditional_festival

The other subclasses of Events are Adventure, Arts, Market, Nature, Nightlife, Religion, Sport and Trade. The complete RDFS type hierarchy is shown in Appendix E. Each instance of the above classification for the Events subdomain is represented as FOAF-like profiles (cf. Appendix B). A FOAF-like profile of an instance of the Events

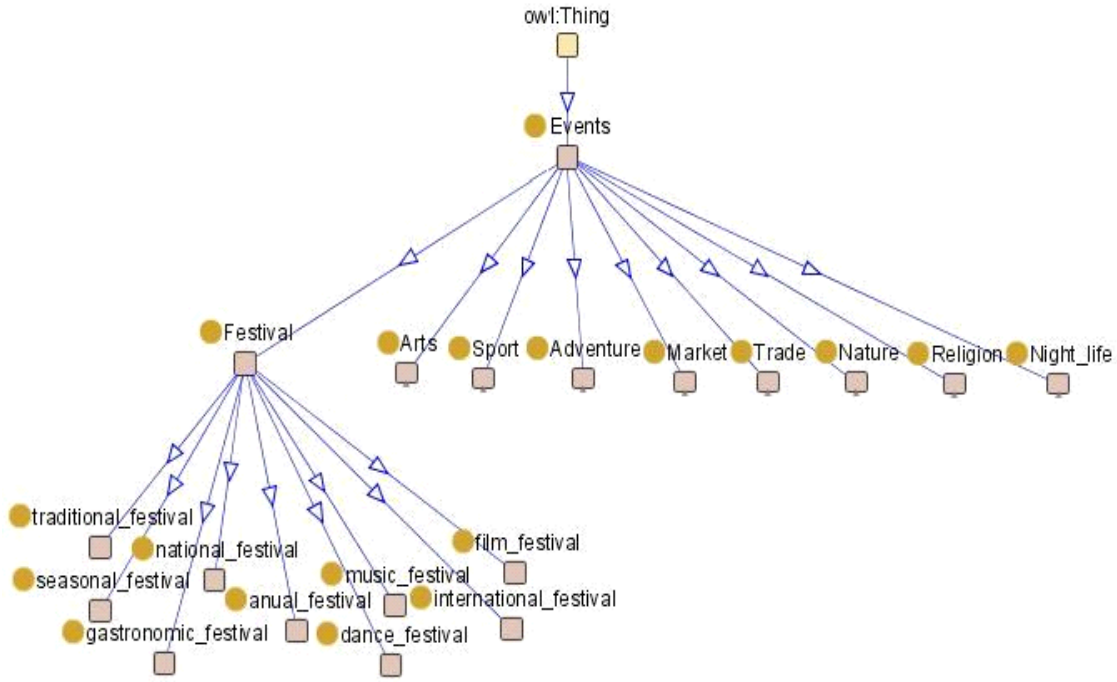

Figure 4.2: Events class (full version in Appendix E)

class of type Annual_festival is shown below:

Profile of Thimphu_Tshechu

```
event(Thimphu_Tshechu:Annual_festival^
                      hs.url->" ";
                      hs.startDate->date[2008:Real,10:Real,09:Real];
                      hs.endDate->date[2008:Real,10:Real,11:Real];
                      et.theme->Cultural_Religious_Heritage;
                      hs.location->Tashichoe_Dzong:Fortress;
                      et.province->Thimphu:Province;
                      hs.description->"It is a popular festival in Thimphu";
                      hs.relatedTo->Thimphu_Drupchen:Annual_festival).
```

The above slotted fact describes an event in Bhutan, "Thimphu_Tshechu" under the type Annual_festival, along with other slots describing the event details. The "hs.relatedTo" slot relates this event profile to the next upcoming event, that is closest to this periodic event schedule.

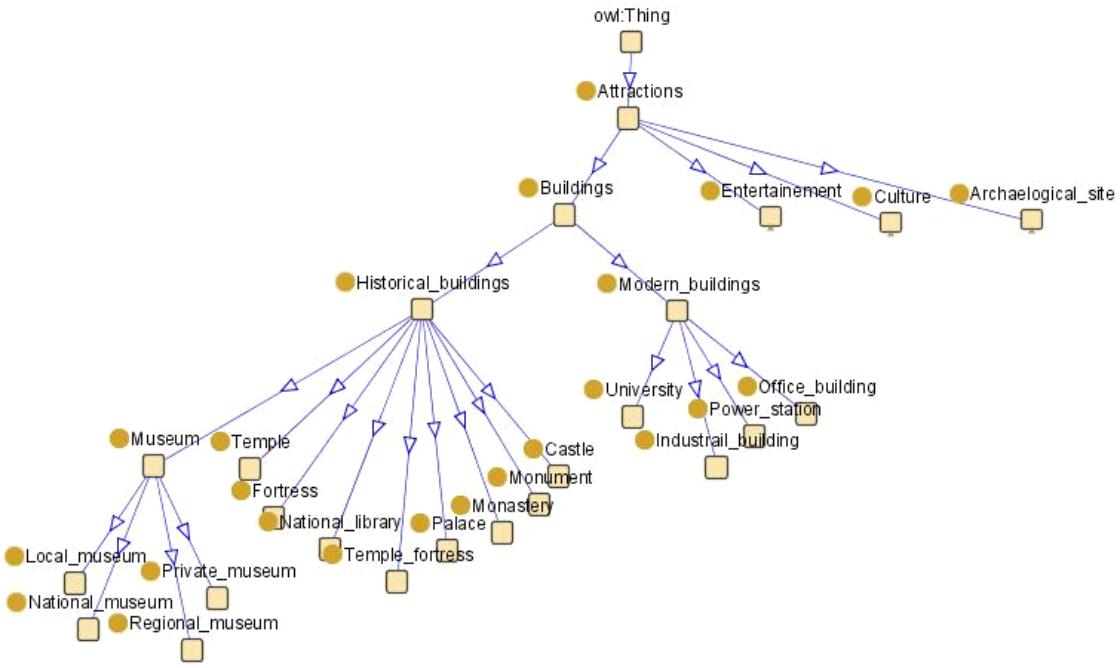

Figure 4.3: Attractions class (full version in Appendix E)

4.2.3 Attractions Class

This section presents the RDFS ontology of attractions. The Attraction subdomain considers attractions that are open to the public at a fixed location and are generally accessible throughout the year. An example of such an attraction could be a museum, a waterfall, or a park. The light-weight ontology for the attraction subdomain is shown in Figure 4.3. An excerpt from the RDFS taxonomy describing a portion of the subClassOf hierarchy for the attraction subdomain is shown below:

Attractions
- Buildings
  - Historical_buildings
    - Museum
      - Local_museum
      - National_museum
      - Regional_museum
      - Private_museum

A FOAF-like profile for an instance of the Attractions class is shown below:

Profile of Ta_Dzong

```
attraction(Ta_Dzong:National_museum^
                    hs.url->"www.nationalmuseum.gov.bt/";
                    et.subblock->Goepay:Village;
                    et.province->Paro:Province;
                    et.theme->Cultural_Religious_Heritage;
                    et.open->Open[DaysOfWeek[Tue, Wed, Thu, Fri, Sat, Sun],
                                  Period[10:Real, 16:Real]];
                    et.capitalDistance->0.5:Real;
                    hs.description->"It is the biggest and the oldest museum
                                     in Bhutan";
                    hs.contact->" ";
                    hs.schedule->"12 months";
                    hs.relatedTo->Tashichoe_Dzong:Fortress).
```

The above slotted fact describes an attraction site, "Ta_Dzong", under the class National_museum as its type, along with other slots describing the attraction. The "hs.relatedTo" slot name in attraction profiles plays a very important role in connecting all the popular neighbouring attractions in the country. This property of our KB is later used to recommend and plan attraction-based travels, as discussed later (in Chapter 5).

An attraction has much in common with an accommodation and has many similar properties because both are static (i.e., they do not move from their place) and both are accessible most of the time. Both could have a price attribute but in this thesis, we have described price only for accommodations.

4.2.4 Accommodations Class

The accommodation class is subclassified into five main categories as shown in Figure 4.4. Our classification is a subclass of the Harmonise accommodation type definition. We have considered five related types in this thesis. A FOAF-like profile of an accommodation is shown below:

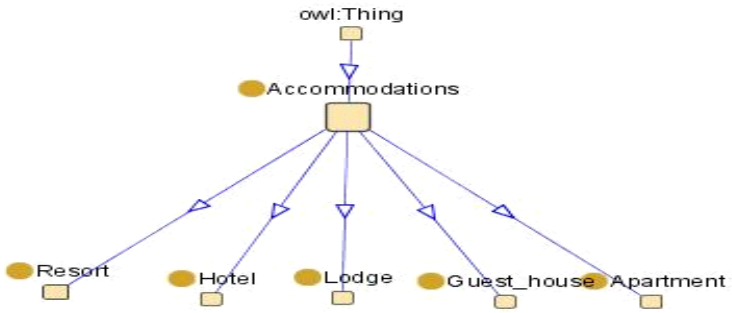

Figure 4.4: Accommodations class

Profile of Wangdicholing_Lodge

```
accommodation(Wangdicholing_Lodge:Lodge^
                              hs.url->"http://www.wangdicholing.bt/";
                              et.rating->3:Real;
                              et.minPrice->800:Real;
                              et.subblock->Chamkhar:Town;
                              et.province->Bumthang:Province;
                              hs.telecoms->Telecoms[
                                            et.landline->9753631452;
                                            et.cell->97517682948];
                              hs.contact->"manager@wangdicholing.bt";
                              hs.relatedTo->Yangphel_Guest_house:Guest_house).
```

The above slotted fact describes an accommodation, "Wangdicholing_Lodge" under the class 'Lodge'as its type, along with slots describing the accommodation such as the star rating, URL, minimum price, and a related accommodation. The "hs.relatedTo" slot in the accommodation subdomain is used to link a profile to an accommodation of similar (ideally equal or nearly equal) rating within minimum distance in the same province.

We will now proceed to how the well-structured knowledge in our KB is used for implementing rules that operate on the aforementioned facts.

Chapter 5

Knowledge Base Design: Rules

The eTourplan rule system has three distinct top-level operations as shown in Figure 5.1. Using the fact base structured with ontologies described in Chapter Four, the current Chapter examines the main functionalities that we have implemented using rules in POSL syntax. The classification of these operations is as follows:

- **Search Engine**: Our KB consists of facts about the five main subdomains of tourism: events, attractions, accommodations, transportation, and regions. This KB can be used for a semantic information search on these subdomains. For example, users can query for detailed information about any specific province, event, attraction, accommodation, or route. The search engine allows parametric searches of information based on a user's preferred parameters, such as search by name, theme, or location. Another flexibility offered to our users is that they can find information about any tourist entity, at any sublevel of the country, from the largest search space (country) to the lowest level (subblock). The details of search options are discussed in Section 5.2.

- **Recommender System**: The FOAF-like profiles of neighbouring provinces in our KB are linked together from a touristic point of view as shown in Section

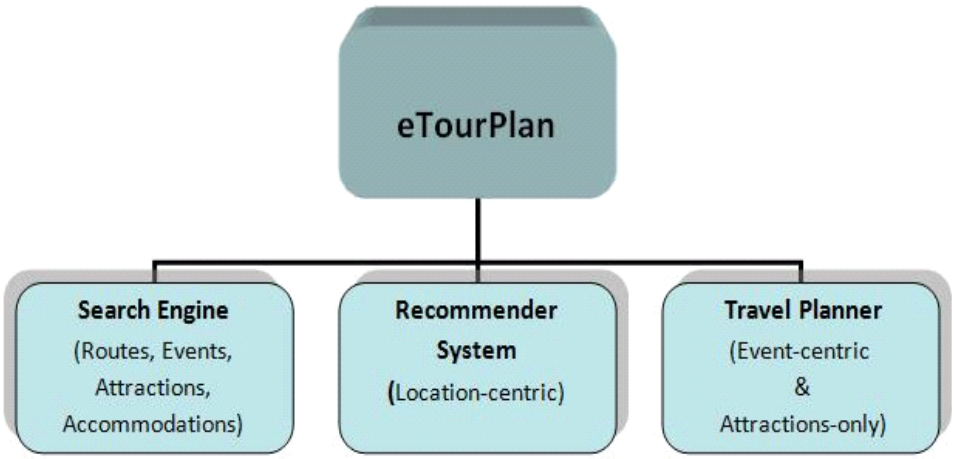

Figure 5.1: The top-level architecture of the eTourPlan

4.2.1. This relation among the provinces in our KB allows us to provide system recommendation of route and activity by chaining through one province to the next related province. It basically uses the FOAF concept to provide location-centric recommendations of activities. The two modes of the location-centric travel recommender are discussed in Section 5.2.4.

- **Travel Planner**: The main travel planner rule system integrates the functionalities of the search and location-centric recommender subsystems. In order to build a travel planner, we need to take into account crucial constraints such as time and distance. The planner employs the various predicates for different operations such as date validation, route finding, and event search in the KB. We have implemented two modes of planning, event-centric and attraction-only planning (cf. Table 2.1 in Section 2.1). The event-centric planning is based on a temporal-geographic search criterion. The FOAF relation between attractions is used as the primary basis for planning a travel in attraction-only mode. These are discussed in detail in Section 5.3.

Our eTourPlan rule system offers the following options to our users:

- Parametric search of tourist information

- System route planning based on province profiles
- Route planning for user-preferred provinces
- Location-centric recommendation by the system
- Location-centric recommendation for user-preferred provinces
- Attraction-only travel planning
- Event-centric travel planning with attraction recommendations

Following the design and structure of the KB facts described in Chapter Four, we now discuss the rule subsystems that operate on top of these facts. In Section 5.1, we discuss two main auxiliary rules, for the administrative partonomy and route computation. We discuss the differences and the interrelations between a partonomy and a taxonomy, both of which are used for defining the Regions subdomain. This is followed by a section discussing the route computation rule subsystem. Each of the top-level rule subsystems (cf. Figure 5.1) that are integrated in our eTourPlan prototype are discussed in Section 5.2. Finally, we describe our main travel planning rule system in Section 5.3.

5.1 Auxiliary Rules

5.1.1 Partonomy rules

Partonomy is a classification system based on the partOf relation between subparts and superparts. We did not implement a RDFS ontology for the Regions subdomain because the partOf relation is different from the subClassOf property that is used in the RDFS for the rest of the subdomains discussed earlier (in Section 4.2). The Regions subdomain is instead structured in our KB using partonomy rules. The

superpart (country) is categorized into four subregions: the Eastern_region, the Central_region, the Western_region, and the Southern_region (There is no Northern_region since northern Bhutan is covered with mountains). These subregions are classified into provinces, and provinces in turn are subclassified into blocks, which are further classified into four subblock types: city, town, village, or place. These subblocks, then, contain attractions, event locations, and accommodations, which are described as their elements. We will now look at two different approaches to define the partonomy of a region.

5.1.1.1 Classification of Regions

Before discussing the classification of regions using partonomy rules, we would like to briefly discuss the distinction and the interrelation between a partonomy and a taxonomy. A taxonomy is a hierarchical classification of representing a partial order classes at different levels of abstraction. A taxonomy can be regarded as an "order-sorted type hierarchy". A partonomy, as previously discussed, is a hierarchical division according of superparts into subparts. Both classifications are asymmetric as well as transitive, and both can form hierarchies [40].

However, there are several differences between partonomy and taxonomy classifications. First, since a taxonomy is the hierarchy of classes whereby lower levels are related to upper levels by inclusion, the subclasses generally allow the inference of properties from superclasses. Part-of relations do not support such an inheritance; i.e., properties of the whole are not necessarily also properties of its parts. Second, a taxonomy describes the division of a set of objects according to various categories, each one containing a set of objects. A partonomy, on the other hand, describes the part of an object with its subparts. Tversky in [57] said that "A partonomic analysis reveals subcomponents and the relations among them, whereas a taxonomic investigation reveals features shared by a number of instances and their range of variability."

A partonomy is a one-to-many relation. On the other hand, a taxonomy can be

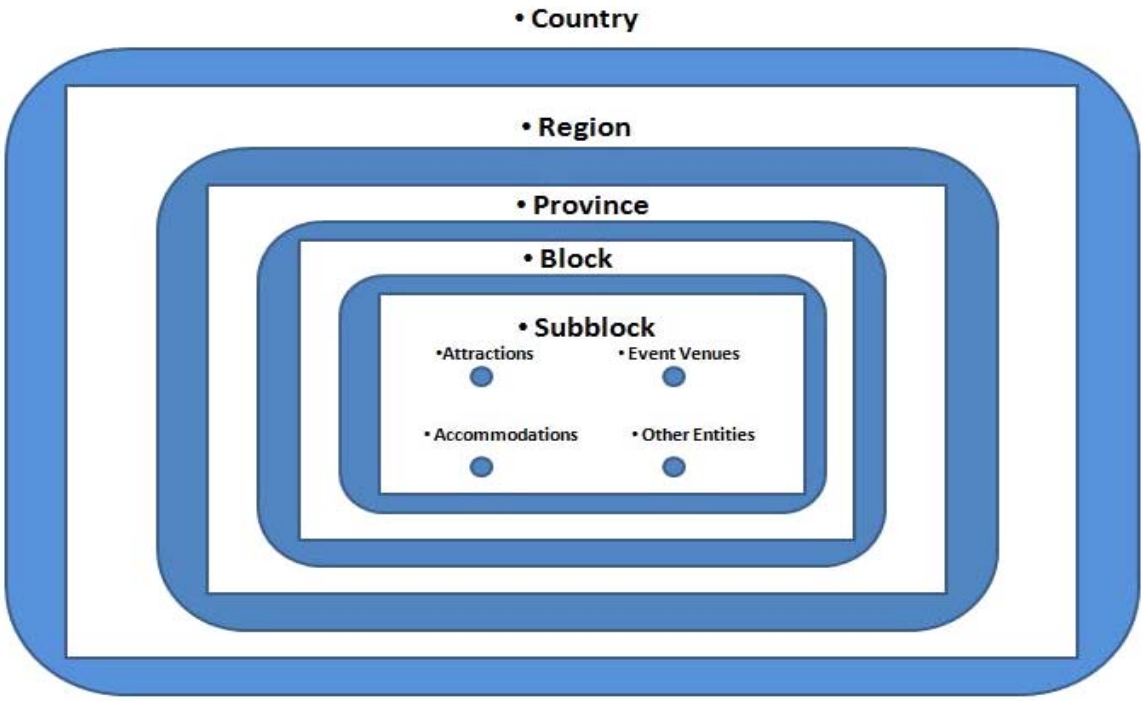

Figure 5.2: Subparts of a Country

either a one-to-many or a many-to-many relation. In most cases, a taxonomy is the product of a bottom-up investigation, in which objects are grouped on the basis of common and distinctive features [40]

Both taxonomic and partonomic knowledge are essential parts of our KB, and the usefulness of both of these forms of knowledge are shown together in the implementation of the Regions subdomain. We present an administrative subdivision of a country (Figure 5.2) implemented in POSL syntax: domain-specific and general partonomy rules.

5.1.1.2 Domain-Specific Partonomy Rule

An example of the partonomy of Bhutan dividing the country into smaller regions and further locating its smallest instances (attractions, event locations, and accommodations) is shown in Figure 5.3. Protege, an ontology editor and knowledge acquisition

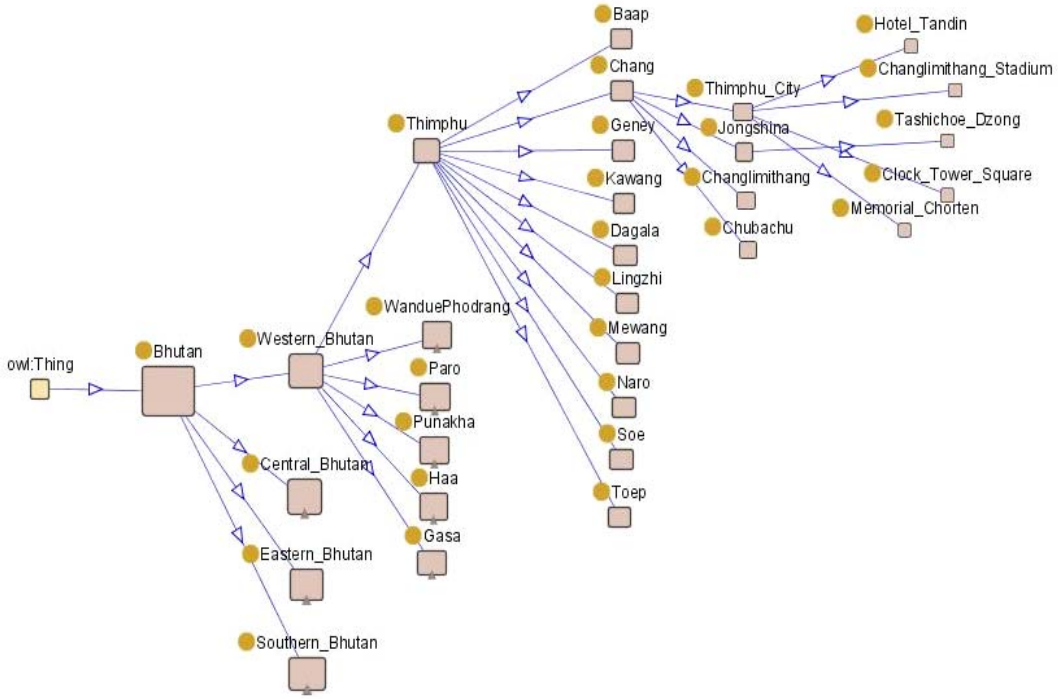

Figure 5.3: Excerpt of the partonomy of Bhutan

system, is used here only for visual demonstration purposes, to illustrate the partonomy of the country. We can see that the lowest node of the tree are tourist entities. For example, in Figure 5.3, "Tashichoedzong" is an attraction of type "fortress" located in the subblock "Jongshina" of type "town" under the block "Chang" in the province "Thimphu", which is located in the western region of Bhutan. The POSL representation of this example is shown below, along with some other sample facts. The subpart representation of a country is described with an n-ary predicate with respect to a location of an attraction. This is a domain-specific approach to represent the partOf relationships of a country.

Ground Facts

```
%region(?Province, ?Region)

 region(Thimphu, Western).
 region(Bumthang, Central).
```

```
region(Tashiyangtse, Eastern).
region(Samtse, Southern).

%city(?CityC, ?BlockB, ?ProvinceP, ?CountryC, ?AttractionX:Attractions).

city(Thimphu_city,Upper_Thimphu,Thimphu,Bhutan,Tashichoe_Dzong:Fortress).
town(Paro_town,  Paro_Bazar, Paro, Bhutan, Museum:National_museum).
village(Shab, Shaba, Paro, Bhutan, Chu_Dzong:Fortress).
place(Drugedingkha, Shari, Paro, Bhutan, Park:National_park).
```

The ground fact representation of the regions of provinces are constructed by pairing the name of a region and its corresponding province as two arguments of a binary relation. Similarly, attractions are described at the level of subblocks and are used as arguments of one of the five predicates of city, town, village, or place relation. The first ground fact shown above represents the attraction "Tashichoe_Dzong" of type fortress is located in the city "Thimphu_city", in the block "Upper_Thimphu", in the province "Thimphu", in the Country "Bhutan". The inference rules that will operate on the above ground facts are implemented as follows:

KB

```
%An Attraction is in a country if it is in a province of a region
%in that country

country(?Country, ?Attraction:Attractions):-
  province(?Province, ?Region, ?Country, ?Attraction: Attractions).

%An Attraction is in the province of a region if it is in a block that
%is located in province of the same region.

province(?Province, ?Region, ?Country, ?Attraction:Attractions):-
  block(?Block, ?Province, ?Country, ?Attraction:Attractions),
  region(?Province, ?Region).

%An Attraction is in the block of a province if it is either in a city,
%town, village or a place in this block of the same province.

block(?Block, ?Province, ?Country, ?Attraction:Attractions):-
  city(?City, ?Block, ?Province, ?Country, ?Attraction:Attractions).

block(?Block, ?Province, ?Country, ?Attraction: Attractions):-
  town(?Town, ?Block, ?Province, ?Country, ?Attraction:Attractions).
```

```
block(?Block, ?Province, ?Country, ?Attraction:Attractions):-
  village(?Village, ?Block, ?Province, ?Country, ?Attraction:Attractions).

block(?Block, ?Province, ?Country, ?Attraction:Attractions):-
  place(?Place, ?Block, ?Province, ?Country, ?Attraction:Attractions).


%Predicate to get attractions given zero or more constant arguments.

getAttraction(?Block, ?Province, ?Region, ?Attraction:Attractions):-
  block(?Block, ?Province, ?Country, ?Attraction:Attractions),
  region(?Province, ?Region).
```

The above rule can find attraction names located in a specified block, province, or a region. It can also be used to get the name of the block, province, and region of a known attraction. Different modes of queries with different combinations of free variables and constants are shown below. We will assume a user asks for more solutions until none are left.

Sample Queries

```
%QUERY: getAttraction(?Block, ?Province, ?Region, ?A:Attractions)
%Solution: It will return all the attractions in the KB since it is an open
           query (all the arguments are free variables).

%QUERY: getAttraction(?Block, Paro, ?Region, ?A:Attractions)
%Solution: It will return all the attractions in the province "Paro"

%QUERY: getAttraction(?Block, ?Province, Western, ?A:Attractions)
%Solution: It will return all the attractions in the "Western" Bhutan

%QUERY: getAttraction(Thimphu_city, ?Province, ?Region, ?A:Attractions)
%Solution: It will return all the attractions located in the block
           "Thimphu_city"

%QUERY: getAttraction(?Block, Paro, ?Region, Memorial_Chorten:Attractions)
%Solution: It will return the Block of the attraction "Memorial_Chorten:
           Attractions in Paro"
```

An attraction's full address can be computed from the facts. This approach is called domain-specific because the predicate names are related to the subparts of the region. The relation names such as city, town, village, or place can be defined as types in our RDFS type definition and used as types to the subblock argument. This

extension of enriching partOf with an RDFS type taxonomy is discussed in the next section.

5.1.1.3 Enriched Partonomy Rule with Type Definition

We now present an enhanced approach to build the partonomy of a country using chained binary facts. The partonomic differentiation is thus shown as a hierarchy from the highest level of the partonomy (country) to its lowest unit (subblock). Attractions, event locations, and accommodations are located in these subblocks. Every argument in the facts is extended by a type. This adds more knowledge to the arguments. It also shows a clear interrelation between a taxonomy and a partonomy. The main advantage of having this richer representation of knowledge is that it makes our task easier at the rule-implemenation level. We will discuss a good example of this advantage after representating the ground facts. The ground facts shown below are an excerpt from our Bhutan KB; refer to Appendix C to view the complete KB.

Ground Facts

```
% siteOf(?Attraction:Attractions, ?Subblock:Subblock).

 siteOf(Ura_Lhakhang:Temple, Ura_Village:Village).
 siteOf(Bumthang_Dzong:Fortress, Chamkhar:Town).

%partOfBlock(?Subblock:Subblock, ?Block:Block).

  partOfBlock(Buli:Village, Chhoekhor:Block).
  partOfBlock(Tharpaling:Village, Chhoekhor:Block).
  partOfBlock(Prakar:Village, Chhoekhor:Block).
  partOfBlock(Nimalung:Village, Chhoekhor:Block).
  partOfBlock(Domkhar:Village, Chhoekhor:Block).
  partOfBlock(Chamkhar:Town, Chhoekhor:Block).
  partOfBlock(Tangbi:Village, Chhoekhor:Block).
  partOfBlock{Ura_Village:Village, Ura:Block).

%partOfProvince(?Block:Block, ?Province:Province).

  partOfProvince(Chhoekhor:Block, Bumthang:Province).
  partOfProvince(Chumme:Block, Bumthang:Province).
  partOfProvince(Tang:Block, Bumthang:Province).
  partOfProvince(Ura:Block, Bumthang:Province).
```

```
%partOfRegion(?Province:Province, ?Region:Region).

  partOfRegion(Bumthang:Province, Central:Region).

%partOfCountry(?Region:Region, ?Country:Country).

  partOfCountry(Central:Region, Bhutan:Country).
```

The main advantage of this representation of partonomy facts over the first representation is that it avoids redundant information, and it demonstrates chaining through the binary facts. This definition of our partonomy is then used for various information retrieval tasks in a systematic manner. A simple application of the partOf hierarchy is the "getFullAddress" rule shown below. It is used to compute the full address of any attraction in the KB.

KB

```
getFullAddress(?Location, [?Subblock,?Block,?Province,?Region,?Country]):-
  siteOf(?Location, ?Subblock),
  partOfBlock(?Subblock, ?Block),
  partOfProvince(?Block, ?Province),
  partOfRegion(?Province, ?Region),
  partOfCountry(?Region, ?Country).
```

Sample Query

```
getFullAddress(Ta_Dzong:National_museum, ?Address)
```

OO jDREW TD Result

```
 ?Address= [Hungrel:Village,
            Hungrel:Block,
            Paro:Province,
            Western:Region,
            Bhutan:Country]
```

The result shown above is an example that demonstrates how the type definition avoids information ambiguity. The address of the museum has the same name for its subblock and block. In Bhutan, it is often the case that the main subblock of a block has the same name as its block. The two kinds of inferences are distinguished by the type definition. We have further extended this work to provide a more precise

addressing scheme by including the GPS coordinates of any given locations. We implementated rules that employ facts about the GPS locations. Examples using fictitious data are shown below:

Sample GPS Facts

```
address(Bumthang_Dzong:Fortress,
        Loc[latitude->Detail[degree->48:Real; minute->0:Real];
            longitude->Detail[degree->66:Real; minute->40:Real]]).

address(Zorig_Chusum:Crafts_show,
        Loc[latitude->Detail[degree->51:Real; minute->6:Real];
            longitude->Detail[degree->114:Real; minute->1:Real]]).

address(Ugyen_Chholing_Museum:Local_museum,
        Loc[latitude->Detail[degree->49:Real; minute->11:Real];
            longitude->Detail[degree->123:Real; minute->10:Real]]).
```

From such centralized stored facts concerning the longitude and latitude of attraction sites, we can apply the rules "getLatiLoc" and "getLongiLoc" to get the latitude and longitude data points, which will be later used for the computation of the actual latitude and longitude of a location by the "getLatitude" and "getLongitude" relations, respectively.

```
getLatiLoc(?Location, ?Degree, ?Minute) :-
  address(?Location,
          Loc[latitude->Detail[degree->?Degree; minute->?Minute]!?]).

getLongiLoc(?Location, ?Degree, ?Minute) :-
  address(?Location,
          Loc[longitude->Detail[degree->?Degree; minute->?Minute]!?]).

%Computes longitude and latitude of a location

 getLatitude(?Location, ?Latitude) :-
   getLatiLoc(?Location, ?Degree, ?Minute),
   divide(?Tmp1, ?Minute, 60:Real),
   add(?Tmp2, ?Degree, ?Tmp1),
   multiply(?Latitude, ?Tmp2, 100:Real).

 getLongitude(?Location, ?Longitude) :-
   getLongiLoc(?Location, ?Degree, ?Minute),
   divide(?Tmp1, ?Minute, 60:Real),
   add(?Tmp2, ?Degree, ?Tmp1),
```

```
multiply(?Longitude, ?Tmp2, 100:Real).
```

Considering the GPS points, the enhanced version of the "getFullAddress" relation is shown next, followed by a sample query and the corresponding result generated in OO jDREW TD.

Sample KB

```
getFullAddress(?Location, [?Subblock, ?Block, ?Province, ?Region, ?Country,
               ?Latitude, ?Longitude]):-
  getLatitude(?Location, ?Latitude),
  getLongitude(?Location, ?Longitude),
  siteOf(?Location, ?Subblock),
  partOfBlock(?Subblock, ?Block),
  partOfProvince(?Block, ?Province),
  partOfRegion(?Province, ?Region),
  partOfCountry(?Region, ?Country).
```

Sample Query

```
getFullAddress(Bumthang_Dzong:Fortress, ?Address)
```

OO jDREW TD Result

```
?Address=
  [4800.0:Real,
   6666.666666666667:Real,
   Chamkhar:Town,
   Chhoekhor:Block,
   Bumthang:Province,
   Central:Region,
   Bhutan:Country]
```

The taxonomy-enriched partonomy rules allow us to search attractions at any subpart of a country. In order to provide an easy search of attractions and flexibility in the search domain to our users, we have implemented five different modes of "getAttraction", each dealing with a specific sublevel of a country. The recursive "getAttraction" predicate shown below allows attraction search at any specified subpart of a country.

KB

```
%-----at a subblock level-----------------------------------------------%

getAttraction(?Attraction:Attractions, ?Subblock:Subblock):-
  siteOf(?Attraction:Attractions, ?Subblock:Subblock).

%----at a block level--------------------------------------------------%

getAttraction(?Attraction:Attractions, ?Block:Block):-
  partOfBlock(?Attraction:Attractions, ?Block:Block).

getAttraction(?Attraction:Attractions, ?Block:Block):-
  partOfBlock(?Subblock:Subblock, ?Block:Block),
  getAttraction(?Attraction:Attractions, ?Subblock:Subblock).

%-----at a province level-----------------------------------------------%

getAttraction(?Attraction:Attractions, ?Province:Province):-
  partOfProvince(?Attraction:Attractions, ?Province:Province).

getAttraction(?Attraction:Attractions, ?Province:Province):-
  partOfProvince(?Block:Block, ?Province:Province),
  getAttraction(?Attraction:Attractions, ?Block:Block).

%-----at a region  level------------------------------------------------%

getAttraction(?Attraction:Attractions, ?Region:Region):-
  partOfRegion(?Attraction:Attractions, ?Region:Region).

getAttraction(?Attraction:Attractions, ?Region:Region):-
  partOfRegion(?Province:Province, ?Region:Region),
  getAttraction(?Attraction:Attractions, ?Province:Province).

%----at a country level-------------------------------------------------%

getAttraction(?Attraction:Attractions, ?Country:Country):-
  partOfCountry(?Attraction:Attractions, ?Country:Country).

getAttraction(?Attraction:Attractions, ?Country:Country):-
  partOfCountry(?Region:Region, ?Country:Country),
  getAttraction(?Attraction:Attractions, ?Region:Region).
```

By querying the above rule set, we can find names of attractions at any level of the partonomy, computed one at a time. A remark on the binary "getAttraction" relation is that it takes care of facts that will not necessarily be at the lowest possible

level in the hierarchical partOf definition of a country. For example, an attraction can be directly placed under any subpart of a country. This gives flexibility to knowledge designers, as it is not always possible to know the complete hierarchy levels for some entity locations. For example, the attraction "Niagra_Fall" can be directly placed as a partOf Canada (rather than "Ontario"), and yet the system can find the correct solution.

Sample search queries and the corresponding results as run in OO jDREW TD for the binary "getAttraction" relation are shown below:

Sample Queries

```
%To get all the attractions located in the country Bhutan
 getAttraction(?Attraction:Attractions, Bhutan:Country)

%To get all the attractions located in the Western region of Bhutan
 getAttraction(?Attraction:Attractions, Western:Region)

%To get all the attractions located in the province Bumthang
 getAttraction(?Attraction:Attractions, Bumthang:Province)

%To get all the attractions located in the block Chhoekhor
 getAttraction(?Attraction:Attractions, Chhoekhor:Block)

%To get all the attractions located in the subblock Chamkhar of type town
 getAttraction(?Attraction:Attractions, Chamkhar:Town)
```

The solution for the last query "getAttraction" at the subblock (lowest subdivision of a country) "Chamkhar", is shown below. Each of the following attractions are result bindings given one result at a time.

OO jDREW TD Result

```
?Attraction= Bumthang_Dzong:Fortress
?Attraction= Zugney:Textiles
?Attraction= Ugyen_Chholing_Museum:Local_museum
?Attraction= Petseling_Gompa:Temple
```

The search results of the "getAttraction" query can be enriched by retrieving more knowledge from the FOAF-like profiles of each attraction in our KB. In order to

provide detailed information about the attraction and its location, we integrated the getFullAddress predicate with getAttraction. The second argument, "?AttractionDetails", in the query shown after the rule gets bound to the attraction details as shown in the result.

KB

```
getAttractionDetails(address->[?Subblock,?Block,?Province,?Region,?Country];
                     [Name->?Name;
                      WebLink->?Url;
                      Address->[?Subblock,
                                ?Block,
                                ?Province,
                                ?Region,
                                ?Country];
                      Description->?Description;
                      RelatedTo->?RelatedTo]):-
   attraction(?Name^hs.url->?Url;
                    hs.description->?Description;
                    et.open->?OpenHours;
                    et.theme->?Theme;
                    et.subblock->?Subblock;
                    et.province->?Province;
                    hs.relatedTo->?RelatedTo!?),
   getFullAddress(?Name, [?Subblock,?Block,?Province,?Region,?Country]).
```

Sample query

```
getAttractionDetails(address->[?Subblock,
                               Hungrel:Block,
                               ?Province,
                               ?Region,
                               ?Country];
                     ?AttractionDetails)
```

OO jDREW TD Result

```
?AttractionDetails=
  [Name->Ta_Dzong:National_museum;
   WebLink->"www.nationalmuseum.gov.bt/";
   Description->"It is the biggest and the oldest museum in Bhutan";
   Address->[Goepay:Village,
             Hungrel:Block,
             Paro:Province,
             Western:Region,
             Bhutan:Country];
             RelatedTo->Tashichoe_Dzong:Fortress]
```

This "getAttraction" predicate can be generalized to a "getEntity" predicate, which will return the required information for a specified entity. A recursive "getAttractionList" predicate, which accumulates a number of 'N' attractions at a user-specified location is shown below.

KB

```
%to get a number of N attractions at any level of the partonomy.

getAttractionList(location->?location;
                  visited->?Visited;
                  attractionList->[]; num->0).

getAttractionList(location->?location;
                  visited->?Visited;
                  attractionList->[?Attraction|?Rest];
                  num->?N:Integer):-
  greaterThan(?N:Integer, 0:Integer),
  getAttraction(?Attraction, ?location),
  notMember(?Attraction, ?Visited),
  subtract(?Numtleft, ?N:Integer,  1:Integer),
  getAttractionList(location->?location;
                    visited->[?Attraction|?Visited];
                    attractionList->?Rest;
                    num->?Numtleft).
```

Sample query

```
getAttractionList(location->Chamkhar:Town;
                  visited->[];
                  attractionList->?AttractionList;
                  num->3:Integer)
```

OO jDREW TD Result

```
?AttractionList=
  [Bumthang_Dzong:Fortress,
   Zugney:Textiles,
   Ugyen_Chholing_Museum:Local_museum]
```

5.1.1.4 General Partonomy Rule

Partonomy rules, however, can be made more general, so that they are less restricted by the domain of interest. A generic definition of the "partOf" relation can be used for describing the entire partonomy of any domain. The transitive closure

of the binary "partOf" relation allows us to do any depth of chaining, from a given superpart of the hierarchy to the lowest subpartOf property. An example of such a derivation of facts is shown below. In this design, we need to describe every subpart relation by the same binary predicate name "partOf". We can define partOfTrans for chaining through pairs of partOf ground facts.

KB

```
partOfTrans(?X, ?Y):-
  partOf(?X, ?Y).

partOfTrans(?X, ?Z):-
  partOf(?X, ?Y),
  partOfTrans(?Y, ?Z).
```

Assuming, our facts are all defined with one common generic name, "partOf", the partOfTrans predicate can explore the entire partonomy as well as check the subparts of each relation by instantiating one of the two arguments. Similar to the first approach, we can generate new variable bindings by using different combinations of constants and free variables in the query. Some facts, a sample query, and the variable bindings in the output generated by OO jDREW TD are shown below.

Ground Facts

```
%partOf(?SubPart, ?SuperPart).

 partOf(Chamkhar:Town, Chhoekhor:Block).
 partOf(Chhoekhor:Block, Bumthang:Province).
 partOf(Bumthang:Province, Central:Region).
 partOf(Central:Region, Bhutan:Country).
```

Sample Query

```
partOfTrans(?SubPart, ?SuperPart)
```

OO jDREW TD Result

```
First Solution:
?SubPart= Chamkhar:Town
?SuperPart= Chhoekhor:Block

Next Solution:
```

```
?SubPart= Chhoekhor:Block
?SuperPart= Bumthang:Province

Next Solution:
?SubPart= Bumthang:Province
?SuperPart= Central:Region

Next Solution:
?SubPart= Central:Region
?SuperPart= Bhutan:Country

Next Solution:
?SubPart= Chamkhar:Town
?SuperPart= Bumthang:Province

Next Solution:
?SubPart= Chhoekhor:Block
?SuperPart= Central:Region

Next Solution:
?SubPart= Bumthang:Province
?SuperPart= Bhutan:Country

Next Solution:
?SubPart= Chamkhar:Town
?SuperPart= Central:Region

Next Solution:
?SubPart= Chhoekhor:Block
?SuperPart= Bhutan:Country

Last Solution:
?SubPart= Chamkhar:Town
?SuperPart= Bhutan:Country
```

This query will bind its variables to every pair of partOf instances in the KB. Using the same facts, we can derive more specific facts by instantiating one of the arguments in the query:

```
partOfTrans(?SubPart, Chhoekhor:Block)

Variable binding:
 ?SubPart=  Chamkhar:Town
```

This query has its second argument instantiated to the constant "Chhoekhor:Block", and the system returns all variable bindings of "?SubPart" that are parts of the block "Chhoekhor:Block". Similarly, we can find parts of any province, region, or the country. We can also perform all the domain-specific queries that were shown in our first approach such as get attraction address. Like in our type-enriched domain-specific partonomy, every part has either exactly one or no superpart etc, down to the leaves level. The highest level of our hierarchy, "country", has no superpart and the lower sublevels have exactly one unique superpart.

However, domain-specific partonomy rules are used in this thesis, so as to speed up the execution time by narrowing down the search space for each specific rule set. Please refer to Appendix C to view the complete partonomy of Bhutan.

5.1.2 Distance Computation

Taking into account the importance of measuring distance for any kind of travel planning, we considered the aspect of distance computation as one of the two primary auxiliary rule sets.

The implementation of the distance computation KB has been first tested solely in the OO jDREW Top-down Backward Reasoner. From the various architectural features of OO jDREW described in Section 3.3, we found the TD FindAll Solutions architecture the most suitable for the precomputation of routes and distance times.

5.1.2.1 Precomputation of All Routes

The transitive closure database consists of ternary "distanceTime" ground facts, which represent the distances between adjacent nodes measured in bus hours. The distance computation rule subsystem has been tested for sample graphs as shown in Figure 5.4. In this example, we have only considered a single mode of transportation. However, in our complete KB (cf. Appendix C), we considered four modes of transportation, namely bus, taxicab, bicycle, and hiking. The bus distance time, measured

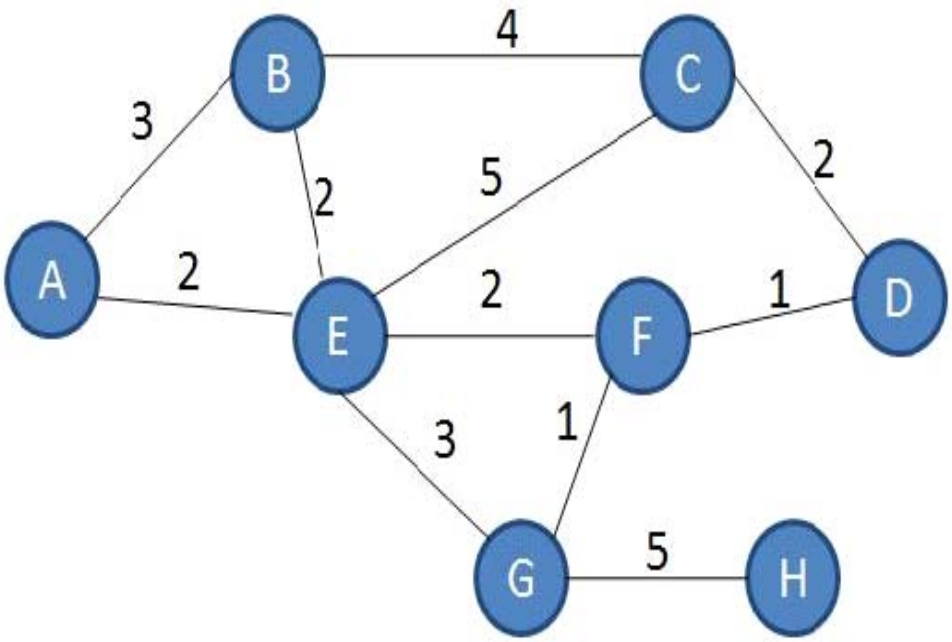

Figure 5.4: Connected graph for distance computation

in hours, for each connected pair of nodes in Figure 5.4 is represented as shown below:

Ground facts

```
%distanceTime(startPoint->?Province1; endPoint->?Province2; bus->?Bus: Real)

 distanceTime(startPoint->A; endPoint->B; bus->3:Real).
 distanceTime(startPoint->A; endPoint->E; bus->2:Real).
 distanceTime(startPoint->B; endPoint->C; bus->4:Real).
 distanceTime(startPoint->B; endPoint->E; bus->2:Real).
 distanceTime(startPoint->E; endPoint->F; bus->2:Real).
 distanceTime(startPoint->C; endPoint->D; bus->2:Real).
 distanceTime(startPoint->C; endPoint->E; bus->5:Real).
 distanceTime(startPoint->D; endPoint->F; bus->1:Real).
 distanceTime(startPoint->E; endPoint->G; bus->3:Real).
 distanceTime(startPoint->F; endPoint->G; bus->1:Real).
 distanceTime(startPoint->G; endPoint->I; bus->5:Real).
```

We obtain the bidirectional routes between locations by applying the symmetric rule on the transitive closure ground facts as shown below:

KB

```
distanceTimeSymmetric(startPoint->?Province1;
                      endPoint->?Province2;
                      bus->?Bus:Real):-
  distanceTime(startPoint->?Province1;
               endPoint->?Province2;
               bus->?Bus:Real).

distanceTimeSymmetric(startPoint->?Province1;
                      endPoint->?Province2;
                      bus->?Bus:Real):-
  distanceTime(startPoint->?Province2;
               endPoint->?Province1;
               bus->?Bus:Real).
```

In order to compute the route and the distance time measured in hours between any two nodes in the graph, we implemented the recursive "distanceTimeRoute" predicate. The "notMember" predicate is used to avoid de-touring in the routes. The "dTR" predicate is the interface predicate, which in turn enforces the "distanceTimeRoute" workhorse predicate to compute the route and bus hours between any two nodes. For efficiency reason, all routes between every pair of nodes are precomputed by using the Top-Down FindAll Solutions architecture.

KB

```
%Interface predicate to enforce the distanceTimeRoute main predicate

 dTR(startPoint->?Province1;
     endPoint->?Province2;
     route->?Route;
     totalTime->?TotalBusHours):-
   distanceTimeRoute(startPoint->?Province1;
                     endPoint->?Province2;
                     [],
                     0:Real;
                     route->?Rest;
                     totalTime->?TotalBusHours).

%Workhorse predicate to compute the routes and bushours for any two provinces

distanceTimeRoute(startPoint->?Province;
                  endPoint->?Province;
                  ?Visited,
```

```
                    ?AccumulateTime:Real;
                    route->[?Province];
                    totalTime->?AccumulateTime:Real).

distanceTimeRoute(startPoint->?Province1;
                    endPoint->?ProvinceX;
                    ?Visited,
                    ?AccumulateTime:Real;
                    route->[?Province1|?Rest];
                    totalTime->?TotalBusHours:Real):-
 distanceTimeSymmetric(?Province1, ?Province2; bus->?Bus!?),
 notMember(?Province2, ?Visited),
 add(?NewBus, ?Bus, ?AccumulateTime:Real),
 distanceTimeRoute(startPoint->?Province2;
                    endPoint->?ProvinceX;
                    [?Province1,?Province2|?Visited],
                    ?NewBus;
                    route->?Rest;
                    totalTime->?TotalBusHours:Real).

%Query: dTR(startPoint->?Province1; endPoint->?Province2;
            route->?Route; totalTime->?TotalBusHours)
```

The dTR predicate was executed in the Top-Down FindAll Solutions with the open query (i.e., free variables for all slot fillers) shown above. Top-Down FindAll Solutions performs an iterative deepening of every connected nodes in the graph, which generates a new set of "dTR" facts. Each dTR fact represents a route and corresponding bus hours between a pair of nodes. For example, the newly generated dTR facts for the start point "A" to the end point "D" are as shown below:

OO jDREW FindAll Solutions Result

```
%Query:dTR(startPoint->A;endPoint->D;route->?Route;totalTime->?TotalBusHours)

%Generated Facts:
 dTR(startPoint->A;endPoint->D;route->[A,B,C,D];totalTime->9.0:Real).
 dTR(startPoint->A;endPoint->D; route->[A,E,F,D];totalTime->5.0:Real).
 dTR(startPoint->A;endPoint->D;route->[A,E,C,D];totalTime->9.0:Real).
 dTR(startPoint->A;endPoint->D;route->[A,B,E,F,D];totalTime->8.0:Real).
 dTR(startPoint->A;endPoint->D;route->[A,B,E,C,D];totalTime->12.0:Real).
 dTR(startPoint->A;endPoint->D;route->[A,E,G,F,D];totalTime->7.0:Real).
 dTR(startPoint->A;endPoint->D;route->[A,E,B,C,D];totalTime->10.0:Real).
 dTR(startPoint->A;endPoint->D;route->[A,B,C,E,F,D];totalTime->15.0:Real).
 dTR(startPoint->A;endPoint->D;route->[A,B,E,G,F,D];totalTime->10.0:Real).
 dTR(startPoint->A;endPoint->D;route->[A,B,C,E,G,F,D];totalTime->17.0:Real).
```

```
routeCount(startPoint->A;endPoint->D;totalCount->10:Integer).
```

There are 10 different routes connecting the start point "A" to the end point "D". The total bus hours for all the routes are given by the slot name "totalTime". These generated dTR facts form the basis of route planning operations. For example, they can be used to answer simple queries such as "find all possible routes and their distance times" between any two provinces. Next, we applied some optimization rules on the dTR facts to solve queries such as to find the shortest path from the set of all possible routes. Our approach to solve this problem is described in the next section.

5.1.2.2 Shortest Path Computation

The transitive closure database executed in the Top-down FindAll Solutions Architecture generates all routes between two specified locations and a count of generated routes. The "dTRShortest" interface predicate calls the recursive predicate "dTRList" to compute the shortest route with the shortest bus hours. For example, the shortest route for start point "A" to end point "D" is 5 hours with route->[A, E, F, D].

Since we have precomputed all routes and bus hours, the OO jDREW reasoning engine can compute solutions to any queries on top of these quickly.

KB

```
%Interface predicate that enforces the workhorse predicate "dTRList"
 dTRShortest(startPoint->?Province1;
             endPoint->?Province2;
             routeList->?Routes;
             shortest->?ShortestRoute):-
   routeCount(startPoint->?Province1;
              endPoint->?Province2;
              count->?Count:Integer),
   dTRList(startPoint->?Province1;
           endPoint->?Province2;
           route->?Routes;
           visited->[];
          %Initializes the current minimum route to a very large bus hours
           currentMinRoute->[R, 10000:Real];
           min->?ShortestRoute;
           count->?Count:Integer).
```

```
%Workhorse recursive predicate to compute the min routes from all routes
 dTRList(startPoint->?Province1;
         endPoint->?Province2;
         route->[];
         visited->?Visited;
         currentMinRoute->?NewMinRoute;
         min->?NewMinRoute;
         count->0:Integer).

 dTRList(startPoint->?Province1;
         endPoint->?Province2;
         route->[[?R1,?T1]|?Rest];
         visited->?Visited;
         currentMinRoute->[?RMin, ?TMin];
         min->?FinalMinRoute;
         count->?Count:Integer):-
  dTR(startPoint->?Province1;
      endPoint->?Province2;
      route->?R1;
      totalTime->?T1),
  notMemberList(?R1,?Visited),
  minRoute([[?R1, ?T1], [?RMin, ?TMin]], ?NewMinRoute),
  subtract(?newCount, ?Count:Integer,  1:Integer),
  dTRList(startPoint->?Province1;
          endPoint->?Province2;
          route->?Rest;
          visited->[?R1|?Visited];
          currentMinRoute->?NewMinRoute;
          min->?FinalMinRoute;
          count->?newCount:Integer).

%notMemberList predicate to avoid duplicate list within a list of lists:
 notMemberList( ?X, []).
 notMemberList( ?X, [?H]) :- naf(equalList(?X, ?H)).
 notMemberList( ?X,[?H|?T]) :- naf(equalList(?X, ?H)), notMemberList(?X,?T).

%Checks if two ground lists are equal
 equalList(?L, ?L).

%Simplified minRoute since the list doesnot contain more than two elements
 minRoute([[?R1,?A],[?R2,?B]], [?R1, ?A]) :- lessThan(?A ,?B).
 minRoute([[?R1,?A],[?R2,?B]], [?R2, ?B]) :- lessThan(?B ,?A).

%Query:
 dTRShortest(startPoint->A;
             endPoint->D;
             routeList->?Routes;
             shortest->?ShortestRoute)
```

The workhorse predicate "dTRShortest" returns the shortest route and its distance time measured in hours. In order to use the entire optimised solution as precomputed facts that can provide all routes and their distance times, along with the shortest route, we have precomputed the shortest routes between every pair of nodes by partially transcribing the above "dTRList" predicate in Java and applying the "minRoute" predicate (cf. Appendix C) in Top-Down FindAll Solutions, which returns the new "dTRShortest" facts for every pair of nodes in the graph. The new set of dTRShortest facts are used as the KB for any route information required for route planning or for distance validation in the main planner's rule system. Route computations are usually implemented using imperative programs such as Dijkstra's Algorithm. However, for the purpose of this thesis, we have tried to keep this computation purely declarative as well. Alternatively, we could use any algorithm from existing Java libraries and load its results as precomputed facts into our KB.

5.2 Rule Subsystem for eTourPlan Subdomains

5.2.1 Rule System for Route Planning

The main function of a route planner is to find a route that meets users' needs such as preferred transportation mode and preferred intermediate provinces for a route. Route planning is considered as a separate subsystem but most of its functionalities are interleaved in the main travel planner. The various options for route planning are discussed in the following subsections.

5.2.1.1 Searching Routes Between Provinces

The eTourPlan prototype can function as a search engine to find all alternative routes between any two provinces specified by a user. Users can make a selection of their preferred transportation mode. This search rule subsystem returns all the routes and the distance time measured in hours for each of the routes, along with a recom-

mendation of the shortest route. The subsystem primarily employs the precomputed "dTRShortest" route and distance time facts to find all possible routes connecting the user-specified start and end province. The advantage of using precomputed facts is the fast retrieval of solutions to the user's query.

In the sample rule shown below, the "getRouteDetails" rule will unify the two arguments, route and bus hour by looking through the "dTRShortest" premises in the KB. The sample query shown below asks for all routes that connect the starting province "Paro", a province in the Western region, to a destination province "Trashigang", in the Eastern region of Bhutan. The routes can be seen on the road map of Bhutan shown in Figure 5.5. The transportation mode used for the distance time computation here is "bus".

KB

```
getRouteDetails(startPoint->?ProvinceA;
                endPoint->?ProvinceB;
                ?Routes,
                ?ShortestRoute):-
  dTRShortest(startPoint->?ProvinceA;
              endPoint->?ProvinceB;
              routeList->?Routes;
              shortest->?ShortestRoute).
```

Sample Query

```
getRouteDetails(startPoint->Paro:Province;
                endPoint->Trashigang:Province;
                ?Routes,
                ?ShortestRoute)
```

OO jDREW TD Result

```
?Routes=
 [[[Paro, Chuzom, Thimphu, Lobesa, WangduePhodrang, Trongsa,
    Bumthang, Mongar, Trashigang], 30.7:Real],
 [[Paro, Chuzom, Thimphu, Lobesa, Punakha, WangduePhodrang,
   Trongsa, Bumthang, Mongar, Trashigang], 31.4:Real],
 [[Paro, Haa, Chuzom, Thimphu, Lobesa, WangduePhodrang, Trongsa,
   Bumthang, Mongar, Trashigang], 36.2:Real],
 [[Paro, Haa, Chuzom, Thimphu, Lobesa, Punakha, WangduePhodrang,
    Trongsa, Bumthang, Mongar, Trashigang], 36.9:Real],
 [[Paro, Chuzom, Thimphu, Lobesa, WangduePhodrang, Damphu, Sarpang,
```

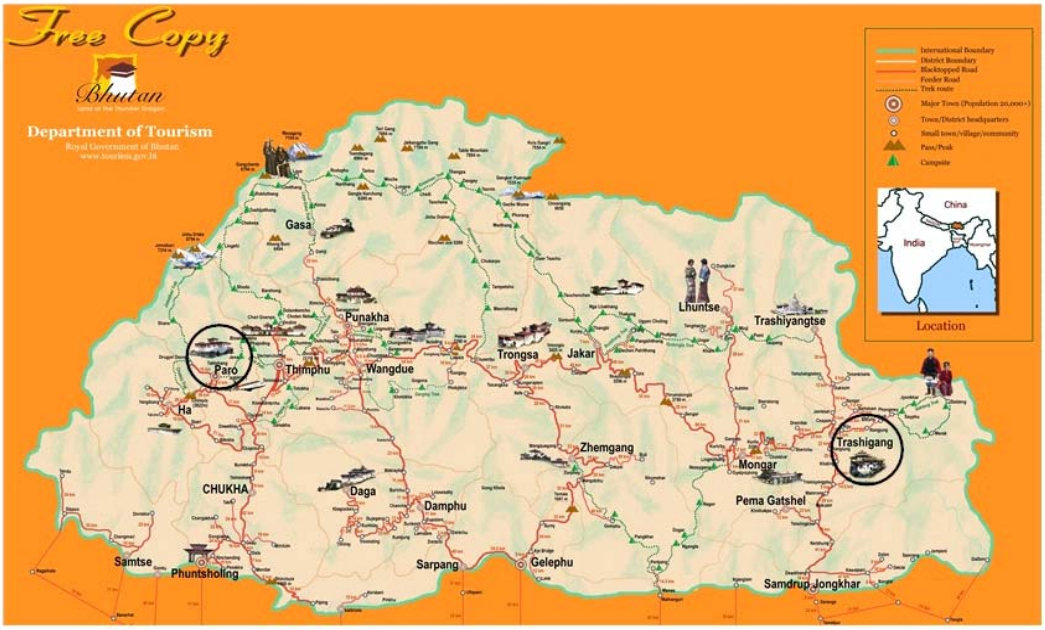

Figure 5.5: Road map of Bhutan (Source:Department of Tourism,Bhutan)

```
  Gelephu, Zhemgang, Trongsa, Bumthang, Mongar, Trashigang], 50.7:Real],
 [[Paro, Chuzom, Thimphu, Lobesa, Punakha, WangduePhodrang, Damphu, Sarpang,
  Gelephu, Zhemgang, Trongsa, Bumthang, Mongar, Trashigang], 51.4:Real],
 [[Paro, Haa, Chuzom, Thimphu, Lobesa, WangduePhodrang, Damphu, Sarpang,
  Gelephu, Zhemgang, Trongsa, Bumthang, Mongar, Trashigang], 56.2:Real],
 [[Paro, Haa, Chuzom, Thimphu, Lobesa, Punakha, WangduePhodrang, Damphu,
  Sarpang, Gelephu, Zhemgang, Trongsa, Bumthang, Mongar, Trashigang],
  56.9:Real]]

?ShortestRoute=
[[Paro, Chuzom, Thimphu, Lobesa, WangduePhodrang,
    Trongsa, Bumthang, Mongar, Trashigang], 30.7:Real]
```

The system returns eight alternative routes connecting "Paro" to "Trashigang", and the shortest route between these two provinces takes 30.7 hours. It takes only 260 milliseconds to retrieve this route information from the KB. This part of our work can also be useful for other organizations, such as road and transportation service providers.

5.2.1.2 System Route Planning based on Province Profiles

Another option offered to our users for route planning is the system's recommendation. We use FOAF-like profiles of provinces described in our KB (cf. Figure 4.1) by chaining through the related provinces to select the next province to visit. Recommendations of locations or touristic provinces are made by referring to the slot filler of the slot name "hs.relatedTo" in the FOAF-like province profiles in the KB. This slot represents the related neighbouring provinces from a touristic point of view. Therefore, we use this concept to provide recommendations of most related provinces and find a touristic route for our users.

Users need to specify the number of provinces, a starting province from the list of FOAF-related provinces and an end province, which can be anywhere in the country. The system finds the best recommended route based on FOAF-like profiles, and it also computes the distance time as it moves from the starting province to the next related province. The resulting route is a chain of related touristic provinces. This predicate is used solely for route planning because we do not yet integrate any recommendation of activities or accommodation facilities in this rule subsystem. This route planner is extended to a location-centric recommender of tourist activities (discussed in Section 5.2.4).

Let us look at the main route planning predicate here. The interface rule "routePlanner" calls the recursive "systemRoutePlanner" predicate (cf. Appendix C) and the "shortestRoute" predicate. The "shortestRoute" predicate finds the shortest route from the last visited province to a user's specified end point. A sample query, which queries the system to recommend 5 provinces to visit, starting at "Paro" and ending at "Gelephu", is shown after representing the route planner rule. The result from our KB is a list of 5 FOAF-related provinces with details of route and distance time for each hop. It also provides the final route to the user's specified end point.

KB

```
routePlanner(typeOfPlan->SystemRecommend;
             startPoint->?Province1;
             endPoint->?ProvinceN;
             routeDetails->[RecommendedRoute->?Route;
                            TotalTime->?TotalBusHours;
                            ReturnRoute->?ReturnRoute];
             numofProvinces->?Num:Integer):-
  systemRoutePlanner(startPoint->?Province1;
                     nextPoint->?Province2;
                     toRoute->?Route;
                     starttime->0:Real;
                     totalTime->?TotalBusHours:Real;
                     numofProvinces->?Num:Integer ),
  shortestRoute(?Province2, ?ProvinceN, ?ReturnRoute).
```

Sample Query

```
routePlanner(typeOfPlan->SystemRecommend;
             startPoint->Paro:Province;
             endPoint->Gelephu:Province;
             routeDetails->?RouteDetails;
             numofProvinces->5:Integer)
```

OO jDREW Result

```
 ?RouteDetails=
  [RecommendedRoute->[[[Paro, Chuzom, Thimphu], 4.5:Real],
                      [[Thimphu, Lobesa, Punakha], 4.2:Real],
                      [[Punakha, WangduePhodrang], 0.7:Real],
                      [[WangduePhodrang, Trongsa], 3.5:Real]];
  ReturnRoute->[[Trongsa,Zhemgang, Gelephu], 8.5:Real];
  TotalTime->12.90:Real]
```

5.2.1.3 Route Planning via User-Preferred Provinces

The system route planner discussed in the previous section has the advantage of finding a complete route guide for users. On the other hand, it has the restriction of providing only for limited preference by users. Therefore, we considered another option for route planning that allows users to include a list of their preferred provinces as intermediate stops between their specified start and end points. The system returns

the route and the distance time in bus hours between each of the provinces, along with the total distance time for the entire trip.

The interface rule "routePlanner" employs the workhorse "userPrefRoutePlanner" predicate (cf. Appendix C), which includes the list of user-preferred provinces as intermediate stops on the route. The rule is followed by a sample query and its output generated by OO jDREW TD:

KB

```
%Interface predicate
routePlanner(typeOfPlan->UserPrefBased;
             startPoint->?Province1;
             endPoint->?Province2;
             userPref->?PrefList;
             routeDetails->[Route->?Route;
                            TotalTime->?TotalBusHours]):-
  userPrefRoutePlanner(startPoint->?Province1;
                       endPoint->?Province2;
                       userPref->?PrefList;
                       route->?Routes;
                       accumulateTime->0:Real;
                       totalTime->?TotalBusHours).
```

Sample Query

```
routePlanner(typeOfPlan->UserPrefBased;
             startPoint->Paro:Province;
             endPoint->Paro:Province;
             userPref->[Thimphu:Province,
                        Punakha:Province,
                        Bumthang:Province,
                        Trashigang:Province];
             routeDetails->?RouteDetails)
```

OO jDREW Result

```
?RouteDetails=
[Route->[[[Paro, Chuzom, Thimphu], 4.5:Real],
           [[Thimphu, Lobesa, Punakha], 4.2:Real],
           [[Punakha, WangduePhodrang, Trongsa, Bumthang], 8.7:Real],
           [[Bumthang, Mongar, Trashigang], 14.0:Real],
            [Trashigang, Mongar, Bumthang, Trongsa, WangduePhodrang,
 Lobesa, Thimphu, Chuzom, Paro], 30.7:Real];
  TotalTime->62.10:Real]
```

The route planning subsystem returns routes connecting all the provinces in the user's preference list, starting at the specified start point and ending at the specified end point. It also computes the total bus hours for the entire trip.

5.2.2 Rule System for Activity and Accommodation Search

"Activity", in the context of our KB, subsumes both the events and attractions subdomains. In this section, we will discuss rules that provide our users with different options to get tourist information from the KB. The parametric search operations that can be performed for any of the tourist entities are explained.

5.2.2.1 Parametric Search for Activity Details

The activity subdomain of our KB consists of well-structured profiles for activities found in a province. This aspect of our KB contributes to providing detailed search results to our users. The first option offered to users is the ability to find the key information concerning any activity by specifying the activity name. For example, for users who are seeking to learn more about certain festivals or attraction sites, our system can provide all the necessary information from the KB. The main predicate "getActivityDetails" (cf. Appendix C) provides parametric search of activities (i.e., events and attractions). Users can query this predicate to get the details of a specified activity or to search activities by type or theme in some part of the country. A sample query to get activity details by name and its result are shown below:

Sample Query

```
getActivityDetails(actName->Paro_Tshechu:Events; ?ActivityDetails!?)
%Remark: "!?" is used to ignore irrelevant slots in the query.
```

OO jDREW TD Result

```
?ActivityDetails=
  [ActName->Paro_Tshechu:Events;
    Theme->Cultural_Religious_Heritage;
    EventDates->[StartDate->date[2008:Real, 03:Real, 17:Real];
```

```
            EndDate->date[2008:Real, 03:Real, 21:Real]];
WebLink->" ";
Description->"Paro Tshechu (Festival) is one of most important
festival of Bhutan which starts in spring. The most magnificient
thing about it is the unraveling of a large Thanka (Thongdrol).";
Address->[Hungrel:Village,
          Hungrel:Block,
          Paro:Province,
          Western:Region,
          Bhutan:Country];
RelatedTo->"Ura_Yakchoe:Annual_festival"]
```

Activities has two top-level types, attractions and events. Users can provide their preferred type of activity at the top-level or be more specific by specifying a subclass of one of the two main types. For example, a user might want to search only for events of type "Annual_festival" or attractions of type "Historical_buildings". These type classifications of tourist entities are according to the standards followed by the Harmonise ontology discussed earlier (in Section 2.3). Activities in our KB are categorized into three main themes: Cultural_Religious_Heritage, Nature, and Recreation. This categorization is officially used by the Department of Tourism of Bhutan. Our users can perform top-level searches of activities by their preferred theme. Since we have object-centric profiles for each of the activity entities, the system can provide detailed activity information according to the user's specified theme.

In addition to the above two parameters, users can be more specific about the search area (i.e., Country, Region, Province, Block, Subblock) from the administrative partonomy of a country discussed earlier (in Section 5.1). For example, users can search for activities from a very specific search space of subblocks or broaden the search space, such as searching in a province or within a country as a whole.

Therefore, the system provides flexibility of search space for users' queries. The complete predicate definition is shown in Appendix C. A sample query shown below queries for activities of type "Annual_festival", in the block "Chhoekhor:Block". The system will return all the activities that meet the user's specified activity type in a given search area. The first solution to this query is shown below (further solutions

would include detailed information of rest of the activities of type "Annual_festival" found in the block Chhoekhor).

Sample Query

```
getActivityDetails(actName->?Name:Annual_festival;
                   theme->?Theme;
                   address->[?Subblock,
                             Chhoekhor:Block,
                             ?Province,
                             ?Region,
                             ?Country];
                  ?ActivityDetails)
```

OO jDREW TD Result

```
?Country= Bhutan:Country
?Region= Central:Region
?Block= Chhoekhor:Block
?Subblock= Tharpaling:Village

?Name:Annual_festival=
  ActName->Jambay_Lhakhang_Drup:Annual_festival
?ActivityDetails=
  [ActName->Jambay_Lhakhang_Drup:Annual_festival;
   Description->"One of the most amazing festivals in Bumthang";
   Theme->Cultural_Religious_Heritage;
   Location->Jambay_Lhakhang:Temple;
   WebLink->" ";
   EventDates->[StartDate->date[2008:Real, 11:Real, 12:Real];
                EndDate->date[2008:Real, 11:Real, 16:Real]];
   RelatedTo->Prakhar_Duchhoed:Annual_festival]
```

Solely searching for activity information at a specific level of the hierarchy of a country requires the user to instantiate that part of the country. In this activity-search rule subsystem, we integrated the "getFullAddress" predicate discussed earlier (in Section 5.1.1) to get the full address of each of the activity locations.

5.2.2.2 Parametric Search for Accommodation Details

Accommodation is one of five primary subdomains of the tourism domain. Facilities to search and recommend accommodations provide daily 'anchor points' in any tourism search engine or travel planner. Equally important is how the system

can provide both general information and options to filter the search according to user-specified preferences.

The accommodation subdomain, like the activity subdomain, consists of well-structured profiles for each of the accommodations found in each of the provinces. We implemented simple yet useful rules to help our users in retrieving accommodation information from these profiles. Users can search accommodations with options such as searching accommodations by type or rates (i.e., price range) in a specific area from the administrative partonomy of a country. Users can get information about a particular accommodation by specifying the name of the accommodation. The predicate "getAccommodationDetails" is available in Appendix C. A sample query where the user queries by name is shown below, followed by its solution:

Sample Query

```
getAccommodationDetails(accName->Kaila:Guest_house; ?AccommodationDetails!?)
```

OO jDREW TD Result

```
?AccommotionDetails=
  [WebLink->"http://www.kaila.bt/";
   Address->[Chamkhar:Town,
             Chhoekhor:Block,
             Bumthang:Province,
             Central:Region,
             Bhutan:Country];
   Name->Kaila:Guest_house;
   Standard->[StarRating->2:Real;
              MinPrice->400:Real];
   ContactDetails->[LandLine->9753631219;
                    CellLine->97517682948];
                    Email->"manager@kaila.bt"];
   RelatedTo->Yangphel:Guest_house]
```

The main classification under the accommodation class are Resort, Hotel, Lodge, Guest_house, and Apartment (in Section 4.2.3). Users can search accommodations by specifying their preferred type of accommodation within a specific area from the partonomic division of a country. Based on these properties, the search result provides detailed information on each accommodation found to meet a user's specification.

The example query shown below queries for accommodations of type "Guest_house" in the Province "Bumthang:Province" with the maximum price affordable set to 800 Ngultrum (Bhutanese currency).

Sample Query

```
getAccommodationDetails(accName->?Name:Guest_house;
                        address->[?Subblock,
                                  ?Block,
                                  Bumthang:Province,
                                  ?Region,
                                  ?Country];
                        setMaxPrice->[Yes, 800:Real];
                        ?AccommodationDetails)
```

OO jDREW TD Result

```
?Name:Guest_house=
  accName->Kaila:Guest_house
?AccommodationDetails=
  [WebLink->"http://www.kaila.bt/";
   Address->[Chamkhar:Town,
             Chhoekhor:Block,
             Bumthang:Province,
             Central:Region,
             Bhutan:Country];
   Name->Kaila:Guest_house;
   Standard->[StarRating->2:Real;
              MinPrice->400:Real];
   ContactDetails->[LandLine->9753631219;
                    CellLine->97517682948];
                    Email->"manager@kaila.bt"];
   RelatedTo->Yangphel:Guest_house]
?Country= Bhutan:Country
?Subblock= Chamkhar:Town
?Block= Chhoekhor:Block
?Region= Central:Region
```

The system returns two accommodations to this query specification, which are of type "Guest_house", located in Bumthang province and do not cost more than the user's maximum affordable price. Users can narrow or broaden the search space either by specifying the lowest level of partonomic division (i.e., subblock) or by specifying none, which makes the search cover the entire Country.

5.2.2.3 Search "N" Activities at a Specific Province

In order to search for a number of "N" activities at a specific province, we enforce a recursive predicate to perform such an operation. Readers are by now aware that activity subsumes both the attraction and event classes, so the recursive "getActivityList" predicate will collect a blend of events and attractions. The occurrence of duplicates are avoided by the "notMember" auxiliary predicate (see Auxiliary predicates in Appendix D). Enhanced versions of this predicate for specifically collecting all attractions ("getAllAttractions"), all events ("getAllEvents"), and all accommodations ("getAllAccommodations") of the province and subblock levels are shown in Appendix D. These find-all predicates are later integrated to our planner to recommend tourist entities at each of the event-occurring subblocks.

KB

```
getActivityList(?Province, [], ?Visited, 0:Integer).

getActivityList(?Province, [?Name|?Rest], ?Visited, ?NumActivities:Integer):-
  greaterThan(?NumActivities:Integer, 0:Integer),
  activity(?Name^hs.description->?Description;
                hs.url->?Url;
                et.theme->?Theme;
                hs.relatedTo->?RelatedTo;
                et.province->?Province!?),
  notMember(?Name, ?Visited),
  subtract(?Numtleft, ?NumActivities:Integer,  1:Integer),
  getActivityList(?Province,  ?Rest,  [?Name|?Visited], ?Numtleft).
```

Sample Query

```
 getActivityList(Bumthang:Province,  ?ActivityList, [],  6:Integer)
```

OO jDREW TD Result

```
?ActivityList=
  activityList->[Tamshingphala_Choepa:Traditional_festival,
                Tangbi_Mani:Traditional_festival,
                Kurjey_Tshechu:Annual_festival,
                Tamshing_Lhakhang:Temple,
                Petseling_Gompa:Temple,
                Kharchu_Monastery:Monastery]
```

5.2.3 Rule System for Location-Centric Travel Recommender

We will now look at how we integrated the aspects of the aforementioned rule subsystems into our location-centric recommender system. Users can get recommendations for the three main tourism subdomains: activity, transportation, and accommodation. The location-centric travel recommender offers two modes of operations. It either provides tourist information for user-specified list of provinces or provides a recommended list of provinces with some tourist information. These two types are distinguished by the first slot name "typeOfRecommend", which is to be filled with either "UserPrefBased" or "SystemRecommendation". The two main operations are discussed in the following subsections.

5.2.3.1 Location-Centric Tour (Via User-Preferred Provinces)

The "locCentricRecommend" interface rule, shown below enforces the recursive workhorse predicate "userPrefLocCentricRecommend" to accumulate events, attractions and routes for a list of provinces selected by the user. It also computes the total travel time of the trip measured in bus hours. This rule subsystem calls "getAllAttractions", "getAllEvents" and "getAllAccommodations" predicates, which are special cases of the "getAllActivity" predicate. Other options of our users are that they can provide a preferred activity theme and set the maximum price affordable for accommodations. The "locCentricRecommend" interface predicate is shown below and the complete "userPrefLocCentricRecommend" predicate is given in Appendix A:

KB

```
locCentricRecommend(typeOfRecommend->UserPrefBased;
                    userInputs->[startPoint->?P1;
                                 userPref->?PrefList;
                                 endPoint->?P2];
                    [?Route,
                     ?Recommendations,
                     ?TotalBusHours:Real]):-
  userPrefLocCentricRecommend(startPoint->?P1;
                              endPoint->?P2;
```

```
                                userPref->?PrefList;
                                route->?Route;
                                recommendations->?Recommendations;
                                accumulateTime->0:Real;
                                totalTime->?TotalBusHours:Real).
```

Location-centric travel recommendation is not concerned with, nor constrained by, dates. It can recommend events, attractions, accommodations, and routes for a list of user-specified provinces. It also computes the total travel time for the entire travel.

Sample Query

```
locCentricRecommend(typeOfRecommend->UserPrefBased;
                    userInputs->[startPoint->Paro:Province;
                                 userPrefList->[Paro:Province];
                                 endPoint->?Thimphu:Province];
                    [?Routes, ?Recommendations, ?TotalBusHours])
```

OO jDREW TD Result

```
?Routes=
  [[Paro:Province], 0:Real,
   [Paro, Chuzom, Thimphu], 4.5:Real]

?TotalBusHours=
  4.5:Real

?Recommendations=
  [[Paro:Province;
    EventList->[
      Paro_Tshechu:Annual_festival;
      EventDates->[StartDate->date[2008:Real, 03:Real, 17:Real];
      EndDate->date[2008:Real, 03:Real, 21:Real]];
      Description->"Paro Tshechu (Festival) is one of most important festi-
      val of Bhutan which is in the start of spring. The most magnificient
      of all at Paro festival is the unraveling of a large painting."];
    AttractionList->[
      Rinpung_Dzong:Fortress;
      Theme->Cultural_Religious_Heritage;
      Description->"It is one of the most beautiful attractions in the
                    Southern region"];
    Accommodations->[
      [Aman_Resort:Resort;
```

```
Url->"http://www.AmanBhutan.bt";Rating->5:Real;MinPrice->4500:Real],
Tiger_Nest:Resort;
Url->"http://www.TigerNest.bt";Rating->3:Real;MinPrice->2500:Real],
[Rangen_Resort:Resort;
 Url->"http://www.Rangen.bt";Rating->2:Real;MinPrice->1000:Real]]]]
```

5.2.3.2 Location-Centric Tour (System-Recommended)

This functionality of the system is an enhancement of the "systemRoutePlanner" rule subsystem, discussed earlier (in subsection 5.2.1.3). We primarily employ FOAF-relation between province profiles to recommend a tour route for travelling. We have extended the "systemRoutePlanner" rule subsystem to the "systemLocRecommender" rule subsystem. This predicate defines routes through the related provinces from their FOAF-like profiles and collects tourist information such as the number of tourist entities at each of the provinces. It requires the user to input the number of provinces that he or she wants in the output.

The interface rule "LocCentricRecommend", followed by a sample query, and its result are shown below. The complete rule subsystem is given in Appendix A.

KB

```
locCentricRecommend(typeOfRecommend->SystemRecommendation;
                    userInputs->[startPoint->?Province;
                                 numProvinces->?NumProvinces:Integer];
                       [Routes->?Routes;
                        ProvinceInfo->?Recommendations;
                        TotalBusHours->?TotalBusHours:Real]):-
  systemLocRecommender(startPoint->?Province;
                       nextPoint->?Province2;
                       route->?Routes;
                       recommendations->?Recommendations;
                       starttime->0:Real;
                       totalTime->?TotalBusHours: Real;
                       numProvinces->?NumProvinces:Integer).
```

The recursive"systemLocRecommender" predicate chains along the system- recommended provinces, collecting tourist information at each hop. As shown in the sample query, the user has to input a starting point and number of provinces to visit.

Sample Query

```
locCentricRecommend(typeOfRecommend->SystemRecommendation;
                    userInputs->[startPoint->Paro:Province;
                                 numProvinces->1:Integer];
                    [Routes->?Routes;
                     ProvinceInfo->?Recommendations;
                     TotalBusHours->?TotalBusHours:Real])
```

OO jDREW TD Result

```
?Routes=
  [Paro, Chuzom, Thimphu], 4.5],
?Recommendations=
  [[Paro:Province;
      WebLink->"http://www.paro.gov.bt/";
      TouristInfo->
        NumAttractions->3:Integer;
        NumEvents->1:Integer;
        NumAccommodations->3:Integer]],
?TotalBusHours=
  4.5:Real
```

5.3 Rule System of the eTourPlan Travel Planner

Referring back to the dimensions of travel planning discussed earlier (in Section 2.4), we have implemented the three aspects of complete travel planning: attraction-only, events-only, and event-centric with attraction recommendations. Based on the domain of interest, we have categorized them into two main planning criterions. The two main criteria for travel planning considered in this thesis are a purely geographic and a temporal-geographic search criterion.

The first approach of the two is most suitable for planning a travel that is not constrained by time. For example, planning a solely attraction-based or attraction-centric travel would be more benefited by using this criterion, where users' key measure of picking attractions is the physical distance between them. In this way, the arrangement of the attractions into a travel package would be on the basis of saving travel time between attraction sites. Bearing in mind this selection criterion, we do not want

our users to visit only on the basis of neighbouring attractions in a particular province. Therefore, in order to provide a good package of popular attractions, we bring in the use of relations between FOAF-like profiles of attractions that we have captured in our KB. The sequence of chaining that we have created among popular attractions, trying to minimize distance, therefore becomes the key element for selection. By using this aspect of tour planning for attractions, the planner will not miss any popular attractions within reach from the user-specified start point.

Our second approach to planning is by way of a temporal-geographic search criterion, where we have multiple temporal constraints that must be met before making any selection. The system is capable of executing conjunctive queries in order to validate each of the temporal constraints over a given time frame. Precomputed routes and distance times between provinces (from the dTRShortest facts in the KB) are used to significantly reduce the planning time for locating temporal events. We found that this search approach is primarily suitable for our second and third aspects of complete planning, event-centric planning. From the profile description of events and attractions in Chapter 4, the main property that differentiates an event from an attraction is the time or date of the event. Events are (more) temporal and therefore, in order to plan events, it is crucial to plan accordingly with respect to their occurrence. Second to the time/event dates measure are the distances between the event locations. With these constraints in mind, we allow users to specify an upper bound (i.e., maxBreak between events) and a lower bound (i.e., minBreak between events). Considering the dates and the distances between event locations as the two main constraints, a temporal-geographic search criterion should first find events between minBreak and maxBreak from the current time, and then find a route (from the precomputed routes and distance time facts) to the next province of the event location within reach, and recursively find more events from that time and place. We will now look at each of attraction-only planning (i.e., purely geographic planning) and event-centric or event-only planning (i.e., temporal-geographic search planning) in the following subsections.

5.3.1 eTourPlan Attraction-Only Planning

The attraction-only planning is based on chaining attractions which are connected to one another through the "hs.related" slot name in the attraction profiles. This concept of FOAF, derived from social networking, makes good sense in forming a chain of attractions linked together on the basis of distances between them. With the prior knowledge that popular attractions in any province are linked together, the system can provide a good plan of visiting attractions starting at any province of a user's choice. The system also keeps a record of the actual total travel time and validates that the actual total travel time does not exceed the user's specified travel time. The travel time also includes the time spent at each attraction site, which is derived from a global constant "maxHoursAtAnAttractionSite" (set as four hours in this thesis). The flowchart in Figure 5.6 illustrates the top-level attraction-only planning strategy. The complete attraction planning rule system is given in Appendix A.

Users must input the following slot fillers:

- ?TypeOfPlanning: AttractionOnly
- ?StartPoint: Starting province
- ?NumAttractions: Number of attractions to be included in the plan
- ?UserTotalTravelTimeInDays: Users total travel time in days

The system outputs the following slot fillers:

- ?AttractionList: List of attractions to visit
- ?TotalActivityTimeInHours: Actual total time of the tour

The planner performs the following steps in sequence:

- From the user-specified starting point, an attraction is selected and chains to the next related attraction.

- Compute and validate route and travel-time constrained by global constants:

eTourPlan Attraction-only

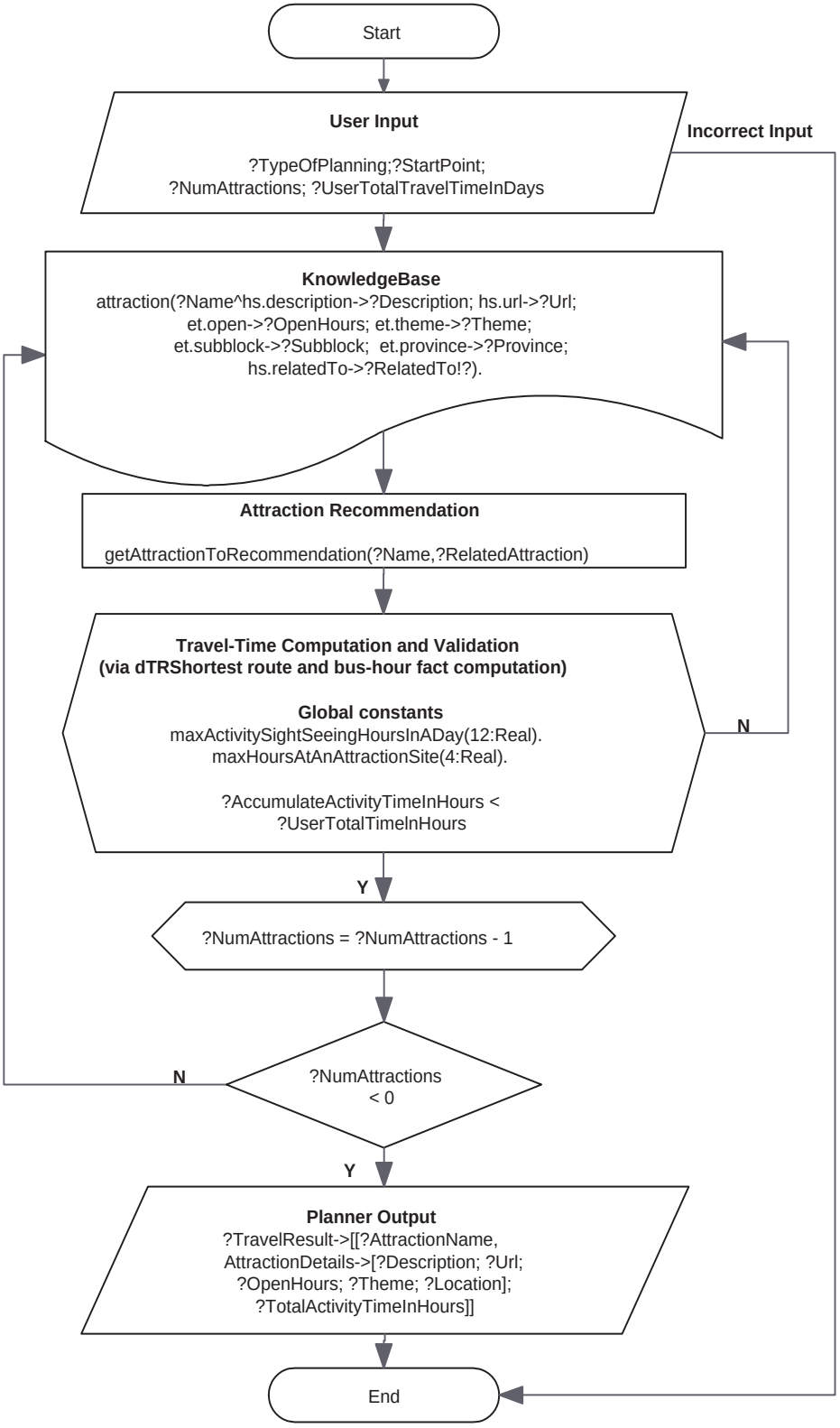

Figure 5.6: The top-level of attraction-only planning

  - maxActivitySightSeeingHoursInADay(12:Real).
  - maxHoursAtAnAttractionSite(4:Real).

- On successful validation of distance and remaining time, add detailed information of the selected attraction to the travel plan.

A sample query of the top-level eTourPlan attraction-only rule system is shown below:

Sample Query

```
eTourPlan(typeOfPlanning->AttractionOnly;
          userInputs->[startPoint->Bumthang:Province;
                       numAttractions->4:Integer;
                       userTotalTravelTimeInDays->3:Real];
          travelResult->?TravelPlan)
```

The solution for the above eTourPlan query:

OO jDREW TD Result

```
?TravelPlan=
[[[Bumthang_Dzong:Fortress;
       AttractionDetails->[
          Description->"The Bumthang Dzong or the Dzong of the white bird.
                       It is perched on the hillock over looking Chamkhar
                       town & places surrounding it. The interesting thing
                       about the Dzong is, that there is a water tower four
                       stairs down behind the Dzong.";
          Url->" ";
          OpenHours->9amTo4pm;
          Theme->Cultural_Religious_Heritage;
          Location->[Chamkhar:Town, Bumthang:Province]]],

[Samchholing_Palace:Popular_architecture;
       AttractionDetails->[
          Description->"Samchholing was built by the Second King. It was
                        later handed over to Ashi Pem Dechen, mother of
                        HRH Namgyel Wangchuck. It is about to be in ruins,
                        but it has a beautiful architecture.";
          Url->" ";
          OpenHours->9amTo4pm;
          Theme->Cultural_Religious_Heritage;
          Location->[Samcholing:Village, Trongsa:Province]]],

 [Trongsa_Dzong:Fortress;
        AttractionDetails->[
```

```
          Description->"It was built by Zhabdrung Rimpochhe. It is one of
                        the biggest fortress in Bhutan";
          Url->" ";
          OpenHours->9amTo4pm;
          Theme->Cultural_Religious_Heritage;
          Location->[Samcholing:Village, Trongsa:Province]]],

Wangdue_Dzong:Fortress;
  AttractionDetails->[
     Description->"It is one of the most beautiful attractions in the
                   Southern region";
     Url->" ";
     OpenHours->9amTo4pm;
     Theme->Cultural_Religious_Heritage;
         Location->[Wangdue_Town:Town, WangduePhodrang:Province]]];

TotalActivityTimeInHours->26.0:Real]
```

5.3.2 eTourPlan Event-Centric Planning

The event-centric planning takes into account the time constraint of event dates, in accordance to the user's preferred time frame for a vacation. We perform the selection of events by date validation, followed by the validation of distance between the event locations. The planner recommends attractions that are located in the subblock of each selected event. An additional option for on-route attraction recommendation is also provided. In a similar manner, we could integrate the accommodation search predicate discussed in Section 5.2.3.3, to provide accommodation recommendation in each of the event-containing subblocks.

The eTourplan event-centric planner considers the following needs and constraints of users:

- **Preferences**: Users would have their preference of time for making a trip. For instance, a tourist might want to take a month's vacation in February. Users can provide preferences for the attraction theme and type of accommodation to be recommended.

- **Constraints**: Maximum break between events: This is considered as the upper

eTourPlan Event-Centric

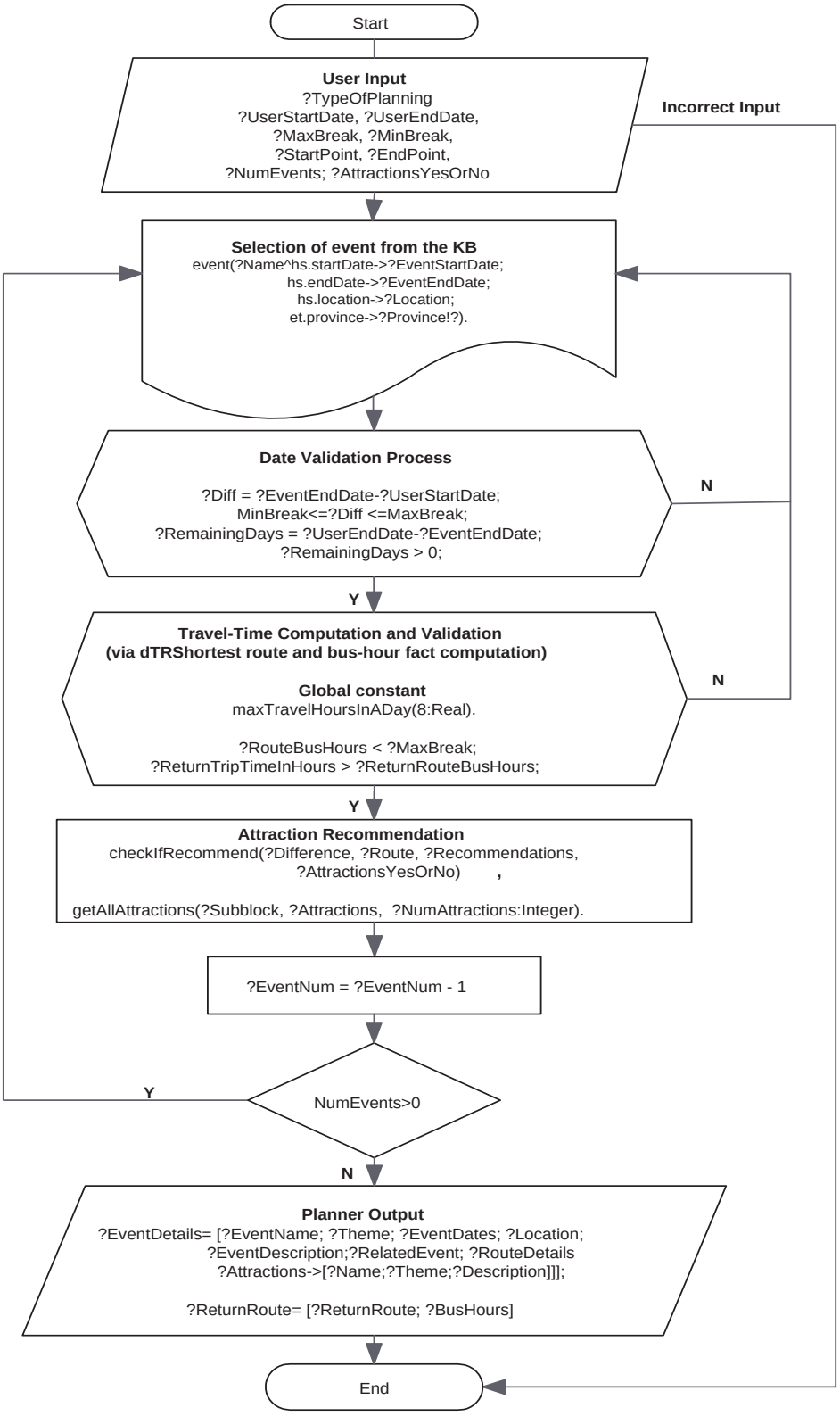

Figure 5.7: The top-level of event-centric planning

bound on the break (time) between two events. This is also considered as the maximum travel time between two event locations for a particular user in the context of this thesis. The second constraint is the minimum break, which is the lower bound on the break between events. Many users would not mind experiencing two events on two consecutive days but we are still allowing users to specify the minimum break between events for flexibility.

The event-centric planner generates the travel plan by selecting a specified number of N events between the user's specified travel dates. The rule system mainly validates the time and distance of each event. The flowchart in Figure 5.7 illustrates the top-level event planning strategy. The complete system is given in Appendix A.

Users must input the following slots:

- ?UserStartDate: User's preferred start date of travel
- ?UserEndDate: User's preferred end date of travel
- ?StartProvince: Start point of the travel
- ?EndProvince: End point of the travel
- ?MaxBreak: Maximum break between events
- ?MinBreak: Minimum break between events
- ?NumEvents: Number of events to be included in the plan
- ?AttractionYesOrNo: User selects the optional on-route attraction recommendation

The system outputs the following slots:

- ?EventDetails: Detail information for each selected event with route information
- ?ReturnRoute: Route and distance time to the specifed EndProvince

The planner performs the following steps in sequence:

- Events are selected by validating the event dates against the users travel dates, minimum break, and maximum break.

- Compute and validate route and bus hours between event locations constrained by the specified maximum break.
- On successful validation of distance and remaining time, add detailed information of the selected event to the travel plan.
- Recommend attractions located in the subblock of the selected event.
- Provide on-route attraction recommendation if the user selects the option (constrained by the global constant maxTimeGapBetweenEvents").

A sample query for the top-level event-centric eTourPlan rule system is shown below. The planner generates an event tour for a given time frame based on the needs and constraints provided by the user.

Sample Query

```
eTourPlan(typeOfPlanning->EventCentric;
          userInputs->[userStartDate->date[2008:Real, 10:Real, 01:Real];
                       userEndDate->date[2008:Real, 11:Real, 10:Real];
                       maxBreak->11:Real;
                       minBreak->0:Real;
                       startPoint->Paro:Province;
                       endPoint->Thimphu:Province;
                       attractionRecommendation->Yes;
                       eventNum->1:Integer];
          ?TravelResult)
```

A sample solution for the above eTourPlan query is shown below:

OO jDREW TD Result

```
?TravelResult=
[[[EventName->Thimphu_Tshechu:Annual_festival;
     EventDates->[Startdate->date[2008:Real, 10:Real, 09:Real];
                  Enddate->date[2008:Real, 10:Real, 11:Real]];
     Theme->Cultural_Religious_Heritage;
     Location->[Tashichoe_Dzong:Fortress,
                Jongshina:Town,
                Thimphu:Province];
     EventDescription->"It is a popular festival in Thimphu";
     RelatedEvents->Tangbi_Mani:Traditional_festival;
```

```
    RouteDetails->[
     [Paro:Province, Chuzom:Province, Thimphu:Province];
      RouteBusHours->4.5:Real;
      RecommendedAttractions->[
       [Paro:Province;
         Attraction->[AttractionName->Rinpung_Dzong:Fortress;
                      RelatedTo->Taktshang:Monastery;
                      Theme->Cultural_Religious_Heritage;
                      Subblock->Phatsana:Village]],

       [Chuzom:Province;
         Attraction->[]],
       [Thimphu:Province;
         Attraction->[AttractionName->Tashichoe_Dzong:Fortress;
                      RelatedTo->"Memorial_Chorten:Temple";
                      Theme->Cultural_Religious_Heritage;
                      Subblock->Jongshina:Town]]]]];
ReturnRoute->[[Thimphu:Province]; Returntime->0:Real]]
```

The system finds three events from the KB, one at a time, each one of which is validated w.r.t. time and distance. The planner offers three different events occurring at different dates between the user-specified travel dates. The planner also provides on-route attraction recommendations if the time gap between the event dates is more than five days (global constant "maxTimeGapBetweenEvents"). Setting "eventNum" to 3 in the above query will go travel plans consisting of alternative combinations of three events with respect to thier times and distances. The complete eTourPlan planning rule system is given in Appendix A.

Attraction recommendations can be done at any level of the partonomy. In this thesis, attractions can be recommended both at the province and the subblock level, but we have tested the prototype with attraction recommendations at the subblock level to obtain maximum granularity. Similarly, accommodation recommendation at any level of the partonomy can be integrated as another optional feature into the planner. The integration of recommendations for routes, attractions, and accommodations into the event-centric planning amounts to packaging a complete travel for users.

Chapter 6

eTourPlan And Its Evaluation on the Bhutan KB

In this chapter, we describe the functionalities of our eTourPlan prototype considering different scenarios. We also demonstrate some experimental results of the key operations. Our illustrations of the operations are based on Bhutan KB.

Bhutan is a popular tourist destination in Asia. Our Bhutan KB, described in Chapter 4, consists of profiles for 10 most popular provinces of 20 total in Bhutan, detailing a total of 23 attractions, 18 events, and 18 accommodations in these 10 provinces. The profile descriptions are based on real information, although we did not include each and every attractions and accommodations located in all the provinces. On top of the fact base on Bhutan, we have the rule subsystems described in Chapter 5, enriching the KB. We show how eTourPlan KB can be used to fulfill the need for retrieving precise tourist information of tourist entities in order to package a tour based on tourist's preferences. We also show how some of the rule subsystems can function as independent modules for various purposes. For example, the administrative partonomy of a country and the route planning module can be used as separate systems, not only for tourist services but also as a general knowledge portal for Bhutan.

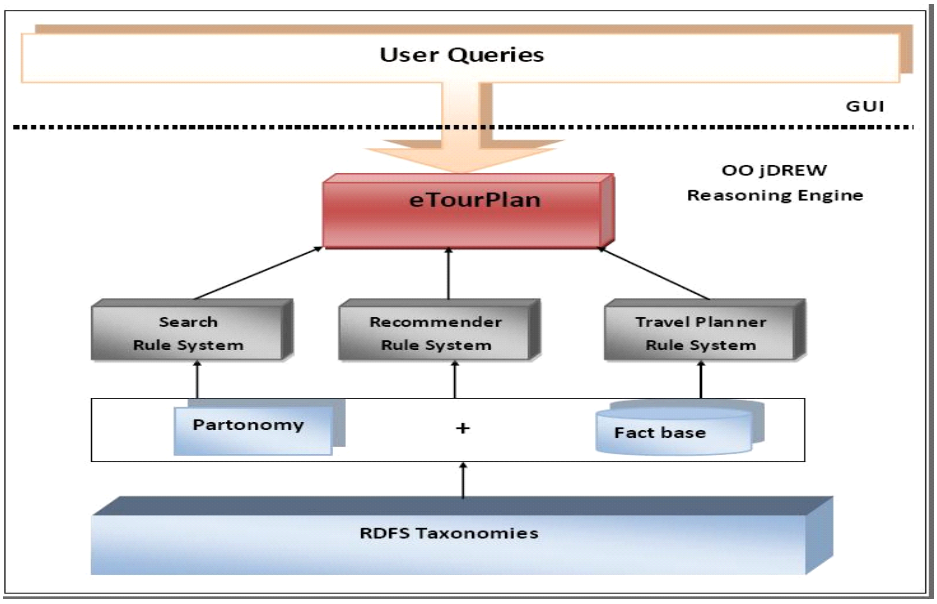

Figure 6.1: eTourPlan Architecture

The eTourPlan architecture is shown in Figure 6.1. The system is built on top of the rule engine, OO jDREW. This thesis is focused on knowledge acquisition and the application of rules on KB. However, future work of designing a convenient Graphical User Interface would help to emphasize the usefulness of the KB. We illustrate all the operations as tested using different queries on OO jDREW TD. First, the user posts a query to our eTourPlan system. The eTourPlan system then automatically applies the corresponding rules to the related tourism subdomain facts. After processing facts with rules, the result of the eTourPlan rule system may turn out be either a success or a failure. When the query is fullfiled by the system, eTourPlan shows the result bindings matched to the user's preferences. Otherwise, eTourPlan notifies the user that there is "No Solution".

6.1 Key Operations of the Prototype

eTourplan prototype has three top-level categories of operations as shown in Figure 5.1. The operations are as follows:

- Search for information concerning tourist entities: provinces, attractions, events, and accommodation based on some preferences.
- Search route information between any two provinces.
- Provide travel route tailored to visiting user-preferred provinces
- Recommend touristic route of provinces
- Recommend a location-centric tour for a user-preferred provinces
- Generate an attraction-only travel plan
- Generate events-only or event-centric with attraction recommendation travel plan

6.2 OO jDREW TD User Interface

The eTourPlan system is executed using the built-in OO jDREW TD interface (cf. the screen shot in Figure 6.2). Brief directions for using the interface are given below:

1. Load the RDFS type definition file (by hand, from file) in the Type window (tab)
2. Load the KB in the Knowledge Base window (tab)
3. Post your queries in the uppermost box in the Query Tab
4. Click "Issue Query" button to execute the eTourPlan system.
5. After the eTourPlan computation is complete, result bindings will be shown on the righthand side of the interface. We can also trace how the results are bound in the result tree on the lefthand side of the interface.

6. However, if eTourPlan system fails to find matching result bindings to the user's query, the lefthand side box for showing traces will remain "No Solution". The box showing the result bindings will remain blank.

6.3 Experimental Results under Typical Operations

In this section, we discuss how the eTourPlan system is tested under typical operations using varying types of different queries. Each of the subdomains are tested as separate module at first, and then together as a whole system, in order to verify the system operations. The following examples are used to demonstrate the typical operations in a bottom up fashion.

6.3.1 Search operations

Our KB consists of facts about the five main subdomains of tourism: events, attractions, accommodations, transportation, and geographic regions. This KB can be used for semantic information search on the tourism subdomains. For example, users can query for the details about any specific province, events, attractions, and accommodation from the KB. The search engine returns the result of the query based on user's preferences such as name or theme pertaining to a specific part of the country. One particularly good use of the search engine is for retrieving information concerning tourist entities pertaining to a specific subpart of the country. The search engine also provides route information between any two provinces. In this section, we describe some of typical search queries that can be issued against the different subdomains:

6.3.1.1 Search for Provinces

The first query in Table 6.1 returns a unique solution as it queries for provincial details, by specifying the name of the province. The system binds the detailed infor-

Table 6.1: Queries of different input/output modes for Province search

Query	**User Input Values**	**Query Formats** (Input values are bold-faced)
1	?Name	getProvinceDetails(region->?Region:Region; name->**Bumthang**:Province; ?ProvinceDetails)
2	?Region	getProvinceDetails(region->**Central**:Region; name->?Name:Province; ?ProvinceDetails)
3	None	getProvinceDetails(region->?Region:Region; name->?Name:Province; ?ProvinceDetails)

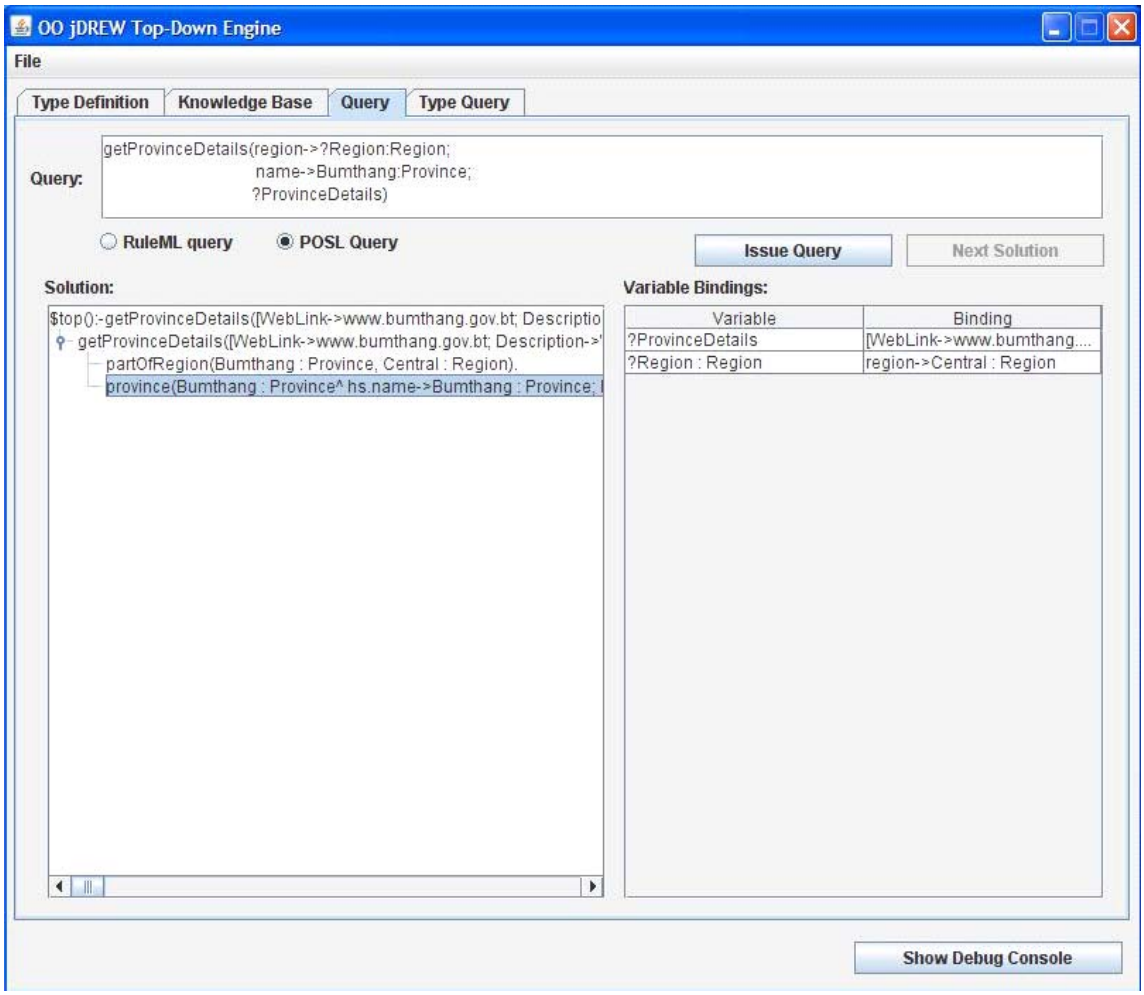

Figure 6.2: Screenshot for Query 1 result

mation of the unifying province profile to the free variable "?ProvinceDetails". The result for the query in OO jDREW TD is shown in Table 6.2 (cf. the screen shot in Figure 6.2).

Table 6.2: Province search result of Query 1

Output Variables	**Variable Bindings**
?ProvinceDetails	[WebLink->"http://www.bumthang.gov.bt/"; Description->"Bumthang is one of the most attractive touristic province with several festivals throughout the year"; Capital->Chamkhar:Town; Geography->[Area->"1,819 sq.km"; Elevation->"1,300 to 7300 meters"]; TouristInfo->[NumAttractions->16:Integer; NumEvents->13:Integer; NumAccommodations->10 :Integer]; Contact->"admbumthang@druknet.bt"]
?Region:Region	Central:Region

Users can retrieve province details pertaining to a specfic region (i.e.,"Central", "Western", "Eastern", "Southern"). Query 2 in Table 6.1 queries for provincial details in the "Central:Region". The system will bind the free slot filler for the slot name"name" and free variable "?ProvinceDetails" to all the provinces in the central region of Bhutan. The details of each province in the central region are returned to the user, one at a time, in the output format shown in the Table 6.2. Users can retrieve all the results by clicking on the "Next Solution" button on the OO jDREW interface.

The third mode of querying is useful when users do not have any input and yet wants to retrieve some provincial details of Bhutan. In this case, the system returns the details of all the provinces, one by one, as the two slots "name" and "region" are both free to be bound to any province in the KB.

6.3.1.2 Search for Routes between Any Two Provinces

Another province-centric operation is route planning. User can retrieve all the alternative routes connecting two specific provinces, along with recommendation of the shortest route. A sample query is shown in Table 6.3 and its solution in Table 6.4 (cf. the screenshot in Figure 6.2).

Table 6.3: Query mode(input/output) for Route search

Query	**User Input Values**	**Query Formats** (Input values are bold-faced)
1	startPoint endPoint	getRouteDetails(startPoint->**Chukha**:Province; endPoint->**Punakha**:Province; ?RouteDetails, ?ShortestRoute).

Table 6.4: Route search result for Query 1

Output Variables	**Variable Bindings**
?RouteDetails	[[[[Chukha, Chuzom, Thimphu, Lobesa, Punakha], 11.2:Real], [[Chukha, Chuzom, Thimphu, Lobesa, WangduePhodrang, Punakha], 11.90:Real]]; numRoutes->2:Integer]
?ShortestRoute	[[[Chukha, Chuzom, Thimphu, Lobesa, Punakha], 11.2 : Real]

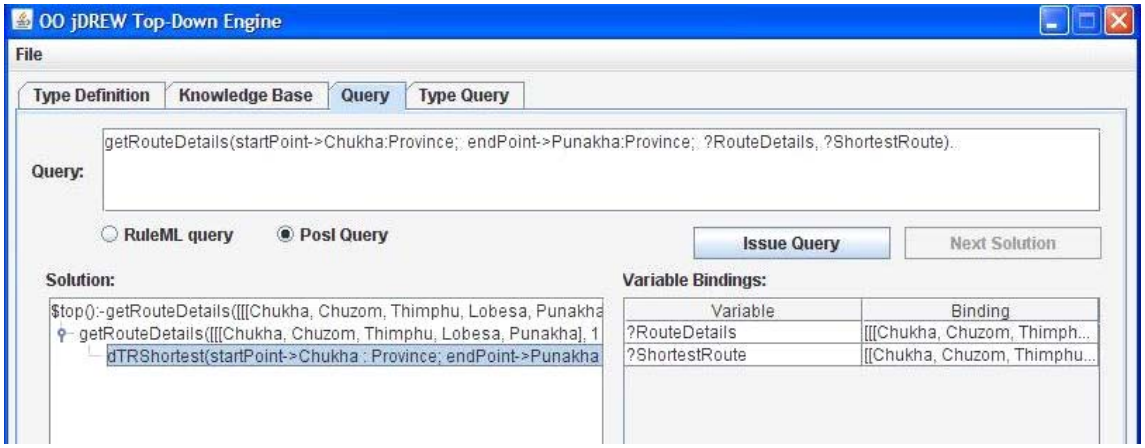

Figure 6.3: Screenshot for Query 1 result

6.3.1.3 Search for Activities

The term "Activity" subsumes both "event" and "attraction" classes introduced in Section 3.1. For each of the selected province profile of Bhutan, the fact base of our KB consists of stored FOAF-like profiles of attractions, events, and accommodations. These tourist entities are subclassified by the use of RDFS light-weight ontology type definitions (cf. Appendix E). The typed and well-formed KB offers a good powerful resource as a search engine to our users. Table 6.5 shows the different modes of searching for an activity. The systematic variation of these queries lies in the user's selected input slots.

Table 6.5: Queries of different input/output modes for Activity search

Query	**User Input Values**	**Query Formats** (Input values are bold-faced)
1	actName	getActivityDetails(actName->**Paro_Tshechu:Events**; theme->?Theme; address->?Address; ?ActivityDetails)
2	actName:*type* and/or address element	getActivityDetails(actName->?Name:**Festivals**; theme->?Theme; address->[?Subblock, **Chhoekhor:Block**, ?Province, ?Region, ?Country]; ?ActivityDetails)
3	theme and/or address element	getActivityDetails(actName->?Name; theme->**Cultural_Religious_Heritage**; address->[?Subblock, ?Block, **Paro:Province**, ?Region, ?Country]; ?ActivityDetails)
4	theme actName:*type* address element	getActivityDetails(actName->?Name:**Events**; theme->**Recreation**; address->[?Subblock, ?Block, ?Province, **Southern:Region**, ?Country]; ?ActivityDetails)
5	None	getActivityDetails(actName->?Name; theme->Theme; address->?Address; ?ActivityDetails)

The main options given to our users for searching an activity are primarily activity theme or type pertaining to a specific search area. The first option is useful if a user wants to retrieve more information about a specific event or attraction. The "type" relates to the RDFS type definition from the classfication of events and attractions collected from the Harmonise ontology (cf. Section 4.2). The type is attached to the activity name with a colon infix (':'). This allows our users to make a more specific choice of the type of activity. The next input slot name, "theme" is a broader sense of classification of the whole activity domain into three main themes, Cultural_Religious_Heritage, Nature, and Recreation. This classification of activity by theme is according to the Department of Tourism, Bhutan [58]. Therefore, we allow users to search activities by either of the two classification or by specifying both. Query 4 in Table 6.5 is a good example, where users can get all activity details of type "Events", belonging to the theme "Recreation" in the "Southern" region of the country. Query 5 is a special case where users specify nothing and the system binds the free slots to every activity profile in the KB. This is the most general form of querying compared to the most specific activity search in Query 4. To get a clear picture of the system output, we show the result to Query 4 in Table 6.6 (cf. the screenshot in Figure 6.4). The given solution is the only solution that meets the user's input specification for Query 4.

Table 6.6: Activity search result of Query 4

Output Variables	**Variable Bindings**
?ActivityDetails	[ActName->Yangphel_Archery_Tournament:Sport_archery; WebLink->"http://www.bhutanarchery.com/default.asp"; EventDates->[StartDate->date[2008:Real, 08:Real, 23:Real]; EndDate->date[2008:Real, 10:Real, 02:Real]]; Description->"11TH Yangphel open archery tournament"; Address->[Phuentsholing_Upper_Town:Town, Phuentsholing:Block, Chukha:Province, Southern:Region, Bhutan:Country]; Theme->Recreation; RelatedTo->"Thimphu_Drupchen:Annual_festival"]

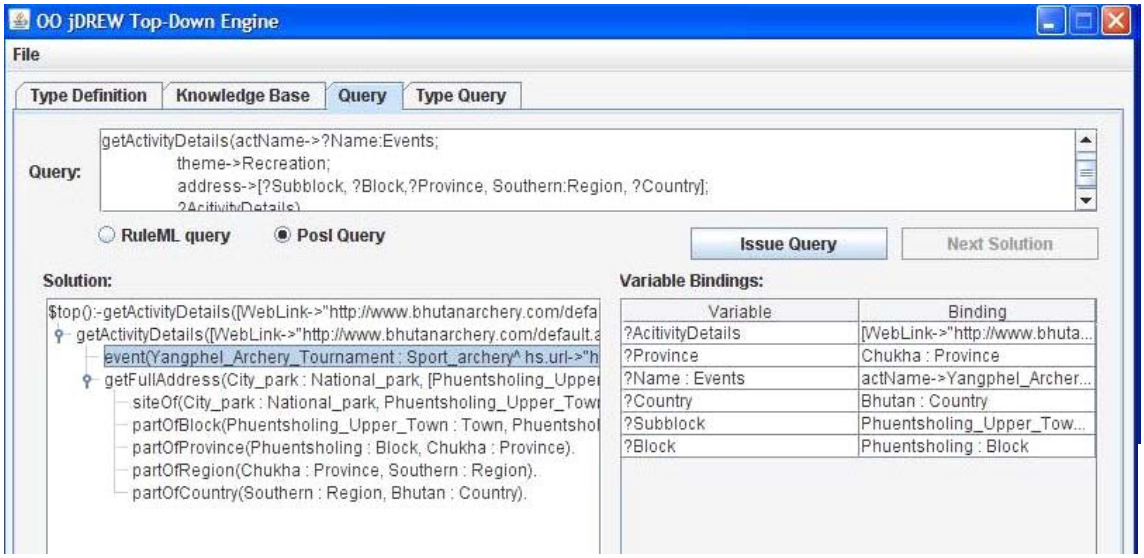

Figure 6.4: Screenshot for Query 1 result

6.3.1.4 Search for Accommodations

Similar to the activity search operations, the accommodation profiles structured with the RDFS light-weight ontology (described in Section 3.2) are used to provide users with some options for retrieving accommodation details for Bhutan. The main parameters for retrieving accommodation details are by name, type, price, or location. The systematic variation of queries are shown in Table 6.7.

Accommodation information retrieval can be performed by name (i.e., the most specific), type, or user maximum affordable price (userMaxPrice) pertaining to a specific search area. Similar to what we have discussed earlier for province and activity search, Query 1 shows a mode of querying for accommodation by name. It is of interest to only users who want to retrieve the detailed information about a particular object of interest. The solution to this is always unique as names are the primary identifier of an entity in our KB. The input "type" for the accommodation subdomain relates to the RDFS type definition of accommodation (cf. Section 4.2.3). We considered a simple type hierarchy for the accommodation consisting of only 4 main types, Guest_house, Lodge, Hotel, and Resort (cf. Section 2.4). This allows our users to define a preference

Table 6.7: Queries of different input/output modes for Accommodation search

Query	**User Input Values**	**Query Formats** (Input values are bold-faced)
1	accName	getAccommodationDetails(accName->**Aman_Resort:Resort**; address->?Address; setMaxPrice->?SetMaxPrice; ?AccommodationDetails)
2	accName:*type* and/or address element	getAccommodationDetails(accName->?Name:**Guest_house**; address->[**Chamkhar:Town**, ?Block, ?Province, ?Region, ?Country]; setMaxPrice->?SetMaxPrice; ?AccommodationDetails)
3	setMaxPrice and/or address element	getAccommodationDetails(accName->?Name; address->[**Chamkhar:Town**, ?Block, ?Province, ?Region, ?Country]; setMaxPrice->[**Yes**, **2000**:Real]; ?AccommodationDetails)
4	accName:*type* setMaxPrice address element	getAccommodationDetails(accName->?Name:**Resort**; address->[**Tsento_Shari:Village**, ?Block, ?Province, ?Region, ?Country]; setMaxPrice->[**Yes**, **1500**:Real]; ?AccommodationDetails)
5	None	getAccommodationDetails(accName->?Name; address->?Address; setMaxPrice->?SetMaxPrice; ?AccommodationDetails)

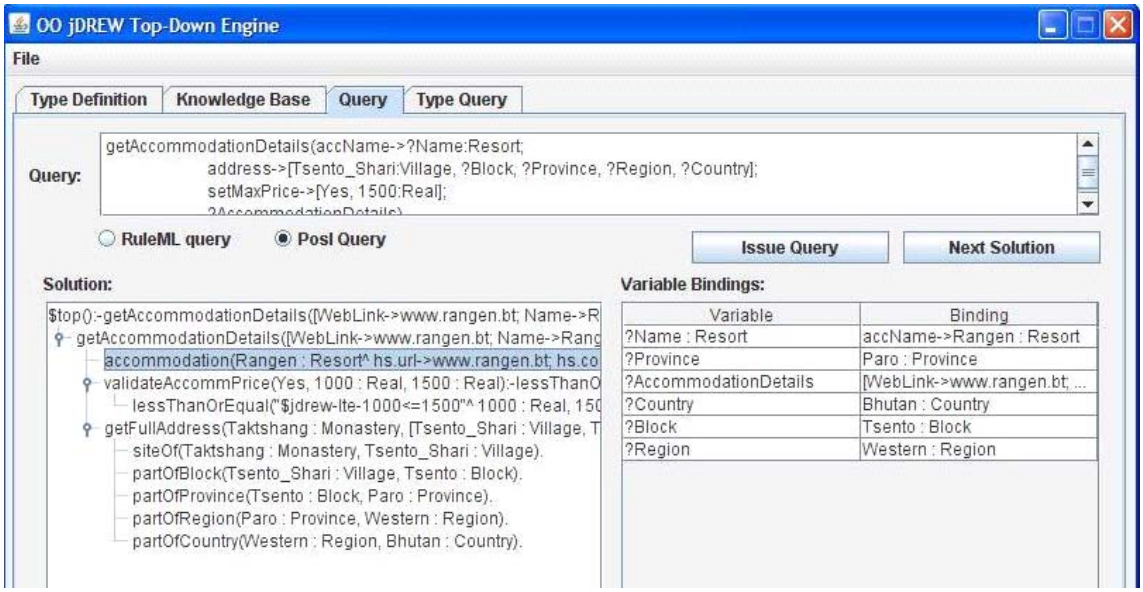

Figure 6.5: Screenshot for Query 4 result

for the type of accommodation. Query 2 searches for a specfic type of accommodation pertaining to a specific part of the country. Another parameter to get accommodations is by a certain price bound within a specific search area (?Subblock, ?Block, ?Province, ?Region, ?Country). Query 3 retrieves two solutions as there are two accommodation profiles that are located in the subblock "Chamkhar:Town" and does not cost more than "2000 Ngultrum" (Ngultrum, Bhutanese currency). Then, in Query 4, we show a query which is specific about all the three options: type, userMaxPrice, and search area. There is only one solution that fulfills Query 4's specification (cf. the screenshot in Figure 6.5). The output variable bindings for this query are shown in Table 6.8. The last mode of query in Table 6.5 is the most general form of query (open query) where the free slots would be bound to every accommodation profile in the KB.

6.3.1.5 Execution Times

The experiments are performed in ITC 315, graduate student lab. The execution time is measured under Windows XP with Intel Core 2 Duo 2.66 GHz processor, 2.95

Table 6.8: Accommodation search result of Query 4

Output Variables	**Variable Bindings**
?AccommodationDetails	[AccName->Rangen:Resort; WebLink->"www.rangnen.bt "; Address->[Tsento_Shari:Village, Tsento:Block, Paro:Province, Western:Region, Bhutan:Country]; Standard->[StarRating->2:Real; MinPrice->1000:Real]; ContactDetails-> [Telecoms->[Landline->9758211452; Cell->97517682948]; Email->"manager@rangen.bt"]; RelatedTo->"Holiday_Home:Hotel"];

GB of RAM. The 00 jDREW engine version 0.96 is used on Eclipse platform with Java Runtime Environment version jre1.6.

The eTourPlan prototype, tested for a KB consisting of 73 facts and 37 rules, can perform any of the preceding search operations in reasonable time. The maximum time it takes for searching any of the tourist information is 4817 milliseconds. From our experiments, the engine takes longer time for open queries (all free slots) than for those with some fixed input (constants) from the user. Firing an open query forces the system to unify with all the facts as the query does not provide any specific unifying knowledge. On the other hand, for queries with some or all fixed arguments (constants), the engine instantly finds the exact match and returns the solution very fast.

The search performance of any of the preceeding subdomains are good due to two main reasons: 1) The object-centric profile descriptions of each of the subdomains are well-structured with RDFS type definitions and partonomy rules, which provides the flexibility to narrow down (or broaden) the search space. 2) Our search rules are object-centric and therefore search is localised to a specific domain.

6.3.2 Location-Centric Recommendation

The location-centric recommender rule subsystem has been described earlier (in Section 5.2.4). The eTourPlan prototype offers two main options, "UserPrefBased" and "SystemRecommendation". The two modes of querying for location-centric recommendations are shown in Table 6.9. For the first option, the system provides a simple recommendation of provinces and the routes between these provinces, which are selected by looking up in the FOAF relation between the provinces. This concept was described earlier for system route planning in Section 5.2.1.2. The user needs to provide the number of provinces and the starting province as shown in Query 1 in Table 6.9. The output from the system is a list of specified number of touristic provinces with some tourist information for each of the provinces. Tourist information includes the URL of the province, number of attractions, events, and accommodations for each of the provinces as shown in Table 6.10 (cf. the screenshot in Figure 6.6). It also provides the routes between these provinces. Similar concept is used for planning a travel in attraction-only mode, discussed in the next section.

Although our system-based recommendation provides good travel recommendation to our users, it is not enough for users who want to visit provinces of their own personal choice. This mode of querying for recommendation is described in the second row of Table 6.9. Users must provide their start and end point along with their preferred list of provinces to visit. The system returns a list of tourist entities at each of the provinces along with routes between the selected provinces. The recommendation process collects all the events, attractions, and accommodations in each of the provinces and returns this accumulated list as the final solution. Sample output of the second mode of recommendation is shown in Table 6.11 (cf. the screenshot in Figure 6.7).

In Query 2, the userPrefList is a singleton list with one province, "Chukha". The recommender system accumulates all the attractions, events, and accommodation

Table 6.9: Queries of different input/output modes for Recommendation)

	User Input Values	**Query Formats** (Input values are bold-faced)
1	typeOfRecommend numProvinces	locCentricRecommend(typeOfRecommend->**SystemRecommendation**; userInputs->[startPoint->**Paro:Province**; numProvinces->**3**:Integer]; [?Routes, ?Recommendations, ?TotalBusHours])
2	typeOfRecommend startPoint userPrefList endPoint	locCentricRecommend(typeOfRecommend->**UserPrefBased**; userInputs->[startPoint->**Paro:Province**; userPrefList->[**Chukha:Province**]; endPoint->**Thimphu:Province**]; [?Routes, ?Recommendations, ?TotalBusHours])

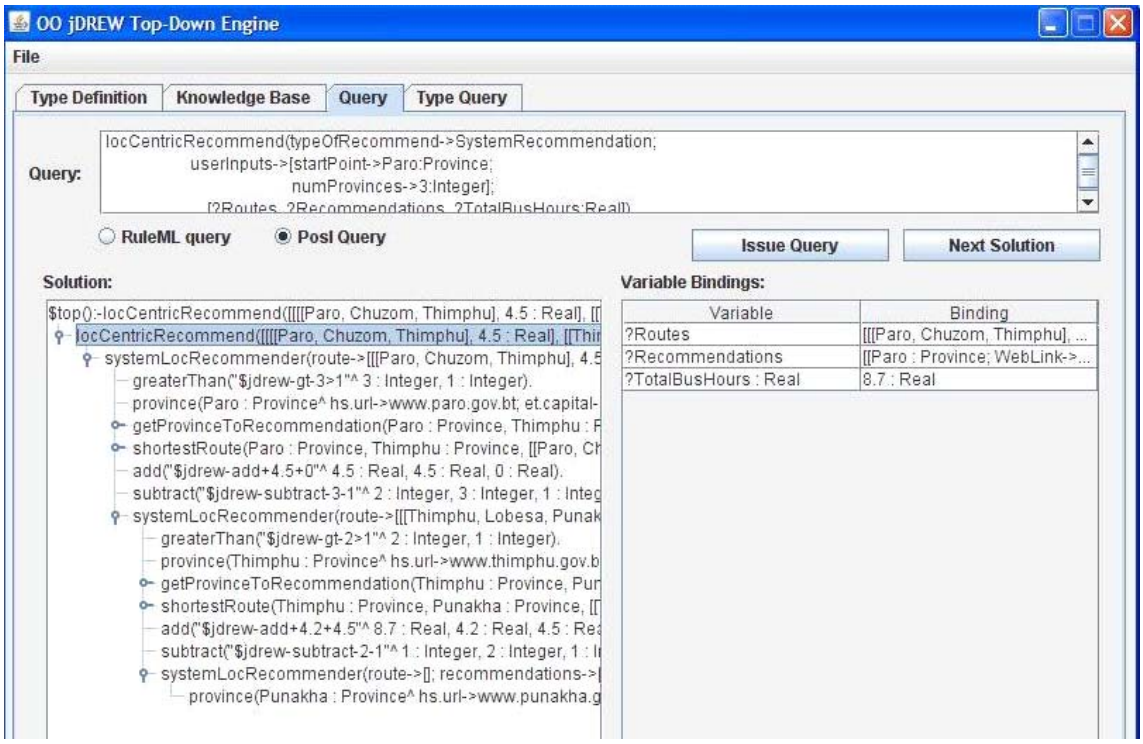

Figure 6.6: Screenshot for Test Query 1

Table 6.10: Location-centric recommendation by the system

Output Variables	**Variable Bindings**
?Routes	[[Paro, Chuzom, Thimphu], 6.5 :Real], [[Thimphu, Lobesa, Punakha], 4.2:Real]]
?Recommendations	[[Paro:Province; WebLink->"http://www.paro.gov.bt/"; TouristInfo->NumAttractions->3:Integer; NumEvents->1:Integer; NumAccommodations->3:Integer]], [Thimphu:Province; WebLink->"http://www.thimphu.gov.bt/"; TouristInfo->NumAttractions->3:Integer; NumEvents->1:Integer; NumAccommodations->3:Integer]], [Punakha:Province; WebLink->"http://www.punakha.gov.bt/"; TouristInfo->NumAttractions->2:Integer; NumEvents->1:Integer; NumAccommodations->0:Integer]]
?TotalBusHours	8.7:Real

facilities. It also provides the shortest route from the starting point (Paro:Province) to the preferred location (Chukha:Province) and the shortest route from the last province in the user's preference list to the specified end point (Thimphu:Province).

6.3.2.1 Execution Times

The execution of Query 1 and Query 2 (cf. Table 6.9) takes 16218 and 1523781 milliseconds, respectively. As we can see in the outputs of the two modes of operations, the location-centric recommendation for user-preferred provinces (Query 2) takes comparatively large amount of time to system recommendation of provinces (Query 1). This is because it is only in the second mode of recommendation that we integrate "getAllAttractions", "getAllEvents", and "getAllAccommodations" recursive predicates. For each recursive predicate, as the number of iteration increases from 1 to N, the execution time gets exponentially high. Another source of inefficiency is from the fact that the textual order between rule is not exploited by our pure logic programs, thereby performing unnecessary searches for candidate bindings.

Table 6.11: Location-centric recommendation for user-preferred Provinces

Output Variables	**Variable Bindings**
?Routes	[[Paro, Chuzom, Chukha], 6.5 :Real, [Chukha, Chuzom, Thimphu], 7.0 :Real]
?Recommendations	[Chukha; EventList->[[EventName->Chukha_Tshechu:Annual_festival; Description->"One of the most amazing festivals in Chukha"; Address->[Chukha_Town:Town, Gelling:Block, Southern:Region, Bhutan:Country, EventDates->[StartDate->date[2008:Real, 03:Real, 19:Real]; EndDate->date[2008:Real, 03:Real, 21:Real]]], [EventName->Yangphel_Archery_Tournament:Sport_archery; Description->"11TH Yangphel open archery tournament"; Address->[Phuentsholing_Upper_Town:Town, Phuentsholing:Block,, Southern:Region, Bhutan :Country, EventDates->[StartDate->date[2008:Real, 08:Real, 23:Real]; EndDate->date[2008:Real, 10:Real, 02 :Real]]]]; AttractionList->[[AttractionName->Chukha_Dzong:Fortress; WebLink->" "; Description->"It is one of the most beautiful attractions."; Address->[Chukha_Town:Town, Gelling:Block, Southern:Region, Bhutan :Country, Theme->Cultural_Religious_Heritage; AccommodationList->[[Hotel_Druk_Phuentsholing:Hotel; WebLink->"www.drukhotels.com/"; MinPrice->"2700:Real"; Rating->4:Real], [Hotel_Namgay:Hotel; WebLink->"www.hotelNamgay.bt/"; MinPrice->"1800:Real"; Rating->3:Real]]]]
?TotalBusHours	13.5:Real

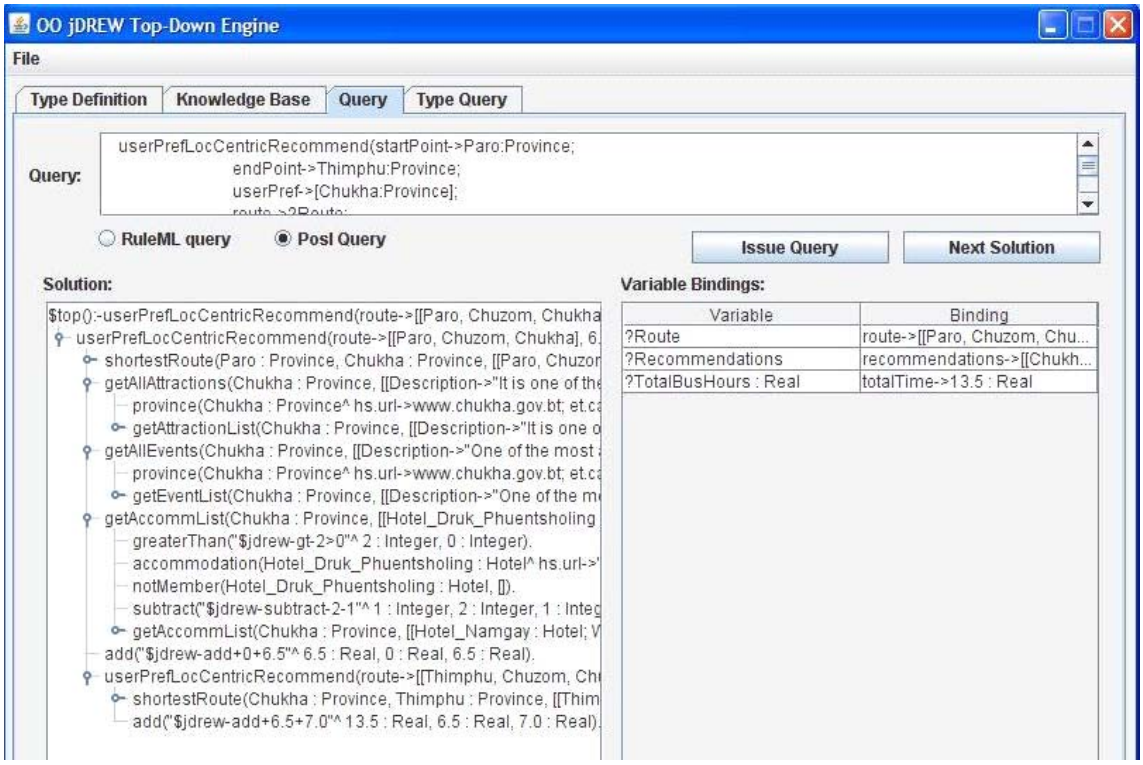

Figure 6.7: Screenshot for Test Query 2

6.3.3 Complete Travel Planning Operation

We will now look at different scenarios of planning and the corresponding output results. eTourPlan offers two main planning options, "Attractions-only" and "Event-centric", as shown in Table 6.12.

6.3.3.1 Attraction-Only Planning

The first option for travel planning is the attraction-based planning. It uses the FOAF relation between attractions similar to related provinces in location-centric planning. Users has to input the starting point, the number of attractions and the total trip time in days. The system accumulates the FOAF-related attractions from the given starting province and chains along until it finds the specified number of N attractions for the user. The output variable "?TravelResult" binds to a detailed attraction list, whicn includes route information as well. Currently, the routes are im-

plemented only between provinces. However, the route computation can be done at a finer granularity, at the subblock level for provinces, by applying the same route computation approach. The last output argument in the list that binds to "?TravelResult" is the ?TotalActivityTimeInHours. This is the sum of the route hours and the total of "?MaxHoursAtAnAttractionSite" for N attractions. A screenshot of the eTourPlan "Attraction0nly" planning is shown in Figure 6.8. The resulting output variable bindings shown in Table 6.13 describe the travel result. The result is a trip scheduled for visiting 4 attractions (two fortresses, a monastery and a national museum). It provides the important details, such as the URL, description, theme, opening hours, and location for each attraction selected through the FOAF relations of attraction profiles. It also provides the provincial route information between the provinces. The total time (includes both travel and activity hours) for the entire trip is also computed.

Table 6.12: Queries of different input/output modes for Travel Planning

	User Input Values	**Query Formats** (Input values are bold-faced)
1	typeOfPlanning startPoint endPoint numAttractions userTotalTravelTimeInDays	eTourPlan(typeOfPlanning->**AttractionOnly**; userInputs->[startPoint->**Paro:Province**; endPoint->**Thimphu:Province**; numAttractions->**4**:Integer; userTotalTravelTimeInDays->**4**:Integer]; ?TravelResult)
2	typeOfPlanning startPoint endPoint userStartDate userEndDate maxBreak minBreak attractionRecommendation eventNum	eTourPlan(typeOfPlanning->**EventCentric**; userInputs->[startPoint->**Paro:Province**; endPoint->**Thimphu:Province**; userStartDate->**date[2008:Real,10:Real,01:Real]**; userEndDate->**date[2008:Real,11:Real,10:Real]**; maxBreak->**10:Real**; minBreak->**0:Real**; attractionRecommendation->**No**; eventNum->**2:Integer**]; ?TravelResult)

6.3.3.2 Scenarios of Event Planning

eTourPlan's event-centric planning is the primary operation that integrates the other rule subsystems (cf. Figure 6.1). The event planner handles the event schedul-

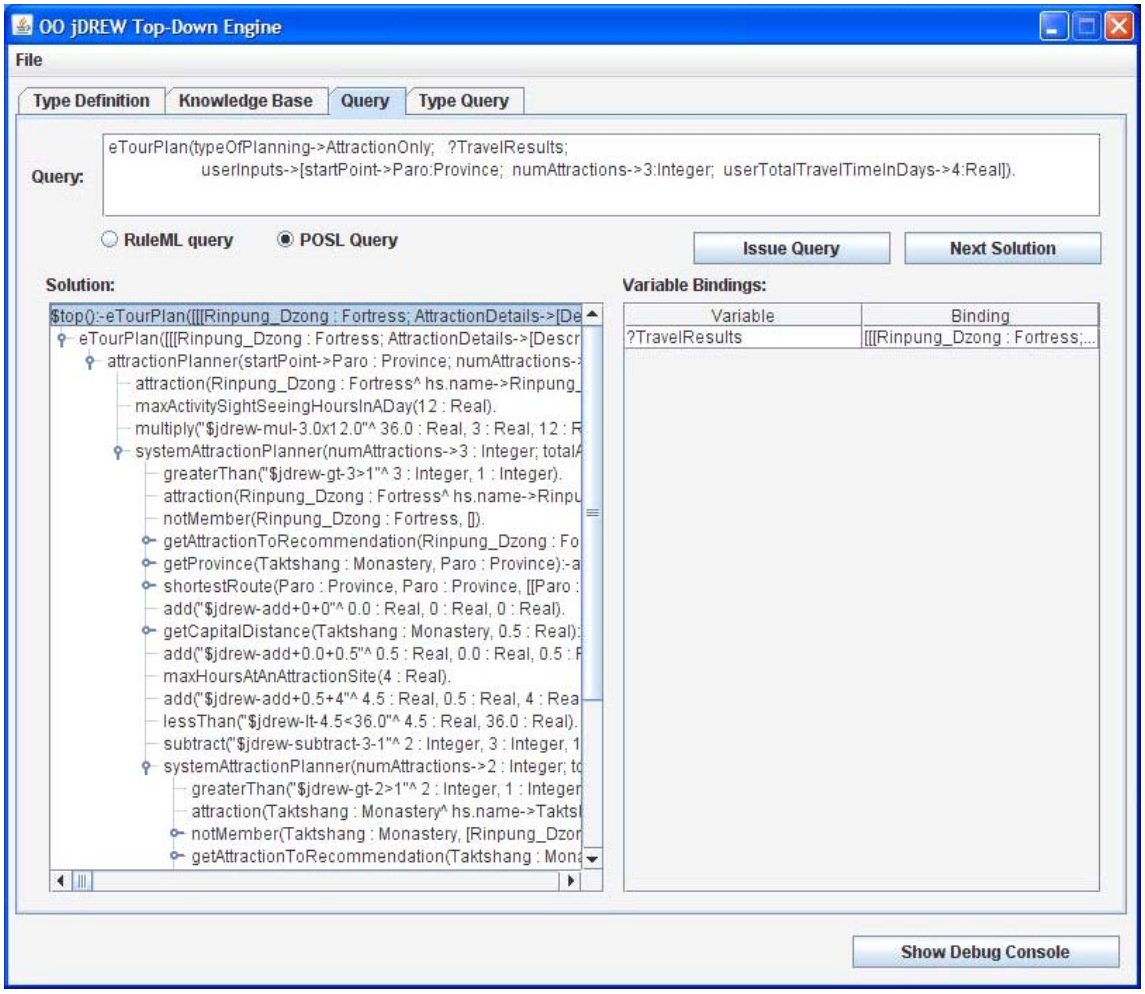

Figure 6.8: Screenshot for Querying "AttractionOnly" travel

Table 6.13: Attraction-only travel result

Output Variables	**Variable Bindings**
?TravelResult	[Name->Rinpung_Dzong:Fortress; AttractionDetails->[Url->“ ”; Description->“It is one of the most beautiful”; attractions in the Western region.”; Theme->Cultural_Religious_Heritage; OpenHours->Open[DaysOfWeek[Mon,Tue,Wed,Thu,Fri,Sat,Sun], Period[9:Real, 16:Real]]; Location->[Phatsana:Village, Paro:Province]]; Route->[[Paro:Province], 0:Real], [Name->Taktshang:Monastery; AttractionDetails->[Url->“ ”; Description->“It is known as the tiger’s Nest,; one of the seven wonders of in the world.”; Theme->Cultural_Religious_Heritage; OpenHours->Open[DaysOfWeek[Mon,Tue,Wed,Thu,Fri,Sat,Sun], Period[8:Real, 18:Real]]; Location->[Tshento_Shari:Village, Paro:Province]]; Route->[[Paro:Province], 0:Real], [Name->Ta_Dzong: National_museum; AttractionDetails->[Url->“www.nationalmuseum.gov.bt”; Description->”It is the biggest and the oldest; museum in Bhutan”; Theme->Cultural_Religious_Heritage; OpenHours->Open[DaysOfWeek[Tue,Wed,Thu,Fri,Sat,Sun], Period[10:Real, 16:Real]]; Location->[Goepay:Village, Paro:Province]]; Route->[[Paro, Thimphu], 4.5:Real]], [Name->Tashichoe_Dzong:Fortress; AttractionDetails->[Url->“ ”; Description->“It is one of the most beautiful; attractions in the Capital”; Theme->Cultural_Religious_Heritage; OpenHours->Open[DaysOfWeek[Tue,Wed,Thu,Fri,Sat,Sun], Period[9:Real, 16:Real]]; Location->[Jongshina:Town, Thimphu:Province]]; Route->[[Thimphu:Province], 0:Real]]]; TotalActivityTimeInHours->22.5: Real

Table 6.14: Execution times for Attraction-only Planning

?NumAttractions:Integer	Execution Times (in Milliseconds)
1	250
2	18812
3	313250
4	3453297

ing with respect to event schedules and distances between event locations for a user-specified travel dates. The planner provides attraction recommendations at the subblocks of the event locations. It also provides an additional option of on-route attraction recommendations between event locations, considering the possibility of a big time gap between two events. This optimises the resulting travel output to users incase there is only one event within the user-specified dates and it has a long route to the event location. User can either check "Yes" or "No" to this option. We will now look at some examples of querying the eTourPlan event-centric planner with varying test queries.

Example 1:*User queries for an event-centric plan of 1 events between the 1st of October and the 10th of November and specifies a "maxBreak" of 10 days and "minBreak" of 0 days between main events. User also specifies the starting province, "Paro:Province", and the final destination province, as "Thimphu:Province". User checks "No" for on-route attraction recommendation, knowing that the planner provides recommendation of attractions at the subblock of event location.*

Test Query 1:

```
eTourPlan(typeOfPlanning->EventCentric;
          ?TravelResult;
          userInputs->[userStartDate->date[2008:Real, 11:Real, 01:Real];
                       userEndDate->date[2008:Real, 11:Real, 04:Real];
                       maxBreak->4:Real;
                       minBreak->0:Real;
                       startPoint->Paro:Province;
                       endPoin->Thimphu:Province;
                       attractionRecommendation->No;
                       eventNum->1:Integer]).
```

The system returns "No solution" since there is no successful binding to this query, meaning the system fails to find any event in the KB that is occuring between the above dates.

Example 2: *User queries for an event-centric plan of 2 events between the 1st of October and the 10th of November. The user specifies a "maxBreak" of 10 days and "minBreak" of 0 days between main events. User specifies the starting province, "Paro: Province", and the final destination province, as "Thimphu:Province". The user selects "No" for on-route attraction recommendation, knowing that the planner provides recommendation of attractions at the subblock of event location.*

Test Query 2:

```
eTourPlan(typeOfPlanning->EventCentric; ?TravelResult;
          userInputs->[userStartDate->date[2008:Real, 10:Real, 01:Real];
                       userEndDate->date[2008:Real, 11:Real, 10:Real];
                       maxBreak->10:Real;
                       minBreak->0:Real;
                       startPoint->Paro:Province;
                       endPoin->Thimphu:Province;
                       attractionRecommendation->No;
                       eventNum->2:Integer]).
```

The system returns 6 alternative solutions to this query. There are 5 events occuring between the user-specified time interval. The planner takes care of time and distance constraint validation, resulting in selective combinations of events. The Screenshots showing the first two test results for this query are shown in Figures 6.9 and 6.10. The selected event combinations in the output are recorded in Table 6.16. The second column contains the list of all events that are found within the user's specified trip dates. The third column corresponds to each of the event combinations generated by the system. For example, the first result generated by the system selects event 1 (Tamshingphala_Choepa:Traditional_festival) and event 2 (Tangbi_Mani:Traditional_festival), which are both located in Bumthang province and validated with user's time specification for a travel. The complete output variable bindings shown in Table 6.15 describe the travel result. It provides important details, such as the URL, description, theme, event dates, and location for each selected event. It also provides recommendation of attractions sited at the event-occuring subblock

and the provincial route information between the provinces.

Table 6.15: Event-Centric travel results

Output Variables	**Variable Bindings**
?TravelResult	[[[EventName->Tamshingphala_Choepa:Traditional_festival; EventDates->[Startdate->date[2008:Real, 10:Real, 08:Real]; Enddate->date[2008:Real, 10:Real, 10:Real]]; Theme->Cultural_Religious_Heritage; EventDescription->"One of the most amazing festivals in Bumthang" Location->[Tamshing_Lhakhang:Temple, Tamshing_Village:Village, Bumthang:Province]; RelatedEvent->Tangbi_Mani:Traditional_festival; RouteDetails->[[Paro:Province, Chuzom:Province, Thimphu:Province, Lobesa:Province, WangduePhodrang:Province, Trongsa:Province, Bumthang:Province], []; RouteBusHours->16.7:Real]; RecommendedAttractions->[Tamshing_Lhakhang:Temple, "It was built by Pema Lingpa,the Treasure Revealer in 1505."]], [EventName->Tangbi_Mani:Traditional_festival; EventDates->[Startdate->date[2008:Real, 10:Real, 13:Real; Enddate->date[2008:Real, 10:Real, 15:Real]]; Theme->Cultural_Religious_Heritage; EventDescription->"A prestigious annual festival in Bumthang" Location->[Tangbi_Monastery:Monastery, Tangbi:Village, Bumthang:Province]; RelatedEvent->Wangdue_Tshechu:Annual_festival; RouteDetails->[Bumthang:Province], []; RouteBusHours->0:Real]]]; RecommendedAttraction->[Tangbi_Monastery:Monastery "Located in upper Tang valley."]], ReturnRoute->[[Bumthang:Province, Trongsa:Province, WangduePhodrang:Province,Lobesa:Province, Thimphu:Province]; Returntime->12.2:Real]]

The results in column 3 (Table 6.17) are shown are in the order of executed results in OO jDREW TD. The highlighted area of the trace in the output Screenshot (cf. Figure 6.9) shows the next selected event.

Example 3: *User queries for 3 events within the same dates and constraints as in Example 2, setting dates from 1st of October to the 10th of November, with a "maxBreak"' of 10 days and "minBreak" of 0 days between main events. User selects "No" for on-route attraction recommendation.*

Table 6.16: Evaluation of event-centric travel results

Event	Event Schedules	Event Sequences of length ?EventNum= 2
1	Tamshingphala_Choepa:Traditional_festival startDate->date[2008:Real,10:Real,08:Real] endDate->date[2008:Real,10:Real,10:Real] province->Bumthang	**1,2** **1,5**
2	Tangbi_Mani:Traditional_festival startDate->date[2008:Real,10:Real,13:Real] endDate->date[2008:Real,10:Real,15:Real] province->Bumthang	
3	Thimphu_Drupchen:Annual_festival startDate->date[2008:Real,10:Real,04:Real] endDate->date[2008:Real,10:Real,08:Real] province->Thimphu	**3,2** **3,4**
4	Thimphu_Tshechu:Annual_festival startDate->date[2008:Real,10:Real,09:Real] endDate->date[2008:Real,10:Real,11:Real] province->Thimphu	**4,2** **4,5**
5	Wangdue_Tshechu:Annual_festival startDate->date[2008:Real,10:Real,20:Real] endDate->date[2008:Real, 10:Real, 29:Real] province->WangduePhodrang	

Table 6.17: Evaluation of event-centric travel results

Event	Event Schedules	Event Sequences of length ?EventNum= 3
1	Tamshingphala_Choepa:Traditional_festival startDate->date[2008:Real,10:Real,08:Real] endDate->date[2008:Real,10:Real,10:Real] province->Bumthang	**1,2,5**
2	Tangbi_Mani:Traditional_festival startDate->date[2008:Real,10:Real,13:Real] endDate->date[2008:Real,10:Real,15:Real] province->Bumthang	
3	Thimphu_Drupchen:Annual_festival startDate->date[2008:Real,10:Real,04:Real] endDate->date[2008:Real,10:Real,08:Real] province->Thimphu	**3,2,5** **3,4,2** **3,4,5**
4	Thimphu_Tshechu:Annual_festival startDate->date[2008:Real,10:Real,09:Real] endDate->date[2008:Real,10:Real,11:Real] province->Thimphu	**4,2,5**
5	Wangdue_Tshechu:Annual_festival startDate->date[2008:Real,10:Real,20:Real] endDate->date[2008:Real, 10:Real, 29:Real] province->WangduePhodrang	

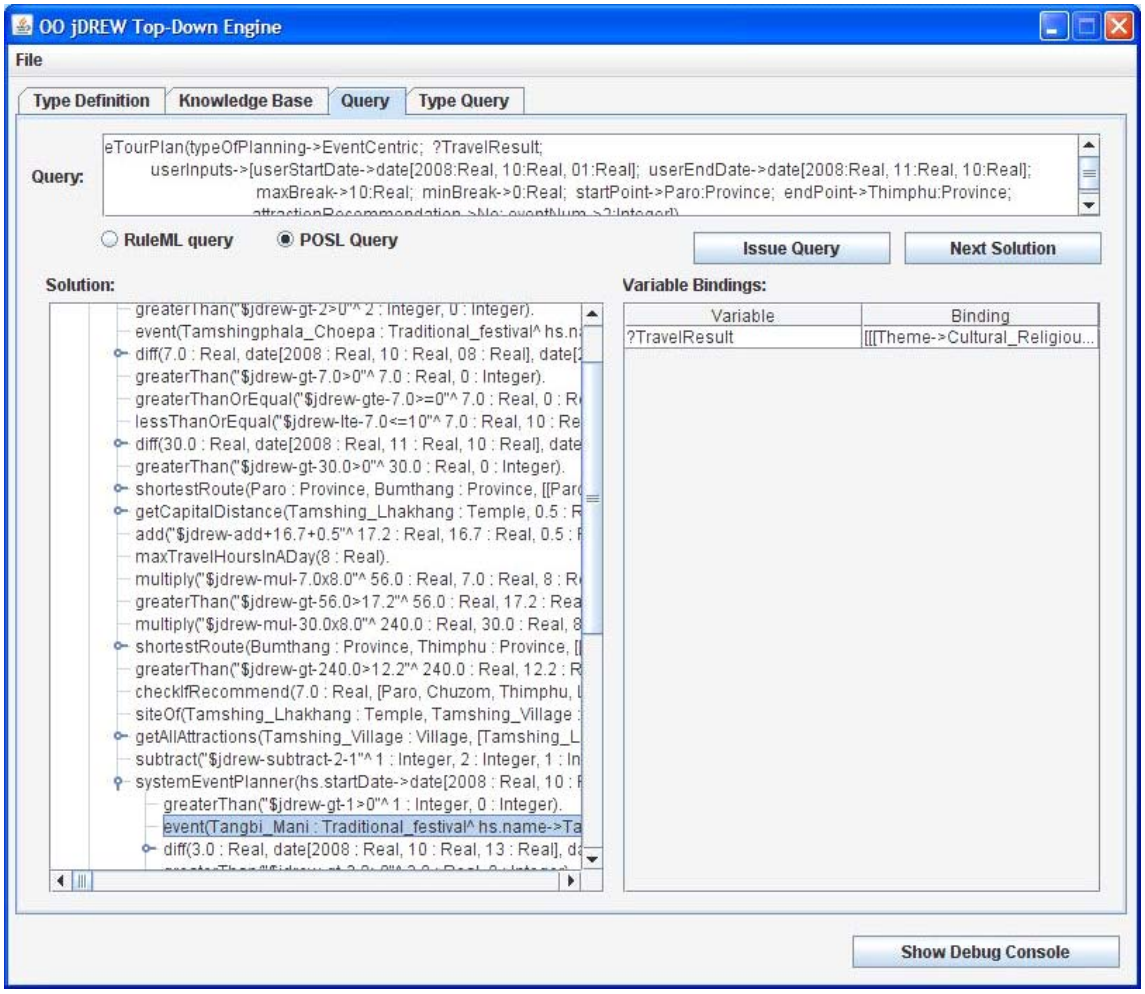

Figure 6.9: Screenshot for Test Query 2, First Solution

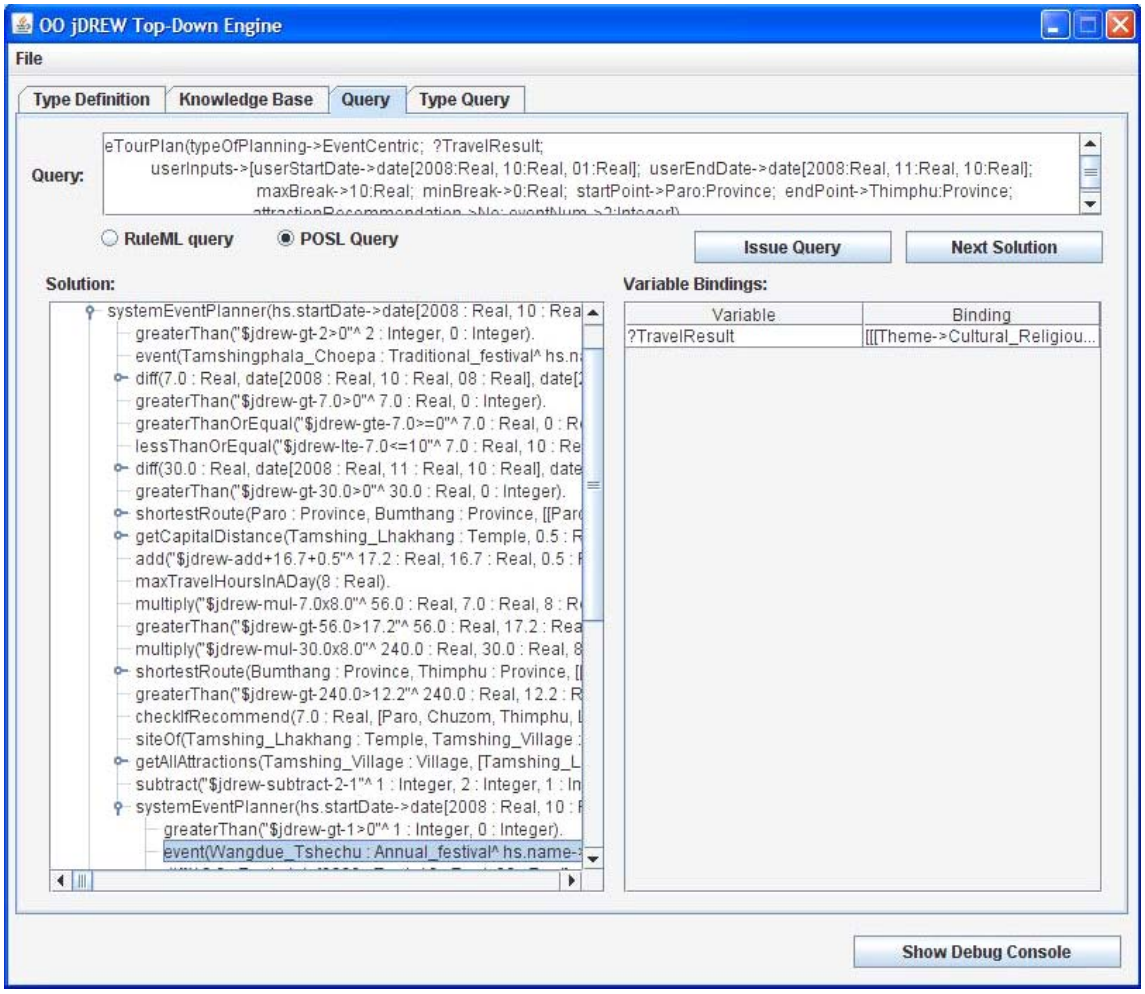

Figure 6.10: Screenshot for Test Query 2, Next Solution

Test Query 3:

```
eTourPlan(typeOfPlanning->EventCentric; ?TravelResult;
          userInputs->[userStartDate->date[2008:Real, 10:Real, 01:Real];
                       userEndDate->date[2008:Real, 11:Real, 10:Real];
                       maxBreak->10:Real;
                       minBreak->0:Real;
                       startPoint->Paro:Province;
                       endPoin->Thimphu:Province;
                       attractionRecommendation->No;
                       eventNum->3:Integer]).
```

Table 6.17 provides the 5 possible combinations of three events within the user's specified trip dates. Increasing the eventNum to "four" enforces the system to return a unique solution with four events. The evaluation result for four events is shown in Table 6.18.

Test Query 4:

```
eTourPlan(typeOfPlanning->EventCentric; ?TravelResult;
          userInputs->[userStartDate->date[2008:Real, 10:Real, 01:Real];
                       userEndDate->date[2008:Real, 11:Real, 10:Real];
                       maxBreak->10:Real;
                       minBreak->0:Real;
                       startPoint->Paro:Province;
                       endPoin->Thimphu:Province;
                       attractionRecommendation->No;
                       eventNum->4:Integer]).
```

6.3.3.3 Execution Times

Although the current version of OO jDREW 0.96 has additional new features (cf. Section 3.3) to support various modes of operations, the reasoning engine performs complete iterative-deepening for each of the predicates, resulting in longer execution times. The OO jDREW team is currently working towards upgrading the current engine to a system using argument indexing. This will improve the run-time of the eTourPlan rule system. Therefore, complete testing of eTourPlan's functionalities on a faster running OO jDREW is left as future work at this point.

Table 6.18: Evaluation of event-centric travel results

Event	Event Schedules	Event Sequences of length ?EventNum= 4
1	Tamshingphala_Choepa:Traditional_festival startDate->date[2008:Real,10:Real,08:Real] endDate->date[2008:Real,10:Real,10:Real] province->Bumthang	
2	Tangbi_Mani:Traditional_festival startDate->date[2008:Real,10:Real,13:Real] endDate->date[2008:Real,10:Real,15:Real] province->Bumthang	
3	Thimphu_Drupchen:Annual_festival startDate->date[2008:Real,10:Real,04:Real] endDate->date[2008:Real,10:Real,08:Real] province->Thimphu	**3,4,2,5**
4	Thimphu_Tshechu:Annual_festival startDate->date[2008:Real,10:Real,09:Real] endDate->date[2008:Real,10:Real,11:Real] province->Thimphu	
5	Wangdue_Tshechu:Annual_festival startDate->date[2008:Real,10:Real,20:Real] endDate->date[2008:Real, 10:Real, 29:Real] province->WangduePhodrang	

Table 6.19: Execution times for Event-centric Planning

?NumEvents:Integer	Execution Times (in Milliseconds)
1	First Solution: 29110 Next Solution time: 672 Next Solution time: 719
2	First Solution: 428563 Next Solution time: 3563 Next Solution time: 4797 Next Solution time: 1282 Next Solution time: 1359 Next Solution time: 1250
3	First Solution: 4873875 Next Solution time: 9656 Next Solution time: 24469 Next Solution time: 12985
4	First Solution: 5428250

Chapter 7

Conclusion

Our eTourPlan prototype is a knowledge-based tourist route and activity planner. The KB is comprised of object-centric facts of tourist entities, which are structured by light-weight ontologies of the tourism subdomains. This well-structured and comprehensive KB is complemented with rule subsystems needed for providing various tourist services such as precise search, tour recommendations, and travel plans. In order to show a real-world implementation of this prototype, we have developed and evaluated a KB based on the tourism information of Bhutan.

7.1 Contributions

This thesis was mainly focused to design, implement, and evaluate an eTourism prototype for Bhutan. We have designed light-weight ontologies (adapted from the Harmonise eTourism ontology) using RDFS to capture the tourism subdomains. We then built a fact base based on Bhutan tourist information for describing tourist entities using FOAF-like profile facts. The ontology-structured fact base in our KB is enriched with various rule subsystems needed for generating a travel plan. Query rules were used to perform semantic searches. Partonomy rules were implemented for the administrative subdivision of a country. Derivation rules were used to deduce tran-

sitive closure facts used for route and distance computation. Finally, inference rules were implemented for providing planning and recommendation modes of services. The eTourPlan's three distinct operations: search, recommendation, and planning are evaluated with varying user preferences in the OO jDREW reasoning engine.

Our eTourPlan prototype has explored the reasoning potentials of rules on a complete KB of a multi-dimensioned domain (i.e. the tourism domain). Our KB consists of 115 classes, 73 facts about Bhutan tourist information, and 37 rules in total, which cover Bhutan's entire administrative partonomy, route computation for the entire roadmap between provinces, profiles of 10 (of 20) provinces and tourist information for these. Results of running the eTourPlan rule system in the rule engine OO jDREW show that our knowledged-based eTourPlan prototype can offers multiple options for a diversity of travel plans with recommendations. Each of the resulting travel plan provides a detailed information of selected events, recommended attractions and routes between event locations. Evaluation of the prototype also shows that it can provide precise parametric search results for queries on the tourism KB.

7.2 Future Work

The eTourPlan prototype currently only supports complete planning where users provide a fixed number of preferences and the planner schedules the entire trip for the user. We have discussed other planning strategies such as partial planning and sequence planning in this thesis. A good example of partial planning is where users specify some individual events or attractions of their choice and ask for N activities in total. An example of sequence planning is where user provides a list of activites and wants the system to order their trip. Adding such sophisticated planning options would provide a very flexible service to our user.

In this thesis, we have considered the cost measure only for the accommodation subdomain. An additional feature of cost estimation for the total travel will comple-

ment the current planner. Similar approaches used for validation of total number of days for the travel can be implemented.

Our eTourPlan prototype is an executable "specification" of a complete travel planning using Semantic Web techniques. Although, it was tested successfully on the OO jDREW TD reasoning engine, the execution times of the top-level predicates are relatively slow mainly because OO jDREW TD performs a complete search of all subpredicates for each iteration of recursive predicates. The OO jDREW team is currently working towards upgrading the reasoning engine to a more efficient indexing system. Therefore, when this new version of OO jDREW is complete, a complete re-evaluation of the eTourPlan on this improved reasoning engine is highly recommended.

This executable specification of a knowledge-based tourist route and activity planner implemented for this thesis can also be integrated with a database or translated to a self-contained database application.

The facts in the KB used in this thesis are handcrafted. These facts could be either derived from some databases or extracted from RDFWeb pages. Currently, RDFWeb is used as a means to describe machine-understandable person-centric profiles [20]. This approach can be used to semantically describe a Web of inter-related homepages of tourist entities using the W3C's RDF technology to integrate information from different homepages and connect them with respect to different relationships.

Lastly, designing a user-friendly graphical user interface would increase the utility of the key operations of the eTourPlan prototype. The semantic model and search of eTourPlan can be extended to a Semantic Bhutan" portal (and transferred to other regions such as New Brunswick).

References

[1] *Technology reports: Rule markup language (ruleml)*, `http://xml.coverpages.org/ruleML.html`, 2002.

[2] *Mandarax*, `http://java-source.net/open-source/rule-engines/mandarax`, September 2005.

[3] *SweetRules*, `http://sweetrules.projects.semwebcentral.org/`, September 2005.

[4] Shin Adachi, Masahiro Hamasaki, Kosuke Numa, Ikki Ohmukai1, and Hideaki Takeda, *Metadata-driven personal knowledge publishing*, ISWC 2004, LNCS 3298, Springer-Verlag, Berlin, 2004, pp. 591–604.

[5] Michelle Anderson, Marcel Ball, Harold Boley, Stephen Greene, Nancy Howse, Daniel Lemire, and Sean McGrath, *Racofi: A rule-applying collaborative filtering system*, Proceedings of COLA'03, IEEE/WIC, October 2003.

[6] Juan D. Arias, Eva Armengol, Daniel Borrajo, Luis A. Castillo, Susana Fernández, Jesús González-Boticario, Eva Onaindia, Antonio Rodríguez, and Laura Sebastia, *samap: An user-oriented adaptive system for planning tourist visits*, Expert Syst. Appl. **34** (2008), no. 2, 1318–1332.

[7] Daniel Bachlechner and Katharina Siorpaes, *Applying Semantic Web Technologies in the Tourism Industry*, DERI Innsbruck, University of Innsbruck, Austria, 2004.

[8] Marcel Ball, *Oo jdrew*, http://www.jdrew.org/oojdrew/, September 2005.

[9] Marcel Ball, Harold Boley, David Hirtle, Jing Mei, and Bruce Spencer, *The OO jDREW Reference Implementation of RuleML*, Proc. Rules and Rule Markup Languages for the Semantic Web (RuleML-2005), LNCS 3791, Springer-Verlag, November 2005, pp. 218–223.

[10] T. Berka and M. Plnig, *Designing recommender systems for tourism.*, The Eleventh International Conference on Information Technology in Travel & Tourism, ENTER, Cairo, Egypt, 2004.

[11] Tim Berners-Lee, *Semantic Web - XML2000*, http://www.w3.org/2000/Talks/-1206-xml2k-tbl.

[12] Yevgen Biletskiy, Harold Boley, and Girish R Ranganathan, *Ruleml-based learning object interoperability on the semantic web*, Interactive Technologies and SMART Education, Emerald, volume 5 issue 1, 2008, pp. 39–58.

[13] Yevgen Biletskiy, J Anthony Brown, and Girish R Ranganathan, *Information extraction from syllabi for e-advising, expert systems with applications*, Elsevier (in press), 2008.

[14] Harold Boley, *RuleML for the Semantic Web*, http://www.dfki.uni-kl.de/~boley/-RuleML-RDF-OntoWeb/sld001.htm, June 2001.

[15] Harold Boley, *The Rule Markup Language: RDF-XML Data Model, XML Schema Hierarchy, and XSL Transformations*, Proc. 14th International Conference on Applications of Prolog (INAP2001), The University of Tokyo, LNAI 2543, Springer-Verlag, October 2002.

[16] Harold Boley, *Integrating positional and slotted knowledge on the semantic web*, `http://www.ruleml.org/posl/poslintweb-talk.pdf`, March 2005.

[17] Harold Boley, Virendrakumar C. Bhavsar, Marcel Ball, and Lu Yang, *Weighted Partonomy-Taxonomy Trees with Local Similarity Measures for Semantic Buyer-Seller Match-Making*, Business Agents and the Semantic Web (BASeWEB) Workshop, 2005.

[18] Harold Boley, Virendrakumar C. Bhavsar, Jie Li, and Jing Mei, *Expert Finding in eCollaboration Using FOAF with RuleML Rules*, Montreal Conference of eTechnologies 2006, 2006, pp. 53–65.

[19] Harold Boley and Michael Kifer, *RIF Basic Logic Dialect*, W3C working draft, W3C, July 2008, http://www.w3.org/TR/2008/WD-rif-bld-20080730/.

[20] Dan Brickley, *RDFWeb and Friend of a Friend (FOAF)– Getting started with FOAF*, `http://rdfweb.org/mt/foaflog/archives/000051.html`, 2003.

[21] Dan Brickley, *The friend of a friend (foaf) project*, http://www.foaf-project.org/, November 2005.

[22] R. Burke, *Encyclopedia of library and information systems, vol.69*, ch. Knowledge-Based Recommender Systems, A.Kent, ed., Marcel Dekker.

[23] R. Burke, *Encyclopedia of library and information systems, vol.69*, ch. Hybrid Recommender Systems: Survey and Experiments., A.Kent, ed., Marcel Dekker, 2002.

[24] Jorge Cardoso, *E-tourism: Creating dynamic packages using semantic web processes*, http://www.w3.org/, April 2005.

[25] Jorge Cardoso, *Semantic web services, theory, tools, and applications*, ch. 1: The Syntactic and the Semantic Web, Information Science Reference, 2007.

[26] William F. Clocksin and Christopher S. Mellish, *Programming in PROLOG*, Springer Verlag, New York Inc., 2004.

[27] R. Scott Cost, Tim Finin, Anupam Joshi, James Mayfield, and Urvi Shah, *Information Retrieval On The Semantic Web*, Proceedings of the eleventh international conference on Information and knowledge management, ACM Press, 2002, pp. 461–468.

[28] R. Scott Cost, Clay Fink, Tim Finin, Anupam Joshi, and James Mayfield, *Information Retrieval and the Semantic Web*, Proceedings of the 38th International Conference on System Sciences, 2005.

[29] John Debenham, *Integrating knowledge base and database*, ACM Press, 2007, pp. 28–32.

[30] John Debenham, *Normalising knowledge objects*, ACM Press, 2007, pp. 335–343.

[31] Ying Ding, Dieter Fensel, Michel Klein, and Borys Omelayenko, *The semantic web: yet another hip?*, Elsevier Science, 2002.

[32] Leigh Dodds, *An Introduction to FOAF*, http://www.xml.com/pub/a/2004/-02/04/foaf.html, 2004.

[33] Oren Etzioni, Steven D. Gribble, Alon Halevy, Henry Levy, Luke McDowell, William Pentney, Deepak Verma, and Stani Vlasseva, *Evolving the semantic web with mangrove*, Tech. Report UWCSE030201, Department of Computer Science and Engineering, University of Washington, Seattle, WA, February 2003.

[34] Alexander Felfernig and Gerhard Friedrich, *Recommender systems*, IEEE Computer Society, 2007, pp. 18–20.

[35] D.R. Fesenmaier, N. Mirzadeh, F. Ricci, H. Rumetshofer, E. Schaumlechner, A. Venturini, K.W. Wber, and A.H. Zins, *Dietorecs: A case-based travel advisory system, destination recommendation systems: Behavioral foundations and applications*, CAB International, London, 2006, pp. 227–239.

[36] Farshad Fotouhi, Ming Dong, and Shiyong Lu, *The Semantic Web: opportunities and challenges for next-generation Web applications*, Information Research 7(4), 2002.

[37] Antonella Fresa, *Harmo-ten: A cost effective solution for information exchange*, http://www.harmo-ten.info, the 13th International conference on Communication Technologies in Tourism, 2006.

[38] T. R. Gruber, *Towards Principles for the Design of Ontologies Used for Knowledge Sharing*, Formal Ontology in Conceptual Analysis and Knowledge Representation (Deventer, The Netherlands) (N. Guarino and R. Poli, eds.), Kluwer Academic Publishers, 1993.

[39] Elliotte Rusty Harold, *The xml bible, 2nd edition*, ch. 17: XSL Transformations, John Wiley & Sons, 2001.

[40] Orit Hazzan and Mariana Teif, *Partonomy and taxonomy in object-oriented thinking: junior high school students' perceptions of object-oriented basic concepts*, ACM SIGCSE Bulletin, 2006, pp. 55–60.

[41] Jafar Jafari, *Encyclopedia of Tourism*, Routledge Taylor & Francis Group, 2003.

[42] Anja Jentzsch, *Tourism Standards*, XML Clearinghouse Report 12, Freie Universitt Berlin, 2005.

[43] Yongil Jeong, Jaehun Joo, and Sang M. Lee, *Semantic web technologies and e-business*, ch. 12: Applications of Semantic Web Based on the Domain-Specific Ontology for Global KM, IGP, Idea Group Publishing, 2007.

[44] Michael Kifer, George Lausen, and James Wu, *Logic Foundations of Object Oriented and Frame Based Languages*, Journal of the Association for Computing Machinary, 1995.

[45] Jie Li, *Rule-based social networking for expert finding*, Master thesis, University of New Brunswick, September 2006.

[46] Anna Maclachlan and Harold Boley, *Semantic web rules for business information*, Proc. International Conference on Web Technologies, Applications, and Services (WTAS 2005), Calgary, Canada, IASTED, 2005.

[47] Alexander Maedche and Steffen Staab, *Applying Semantic Web Technologies for Tourism Information Systems*, Proceedings of the International Conference on Information and Communication Technologies in Tourism, Research Center for Information Technologies at the University of Karlsruhe, Research Group Knowledge Management(WIM), January 2002, pp. 218–223.

[48] Michele Missikoff, *Harmonise: An Ontology-Based Approach for Semantic Interoperability*, http://www.harmonise.org, July 2002.

[49] National Information Standards Organization, *Understanding metadata*, NISO Press, 2004.

[50] Sean B. Palmer, *The Semantic Web: An Introduction*, `http://infomesh.net/2001/swintro/`, 2001.

[51] W3C Recommendation, *RDF Vocabulary Description Language 1.0: RDF Schema*, http://www.w3.org/TR/rdf-schema/, February 2004.

[52] Francesco Ricci, *Travel recommender systems*, IEEE Intelligent Systems, 2002, pp. 55–57.

[53] Francesco Ricci and Hannes Werthner, *E-commerce and tourism*, Commun. ACM Press, 2004, pp. 101–105.

[54] Bruce Spencer, *jDREW*, http://www.jdrew.org/jDREWebsite/jDREW.html, October 2005.

[55] Said Tabet, Gerd Wagner, Silvie Spreeuwenberg, Paul D. Vincent, Gonzaques Jacques, Christian de Sainte Marie, Jon Pellant, Jim Frank, and Jacques Durand, *Omg production rule representation - context and current status*, Rule Languages for Interoperability, 2005.

[56] C Tisdell, *Tourism Economics, the Environment and Development*, EDWARD ELGAR, INC, 2001.

[57] B. Tversky, *Where partonomies and taxonomies meet: Meanings and prototypes: Studies on linguistic categorization*, Routledge,London, 1990, pp. 334–344.

[58] Martin Zeppezauer, *Sustainable Tourism Development Strategy*, Department of Tourism, Bhutan Media Services, Oct 2005.

Appendix A: Main Planner and Recommender Rule Set

Recommendation Rule Subsystem

```
% eTourPlan offers two main options for location-centric recommendation
% 1. Recommendations for user-Preferred Locations
% 2. System's Recommendation of Touristic Locations

% location-centric Recommendation
 locCentricRecommend(typeOfRecommend->UserPrefBased;
                     userInputs->[startPoint->?P1;
                                  userPref->?PrefList;
                                  endPoint->?P2];
                         [?Route,
                          ?Recommendations,
                          ?TotalBusHours:Real]):-
   userPrefLocCentricRecommend(startPoint->?P1;
                               endPoint->?P2;
                               userPref->?PrefList;
                               route->?Route;
                               recommendations->?Recommendations;
                               accumulateTime->0:Real;
                               totalTime->?TotalBusHours:Real).

% Workhorse predicate for location centric recommendation
 userPrefLocCentricRecommend(startPoint->?P1;
                             endPoint->?P1;
                             userPref->[];
                             route->[];
                             recommendations->[];
                             accumulateTime->0:Real;
                             totalTime->0:Real).

 userPrefLocCentricRecommend(startPoint->?P1;
                             endPoint->?P2;
                             userPref->[];
                             route->[?Route, ?BusHours];
```

```
                              recommendations->[];
                              accumulateTime->?AccumulateTime:Real;
                              totalTime->?NewBus:Real):-
   shortestRoute(?P1, ?P2, [?Route,?BusHours]!?),
   add(?NewBus, ?AccumulateTime:Real, ?BusHours:Real).


   userPrefLocCentricRecommend(startPoint->?P1;
                               endPoint->?P2;
                               userPref->[?Pref|?PrefRest];
                               route->[?Route, ?BusHours|?RouteRest];
                               recommendations->[[?Pref;
                                    EventList->?Events;
                                    AttractionList->?Attractions;
                                    Accommodations->?AccommodationList]|
                                    ?RestRecommendations];
                               accumulateTime->?AccumulateTime:Real;
                               totalTime->?TotalBusHours:Real):-
    shortestRoute(?P1, ?Pref, [?Route,?BusHours]!?),
    getAllAttractions(?Pref, ?Attractions, ?NumAttractions:Integer),
    getAllEvents(?Pref,  ?Events, ?NumEvents:Integer),
    getAccommList(?Pref, ?AccommodationList, [], 2:Integer),
    add(?NewBus, ?AccumulateTime:Real, ?BusHours),
    userPrefLocCentricRecommend(startPoint->?Pref;
                                endPoint->?P2;
                                userPref->?PrefRest;
                                route->?RouteRest;
                                recommendations->?RestRecommendations;
                                accumulateTime->?NewBus;
                                totalTime->?TotalBusHours:Real).

% System's Recommendation of Locations
 locCentricRecommend(typeOfRecommend->SystemRecommendation;
                     userInputs->[startPoint->?Province;
                                  numProvinces->?NumProvinces:Integer];
                        [?Routes, ?TouristInfo, ?TotalBusHours:Real]):-
   systemLocRecommender(startPoint->?Province;
                        nextPoint->?Province2:Province;
                        route->?Routes;
                        recommendations->?TouristInfo;
                        starttime->0:Real;
                        totalTime->?TotalBusHours:Real;
                        numProvinces->?NumProvinces:Integer).

% Work horse predicate
 systemLocRecommender(startPoint->?Province;
                      nextPoint->?Province2;
                      route->[];
                      recommendations->[?Province;
                                        WebLink->?Url;
                                        TouristInfo-> [
                                          NumAttractions->?NumAttractions;
                                          NumEvents->?NumEvents;
                                          NumAccommodations->?NumAccommodations]];
                           starttime->?Starttime:Real;
```

```
                          totalTime->?Starttime:Real;
                          numProvinces->1:Integer):-
      province(?Province^hs.url->?Url;
                    et.numBlocks->?NumBlocks;
                    et.numAttractions->?NumAttractions;
                    et.numEvents->?NumEvents;
                    et.numAccommodations->?NumAccommodations!?).


  systemLocRecommender(startPoint->?Province1;
                       nextPoint->?Province2;
                       route->[[?Route,?BusHours]|?Rest];
                       recommendations->[[?Province1;
                                          WebLink->?Url;
                                          TouristInfo->[
                                            NumAttractions->?NumAttractions;
                                            NumEvents->?NumEvents;
                                            NumAccommodations->?NumAccommodations]]
                                            |?RestProvinces];
                       starttime->?Starttime:Real;
                       totalTime->?TotalBusHours:Real;
                       numProvinces->?NumProvinces:Integer):-

     greaterThan(?NumProvinces:Integer, 1:Integer),
     province(?Province1^hs.url->?Url;
                       et.numBlocks->?NumBlocks;
                       et.numAttractions->?NumAttractions;
                       et.numEvents->?NumEvents;
                       et.numAccommodations->?NumAccommodations!?),
      getProvinceToRecommendation(?Province1,?R1),
      shortestRoute(?Province1, ?R1,  [?Route, ?BusHours]),
      add(?NewBus:Real,?BusHours:Real,?Starttime:Real),
      subtract(?RouteLeft:Integer,  ?NumProvinces:Integer, 1:Integer),
      systemLocRecommender(startPoint->?R1;
                        nextPoint->?Province2;
                        route->?Rest;
                        recommendations->?RestProvinces;
                        starttime->?NewBus:Real;
                        totalTime->?TotalBusHours:Real;
                        numProvinces->?RouteLeft:Integer).
```

Planning Rule Subsystem

```
% eTourPlan has two modes of travel planning operation:
% 1. Attraction-only
% 2. Event-centric(Optional attraction recommendation)

% Attraction-Only mode of eTourPlan
 eTourPlan(typeOfPlanning->AttractionOnly;
           userInputs->[startPoint->?StartPoint;
                        numAttractions->?NumAttractions:Integer;
                        userTotalTravelTimeInDays->?UserTotalTravelTimeInDays:Real];
                       [?AttractionList;
                        TotalActivityTimeInHours->?TotalActivityTimeInHours]):-
```

```
  attractionPlanner(startPoint->?StartPoint;
                    numAttractions->?NumAttractions:Integer;
                    userTotalTravelTimeInDays->?UserTotalTravelTimeInDays:Real;
                    totalActivityTimeInHours->?TotalActivityTimeInHours:Real;
                    attractionList->?AttractionList).

% Event-Only/Centric mode of eTourPlan
  eTourPlan(typeOfPlanning->EventCentric;
            userInputs->[userStartDate->?UserStartDate;
                         userEndDate->?UserEndDate;
                         maxBreak->?MaxBreak:Real;
                         minBreak->?MinBreak:Real;
                         startPoint->?StartPoint;
                         endPoint->?EndPoint;
                         attractionRecommendation->?YesOrNo;
                         eventNum->?NumOfEvents:Integer];
                        [?EventDetails;
                         ReturnRoute->[?ReturnRoute; Returntime->?ReturnBusHours]]):-

   systemEventPlanner(hs.startDate->?UserStartDate;
                      hs.endDate->?UserEndDate;
                      maxBreak->?MaxBreak:Real;
                      minBreak->?MinBreak:Real;
                      et.startPoint->?StartPoint;
                      et.lastProvince->?LastProvince;
                      et.endPoint->?EndPoint;
                      attractionRecommendation->?YesOrNo;
                      eventDetails->?EventDetails;
                      eventNum->?NumOfEvents:Integer),
      shortestRoute(?LastProvince, ?EndPoint, [?ReturnRoute, ?ReturnBusHours]).

%Attraction Planner predicate

 attractionPlanner(startPoint->?StartPoint;
                   numAttractions->?NumAttractions:Integer;
                   userTotalTravelTimeInDays->?UserTotalTravelTimeInDays:Real;
                   totalActivityTimeInHours->?TotalActivityTimeInHours:Real;
                   attractionList->?AttractionList):-
   attraction(?Name^et.province->?StartPoint!?),

  %Computing user's total available time in hours
   maxActivitySightSeeingHoursInADay(?MaxActivityHoursInADay:Real),
   multiply(?UserTotalTravelTimeInHours:Real, ?UserTotalTravelTimeInDays:Real,
                  ?MaxActivityHoursInADay:Real),

   systemAttractionPlanner(attraction->?Name;
                           attractionList->?AttractionList;
                           visited->[];
                           accummulateTime->0:Real;
                       userTotalTravelTimeInHours->?UserTotalTravelTimeInHours:Real;
                           totalActivityTimeInHours->?TotalActivityTimeInHours:Real;
                           numAttractions->?NumAttractions:Integer).

% Workhorse recursive predicate
```

```
systemAttractionPlanner(attraction->?Name;
                        attractionList->[];
                        visited->?Visited;
                        accummulateTime->?AccummulateTime:Real;
                        userTotalTravelTimeInHours->?UserTotalTimelnHours:Real;
                        totalActivityTimeInHours->?AccummulateTime:Real;
                        numAttractions->0:Integer).

systemAttractionPlanner(attraction->?Name;
                        attractionList->[Name->?Name;
                        AttractionDetails->[
                          Description->?Description;
                          Url->?Url;
                          OpenHours->?OpenHours;
                          Theme->?Theme;
                          Location->[?Subblock, ?Province]]];
                        visited->?Visited;
                        accummulateTime->?AccummulateTime:Real;
                        userTotalTravelTimeInHours->?UserTotalTimelnHours:Real;
                        totalActivityTimeInHours->?TotalActivityTimeInHours:Real;
                        numAttractions->1:Integer):-
   attraction(?Name^
              hs.description->?Description;
              hs.url->?Url;
              et.open->?OpenHours;
              et.theme->?Theme;
              et.subblock->?Subblock;
              et.province->?Province;
              hs.relatedTo->?RelatedTo!?),
  getCapitalDistance(?Name, ?CapitalDistance),
  add(?TotalRouteHours:Real, ?AccummulateTime:Real, ?CapitalDistance:Real),

 %Assuming the user spends maximum of 4 hours at each attraction site,
 %which is stored as a global constant in the KB
  maxHoursAtAnAttractionSite(?MaxHoursAtAnAttractionSite:Real),
  add(?TotalActivityTimeInHours,
      ?TotalRouteHours:Real,
      ?MaxHoursAtAnAttractionSite:Real).

 systemAttractionPlanner(attraction->?Name;
                         attractionList->[[?Name;
                         AttractionDetails->[
                           Description->?Description;
                           Url->?Url;
                           OpenHours->?OpenHours;
                           Theme->?Theme;
                           Location->[?Subblock, ?Province]];
                           Route->?Route]|?Rest];
                        visited->?Visited;
                        accummulateTime->?AccummulateTime:Real;
                        userTotalTravelTimeInHours->?UserTotalTimelnHours:Real;
                        totalActivityTimeInHours->?TotalActivityTimeInHours:Real;
                        numAttractions->?NumAttractions:Integer):-

  greaterThan(?NumAttractions:Integer, 1:Integer),
```

```
 attraction(?Name^
             hs.description->?Description;
             hs.url->?Url;
             et.open->?OpenHours;
             et.theme->?Theme;
             et.subblock->?Subblock;
             et.province->?Province;
             hs.relatedTo->?RelatedTo!?),
 notMember(?Name, ?Visited),

%To get the next related attraction from the current attraction
 getAttractionToRecommendation(?Name,?RelatedAttraction),
 getProvince(?RelatedAttraction, ?Province1),

%Route and distance computation and validation
 shortestRoute(?Province, ?Province1, [?Route,?BusHours]),
 add(?NewBusHours, ?AccummulateTime:Real, ?BusHours:Real),
 getCapitalDistance(?RelatedAttraction, ?CapitalDistance),
 add(?TotalRouteHours, ?NewBusHours:Real, ?CapitalDistance:Real),

 maxHoursAtAnAttractionSite(?MaxHoursAtAnAttractionSite:Real),
 add(?AccumulateActivityTimeInHours,
     ?TotalRouteHours:Real,
     ?MaxHoursAtAnAttractionSite:Real),
 lessThan(?AccumulateActivityTimeInHours, ?UserTotalTimeInHours),

 subtract(?AttractionLeft:Integer,  ?NumAttractions:Integer, 1:Integer),
 systemAttractionPlanner(attraction->?RelatedAttraction;
                         attractionList->?Rest;
                         visited->[?Name|?Visited];
                         accummulateTime->?AccumulateActivityTimeInHours;
                         totalActivityTimeInHours->?TotalActivityTimeInHours:Real;
                         userTotalTravelTimeInHours->?UserTotalTimeInHours:Real;
                         numAttractions->?AttractionLeft:Integer).

% Workhorse predicate for event-centric planning
 systemEventPlanner(hs.startDate->?UserStartDate;
                    hs.endDate->?UserEndDate;
                    maxBreak->?MaxBreak:Real;
                    minBreak->?MinBreak:Real;
                    et.startPoint->?StartPoint;
                    et.lastProvince->?StartPoint;
                    et.endPoint->?EndPoint;
                    attractionRecommendation->?YesOrNo;
                    eventDetails->[];
                    eventNum->0:Integer).

systemEventPlanner(hs.startDate->?UserStartDate;
                    hs.endDate->?UserEndDate;
                    maxBreak->?MaxBreak:Real;
                    minBreak->?MinBreak:Real;
                    et.startPoint->?StartPoint;
                    et.lastProvince->?LastProvince;
                    et.endPoint->?EndPoint;
                    attractionRecommendation->?YesOrNo;
```

```
                    eventDetails->[[EventName->?Name:Events;
                                    EventDates->[Startdate->?EventStartDate;
                                                 Enddate->?EventEndDate];
                                    Theme->?Theme;
                                    Location->[?Location, ?Subblock, ?Province];
                                    EventDescription->?Description;
                                    RelatedEvents->?RelatedTo;
                                    RouteDetails->[?Route, ?Recommendations;
                                                   RouteBusHours->?BusHours];
                                    RecommendedAttractions->?Attractions]
                                    |?EventList];
                    eventNum->?N:Integer) :-

  greaterThan(?N:Integer,0:Integer),
  event(?Name^
        hs.startDate->?EventStartDate;
        hs.endDate->?EventEndDate;
        hs.description->?Description;
        hs.url->?Url;
        et.theme->?Theme;
        hs.location->?Location;
        et.province->?Province;
        hs.relatedTo->?RelatedTo!?),

 % Date validation of the event with the user's travel dates
    diff(?Difference, ?EventStartDate,?UserStartDate),
    greaterThanOrEqual(?Difference:Real, ?MinBreak:Real),
    lessThanOrEqual(?Difference:Real, ?MaxBreak:Real),
    diff(?RemainingDays, ?UserEndDate, ?EventEndDate),
    greaterThan(?RemainingDays, 0:Integer),

 % Route and distance time computation and validation
    shortestRoute(?StartPoint, ?Province, [?Route, ?BusHours]),
    getCapitalDistance(?Location, ?CapitalDistance),
    add(?RouteBusHours, ?BusHours:Real, ?CapitalDistance:Real),

% The time difference between the event dates has to be greater than
% the route bus hour from one event location to the next event location
  maxTravelHoursInADay(?MaxTravelInHours),
  multiply(?DifferenceInHours, ?Difference, ?MaxTravelInHours),
  greaterThan(?DifferenceInHours, ?RouteBusHours),

% Checking if there is enough remaining time to reach the user's specified
% endPoint from the current event location
  multiply(?ReturntripTimeInHours:Real, ?RemainingDays, ?MaxTravelInHours),
  shortestRoute(?Province, ?EndPoint, [?ReturnRoute, ?ReturnBusHours]),
  greaterThan(?ReturntripTimeInHours:Real, ?ReturnBusHours:Real),

% Check if the route needs attraction recommendation
  checkIfRecommend(?Difference, ?Route, ?Recommendations, ?YesOrNo),

% The getAttraction subpredicate with a specified location would
% also work for this purpose.
% getAttraction(?Location, ?Subblock),
  siteOf(?Location, ?Subblock),
```

```
% Getting all the attractions in the subblock of the event-location
  getAllAttractions(?Subblock,  ?Attractions, ?Num:Integer),

 subtract(?Eventleft:Integer,?N:Integer, 1:Integer),
 systemEventPlanner(hs.startDate->?EventEndDate;
                    hs.endDate->?UserEndDate;
                    maxBreak->?MaxBreak:Real;
                    minBreak->?MinBreak:Real;
                    et.startPoint->?Province;
                    et.lastProvince->?LastProvince;
                    et.endPoint->?EndPoint;
                    attractionRecommendation->?YesOrNo;
                    eventDetails->?EventList;
                    eventNum->?Eventleft:Integer).
```

Appendix B: Profiles of Provinces and thier Tourist Entities

This appendix gives the province-centric touristic profiles of Bhutan, which includes profiles of attractions, accommodations and events found in it.

```
% Fact profiles of Bumthang province and its Tourist Entities
% The facts are collected from Tourism Resources Inventory
% of Bhutan, 2005.

% FOAF-Profile of Attractions, Accommodations, and Events in Bumthang Province
 province(Bumthang:Province^
          hs.url->"www.bumthang.gov.bt";
          et.capital->Chamkhar:Town;
          et.area->"1,819 sq.km";
          et.elevation->"1,300 to 7300 meters";
          et.numBlocks->10:Integer;
          et.numAttractions->16:Integer;
          et.numEvents->13:Integer;
          et.numAccommodations->10:Integer;
          hs.languagesSpoken->"Dzongkha, Bumthap";
          hs.contact->"admbumthang@druknet.bt";
          hs.description->"Bumthang is one of the most attractive touristic
          province with several festivals throughout the year";
          hs.relatedTo->Mongar:Province).

 attraction(Bumthang_Dzong:Fortress^
            hs.url->" ";
            et.subblock->Chamkhar:Town;
            et.province->Bumthang:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                          Period[9:Real, 16:Real]];
            et.capitalDistance->0.5:Real;
            hs.description-> "The Bumthang Dzong or the Dzong of the white
            bird. The interesting thing about the Dzong is,that there is a
            water tower four stairs down behind the Dzong.";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->Ugyen_Chholing_Museum:Local_museum).

 attraction(Ugyen_Chholing_Museum:Local_museum^
            hs.url->" ";
            et.subblock->Chamkhar:Town;
            et.province->Bumthang:Province;
```

```
            et.theme->Cultural_Religious_Heritage;
            et.open->[Open[DaysOfWeek[Mon, Tue, Wed], Period[9:Real, 16:Real]],
                        Open[DaysOfWeek[Fri], Period[9:Real, 14:Real]],
                        Open[DaysOfWeek[Sun], Period[9:Real, 14:Real]]];
            et.capitalDistance->0.5:Real;
            hs.description->"Located in upper Tang valley on a commanding spur of
            3000m. The museum was opened in May 2001. It will give visitors a his-
            toric knowledge of bhutanese culture, religion & tradition.";
            hs.contact->" ";
            hs.schedule-> "12 months";
            hs.relatedTo->Petseling_Gompa:Temple).

 attraction(Petseling_Gompa:Temple^
            hs.url->" ";
            et.subblock->Chamkhar:Town;
            et.province->Bumthang:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed], Period[9:Real, 16:Real]];
            et.capitalDistance->0.5:Real;
            hs.description-> "It is situated about 3 to 4 hours walk uphill through
            pine forest from Chamkhar town.";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->Zugney:Textiles).

attraction(Zugney:Textiles^
           hs.url->" ";
           et.subblock->Chamkhar:Town;
           et.province->Bumthang:Province;
           et.theme->Cultural_Religious_Heritage;
           et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                        Period[10:Real, 18:Real]];
           et.capitalDistance->0.5:Real;
           hs.description-> "Two handicraft shops at Zugney, selling mostly the
           local woolen fabric Bumthang yathra.";
           hs.contact->" ";
           hs.schedule->"12 months";
           hs.relatedTo->Kharchu_Monastery:Monastery).


 attraction(Kharchu_Monastery:Monastery^
            hs.url->" ";
            et.subblock->Chamkhar:Town;
            et.province->Bumthang:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                         Period[8:Real, 15:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"It is a new monastery established by Lam Namkhai
            Nyingpo It has beautiful sight of Chamkhar town.";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->Kurje_Lhakhang:Temple).

 attraction(Kurjey_Lhakhang:Temple^
```

```
            hs.url->" ";
            et.subblock->Kurjey:Village;
            et.province->Bumthang:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                        Period[8:Real, 15:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"It is very important in the history of Bhutan
            associated with Guru Rimpoche conversion to buddhism of Bumthang.";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->Thrasephel_Lhakhang:Temple).

attraction(Jambay_Lhakhang:Temple^
            hs.url->" ";
            et.subblock->Kurjey:Village;
            et.province->Bumthang:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                        Period[8:Real, 15:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"It is a very religious temple.";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->Thrasephel_Lhakhang:Temple).

attraction(Thrasephel_Lhakhang:Temple^
            hs.url->" ";
            et.subblock->Tharpeling:Village;
            et.province->Bumthang:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                        Period[8:Real, 15:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"A small temple on a ridge & on the left of the road
            leading to Chhokhortoe.";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->Tharpeling_Monastery:Monastery).

attraction(Tharpaling_Monastery:Monastery^hs.name->Tharpaling_Monastery:Monastery;
            hs.url->" ";
            et.subblock->Tharpaling:Village;
            et.province->Bumthang:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                        Period[8:Real, 15:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"It was built by the great saint Lonchen Rabjam in the
            14th century. It has beautiful paintings, monastic school above the
            main temple and it has fabulous view on Chumme valley.";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->Tangbi_Monastery:Monastery).
```

```
attraction(Tangbi_Monastery:Monastery^hs.name->Tangbi_Monastery:Monastery;
           hs.url->" ";
           et.subblock->Tangbi:Village;
           et.province->Bumthang:Province;
           et.theme->Cultural_Religious_Heritage;
           et.open->[Open[DaysOfWeek[Mon, Tue, Wed], Period[9:Real, 16:Real]],
                     Open[DaysOfWeek[Fri], Period[9:Real, 14:Real]],
                     Open[DaysOfWeek[Sun], Period[9:Real, 14:Real]]];
           et.capitalDistance->0.5:Real;
           hs.description-> "Located in upper Tang  valley. ";
           hs.contact->" ";
           hs.schedule-> "12 months";
           hs.relatedTo->Tamshing_Lhakhang:Temple).

attraction(Tamshing_Lhakhang:Temple^hs.name->Tamshing_Lhakhang:Temple;
           hs.url->" ";
           et.subblock->Tamshing_Village:Village;
           et.province->Bumthang:Province;
           et.theme->Cultural_Religious_Heritage;
           et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                         Period[8:Real, 15:Real]];
           et.capitalDistance->0.5:Real;
           hs.description->"Built by Pema Lingpa(A Treasure Revealer) in 1505.";
           hs.contact->" ";
           hs.schedule->"12 months";
           hs.relatedTo->Koenchhogsum_Lhakhang:Temple).

attraction(Koenchhogsum_Lhakhang:Temple^
           hs.url->" ";
           et.subblock->Nimalung:Village;
           et.province->Bumthang:Province;
           et.theme->Cultural_Religious_Heritage;
           et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
           Period[8:Real, 15:Real]];
           et.capitalDistance->0.5:Real;
           hs.description->"This is a small temple housing a big bell, a unique
           historical testimony of the 8th century. A big boulder placed in front
           of the temple is beleived to be the lid covering the lake inside.";
           hs.contact->" ";
           hs.schedule->"12 months";
           hs.relatedTo->Ura_Lhakhang:Temple).


attraction(Ura_Lhakhang:Temple^
           hs.url->" ";
           et.subblock->Ura:Village;
           et.province->Bumthang:Province;
           et.theme->Cultural_Religious_Heritage;
           et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                         Period[8:Real, 15:Real]];
           et.capitalDistance->0.5:Real;
           hs.description-> "A big  temple situated in the middle of Ura village.
           It has a statue of Guru Rimpochhe and beautiful paintings.";
           hs.contact->" ";
           hs.schedule-> "12 months";
```

```
            hs.relatedTo->Trumshingla_Park:National_park).


attraction(Trumshingla_Park:National_park^
            hs.url->" ";
            et.subblock->Ura:Village;
            et.province->Bumthang:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->[Open[DaysOfWeek[Mon, Tue, Wed], Period[9:Real, 16:Real]],
                          Open[DaysOfWeek[Sun], Period[9:Real, 14:Real]]];
            et.capitalDistance->0.5:Real;
            hs.description-> "Thrumshingla park is on the way to Mongar from Bum-
            thang. It is a home to many animals expecially the Red Panda. The
            Bengal Tiger has been also sighted. The park has also different
            kinds of rhododendron flowers and a variety of bird species.";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->Gongkhar_Lhakhang:Temple).

attraction(Gongkhar_Lhakhang:Temple^
            hs.url->" ";
            et.subblock->Ura:Village;
            et.province->Bumthang:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                         Period[8:Real, 15:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"It is a temple attributed to the Nun Gelongma Pemo and
            it takes half an hour walk uphill through pine wood.";
            hs.contact->" ";
            hs.schedule-> "12 months";
            hs.relatedTo->Domkhar_Zimchu:Palace).

attraction(Domkhar_Zimchu:Palace^
            hs.url->" ";
            et.subblock->Domkhar:Village;
            et.province->Bumthang:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Wed, Fri, Sun],
                         Period[8:Real, 15:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"The summer palace of the second king of Bhutan.
            This has a good view of Chhume village.";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->Darphel_Dzong:Ruins).

attraction(Darphel_Dzong:Ruins^
            hs.url->" ";
            et.subblock->Domkhar_Village;
            et.province->Bumthang:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                         Period[1:Real, 24:Real]];
            et.capitalDistance->0.5:Real;
```

```
            hs.description-> "The Dzong ruins of Chheokhor Deb(Leader) is about an
            hours walk from the road head at Zhabjethang.";
            hs.contact->" ";
            hs.schedule-> "12 months";
            hs.relatedTo->Mongar_Dzong:Fortress).


accommodation(Kaila:Guest_house^
               hs.url->"www.kaila.bt/";
               et.rating->2:Real;
               et.minPrice->400:Real;
               et.subblock-> Chamkhar:Town;
               et.province->Bumthang:Province;
               hs.telecoms->Telecoms[
                  et.landline->9753631219;
                  et.cell->97517682948];
               hs.contact->"manager@kaila.bt";
               hs.relatedTo->Yangphel:Guest_house).


accommodation(Wangdicholing_Lodge:Lodge^
               hs.url->"www.wangdicholing.bt/";
               et.rating->3:Real;
               et.minPrice->800:Real;
               et.subblock->Chamkhar:Town;
               et.province->Bumthang:Province;
               hs.telecoms->Telecoms[
                  et.landline->9753631452;
                  et.cell->97517682948];
               hs.contact->"manager@wangdicholing.bt";
               hs.relatedTo->Yangphel:Guest_house).


accommodation(Yangphel:Guest_house^
               hs.url->"www.yangphel.bt/";
               et.rating->2:Real;
               et.minPrice->500:Real;
               et.subblock->Tamshing_Village:Village;
               et.province->Bumthang:Province;
               hs.telecoms->Telecoms[
                  et.landline->9753631452;
                  et.cell->97517682948];
               hs.contact->"manager@yangphelbum.bt";
               hs.relatedTo->Wangdicholing_Lodge:Lodge).

accommodation(Hotel_Home:Hotel^
               hs.url->"www.hotelhome.bt/";
               et.rating->2:Real;
               et.minPrice->350:Real;
               et.subblock->Samcholing:Village;
               et.province->Bumthang:Province;
               hs.telecoms->Telecoms[
                  et.landline->9753631452;
                  et.cell->97517682948];
               hs.contact->"manager@hotelhome.bt";
```

```
              hs.relatedTo->Kaila:Guest_house).

accommodation(Swiss:Guest_house^
              hs.url->"www.swissbumthang.bt/";
              et.rating->3:Real;
              et.minPrice->1300:Real;
              et.subblock->Tharpaling:Village;
              et.province->Bumthang:Province;
              hs.telecoms->Telecoms[
                 et.landline->9753631452;
                 et.cell->97517682948];
              hs.contact->"manager@swissbumthang.bt";
              hs.relatedTo->Aryazamlha:Guest_house).

accommodation(Aryazamlha:Guest_house^
              hs.url->"www.Aryazamlha.com/";
              et.rating->3:Real;
              et.minPrice->1500:Real;
              et.subblock->Ura_Village:Village;
              et.province->Bumthang:Province;
              hs.telecoms->Telecoms[
                 et.landline->97555252464;
                 et.cell->97517682948];
              hs.contact->"manager@Aryazamlha.bt";
              hs.relatedTo->Swiss:Guest_house).

event(Tamshingphala_Choepa:Traditional_festival^
      hs.description->"One of the most amazing festivals in Bumthang";
      hs.startDate->date[2008:Real,10:Real,08:Real];
      hs.endDate->date[2008:Real,10:Real,10:Real];
      et.theme->Cultural_Religious_Heritage;
      hs.location->Tamshing_Lhakhang:Temple;
      et.province->Bumthang:Province;
      hs.url->" ";
      hs.relatedTo->Tangbi_Mani:Traditional_festival).

event(Tangbi_Mani:Traditional_festival^
      hs.description->"One of the most amazing festivals in Bumthang";
      hs.startDate->date[2008:Real,10:Real,13:Real];
      hs.endDate->date[2008:Real,10:Real,15:Real];
      et.theme->Cultural_Religious_Heritage;
      hs.location->Tangbi_Monastery:Monastery;
      et.province->Bumthang:Province;
      hs.url->" ";
      hs.relatedTo->Wangdue_Tshechu:Annual_festival).


event(Jambay_Lhakhang_Drup:Annual_festival^
      hs.description->"One of the most amazing festivals in Bumthang";
      hs.startDate->date[2008:Real,11:Real,12:Real];
      hs.endDate->date[2008:Real,11:Real,16:Real];
      et.theme->Cultural_Religious_Heritage;
      hs.location->Jambay_Lhakhang:Temple;
      et.province->Bumthang:Province;
      hs.url->" ";
```

```
      hs.relatedTo->Mongar_Tshechu:Annual_festival).


event(Prakhar_Duchhoed:Annual_festival^
      hs.description->"One of the most amazing festivals in Bumthang";
      hs.startDate->date[2008:Real,11:Real,13:Real];
      hs.endDate->date[2008:Real,11:Real,15:Real];
      et.theme->Cultural_Religious_Heritage;
      hs.location->Prakar_Lhakhang:Temple;
      et.province->Bumthang:Province;
      hs.url->" ";
      hs.relatedTo->Mongar_Tshechu:Annual_festival).

event(Nalakhar_Tshechu:Annual_festival^
      hs.description->"One of the most amazing festivals in Bumthang";
      hs.startDate->date[2008:Real,12:Real,12:Real];
      hs.endDate->date[2008:Real,12:Real,14:Real];
      et.theme->Cultural_Religious_Heritage;
      hs.location->Nalakhar_Lhakhang:Temple;
      et.province->Bumthang:Province;
      hs.url->" ";
      hs.relatedTo->National_day:National_festival).


event(Ura_Yakchoe:Annual_festival^
      hs.description->"One of the most amazing festivals in Bumthang";
      hs.startDate->date[2008:Real,04:Real,16:Real];
      hs.endDate->date[2008:Real,04:Real,20:Real];
      et.theme->Cultural_Religious_Heritage;
      hs.location->Ura_Lhakhang:Temple;
      et.province->Bumthang:Province;
      hs.url->" ";
      hs.relatedTo->National_Film_Festival:Film_festival).


event(Nimalung_Tshechu:Annual_festival^
      hs.description->"One of the most amazing festivals in Bumthang";
      hs.startDate->date[2008:Real,07:Real,10:Real];
      hs.endDate->date[2008:Real,07:Real,12:Real];
      et.theme->Cultural_Religious_Heritage;
      hs.location->Nimalung_Dratshang:Temple_fortress;
      et.province->Bumthang:Province;
      hs.url->" ";
      hs.relatedTo->Kurjey_Tshechu:Annual_festival).


event(Kurjey_Tshechu:Annual_festival^
      hs.description->"One of the most amazing festivals in Bumthang";
      hs.startDate->date[2008:Real,07:Real,12:Real];
      hs.endDate->date[2008:Real,07:Real,13:Real];
      et.theme->Cultural_Religious_Heritage;
      hs.location->Kurjey_Lhakhang:Temple;
      et.province->Bumthang:Province;
      hs.url->" ";
      hs.relatedTo->Yangphel_Archery_Tournament:Sport_archery).
```

```
% FOAF-Profile of Attractions, Accommodations, and Events in Chukha Province
 province(Chukha:Province^
          hs.url->"www.chukha.gov.bt";
          et.capital->Gedu:Town;
          et.area->"1,819 sq.km";
          et.elevation-> "1,300 to 7300 meters";
          et.numBlocks->6:Integer;
          et.numAttractions->1:Integer;
          et.numEvents->1:Integer;
          et.numAccommodations->4:Integer;
          hs.languagesSpoken->"Dzongkha";
          hs.contact->"admchukha@druknet.bt";
          hs.description->"Thimphu is the capital of Bhutan";
          hs.relatedTo->Paro:Province).

 attraction(Chukha_Dzong:Fortress^
            hs.url->" ";
            et.subblock->Chukha_Town:Town;
            et.province->Chukha:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                          Period[9:Real, 16:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"It is one of the most beautiful attractions in the
            Southern region";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->Rinpung_Dzong:Fortress).

 accommodation(Hotel_Druk_Phuentsholing:Hotel^
               hs.url->"www.drukhotels.com/";
               et.rating->4:Real;
               et.minPrice->2700:Real;
               et.subblock->Phuentsholing_Upper_Town:Town;
               et.province->Chukha:Province;
               hs.telecoms->Telecoms[
                  et.landline->9753631452;
                  et.cell->97517682948];
               hs.contact->"manager@hoteldruk.bt";
               hs.relatedTo->Hotel_Namgay:Hotel).

 accommodation(Hotel_Namgay:Hotel^
               hs.url->"www.hotelNamgay.bt/";
               et.rating->3:Real;
               et.minPrice->1800:Real;
               et.subblock->Phuentsholing_Upper_Town:Town;
               et.province->Chukha:Province;
               hs.telecoms->Telecoms[
                  et.landline->9753631452;
               et.cell->97517682948];
               hs.contact->"manager@hoteldruk.bt";
               hs.relatedTo->Central_Hotel:Hotel).
```

```
accommodation(Central_Hotel:Hotel^
              hs.url->"www.CentralHotel.bt/";
              et.rating->2:Real;
              et.minPrice->800:Real;
              et.subblock->Phuentsholing_Upper_Town:Town;
              et.province->Chukha:Province;
              hs.telecoms->Telecoms[
                 et.landline->9753631452;
                 et.cell->97517682948];
              hs.contact->"manager@hoteldruk.bt";
              hs.relatedTo->Hotel_Namgay:Hotel).

accommodation(Blue_Dragon:Hotel^
              hs.url->"www.hotelBlueDragon.bt/";
              et.rating->1:Real;
              et.minPrice->350:Real;
              et.subblock->Phuentsholing_Upper_Town:Town;
              et.province->Chukha:Province;
              hs.telecoms->Telecoms[
                 et.landline->9753631452;
                 et.cell->97517682948];
              hs.contact->"manager@hoteldruk.bt";
              hs.relatedTo->Central_Hotel:Hotel).

event(Chukha_Tshechu:Annual_festival^
      hs.description->"One of the most amazing festivals in Chukha";
      hs.startDate->date[2008:Real,03:Real,19:Real];
      hs.endDate->date[2008:Real,03:Real,21:Real];
      et.theme->Cultural_Religious_Heritage;
      hs.location->Chukha_Dzong:Fortress;
      et.province->Chukha:Province;
      hs.url->" ";
      hs.relatedTo->Ura_Yakchoe:Annual_festival).

event(Yangphel_Archery_Tournament:Sport_archery^
      hs.description->"11TH Yangphel open archery tournament";
      hs.startDate->date[2008:Real,08:Real,23:Real];
      hs.endDate->date[2008:Real,10:Real,02:Real];
      et.theme->Recreation;
      hs.location->City_park:National_park;
      et.province->Chukha:Province;
      hs.url->"http://www.bhutanarchery.com/default.asp";
      hs.relatedTo->"Thimphu_Drupchen:Annual_festival").

% FOAF-Profile of Attractions, Accommodations, and Events in Mongar Province
province(Mongar:Province^
         hs.url->"www.mongar.gov.bt";
         et.capital->Mongar_Town:Town;
         et.area->"1,819 sq.km";
         et.elevation-> "1,300 to 7300 meters";
         et.numBlocks->5:Integer;
         et.numAttractions->1:Integer;
         et.numEvents->1:Integer;
         et.numAccommodations->0:Integer;
         hs.languagesSpoken->"Dzongkha,Sharchopa";
```

```
          hs.contact->"admmongar@druknet.bt";
          hs.description->"The largest province by size and popolution";
          hs.relatedTo->Trashigang:Province).

 attraction(Mongar_Dzong:Fortress^
            hs.url->" ";
            et.subblock->Mongar_Town:Town;
            et.province->Mongar:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                          Period[9:Real, 16:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"It is a beautiful fortress";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->Trashigang_Dzong:Fortress).


 event(Mongar_Tshechu:Annual_festival^
       hs.description->"One of the most amazing festivals in Mongar";
       hs.startDate->date[2008:Real,12:Real,04:Real];
       hs.endDate->date[2008:Real,12:Real,07:Real];
       et.theme->Cultural_Religious_Heritage;
       hs.location->Mongar_Dzong:Fortress;
       et.province->Mongar:Province;
       hs.url->" ";
       hs.relatedTo->Nalakhar_Tshechu:Annual_festival).



% FOAF-Profile of Attractions, Accommodations, and Events in Paro Province
 province(Paro:Province^
          hs.url->"www.paro.gov.bt";
          et.capital->Paro_Town:Town;
          et.area->"1,819 sq.km";
          et.elevation-> "1,300 to 7300 meters";
          et.numBlocks->10:Integer;
          et.numAttractions->3:Integer;
          et.numEvents->1:Integer;
          et.numAccommodations->3:Integer;
          hs.languagesSpoken->"Dzongkha";
          hs.contact->"admparo@druknet.bt";
          hs.description->"Thimphu is the capital of Bhutan";
          hs.relatedTo->Thimphu:Province).

 attraction(Rinpung_Dzong:Fortress^
            hs.url->" ";
            et.subblock->Phatsana:Village;
            et.province->Paro:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                          Period[9:Real, 16:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"It is a beautiful attraction in the Western region";
            hs.contact->" ";
```

```
            hs.schedule->"12 months";
            hs.relatedTo->Taktshang:Monastery).

attraction(Taktshang:Monastery^
            hs.url->" ";
            et.subblock->Tshento_Shari:Village;
            et.province->Paro:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                          Period[8:Real, 18:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"One of the seven wonders in the world";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->Ta_Dzong:National_museum).

attraction(Ta_Dzong:National_museum^
            hs.url->"www.nationalmuseum.gov.bt/";
            et.subblock->Goepay:Village;
            et.province->Paro:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Tue, Wed, Thu, Fri, Sat, Sun],
                          Period[10:Real, 16:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"It is the biggest and the oldest museum in Bhutan";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->Tashichoe_Dzong:Fortress).

accommodation(Aman:Resort^
               hs.url->"www.amanresorts.com/amankora/home.aspx";
               et.rating->5:Real;
               et.minPrice->4500:Real;
               et.subblock->Tsento_Shari:Village;
               et.province->Paro:Province;
               hs.telecoms->Telecoms[
                  et.landline->9758211452;
                  et.cell->97517682948];
               hs.contact->"manager@AmanResort.bt";
               hs.relatedTo->Tiger_Nest:Resort).

accommodation(Tiger_Nest:Resort^
               hs.url->"www.TigerNest.bt";
               et.rating->3:Real;
               et.minPrice->2500:Real;
               et.subblock->Tsento_Shari:Village;
               et.province->Paro:Province;
               hs.telecoms->Telecoms[
                  et.landline->9758211452;
                  et.cell->97517682948];
               hs.contact->"manager@AmanResort.bt";
               hs.relatedTo->Kichu_Resort:Resort).

accommodation(Rangen:Resort^
               hs.url->"www.rangen.bt";
```

```
                et.rating->2:Real;
                et.minPrice->1000:Real;
                et.subblock->Tsento_Shari:Village;
                et.province->Paro:Province;
                hs.telecoms->Telecoms[
                   et.landline->9758211452;
                   et.cell->97517682948];
                hs.contact->"manager@rangen.bt";
                hs.relatedTo->Holiday_Home:Hotel).

 accommodation(KichuResort^
                hs.url->"www.kichuResort.bt/";
                et.rating->3:Real;
                et.minPrice->2000:Real;
                et.subblock->Lamgong:Village;
                et.province->Paro:Province;
                hs.telecoms->Telecoms[
                   et.landline->9758211452;
                   et.cell->97517682948];
                hs.contact->"manager@kichuResort.bt";
                hs.relatedTo->"http://www.TigerNest.bt").


 accommodation(Holiday_Home:Hotel^
                hs.url->"www.kichuResort.bt";
                et.rating->3:Real;
                et.minPrice->2000:Real;
                et.subblock->Gaptey:Village;
                et.province->Paro:Province;
                hs.telecoms->Telecoms[
                et.landline->9758211452;
                et.cell->97517682948];
                hs.contact->"manager@kichuResort.bt";
                hs.relatedTo->Rangen:Resort).


 event(Paro_Tshechu:Annual_festival^
       hs.description->"Paro Tshechu (Festival) is one of most important festival
       of Bhutan which is in the start of spring. The most magnificient
       thing about it is the unraveling of a large Thanka (Thongdrol).";
       hs.startDate->date[2008:Real,03:Real,17:Real];
       hs.endDate->date[2008:Real,03:Real,21:Real];
       et.theme->Cultural_Religious_Heritage;
       hs.location->Rinpung_Dzong:Fortress;
       et.province->Paro:Province;
       hs.url->" ";
       hs.relatedTo->"Ura_Yakchoe:Annual_festival").

% FOAF-Profile of Attractions, Accommodations, and Events in Thimphu Province
 province(Thimphu:Province^
          hs.url->"http://www.thimphu.gov.bt";
          et.capital->Thimphu_City:City;
          et.area->"1,819 sq.km";
          et.elevation-> "1,300 to 7300 meters";
          et.numBlocks->10:Integer;
```

```
          et.numAttractions->3:Integer;
          et.numEvents->1:Integer;
          et.numAccommodations->3:Integer;
          hs.languagesSpoken->"Dzongkha";
          hs.contact->"admthimphu@druknet.bt";
          hs.description->"Thimphu is the capital of Bhutan";
          hs.relatedTo->Punakha:Province).

attraction(Tashichoe_Dzong:Fortressˆ
            hs.url->" ";
            et.subblock->Jongshina:Town;
            et.province->Thimphu:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                          Period[9:Real, 16:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"One of the most beautiful attractions in Thimphu";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo-> "Memorial_Chorten:Temple").

attraction(Memorial_Chorten:Templeˆ
            hs.url->" ";
            et.subblock->Thimphu_City:City;
            et.province->Thimphu:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                          Period[9:Real, 16:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"One of the most beautiful attractions in Thimphu";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->"Botanical_Garden:Garden").


attraction(Botanical_Garden:Gardenˆ
            hs.url->" ";
            et.subblock->Barbesa:Town;
            et.province->Thimphu:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                          Period[9:Real, 16:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"One of the most beautiful attractions in Thimphu";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->"Punakha_Dzong:Fortress").

accommodation(River_View:Hotelˆ
               hs.url->"www.riverview.bt";
               et.rating->4:Real;
               et.minPrice->4000:Real;
               et.subblock->Thimphu_City:City;
               et.province->Thimphu:Province;
               hs.telecoms->Telecom[|
```

```
                et.landline->9758375133;
                et.cell->97517616363];
            hs.contact->"manager@druknet.bt";
            hs.relatedTo->Tandin:Hotel).

accommodation(Tandin:Hotel^
            hs.url->"www.tandin.bt/";
            et.rating->2:Real;
            et.minPrice->1500:Real;
            et.subblock->Thimphu_City:City;
            et.province->Thimphu:Province;
            hs.telecoms->Telecom[
                et.landline->9758375143;
                et.cell->97517663421];
            hs.contact->"manager@druknet.bt";
            hs.relatedTo->River_View:Hotel).

accommodation(Hotel_Druk_Thimphu:Hotel^
            hs.url->"www.drukhotels.com/";
            et.rating->4:Real;
            et.minPrice->2700:Real;
            et.subblock->Thimphu_City:City;
            et.province->Thimphu:Province;
            hs.telecoms->Telecoms[
                et.landline->9753631452;
                et.cell->97517682948];
            hs.contact->"manager@hoteldruk.bt";
            hs.relatedTo->River_View:Hotel).

event(Thimphu_Drupchen:Annual_festival^
      hs.description->"It is a popular festival in Thimphu";
      hs.startDate->date[2008:Real,10:Real,04:Real];
      hs.endDate->date[2008:Real,10:Real,08:Real];
      et.theme->Cultural_Religious_Heritage;
      hs.location->Tashichoe_Dzong:Fortress;
      et.province->Thimphu:Province;
      hs.url->" ";
      hs.relatedTo->Thimphu_Tshechu:Annual_festival).

event(Thimphu_Tshechu:Annual_festival^
      hs.description->"It is a popular festival in Thimphu";
      hs.startDate->date[2008:Real,10:Real,09:Real];
      hs.endDate->date[2008:Real,10:Real,11:Real];
      et.theme->Cultural_Religious_Heritage;
      hs.location->Tashichoe_Dzong:Fortress;
      et.province->Thimphu:Province;
      hs.url->" ";
      hs.relatedTo->Tangbi_Mani:Traditional_festival).

event(GNH_Conference:Events^
      hs.description->"It is the 4th International Conference on Gross
      National Happiness.";
      hs.startDate->date[2008:Real,11:Real,24:Real];
      hs.endDate->date[2008:Real,11:Real,26:Real];
      et.theme->Cultural_Religious_Heritage;
```

```
      hs.location->Tashichoe_Dzong:Fortress;
      et.province->Thimphu:Province;
      hs.url->"http://www.pc.gov.bt/gnh.asp";
      hs.relatedTo->Mongar_Tshechu:Annual_festival).


event(National_Film_Festival:Film_festival^
      hs.description->"Film festival held every year";
      hs.startDate->date[2008:Real,05:Real,02:Real];
      hs.endDate->date[2008:Real,05:Real,04:Real];
      et.theme->Cultural_Religious_Heritage;
      hs.location-> Clock_Tower_Square:Stadium;
      et.province->Thimphu:Province;
      hs.url->" ";
      hs.relatedTo->Nimalung_Tshechu:Annual_festival).

event(Fifth_King_Birth_Anniversary:National_festival^
      hs.description->"King Jigme Khesar Namgyel Wangchuck's birthday";
      hs.startDate->date[2008:Real,02:Real,21:Real];
      hs.endDate->date[2008:Real,02:Real,21:Real];
      et.theme->Cultural_Religious_Heritage;
      hs.location->Changlimithang_stadium:Stadium;
      et.province->Thimphu:Province;
      hs.url->" ";
      hs.relatedTo->Paro_Tshechu:Annual_festival).

event(National_day:National_festival^
      hs.description->"The first hereditary King (Gongsa Ugen Wangchuck) of
      Bhutan was installed on this day in 1907.";
      hs.startDate->date[2008:Real,12:Real,17:Real];
      hs.endDate->date[2008:Real,12:Real,17:Real];
      et.theme->Cultural_Religious_Heritage;
      hs.location-> Changlimithang_stadium:Theatre;
      et.province->Thimphu:Province;
      hs.url->" ";
      hs.relatedTo->Fifth_King_Birth_Anniversary:National_festival).


event(Sunday_market:Market_weekly^
      hs.description->"Sunday farmer's market for fresh organic fruits and
      vegetable shopping.";
      hs.startDate->" ";
      hs.endDate->" ";
      et.theme->Cultural_Religious_Heritage;
      hs.location->Farmers_market:Market_weekly;
      et.province->Thimphu:Province;
      hs.url->" ";
      hs.relatedTo->Sunday_Market_handicraft:Market_handicraft).


event(Space_34:Nightlife_party^
      hs.description->"It is a nice cozy place to hang out with friends.";
      hs.startDate->" ";
      hs.endDate->" ";
      et.theme->Recreation;
```

```
      hs.location->eWorld_Building:Modern_buildings;
      et.province->Thimphu:Province;
      hs.url->" ";
      hs.relatedTo->Bhutan_National_Film_Festival:Film_festival).

% FOAF-Profile of Attractions, Accommodations, and Events in Punakha Province
 province(Punakha:Province^
         hs.url->"www.punakha.gov.bt";
         et.capital->Paro_Town:Town;
         et.area->"1,819 sq.km";
         et.elevation-> "1,300 to 7300 meters";
         et.numBlocks->6:Integer;
         et.numAttractions->2:Integer;
         et.numEvents->1:Integer;
         et.numAccommodations->0:Integer;
         hs.languagesSpoken->"Dzongkha";
         hs.contact->"admpunakha@druknet.bt";
         hs.description->"Thimphu is the capital of Bhutan";
         hs.relatedTo->WangduePhodrang:Province).

 attraction(Punakha_Dzong:Fortress^
           hs.url->" ";
           et.subblock->Thangzona:Village;
           et.province->Punakha:Province;
           et.theme->Cultural_Religious_Heritage;
           et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                        Period[9:Real, 16:Real]];
           et.capitalDistance->0.5:Real;
           hs.description->"It is the head of the monastic body";
           hs.contact->" ";
           hs.schedule->"12 months";
           hs.relatedTo->Chimi_Lhakhang:Temple).

 attraction(Chimi_Lhakhang:Temple^
           hs.url->" ";
           et.subblock->Chimithang:Village;
           et.province->Punakha:Province;
           et.theme->Cultural_Religious_Heritage;
           et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                        Period[9:Real, 16:Real]];
           et.capitalDistance->0.5:Real;
           hs.description->"It is a beautiful temple";
           hs.contact->" ";
           hs.schedule->"12 months";
           hs.relatedTo->Punakha_Dzong:Fortress).


 event(Punakha_Dromche:Annual_festival^
       hs.description->"Punakha Dromche take place in the first month of the
       lunar year and ends with 'Serda', a magnificent procession which re-enacts
       an episode of the war against the Tibetan in the 17th century.";
       hs.startDate->date[2008:Real,02:Real,11:Real];
       hs.endDate->date[2008:Real,02:Real,15:Real];
       et.theme->Cultural_Religious_Heritage;
       hs.location->Punakha_Dzong:Fortress;
```

```
      et.province->Punakha:Province;
      hs.url->" ";
      hs.relatedTo->Wangdue_Dzong:Fortress).


% FOAF-Profile of Attractions, Accommodations, and Events in WangduePhodrang
 province(WangduePhodrang:Province^
          hs.url->"www.wangduephodrang.gov.bt";
          et.capital->Wangdue_Town:Town;
          et.area->"1,819 sq.km";
          et.elevation->"1,300 to 7300 meters";
          et.numBlocks->8:Integer;
          et.numAttractions->1:Integer;
          et.numEvents->1:Integer;
          et.numAccommodations->0:Integer;
          hs.languagesSpoken->"Dzongkha";
          hs.contact->"admtrongsa@druknet.bt";
          hs.description->"Thimphu is the capital of Bhutan";
          hs.relatedTo->Trongsa:Province).

 attraction(Wangdue_Dzong:Fortress^
            hs.url->" ";
            et.subblock->Wangdue_Town:Town;
            et.province->WangduePhodrang:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                          Period[9:Real, 16:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"It is one of the most beautiful attractions";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->Trongsa_Dzong:Fortress).

 event(Wangdue_Tshechu:Annual_festival^
       hs.description->"Very popular festival in western Bhutan";
       hs.startDate->date[2008:Real, 10:Real, 20:Real];
       hs.endDate->date[2008:Real, 10:Real, 29:Real];
       et.theme->Cultural_Religious_Heritage;
       hs.location->Wangdue_Dzong:Fortress;
       et.province->WangduePhodrang:Province;
       hs.url->" ";
       hs.relatedTo->Jambay_Lhakhang_Drup:Annual_festival).

% FOAF-Profile of Attractions, Accommodations, and Events in Trongsa Province
 province(Trongsa:Province^
          hs.url->"www.trongsa.gov.bt";
          et.capital->Samcholing:Village;
          et.area->"1,819 sq.km";
          et.elevation->"1,300 to 7300 meters";
          et.numBlocks->10:Integer;
          et.numAttractions->2:Integer;
          et.numEvents->1:Integer;
          et.numAccommodations->0:Integer;
          hs.languagesSpoken->"Dzongkha";
          hs.contact->"admtrongsa@druknet.bt";
```

```
          hs.description->"Thimphu is the capital of Bhutan";
          hs.relatedTo->Bumthang:Province).

 attraction(Trongsa_Dzong:Fortress^
            hs.url->" ";
            et.subblock->Samcholing:Village;
            et.province->Trongsa:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                          Period[9:Real, 16:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"The biggest fortress built by Zhabdrung Rimpochhe";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo-> Samchholing_Palace:Popular_architecture ).

 attraction(Samchholing_Palace:Popular_architecture^
            hs.url->" ";
            et.subblock->Samcholing:Village;
            et.province->Trongsa:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Wed, Fri, Sun],
                          Period[8:Real, 15:Real]];
            et.capitalDistance->0.5:Real;
            hs.description-> "Samchholing was built by the Second King.
            It is about to be in ruins,but still has a beautiful
            architecture.";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->Bumthang_Dzong:Fortress).



% FOAF-Profile of Attractions and Events in Trashigang Province
 province(Trashigang:Province^
          hs.url->"www.trashigang.gov.bt";
          et.capital->Pam_Town:Town;
          et.area->"1,819 sq.km";
          et.elevation-> "1,300 to 7300 meters";
          et.numBlocks->10:Integer;
          et.numAttractions->1:Integer;
          et.numEvents->1:Integer;
          et.numAccommodations->0:Integer;
          hs.languagesSpoken->"Dzongkha";
          hs.contact->"admtrashigang@druknet.bt";
          hs.description->"Thimphu is the capital of Bhutan";
          hs.relatedTo->TrashiYangtse:Province).

 attraction(Trashigang_Dzong:Fortress^
            hs.url->" ";
            et.subblock->Pam_Town:Town;
            et.province->Tashigang:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                          Period[9:Real, 16:Real]];
```

```
            et.capitalDistance->0.5:Real;
            hs.description->"It is a beautiful fortress";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->ChortenKora:Temple).


% FOAF-Profile of Attractions, Accommodations, and Events in TrashiYangtse Province
 province(TrashiYangtse:Province^
          hs.url->"www.trashiyangtse.gov.bt/";
          et.capital->Gom_Kora:Town;
          et.area->"1,819 sq.km";
          et.elevation->"1,300 to 7300 meters";
          et.numBlocks->8:Integer;
          et.numAttractions->2:Integer;
          et.numEvents->2:Integer;
          et.numAccommodations->0:Integer;
          hs.languagesSpoken->"Dzongkha";
          hs.contact->"admtrashiyangtse@druknet.bt";
          hs.description->"Thimphu is the capital of Bhutan";
          hs.relatedTo->" ").

 attraction(ChortenKora:Temple^
            hs.url->" ";
            et.subblock->Gom_Kora:Town;
            et.province->TrashiYangtse:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                          Period[9:Real, 16:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"It is an attractive temple";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->Zorig_Chusum:Crafts_show).

 attraction(Zorig_Chusum:Crafts_show^
            hs.url->" ";
            hs.subblock->Gom_Kora:Town;
            et.province->TrashiYangtse:Province;
            et.theme->Cultural_Religious_Heritage;
            et.open->Open[DaysOfWeek[Mon, Tue, Wed, Thu, Fri, Sat, Sun],
                          Period[9:Real, 18:Real]];
            et.capitalDistance->0.5:Real;
            hs.description->"It was established in 1997 as a Rigney School and has
            5 areas of studies: Painting, Trezo (Silversmith), Embroidery,
            Patra(sculptures), and Lacquering.";
            hs.contact->" ";
            hs.schedule->"12 months";
            hs.relatedTo->" ").
```

Appendix C: Partonomy and Distance Computation KB

The locations of tourist entities and the predicates for route computation between provinces are given below:

Partonomy Rules

```
% Partonomy rules to structure the subparts of a country.
% Domain:Bhutan

% partOf(?Region, ?Country).

 partOfCountry(Western:Region, Bhutan:Country).
 partOfCountry(Central:Region, Bhutan:Country).
 partOfCountry(Eastern:Region, Bhutan:Country).
 partOfCountry(Southern:Region, Bhutan:Country).

% partOf(?Province, ?Region).
 partOfRegion(Trongsa:Province, Central:Region).
 partOfRegion(Bumthang:Province, Central:Region).
 partOfRegion(Zhemgang:Province, Central:Region).

 partOfRegion(Lhuntse:Province, Eastern:Region).
 partOfRegion(Mongar:Province, Eastern:Region).
 partOfRegion(Trashigang:Province, Eastern:Region).
 partOfRegion(PemaGatshel:Province, Eastern:Region).
 partOfRegion(TrashiYangtse:Province, Eastern:Region).
 partOfRegion(SamdrupJongkhar:Province, Eastern:Region).

 partOfRegion(Thimphu:Province, Western:Region).
 partOfRegion(Haa:Province, Western:Region).
 partOfRegion(Gasa:Province, Western:Region).
 partOfRegion(Paro:Province, Western:Region).
 partOfRegion(Punakha:Province, Western:Region).
 partOfRegion(Dagana:Province, Western:Region).
 partOfRegion(WangduePhodrang:Province, Western:Region).

 partOfRegion(Samtse:Province, Southern:Region).
 partOfRegion(Chukha:Province, Southern:Region).
 partOfRegion(Sarpang:Province, Southern:Region).
 partOfRegion(Gelephu:Province, Southern:Region).
 partOfRegion(Damphu:Province, Southern:Region).
```

```
% partOf(?Block, ?Province).
 partOfProvince(Chhoekhor:Block, Bumthang:Province).
 partOfProvince(Chhumme:Block, Bumthang:Province).
 partOfProvince(Tang:Block, Bumthang:Province).
 partOfProvince(Ura:Block, Bumthang:Province).

 partOfProvince(Athang:Block, WangduePhodrang:Province).
 partOfProvince(Daga:Block, WangduePhodrang:Province).
 partOfProvince(Dangchu:Block, WangduePhodrang:Province).
 partOfProvince(Gangtey:Block, WangduePhodrang:Province).
 partOfProvince(Gasetshog_Gom:Block, WangduePhodrang:Province).
 partOfProvince(Gasetshog_Wom:Block, WangduePhodrang:Province).
 partOfProvince(Kazhi:Block, WangduePhodrang:Province).
 partOfProvince(Nahi:Block, WangduePhodrang:Province).
 partOfProvince(Nyisho:Block, WangduePhodrang:Province).
 partOfProvince(Phangyul:Block, WangduePhodrang:Province).
 partOfProvince(Phobjikha:Block, WangduePhodrang:Province).
 partOfProvince(Rubeisa:Block, WangduePhodrang:Province).
 partOfProvince(Sephu:Block, WangduePhodrang:Province).
 partOfProvince(Thedtsho:Block, WangduePhodrang:Province).

 partOfProvince(Baap:Block, Thimphu:Province).
 partOfProvince(Chang:Block, Thimphu:Province).
 partOfProvince(Dagala:Block, Thimphu:Province).
 partOfProvince(Geney:Block, Thimphu:Province).
 partOfProvince(Kawang:Block, Thimphu:Province).
 partOfProvince(Lingzhi:Block, Thimphu:Province).
 partOfProvince(Mewang:Block, Thimphu:Province).
 partOfProvince(Naro:Block, Thimphu:Province).
 partOfProvince(Soe:Block, Thimphu:Province).
 partOfProvince(Toep:Block, Thimphu:Province).

 partOfProvince(Dzomi:Block, Punakha:Province).
 partOfProvince(Goenshari:Block, Punakha:Province).
 partOfProvince(Guma:Block, Punakha:Province).
 partOfProvince(Kabjisa:Block, Punakha:Province).
 partOfProvince(Lingmukha:Block, Punakha:Province).
 partOfProvince(Shengana:Block, Punakha:Province).
 partOfProvince(Talo:Block, Punakha:Province).
 partOfProvince(Toewang:Block, Punakha:Province).

 partOfProvince(Dogar:Block, Paro:Province).
 partOfProvince(Dopshari:Block, Paro:Province).
 partOfProvince(Doteng:Block, Paro:Province).
 partOfProvince(Hungrel:Block, Paro:Province).
 partOfProvince(Lamgong:Block, Paro:Province).
 partOfProvince(Lungnyi:Block, Paro:Province).
 partOfProvince(Naja:Block, Paro:Province).
 partOfProvince(Shaba:Block, Paro:Province).
 partOfProvince(Tsento:Block, Paro:Province).
 partOfProvince(Wangchang:Block, Paro:Province).

 partOfProvince(Gelling:Block, Chukha:Province).
 partOfProvince(Phuentsholing:Block, Chukha:Province).
```

```
 partOfProvince(Mongar_Gewog:Block, Mongar:Province).
 partOfProvince(Pam:Block, Trashigang:Province).
 partOfProvince(Yangtse:Block,TrashiYangtse:Province).
 partOfProvince(Samtengang:Block,Trongsa:Province).

% partOf(?Subblock, ?Block).
 partOfBlock(Buli:Village, Chhoekhor:Block).
 partOfBlock(Tharpaling:Village, Chhoekhor:Block).
 partOfBlock(Prakar:Village, Chhoekhor:Block).
 partOfBlock(Nimalung:Village, Chhoekhor:Block).
 partOfBlock(Domkhar:Village, Chhoekhor:Block).
 partOfBlock(Chamkhar:Town, Chhoekhor:Block).
 partOfBlock(Tangbi:Village, Chhoekhor:Block).
 partOfBlock(Ura_Village:Village, Ura:Block).
 partOfBlock(Tamshing_Village:Village, Chhumme:Block).
 partOfBlock(Domkhar:Village, Chhumme:Block).
 partOfBlock(Nalakhar:Village, Chhumme:Block).

 partOfBlock(Hungrel:Village, Hungrel:Block).
 partOfBlock(Goepay:Village, Hungrel:Block).
 partOfBlock(Gaptey:Village, Wangchang:Block).
 partOfBlock(Tsento_Shari:Village, Tsento:Block).
 partOfBlock(Lamgong_Village:Village, Lamgong:Block).

 partOfBlock(Chukha_Town:Town,Gelling:Block).
 partOfBlock(Samcholing:Village,Samtengang:Block).

 partOfBlock(Jongshina:Town, Kawang:Block).
 partOfBlock(Thimphu_City:City, Chang:Block).
 partOfBlock(Chubachu:Town, Chang:Block).
 partOfBlock(Changlimithang:Town, Chang:Block).
 partOfBlock(Barbesa:Town, Baap:Block).

 partOfBlock(Thangzona:Village, Shengana:Block).
 partOfBlock(Wangdue_Town:Town, Gasetshog_Gom:Block).
 partOfBlock(Mongar_Town:Town, Mongar_Gewog:Block).
 partOfBlock(Pam_Town:Town, Pam:Block).
 partOfBlock(Gom_Kora:Town, Yangtse:Block).
 partOfBlock(Phuentsholing_Upper_Town:Town, Phuentsholing:Block).

% siteOf(?Attractions, ?Subblock).
 siteOf(Bumthang_Dzong:Fortress, Chamkhar:Town).
 siteOf(Zugney:Textiles, Chamkhar:Town).
 siteOf(Ugyen_Chholing_Museum:Local_museum, Chamkhar:Town).
 siteOf(Petseling_Gompa:Temple, Chamkhar:Town).
 siteOf(Prakar_Lhakhang:Temple, Prakar:Village).
 siteOf(Nimalung_Dratshang:Temple_fortress, Nimalung:Village).
 siteOf(Tamshing_Lhakhang:Temple, Tamshing_Village:Village).
 siteOf(Jambay_Lhakhang:Temple, Kurjey:Village).
 siteOf(Kurjey_Lhakhang:Temple, Kurjey:Village).
 siteOf(Tharpeling_Monastery:Monastery, Tharpeling:Village).
 siteOf(Thrasephel_Lhakhang:Temple, Tharpeling:Village).
 siteOf(Nalakhar_Lhakhang:Temple, Nalakhar:Village).
 siteOf(Domkhar_Zimchu:Palace, Domkhar:Village).
 siteOf(Darphel_Dzong:Ruins, Domkhar:Village).
```

```
 siteOf(Tangbi_Monastery:Monastery, Tangbi:Village).
 siteOf(Ura_Lhakhang:Temple, Ura_Village:Village).
 siteOf(Trumshingla_Park:National_park, Ura_Village:Village).
 siteOf(Chukha_Dzong:Fortress, Chukha_Town:Town).
 siteOf(City_park:National_park, Phuentsholing_Upper_Town:Town).
 siteOf(Rinpung_Dzong:Fortress, Hungrel:Village).
 siteOf(Ta_Dzong:National_museum, Goepay:Village).
 siteOf(Taktshang:Monastery, Tsento_Shari:Village).
 siteOf(Punakha_Dzong:Fortress, Thangzona:Village).
 siteOf(Dzongchu:Temple, Thangzona:Village).
 siteOf(Chimi_Lhakhang:Temple, Chimithang:Village).
 siteOf(Tashichoe_Dzong:Fortress, Jongshina:Town).
 siteOf(Memorial_Chorten:Temple, Thimphu_City:City).
 siteOf(Botanical_Garden:Garden, Barbesa:Town).
 siteOf(Trongsa_Dzong:Fortress, Samcholing:Village).
 siteOf(Mongar_Dzong:Fortress, Mongar_Town:Town).
 siteOf(Trashigang_Dzong:Fortress, Pam_Town:Town).

 siteOf(ChortenKora:Temple, Gom_Kora:Town).
 siteOf(Public_place:Theatre, Gom_Kora:Town).
 siteOf(Zorig_Chusum:Crafts_show, Gom_Kora:Town).
 siteOf(Wangdue_Dzong:Fortress, Wangdue_Town:Town).

% Event Venues
 siteOf(Yangtse_Theatre:Theatre, Gom_Kora:Town).
 siteOf(Changlimithang_stadium:Stadium, Thimphu_City:City).
 siteOf(Clock_Tower_Square:Stadium, Thimphu_City:City).
 siteOf(eWorld_Building:Modern_buildings, Thimphu_City:City).
 siteOf(Farmers_market:Market_weekly, Chubachu:Town).

% Accommodations in the subblock
 siteOf(Kaila:Guest_house, Chamkhar:Town).
 siteOf(Wangdicholing:Lodge, Chamkhar:Town).
 siteOf(Yangphel:Guest_house, Tamshing_Village:Village).
 siteOf(Hotel_Home:Hotel, Samcholing:Village).
 siteOf(Swiss:Guest_house, Tharpaling:Village).
 siteOf(Aryazamlha:Guest_house, Ura_Village:Village).
 siteOf(Hotel_Druk:Hotel, Phuentsholing_Upper_Town:Town).
 siteOf(Hotel_Namgay:Hotel, Phuentsholing_Upper_Town:Town).
 siteOf(Central_Hotel:Hotel, Phuentsholing_Upper_Town:Town).
 siteOf(Blue_Dragon:Hotel, Phuentsholing_Upper_Town:Town).
 siteOf(Aman:Resort, Tsento_Shari:Village).
 siteOf(Tiger_Nest:Resort,Tsento_Shari:Village).
 siteOf(Rangen:Resort, Tsento_Shari:Village).
 siteOf(Kichu_Resort:Resort, Lamgong_Village:Village).
 siteOf(Holiday_Home:Hotel, Gaptey:Village).
 siteOf(Tandin:Hotel, Thimphu_City:City).
 siteOf(Hotel_Druk_Thimphu:Hotel, Thimphu_City:City).
 siteOf(River_View:Hotel, Thimphu_City:City).
%-------------------------------------------------------------%
%Record of Tourist Entities in each Subblock.

% consistsOf(?Subblock, ?NumAtractions, ?NumAccommodations).

% Province Bumthang
```

```
 consistsOf(Tamshing_Village:Village, 1:Integer, 1:Integer).
 consistsOf(Tangbi:Village, 1:Integer, 0:Integer).
 consistsOf(Kurkey:Village, 2:Integer, 1:Integer).
 consistsOf(Prakar:Village, 1:Integer, 0:Integer).
 consistsOf(Nalakhar:Village, 1:Integer, 0:Integer).
 consistsOf(Ura_Village:Village,2:Integer, 1:Integer).
 consistsOf(Nimalung:Village, 1:Integer, 0:Integer).
 consistsOf(Chamkhar:Town, 4:Integer, 2:Integer).
 consistsOf(Samcholing:Village, 1:Integer, 1:Integer).
 consistsOf(Tharpeling:Village, 2:Integer, 1:Integer).
 consistsOf(Domkhar:Village, 2:Integer, 0:Integer).

% Province Mongar
 consistsOf(Mongar_Town:Town, 1:Integer, 0:Integer).

% Province Paro
 consistsOf(Goepay:Village, 1:Integer, 0:Integer).
 consistsOf(Hungrel:Village, 1:Integer, 0:Integer).
 consistsOf(Tsento_Shari:Village, 1:Integer, 0:Integer).

% Province Punakha
 consistsOf(Thangzona:Village, 2:Integer, 0:Integer).
 consistsOf(Chimithang:Village, 1:Integer, 0:Integer).

% Province Chukha
 consistsOf(Chukha_Town:Town, 1:Integer, 0:Integer).
 consistsOf(Phuentsholing_Upper_Town:Town, 1:Integer, 0:Integer).

% Province Thimphu
 consistsOf(Jongshina:Town, 1:Integer, 0:Integer).
 consistsOf(Chubachu:Town, 1:Integer, 0:Integer).
 consistsOf(Thimphu_City:City,2:Integer, 3:Integer).
 consistsOf(Barbesa:Town,1:Integer, 0:Integer).

% Province Trashigang
 consistsOf(Pam_Town:Town, 1:Integer, 0:Integer).

% Province Tashiyangtse
 consistsOf(Gom_Kora:Town,2:Integer, 0:Integer).

% Province Wangduephodrang
 consistsOf(Wangdue_Town:Town, 1:Integer, 0:Integer).

% Additional facts to test the KB with ground fact instances of
% partOf by locating attractions at any part of a country.

 partOfCountry(HopeWellRock:Nature, Canada:Country).
 partOfProvince(Niagra_fall:Water_fall, Toronto:Province).
 partOfProvince(National_Library:Popular_architecture, Thimphu:Province).

% Auxiliary predicates acting on the partonomy facts

getFullAddress( ?Location, [?Subblock, ?Block,?Province, ?Region, ?Country]):-
  siteOf(?Location, ?Subblock),
  partOfBlock(?Subblock, ?Block),
  partOfProvince(?Block, ?Province),
```

```
  partOfRegion(?Province, ?Region),
  partOfCountry(?Region, ?Country).

%To locate attractions at different sublevel of a country
%-----at a subblock level--------------------------------%

getAttraction(?Attraction:Attractions, ?Subblock:Subblock):-
  siteOf(?Attraction:Attractions, ?Subblock:Subblock).

%-----at a block level----------------------------------%

getAttraction(?Attraction:Attractions, ?Block:Block):-
  partOfBlock(?Attraction:Attractions, ?Block:Block).

getAttraction(?Attraction:Attractions, ?Block:Block):-
   partOfBlock(?Subblock:Subblock, ?Block:Block),
   getAttraction(?Attraction:Attractions, ?Subblock:Subblock).

%-----at a province level-------------------------------%

getAttraction(?Attraction:Attractions, ?Province:Province):-
  partOfProvince(?Attraction:Attractions, ?Province:Province).

getAttraction(?Attraction:Attractions, ?Province:Province):-
    partOfProvince(?Block:Block, ?Province:Province),
    getAttraction(?Attraction:Attractions, ?Block:Block).

%------at a region  level-------------------------------%

getAttraction(?Attraction:Attractions, ?Region:Region):-
  partOfRegion(?Attraction:Attractions, ?Region:Region).

getAttraction(?Attraction:Attractions, ?Region:Region):-
   partOfRegion(?Province:Province, ?Region:Region),
   getAttraction(?Attraction:Attractions, ?Province:Province).

%------at a country level--------------------------------%

getAttraction(?Attraction:Attractions, ?Country:Country):-
   partOfCountry(?Attraction:Attractions, ?Country:Country).

getAttraction(?Attraction:Attractions, ?Country:Country):-
    partOfCountry(?Region:Region, ?Country:Country),
    getAttraction(?Attraction:Attractions, ?Region:Region).
```

Distance Computation Facts and Rules

```
% Transitive closure facts describing all adjacent provinces of Bhutan

distanceTime(Haa, Paro; hike->6:Real; bike->5:Real; bus->4:Real; cab->3:Real).
distanceTime(Haa, Chuzom; hike->7.5:Real; bike->5:Real; bus->3.5:Real; cab->2.5:Real).
distanceTime(Paro, Chuzom; hike->4:Real; bike->3:Real; bus->2:Real; cab->1.5:Real).
distanceTime(Chuzom, Thimphu; hike->4.5:Real; bike->3.5:Real; bus->2.5:Real; cab->1.5:Real).
distanceTime(Chuzom, Chukha; hike->7:Real; bike->6.5:Real; bus->4.5:Real; cab->4:Real).
distanceTime(Chukha, Phuntsholing; hike->9:Real; bike->7.0:Real; bus->3.5:Real; cab->3:Real).
distanceTime(Phuentsholing, Samtse; hike->9:Real; bike->7.0:Real; bus->4:Real; cab->3:Real).
distanceTime(Thimphu, Lobesa; hike->9.5:Real; bike->5.0:Real; bus->3.5:Real; cab->3:Real).
distanceTime(Lobesa, WangduePhodrang; hike->1.0:Real; bike->0.5:Real; bus->0.7:Real; cab->0.5:Real).
distanceTime(Lobesa, Punakha; hike->1.0:Real; bike->0.5:Real; bus->0.7:Real; cab->0.5:Real).
distanceTime(Punakha, Gasa; hike->5.5:Real; bike->5.5:Real; bus->4.5:Real; cab->4:Real).
distanceTime(Punakha, WangduePhodrang; hike->1.5:Real; bike->1:Real; bus->0.7:Real; cab->0.5:Real).
distanceTime(WangduePhodrang, Damphu; hike->22:Real; bike->20:Real; bus->8:Real; cab->6.5:Real).
distanceTime(Damphu, Dagana; hike->14:Real; bike->11:Real; bus->8:Real; cab->6.5:Real).
distanceTime(Damphu, Sarpang; hike->7:Real; bike->5.5:Real; bus->4:Real; cab->3:Real).
distanceTime(Sarpang, Gelephu; hike->5.5:Real; bike->4:Real; bus->3:Real; cab->2.5:Real).
distanceTime(Gelephu, Zhemgang; hike->7:Real; bike->5.5:Real; bus->4.5:Real; cab->3.5:Real).
distanceTime(Zhemgang, Trongsa; hike->10:Real; bike->6.5:Real; bus->4:Real; cab->3.5:Real).
distanceTime(WangduePhodrang, Trongsa; hike->9:Real; bike->5.5:Real; bus->3.5:Real; cab->3:Real).
distanceTime(Trongsa, Bumthang; hike->6.5:Real; bike->5.5:Real; bus->4.5:Real; cab->3.5:Real).
distanceTime(Bumthang, Mongar; hike->20:Real; bike->15:Real; bus->7:Real; cab->6:Real).
distanceTime(Mongar, Lhuentse; hike->10:Real; bike->8.5:Real; bus->7.5:Real; cab->6.5:Real).
distanceTime(Mongar, Trashigang; hike->9:Real; bike->8:Real; bus->7:Real; cab->5.5:Real).
distanceTime(Trashigang, TrashiYangtse; hike->10:Real; bike->8.5:Real; bus->7.5:Real; cab->6.5:Real).
distanceTime(Trashigang, PemaGatshel; hike->8.5:Real; bike->7.5:Real; bus->6.5:Real; cab->5.5:Real).
distanceTime(Trashigang, SamdrupJongkhar; hike->10:Real; bike->8:Real; bus->7:Real; cab->5.5:Real).


% Symmetric rule holds the birectionality betweem the adjacent nodes

 distanceTimeSymmetric(?Province1,
                       ?Province2;
                       hike->?Hike;
                       bike->?Bike;
                       bus->?Bus;
                       cab->?Cab):-
   distanceTime(?Province1,
                ?Province2;
                hike->?Hike;
                bike->?Bike;
                bus->?Bus;
                cab->?Cab).

 distanceTimeSymmetric(?Province1,
                       ?Province2;
                       hike->?Hike;
                       bike->?Bike;
                       bus->?Bus;
                       cab->?Cab):-
   distanceTime(?Province2,
                ?Province1;
                hike->?Hike;
                bike->?Bike;
                bus->?Bus;
                cab->?Cab).

% Interface predicate to enforce the distanceTimeRoute main predicate
```

```
 dTR(startPoint->?Province1;
     endPoint->?Province2;
   route->?Route;
   totalTime->?TotalBusHours):-
   distanceTimeRoute(startPoint->?Province1;
                     endPoint->?Province2;
                     [],
                     0:Real;
                     route->?Rest;
                     totalTime->?TotalBusHours).

% Workhorse predicate to compute the routes and bushours
% for any two province on the road-map

 distanceTimeRoute(startPoint->?Province;
                   endPoint->?Province;
                   ?Visited,
                   ?AccumulateTime:Real;
                   route->[?Province];
                   totalTime->?AccumulateTime:Real).

 distanceTimeRoute(startPoint->?Province1;
                   endPoint->?ProvinceX;
                   ?Visited,
                   ?AccumulateTime:Real;
                   route->[?Province1|?Rest];
                   totalTime->?TotalBusHours:Real):-
  distanceTimeSymmetric(?Province1, ?Province2; bus->?Bus!?),
  notMember(?Province2, ?Visited),
  add(?NewBus, ?Bus, ?AccumulateTime:Real),
  distanceTimeRoute(startPoint->?Province2;
                    endPoint->?ProvinceX;
                    [?Province1,?Province2|?Visited],
                    ?NewBus;
                    route->?Rest;
                    totalTime->?TotalBusHours:Real).

%Query: dTR(startPoint->?Province1;
            endPoint->?Province2;
            route->?Rest;
            totalTime->?TotalBusHours:Real).

% This dTR predicate is executed in OO jDREW FindAll Solution archtecture to
generate all the routes and bushours for every pair of provinces on the road-map.

% Predicate to compute the shortest route from the list of alternative routes
% routeCount is generated along with dTR facts in the first level of computation

%Query: dTRShortest(startPoint->A;
                   endPoint->D;
                   route->?Routes;
                   shortestRoute->?Minimum).

% Interface predicate
 dTRShortest(startPoint->?Province1;
```

```
            endPoint->?Province2;
            route->?Routes;
            shortestRoute->?ShortestRoute):-
   routeCount(startPoint->?Province1;
              endPoint->?Province2;
              count->?Count:Integer),
   dTRList(startPoint->?Province1;
           endPoint->?Province2;
           route->?Routes;
           visited->[];
           currentMinRoute->[R, 10000:Real];
           min->?ShortestRoute;
           count->?Count:Integer).

% Workhorse predicate to compute the min from routes recursively

 dTRList(startPoint->?Province1;
           endPoint->?Province2;
           route->[];
           visited->?Visited;
           currentMinRoute->?NewMinRoute;
           min->?NewMinRoute;
           count->0:Integer).

 dTRList(startPoint->?Province1;
           endPoint->?Province2;
           route->[[?R1,?T1]|?Rest];
           visited->?Visited;
           currentMinRoute->[?RMin, ?TMin];
           min->?FinalMinRoute;
           count->?Count:Integer):-
   dTR(startPoint->?Province1;
       endPoint->?Province2;
       route->?R1;
       totalTime->?T1),
   notMemberList(?R1,?Visited),
   minRoute([[?R1, ?T1], [?RMin, ?TMin]], ?NewMinRoute),
   subtract(?newCount, ?Count:Integer,  1:Integer),
   dTRList(startPoint->?Province1;
            endPoint->?Province2;
            route->?Rest;
            visited->[?R1|?Visited];
            currentMinRoute->?NewMinRoute;
            min->?FinalMinRoute;
            count->?newCount:Integer).

% Customized version of min predicate to find the smallest route in a list:
 minRoute([[?R1,?X]] , [?R1,?X]).
 minRoute([[?R1,?A],[?R2,?B]|?T], ?M) :- lessThan(?B ,?A),
                                         minRoute([[?R2,?B]|?T], ?M).
 minRoute([[?R1,?A],[?R2,?B]|?T], ?M) :- lessThan(?A ,?B),
                                         minRoute([[?R1,?A]|?T], ?M).
 minRoute([[?R1,?A],[?R2,?B]|?T], ?M) :- equal(?A ,?B), minRoute([[?R1,?A]|?T], ?M).
```

```
%Simplified minRoute for list consisting of two elements
 minRoute([[?R1,?A],[?R2,?B]], [?R1, ?A]) :- lessThan(?A ,?B).
 minRoute([[?R1,?A],[?R2,?B]], [?R2, ?B]) :- lessThan(?B ,?A).

% In order to have precomputed shortest route for every pair
% of provinces along with precomputed all routes. The dTRShortest
% predicate is partially (route accumulation) implemented
% in java in the OO jDREW engine.
% The package consisting of three main parts is appendixed here

import java.io.BufferedReader;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.util.ArrayList;
import java.util.StringTokenizer;

import jdrew.oo.util.ParseException;
import jdrew.oo.util.SubException;
import nu.xom.ParsingException;
import nu.xom.ValidityException;
import antlr.RecognitionException;
import antlr.TokenStreamException;


public class RouteAccumulation {

 public static void main(String args[]){

 ArrayList<DTRObject> list = new ArrayList<DTRObject>();
ArrayList<VisitedNode> visited = new ArrayList<VisitedNode>();
try{
String out = "";
File f = new File("bhutandTRKB");
FileReader inFile = new FileReader(f);
    BufferedReader in = new BufferedReader(inFile);
    String read ="";
    String contents="";

    while((read = in.readLine()) != null)
     {
       contents = contents + read + '\n';
     }
      in.close();
     StringTokenizer st = new StringTokenizer(contents, "\n");
     while(st.hasMoreTokens()){
       StringTokenizer part = new StringTokenizer(st.nextToken(), ";");
       String start = part.nextToken();
       start = start.substring(start.indexOf("->") + 2);
       String end = part.nextToken();
       end = end.substring(end.indexOf("->") + 2);
       String route = part.nextToken();
       route = route.substring(route.indexOf("->") + 2);

       String time = part.nextToken();
```

```
        time = time.substring(time.indexOf("->") + 2);
        StringTokenizer timeTokenizer = new StringTokenizer(time, ":");
        String time2 = timeTokenizer.nextToken();
        DTRObject dtr = new DTRObject(start, end, route, time2);

        list.add(dtr);
        }
        for(int i = 0; i < list.size(); i++){
         DTRObject dtr = list.get(i);
         String start = dtr.getStart();
         String end = dtr.getEnd();
         VisitedNode node = new VisitedNode(start,end);
         boolean found = false;
         for(int y = 0; y  < visited.size(); y++){

          VisitedNode previous = visited.get(y);

     if((previous.getStart().equalsIgnoreCase(start) &&
         previous.getEnd().equalsIgnoreCase(end)) ||
        (previous.getStart().equalsIgnoreCase(end) &&
         previous.getEnd().equalsIgnoreCase(start))){
         found = true;
        }
      }

    if(!found){
        visited.add(node);
        ArrayList<DTRObject> same = new ArrayList<DTRObject>();

        for(int j = 0; j < list.size(); j++){
        DTRObject dtr2 = list.get(j);

        if((start.equalsIgnoreCase(dtr2.getStart()) &&
            end.equalsIgnoreCase(dtr2.getEnd())))
          {// ||same.add(dtr2);
          }
         }

     out = "dTRList(startPoint->" + same.get(0).getStart() +
           ":Province; endPoint->" + same.get(0)
           .getEnd() + ":Province; routeList->[";

          for(int q = 0; q < same.size(); q++){
             out = out + "[" + same.get(q).getPath()+ ",
             "+ same.get(q).getTime() + ":Real" + "]";
         if(!(same.size()-1 == q)){
           out = out + ",";
         }
          }
          out = out + "]; count->" + same.size() + ":Integer).";
         System.out.println(out);
         }
        }

}catch(Exception e){
```

```
System.out.println(e.toString());
}
}
}


public class DTRObject {

private String start;
private String end;
private String time;
private String path;

DTRObject(String start, String end, String path, String time){
this.start = start;
this.end = end;
this.time = time;
this.path = path;
}
String getStart(){
return start;
}
String getEnd(){
return end;
}
String getTime(){
return time;
}
String getPath(){
return path;
}
}
public class BindingPair {

private String variable;
private String value;

BindingPair(String variable, String value){
this.variable = variable;
this.value = value;
}
public String getVariable(){
return variable;
}
public String getValue(){
return value;
}
}

public class VisitedNode {

String start;
String end;
VisitedNode(String start, String end){
this.start = start;
this.end = end;
```

```
}
String getStart(){
return start;
}
String getEnd(){
return end;
}
}

% This java code accummulate each of routes connecting a specific
% pair of nodes into a list from the dTR facts

% A sample dTRList fact is shown here
 dTRList(startPoint->Punakha:Province;
         endPoint->WangduePhodrang:Province;
         routeList->[[[Punakha, WangduePhodrang],0.7:Real],
                     [[Punakha, Lobesa, WangduePhodrang],1.4:Real]];
         count->2:Integer).

% Route Search between any two provinces
 getRouteDetails(startPoint->?ProvinceA;
                 endPoint->?ProvinceB;
                 ?Routes;
                 ?ShortestRoute):-
    dTRShortest(startPoint->?ProvinceA;
                endPoint->?ProvinceB;
                routeList->?Routes;
                shortest->?ShortestRoute).

% Route and Distance Computation Rule subsystem
% Symmetric is maintained by having  two modes of same predicate

shortestRoute(?P1, ?P2, ?ShortestRoute):-
 dTRShortest(startPoint->?P1;
             endPoint->?P2;
             shortest->?ShortestRoute!?).

shortestRoute(?P2, ?P1, ?ReverseRoute):-
  dTRShortest(startPoint->?P1; endPoint->?P2;  shortest->?ShortestRoute!?),
  reverse(?ShortestRoute, ?ReverseRoute).

% subpredicate to append an element to a list
 append([], ?List, ?List).
 append([?H|?Tail1], ?List2, [?H|?Tail3]) :-
   append(?Tail1, ?List2, ?Tail3) .

% subpredicate to reverse a list
 reverse([], []).
 reverse([?H|?T], ?R) :- reverse(?T, ?R1), append(?R1, [?H], ?R).

%dTRShortest are newly generated facts from the TopDownFindAll Solution,
% after applying the minRoute supredicate on each dTRList facts generated
% by the java code.

% 277 dTRShortest facts are generated, which connects 24 provinces
% A sample fact is shown here
```

```
dTRShortest(startPoint->Punakha:Province;
            endPoint->WangduePhodrang:Province;
            routeList->[[[Punakha, WangduePhodrang], 0.7:Real],
                        [[Punakha, Lobesa, WangduePhodrang], 1.4:Real]];
            shortest->[[Punakha, WangduePhodrang], 0.7:Real]).

% Route Planner has two options:
% i)UserPrefBased
% ii)SystemBased

% Route planner for a list of provinces selected by the user
 routePlanner(typeOfPlan->UserPrefBased;
              startPoint->?Province1;
              endPoint->?Province2;
              userPref->?PrefList;
              routeDetails->[Route->?Route;
                             TotalTime->?TotalBusHours]):-
  userPrefRoutePlanner(startPoint->?Province1;
                       endPoint->?Province2;
                       userPref->?PrefList;
                       route->?Routes;
                       accumulateTime->0:Real;
                       totalTime->?TotalBusHours).
% System Route planner
 routePlanner(typeOfPlan->SystemRecommend;
              startPoint->?Province1;
              endPoint->?ProvinceN;
              routeDetails->[RecommendedRoute->?Route;
                             TotalTime->?TotalBusHours;
                             ReturnRoute->?ReturnRoute];
             numofProvinces->?Num:Integer):-
  systemRoutePlanner(startPoint->?Province1;
                     nextPoint->?Province2;
                     toRoute->?Route;
                     starttime->0:Real;
                     totalTime->?TotalBusHours:Real;
                     numofProvinces->?Num:Integer ),
  shortestRoute(?Province2, ?ProvinceN, ?ReturnRoute).

% Workhorse predicate for option one
 userPrefRoutePlanner(startPoint->?P1;
                      endPoint->?P1;
                      userPref->[];
                      route->[];
                      accumulateTime->0:Real;
                      totalTime->0:Real).

 userPrefRoutePlanner(startPoint->?P1;
                      endPoint->?P2;
                      userPref->[];
                      route->[?Route, ?BusHours];
                      accumulateTime->?AccumulateTime:Real;
                      totalTime->?NewBus:Real):-
   shortestRoute(?P1, ?P2, [?Route, ?BusHours]),
```

```
   add(?NewBus, ?AccumulateTime, ?BusHours).

  userPrefRoutePlanner(startPoint->?P1;
                       endPoint->?P2;
                       userPref->[?Pref|?PrefRest];
                       route->[[?Route, ?BusHours]|?RouteRest];
                       accumulateTime->?AccumulateTime:Real;
                       totalTime->?TotalBusHours:Real):-
                       shortestRoute(?P1, ?Pref, [?Route, ?BusHours]),
                       add(?NewBus, ?AccumulateTime:Real, ?BusHours),
                       userPrefRoutePlanner(startPoint->?Pref;
                                            endPoint->?P2;
                                            userPref->?PrefRest;
                                            route->?RouteRest;
                                            accumulateTime->?NewBus;
                                            totalTime->?TotalBusHours:Real).

% System Route planner
 systemRoutePlanner(startPoint->?Province;
                    nextPoint->?Province;
                    toRoute->[];
                    starttime->?Starttime:Real;
                    totalTime->?Starttime:Real;
                    numofProvinces->1:Integer).

 systemRoutePlanner(startPoint->?Province1;
                     nextPoint->?Province2;
                     toRoute->[[?Route,?BusHours]|?Rest];
                     starttime->?Starttime:Real;
                     totalTime->?TotalBusHours:Real;
                     numofProvinces->?NumOfProvinces:Integer):-

   greaterThan(?NumOfProvinces:Integer, 1:Integer),
   getProvinceToRecommendation(?Province1,?R1),
   shortestRoute(?Province1, ?R1, [?Route, ?BusHours]),
   add(?NewBus:Real,?BusHours:Real,?Starttime:Real),
   subtract(?RouteLeft:Integer,  ?NumOfProvinces, 1:Integer),
   systemRoutePlanner(startPoint->?R1;
                       nextPoint->?Province2;
                       toRoute->?Rest;
                       starttime->?NewBus:Real;
                       totalTime->?TotalBusHours:Real;
                       numofProvinces->?RouteLeft:Integer ).
```

Appendix D: Auxiliary Rule Sets

The Global constants, Date Validation, Duplicate checking and other search subpredicates are given below:

```
% Global constants

% Maximum travel hours in a day
  maxTravelHoursInADay(8:Real).

% Maximum hours of sightseeing in a day
  maxActivitySightSeeingHoursInADay(12:Real).

% Maximum hours at each attraction site
  maxHoursAtAnAttractionSite(4:Real).

% Maximum days between two events inorder to check for attraction
% recommendation on the route
  maxTimeGapBetweenEvents(5:Real).

% Provinces that have attractions
  hasAttractions(Chukha).
  hasAttractions(Paro).
  hasAttractions(Thimphu).
  hasAttractions(Punakha).
  hasAttractions(WangduePhodrang).
  hasAttractions(Trongsa).
  hasAttractions(Bumthang).
  hasAttractions(Mongar).
  hasAttractions(Trashigang).
  hasAttractions(TrashiYangtse).

%Date Validation Rules
%later(?Date2, ?Date1) if ?Date2 is after ?Date1:

later(date[?Y:Real, ?M:Real, ?Day2:Real],
      date[?Y:Real, ?M:Real, ?Day1:Real]) :-
  greaterThan(?Day2:Real, ?Day1:Real).

later(date[?Y:Real, ?Month2:Real, ?],
      date[?Y:Real, ?Month1:Real, ?]) :-
  greaterThan(?Month2:Real, ?Month1:Real).

later(date[?Year2:Real, ?, ?], date[?Year1:Real, ?, ?]) :-
  greaterThan(?Year2:Real, ?Year1:Real).

%?Difference = ?Date2 - ?Date2 (In days)
```

```
diff(?Difference, date[?Y:Real, ?M:Real, ?Day2:Real],
                  date[?Y:Real, ?M:Real, ?Day1:Real]) :-
subtract(?Difference, ?Day2:Real, ?Day1:Real).

diff(?Difference, date[?Y:Real, ?Month2:Real, ?Day2:Real],
                  date[?Y:Real, ?Month1:Real, ?Day1:Real]) :-
  subtract(?DiffMonth, ?Month2, ?Month1),
  multiply(?DiffInDays, ?DiffMonth, 30:Real),
  subtract(?TempDaysLeft, ?DiffInDays, ?Day1),
  add(?Difference, ?TempDaysLeft, ?Day2).

% Auxiliary rules to avoid duplicates in a list

% notMember predicate is used to avoid duplicates in a list
 notMember(?X, []).
 notMember(?X, [?H|?T]) :- notEqual(?X, ?H), notMember(?X, ?T).

% notMemberList predicate is used to avoid duplicate list in a nested list
 notMemberList( ?X, []).
 notMemberList( ?X, [?H]) :- naf(equalList(?X, ?H)).
 notMemberList( ?X,[?H|?T]) :-  naf(equalList(?X, ?H)), notMemberList(?X,?T).

% equalList predicate checks if two list are equal
  equalList(?L, ?L).

% Selecting leaf values from a the tourist entity profiles
 getCapitalDistance(?Name,?CapitalDistance):-
   attraction(?Name^et.capitalDistance->?CapitalDistance:Real!?).

 getTheme(?Name,?Theme):-
   attraction(?Name^et.theme->?Theme!?).

 getProvince(?Name,?Province):-
   attraction(?Name^et.province->?Province!?).

 % Symmetric relation between related Provinces

 getProvinceToRecommendation(?Province,?RelatedProvince):-
   province(?Province^hs.relatedTo->?RelatedProvince!?).

 getProvinceToRecommendation(?Province,?RelatedProvince):-
   province(?RelatedProvince^hs.relatedTo->?Province!?).

 %Symmetry between related attractions

 getAttractionToRecommendation(?Attraction,?RelatedAttraction):-
   attraction(?RelatedAttraction^hs.relatedTo->?Attraction!?).

 getAttractionToRecommendation(?Attraction,?RelatedAttraction):-
   attraction(?Attraction^hs.relatedTo->?RelatedAttraction!?).

 %Activity subsumes both attraction and event profiles
 activity(?Name^hs.startDate->?StartDate;
               hs.endDate->?EndDate;
```

```
              hs.description->?Description;
              hs.url->?Url;
            et.theme->?Theme;
            hs.location->?Location;
            et.province->?Province;
            hs.relatedTo->?RelatedTo):-
   event(?Name^hs.startDate->?StartDate;
             hs.endDate->?EndDate;
           hs.description->?Description;
             hs.url->?Url;
             et.theme->?Theme;
             hs.location->?Location;
           et.province->?Province;
             hs.relatedTo->?RelatedTo!?).

 activity(?Name^hs.description->?Description;
              hs.url->?Url;
              et.theme->?Theme;
                et.open->?OpenHours;
                et.subblock->?Subblock;
                et.province->?Province;
                hs.relatedTo->?RelatedTo):-
   attraction(?Name^hs.description->?Description;
                    hs.url->?Url;
                    et.open->?OpenHours;
                  et.theme->?Theme;
                    et.subblock->?Subblock;
                  et.province->?Province;
                  hs.relatedTo->?RelatedTo!?).
```

Search Rule Subsystem

```
% Subpredicate to retrieve province details within a specific region

  getProvinceDetails(region->?Region;
                      name->?Name;
                    [WebLink->?Url;
                     Description->?Description;
                     Capital->?Capital;
                     Geography->[Area->?Area;
                                 Elevation->?Elevation];
                     TouristInfo->[NumAttractions->?NumAttractions;
                                   NumEvents->?NumEvents;
                                   NumAccommodations->?NumAccommodations];
                   Contact->?Contact]):-

  partOfRegion(?Name, ?Region),
  province(?Name^hs.url->?Url;
                et.capital->?Capital;
                et.area->?Area;
                et.elevation-> ?Elevation;
                et.numBlocks->?NumBlocks;
                et.numAttractions->?NumAttractions;
```

```
                et.numEvents->?NumEvents;
                et.numAccommodations->?NumAccommodations;
                hs.languagesSpoken->?Language;
                hs.contact->?Contact;
                hs.description->?Description;
                hs.relatedTo->?RelatedTo!?).

 % Subpredicate for parametric search of activities

 getActivityDetails(actName->?Name;
                    theme->?Theme;
                    address->[?Subblock, ?Block,?Province, ?Region, ?Country];
                   [Name->?Name;
                    WebLink->?Url;
                    Address->[?Subblock, ?Block,?Province, ?Region, ?Country];
                    Description->?Description;
                    RelatedTo->?RelatedTo]):-
    attraction(?Nameˆhs.url->?Url;
                     hs.description->?Description;
                     et.open->?OpenHours;
                     et.theme->?Theme;
                     et.subblock->?Subblock;
                     et.province->?Province;
                     hs.relatedTo->?RelatedTo!?),
    getFullAddress( ?Name, [?Subblock, ?Block,?Province, ?Region, ?Country]).

getActivityDetails(actName->?Name;
                   theme->?Theme;
                   address->[?Subblock, ?Block,?Province, ?Region, ?Country];
                  [ActName->?Name;
                   WebLink->?Url;
                   Address->[?Subblock, ?Block,?Province, ?Region, ?Country];
                   Theme->?Theme;
                   EventDates->[StartDate->?StartDate;
                                EndDate->?EndDate];
                   Description->?Description;
                   RelatedTo->?RelatedTo]):-

    event(?Nameˆhs.startDate->?StartDate;
                hs.endDate->?EndDate;
                hs.description->?Description;
                hs.url->?Url;
                et.theme->?Theme;
                hs.location->?Venue;
                et.province->?Province;
                hs.relatedTo->?RelatedTo!?),
    getFullAddress( ?Venue, [?Subblock,?Block, ?Province, ?Region, ?Country]).

 % Subpredicate for parametric search of accommodations

 getAccommodationDetails(accName->?Name;
                         address->[?Subblock,
                                   ?Block,
                                   ?Province,
                                   ?Region,
```

```
                                ?Country];
                         setMaxPrice->[?YesOrNo, ?UserMaxPrice:Real];
                        [Name->?Name;
                         WebLink->?Url;
                         Address->[?Subblock,
                                   ?Block,
                                   ?Province,
                                   ?Region,
                                   ?Country];
                         Standard->[StarRating->?Rating;
                                    MinPrice->?AccommMinPrice];
                         ContactDetails->[Telecoms->[LandLine->?LandLine;
                                                     CellLine->?CellLine];
                                                     Email->?ContactInfo];
                         RelatedTo->?Recommendation]):-
   accommodation(?Name^
                 hs.url->?Url;
                 et.rating->?Rating;
                 et.minPrice->?AccommMinPrice;
                 et.subblock->?Subblock;
                 et.province->?Province;
                 hs.telecoms->Telecoms[
                    et.landline->?LandLine;
                    et.cell->?CellLine];
                 hs.contact->?ContactInfo;
                 hs.relatedTo->?Recommendation!?),
                 validateAccommPrice(?YesOrNo,
                                     ?AccommMinPrice:Real,
                                     ?UserMaxPrice:Real),
                 getFullAddress(?AccName,  [?Subblock,
                                            ?Block,
                                            ?Province,
                                            ?Region,
                                            ?Country]).

% Auxiliary predicate for price validation of an accommodation(Optional)

   validateAccommPrice(Yes, ?AccommMinPrice:Real, ?UserMaxPrice:Real):-
      lessThanOrEqual(?AccommMinPrice:Real, ?UserMaxPrice:Real).

    validateAccommPrice(No, ?AccommMinPrice:Real, ?UserMaxPrice:Real).


% Subpredicate to accumulate all the attractions located in the specified province

 getAllAttractions(?Province,  ?Attractions, ?NumAttractions:Integer):-
   province(?Province^et.numAttractions->?NumAttractions!?),
   getAttractionList(?Province,  ?Attractions, [], ?NumAttractions:Integer).

 getAttractionList(?Province, [], ?Visited, 0:Integer).

 getAttractionList(?Province,  [[AttractionName->?Name;
                                 Weblink->?Url;
                                 Theme->?Theme;
                                 Address->?Address;
```

```
                                Description->?Description]|?Rest],
                ?Visited,
                ?NumAttractions:Integer):-
  greaterThan(?NumAttractions:Integer, 0:Integer),
  attraction(?Name^
          hs.description->?Description;
          hs.url->?Url;
          et.theme->?Theme;
          hs.relatedTo->?RelatedTo;
          et.province->?Province!?),
   notMember(?Name, ?Visited),
   getFullAddress(?Name, ?Address),
   subtract(?Numtleft, ?NumAttractions:Integer,  1:Integer),
   getAttractionList(?Province,  ?Rest,  [?Name|?Visited], ?Numtleft).

% Subpredicate to accumulate all the events located in the specified province

%Interface Predicate
 getAllEvents(?Province, ?Events, ?NumEvents:Integer):-
   province(?Province^et.numEvents->?NumEvents!?),
   getEventList(?Province,?Events, [], ?NumEvents:Integer).

% Workhorse subpredicate
 getEventList(?Province, [], ?Visited, 0:Integer).

 getEventList(?Province,
             [[EventName->?Name;
               EventDates->[StartDate->?StartDate;
                            EndDate->?EndDate];
               Address->?Address;
               Description->?Description]|?Rest],
             ?Visited,
             ?NumEvents:Integer):-
   greaterThan(?NumEvents:Integer, 0:Integer),
   event(?Name^
          hs.startDate->?StartDate;
          hs.endDate->?EndDate;
          hs.description->?Description;
          hs.url->?Url;
          et.theme->?Theme;
          hs.location->?Venue;
          hs.relatedTo->?RelatedTo;
          et.province->?Province!?),
   notMember(?Name, ?Visited),
   getFullAddress(?Venue, ?Address),
   subtract(?Numtleft, ?NumEvents:Integer,  1:Integer),
   getEventList(?Province,  ?Rest,  [?Name|?Visited], ?Numtleft).

% Subpredicate to accumulate all the accommodations located in a specified province

 getAllAccomm(?Province,  ?Attractions, ?NumAccommodations:Integer):-
   province(?Province^et.numAccommodations->?NumAccommodations!?),
   getAccommList(?Province, ?Attractions, [], ?NumAccommodations:Integer).

 getAccommList(?Province, [], ?Visited, 0:Integer).
```

```
 getAccommList(?Province, [[?Name;
                            Weblink->?Url;
                            Rating->?Rating;
                            MinPrice->?MinPrice]|?Rest],
               ?Visited,
               ?NumAccommodations:Integer):-
    greaterThan(?NumAccommodations, 0:Integer),
    accommodation(?Name^hs.url->?Url;
                    et.rating->?Rating;
                    et.minPrice->?MinPrice;
                        et.province->?Province!?),
    notMember(?Name, ?Visited),
    subtract(?Numtleft, ?NumAccommodations:Integer,  1:Integer),
    getAccommList(?Province,  ?Rest,  [?Name|?Visited], ?Numtleft).

% Auxiliary "getAttraction subpredicate to get an attraction at each province
 getAttraction(?Province,  [AttractionName->?Name;
                            Theme->?Theme;
                            Subblock->?Subblock;
                            RelatedTo->?RelatedTo]):-
   attraction(?Name^
              et.subblock->?Subblock;
              et.theme->?Theme;
              hs.relatedTo->?RelatedTo;
              et.province->?Province!?).

% To check of a stop on the route is a province or not
 checkIfAttractions(?P, ?Attractions, ?NumAttractions:Integer):-
   hasAttractions(?P),
   getAttraction(?P,  ?Attractions).

 checkIfAttractions(?P, [], ?Num:Integer):-
   naf(hasAttractions(?P)).

% Subpredicate to check if the route between two events require
% attraction recommendations provided the users selects the option.
 checkIfRecommend(?TimeGap, ?Route, [], No).

 checkIfRecommend(?TimeGap, ?Route, ?Recommendations, Yes):-
   maxTimeGapBetweenEvents(?MaxTimeGapBetweenEvents),
   greaterThanOrEqual(?TimeGap, ?MaxTimeGapBetweenEvents),
   routeLocCentric( route->?Route; recommendations->?Recommendations).

% Subpredicate that accumulates some attractions for a specific route
 routeLocCentric( route->[]; recommendations->[]).

 routeLocCentric(route->[?R1|?Rest];
                 recommendations->[[?R1; Attraction->?Attractions]
                                   |?RestRecommendations]):-
   checkIfAttractions(?R1, ?Attractions, 1:Integer),
   routeLocCentric(route->?Rest; recommendations->?RestRecommendations).
```

Appendix E: RDFS Type Definitions

This appendix gives the RDFS type definition of the tourism subdomains, a light-weight ontology collected from the Harmonise ontology. It consists of the classification of events, attractions, accommodations, and geographic divisions of a country.

Events RDFS Type Hierarchy

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns="http://www.owl-ontologies.com/Ontology1164049376.owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xml:base="http://www.owl-ontologies.com/Ontology1164049376.owl">

<rdfs:Class rdf:ID="Events"/>

 <rdfs:Class rdf:about="Adventure">
    <rdfs:subClassOf rdf:resource="Events"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="Bird_watching">
    <rdfs:subClassOf rdf:resource="Adventure"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="Cattle_driving">
    <rdfs:subClassOf rdf:resource="Adventure"/>
  </rdfs:Class>

   <rdfs:Class rdf:about="Rock_climbing">
    <rdfs:subClassOf rdf:resource="Adventure"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="Caving">
    <rdfs:subClassOf rdf:resource="Adventure"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="Trekking">
    <rdfs:subClassOf rdf:resource="Adventure"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="Agriculture">
```

```
  <rdfs:subClassOf rdf:resource="Adventure"/>
</rdfs:Class>

<rdfs:Class rdf:about="Arts">
  <rdfs:subClassOf rdf:resource="Events"/>
</rdfs:Class>

<rdfs:Class rdf:about="Architecture">
  <rdfs:subClassOf rdf:resource="Arts"/>
</rdfs:Class>

<rdfs:Class rdf:about="Decorative">
  <rdfs:subClassOf rdf:resource="Arts"/>
</rdfs:Class>

<rdfs:Class rdf:about="Design">
  <rdfs:subClassOf rdf:resource="Arts"/>
</rdfs:Class>

<rdfs:Class rdf:about="Exhibition">
  <rdfs:subClassOf rdf:resource="Arts"/>
</rdfs:Class>

<rdfs:Class rdf:about="Fashion">
  <rdfs:subClassOf rdf:resource="Arts"/>
</rdfs:Class>

<rdfs:Class rdf:about="Painting">
  <rdfs:subClassOf rdf:resource="Arts"/>
</rdfs:Class>

<rdfs:Class rdf:about="Photography">
  <rdfs:subClassOf rdf:resource="Arts"/>
</rdfs:Class>

<rdfs:Class rdf:about="Sculpture">
  <rdfs:subClassOf rdf:resource="Arts"/>
</rdfs:Class>

<rdfs:Class rdf:about="Festival">
  <rdfs:subClassOf rdf:resource="Events"/>
</rdfs:Class>

<rdfs:Class rdf:about="Annual_festival">
  <rdfs:subClassOf rdf:resource="Festival"/>
</rdfs:Class>

<rdfs:Class rdf:about="Dance_festival">
  <rdfs:subClassOf rdf:resource="Festival"/>
</rdfs:Class>

<rdfs:Class rdf:about="Film_festival">
  <rdfs:subClassOf rdf:resource="Festival"/>
</rdfs:Class>
```

```
<rdfs:Class rdf:about="Gastronomic_festival">
  <rdfs:subClassOf rdf:resource="Festival"/>
</rdfs:Class>

<rdfs:Class rdf:about="International_festival">
  <rdfs:subClassOf rdf:resource="Festival"/>
</rdfs:Class>

<rdfs:Class rdf:about="Music_festival">
  <rdfs:subClassOf rdf:resource="Festival"/>
</rdfs:Class>

<rdfs:Class rdf:about="National_festival">
  <rdfs:subClassOf rdf:resource="Festival"/>
</rdfs:Class>

<rdfs:Class rdf:about="Seasonal_festival">
  <rdfs:subClassOf rdf:resource="Festival"/>
</rdfs:Class>

<rdfs:Class rdf:about="Traditional_festival">
  <rdfs:subClassOf rdf:resource="Festival"/>
</rdfs:Class>

<rdfs:Class rdf:about="Market">
  <rdfs:subClassOf rdf:resource="Events"/>
</rdfs:Class>

 <rdfs:Class rdf:about="Market_daily">
  <rdfs:subClassOf rdf:resource="Market"/>
</rdfs:Class>

 <rdfs:Class rdf:about="Market_fair">
  <rdfs:subClassOf rdf:resource="Events"/>
</rdfs:Class>

 <rdfs:Class rdf:about="Market_flea">
  <rdfs:subClassOf rdf:resource="Events"/>
</rdfs:Class>

 <rdfs:Class rdf:about="Market_handicraft">
  <rdfs:subClassOf rdf:resource="Events"/>
</rdfs:Class>

 <rdfs:Class rdf:about="Market_street">
  <rdfs:subClassOf rdf:resource="Events"/>
</rdfs:Class>

 <rdfs:Class rdf:about="Market_weekly">
  <rdfs:subClassOf rdf:resource="Events"/>
</rdfs:Class>

<rdfs:Class rdf:about="Nature">
  <rdfs:subClassOf rdf:resource="Events"/>
</rdfs:Class>
```

```
<rdfs:Class rdf:about="Nature_animals">
  <rdfs:subClassOf rdf:resource="Nature"/>
</rdfs:Class>

 <rdfs:Class rdf:about="Nature_parks">
  <rdfs:subClassOf rdf:resource="Nature"/>
</rdfs:Class>

<rdfs:Class rdf:about="Nature_plants">
  <rdfs:subClassOf rdf:resource="Nature"/>
</rdfs:Class>

<rdfs:Class rdf:about="Nature_zoos">
  <rdfs:subClassOf rdf:resource="Nature"/>
</rdfs:Class>

<rdfs:Class rdf:about="Night_Life">
  <rdfs:subClassOf rdf:resource="Events"/>
</rdfs:Class>

<rdfs:Class rdf:about="Nightlife_balls">
  <rdfs:subClassOf rdf:resource="Night_Life"/>
</rdfs:Class>

<rdfs:Class rdf:about="Nightlife_food">
  <rdfs:subClassOf rdf:resource="Night_Life"/>
</rdfs:Class>

 <rdfs:Class rdf:about="Nightlife_party">
  <rdfs:subClassOf rdf:resource="Night_Life"/>
</rdfs:Class>

<rdfs:Class rdf:about="Religion">
  <rdfs:subClassOf rdf:resource="Events"/>
</rdfs:Class>

<rdfs:Class rdf:about="Religion_parade">
  <rdfs:subClassOf rdf:resource="Religion"/>
</rdfs:Class>

<rdfs:Class rdf:about="Religion_pilgrimage">
  <rdfs:subClassOf rdf:resource="Religion"/>
</rdfs:Class>

<rdfs:Class rdf:about="Religion_service">
  <rdfs:subClassOf rdf:resource="Religion"/>
</rdfs:Class>

<rdfs:Class rdf:about="Sport">
  <rdfs:subClassOf rdf:resource="Events"/>
</rdfs:Class>

 <rdfs:Class rdf:about="Sport_cycling">
  <rdfs:subClassOf rdf:resource="Sport"/>
```

```
  </rdfs:Class>

   <rdfs:Class rdf:about="Sport_tennis">
    <rdfs:subClassOf rdf:resource="Sport"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="Sport_archery">
    <rdfs:subClassOf rdf:resource="Sport"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="Sport_football">
    <rdfs:subClassOf rdf:resource="Sport"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="Sport_basketball">
    <rdfs:subClassOf rdf:resource="Sport"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="Sport_golf">
    <rdfs:subClassOf rdf:resource="Sport"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="Sport_hiking">
    <rdfs:subClassOf rdf:resource="Sport"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="Sport_biking">
    <rdfs:subClassOf rdf:resource="Sport"/>
  </rdfs:Class>

   <rdfs:Class rdf:about="Trade">
    <rdfs:subClassOf rdf:resource="Events"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="Trade_fair_agriculture">
    <rdfs:subClassOf rdf:resource="Trade"/>
  </rdfs:Class>

 <rdfs:Class rdf:about="Trade_fair_expo">
    <rdfs:subClassOf rdf:resource="Trade"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="Trade_fair_thematic">
    <rdfs:subClassOf rdf:resource="Trade"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="Trade_fair_tourism">
    <rdfs:subClassOf rdf:resource="Trade"/>
  </rdfs:Class>

Attractions RDFS Type Hierarchy

<rdfs:Class rdf:ID="Attractions"/>

 <rdfs:Class rdf:about="Archaeological_site">
    <rdfs:subClassOf rdf:resource="Attractions"/>
```

```
</rdfs:Class>

<rdfs:Class rdf:about="Culture">
  <rdfs:subClassOf rdf:resource="Attractions"/>
</rdfs:Class>

<rdfs:Class rdf:about="Entertainment">
    <rdfs:subClassOf rdf:resource="Attractions"/>
</rdfs:Class>

<rdfs:Class rdf:about="Buildings">
    <rdfs:subClassOf rdf:resource="Attractions"/>
</rdfs:Class>

<rdfs:Class rdf:ID="Cave">
  <rdfs:subClassOf rdf:resource="Archaeological_site"/>
</rdfs:Class>

 <rdfs:Class rdf:ID="Prehistoric_shelter">
  <rdfs:subClassOf rdf:resource="Archaeological_site"/>
</rdfs:Class>

<rdfs:Class rdf:about="Ruins">
    <rdfs:subClassOf rdf:resource="Archaeological_site"/>
</rdfs:Class>

<rdfs:Class rdf:ID="Crafts_show">
  <rdfs:subClassOf rdf:resource="Culture"/>
</rdfs:Class>

<rdfs:Class rdf:ID="Popular_architecture">
  <rdfs:subClassOf rdf:resource="Culture"/>
</rdfs:Class>

<rdfs:Class rdf:ID="Farm_house">
  <rdfs:subClassOf rdf:resource="Culture"/>
</rdfs:Class>

<rdfs:Class rdf:ID="Place_of_pilgrimage">
  <rdfs:subClassOf rdf:resource="Culture"/>
</rdfs:Class>

<rdfs:Class rdf:ID="Series_of_sculptures">
  <rdfs:subClassOf rdf:resource="Culture"/>
</rdfs:Class>

<rdfs:Class rdf:ID="Textiles">
  <rdfs:subClassOf rdf:resource="Culture"/>
</rdfs:Class>

<rdfs:Class rdf:ID="CityOrTown_excavations">
  <rdfs:subClassOf rdf:resource="Entertainment"/>
</rdfs:Class>

<rdfs:Class rdf:ID="Boat_trips">
```

```
  <rdfs:subClassOf rdf:resource="Entertainment"/>
 </rdfs:Class>

 <rdfs:Class rdf:ID="Garden">
  <rdfs:subClassOf rdf:resource="Entertainment"/>
 </rdfs:Class>

 <rdfs:Class rdf:ID="National_park">
  <rdfs:subClassOf rdf:resource="Entertainment"/>
 </rdfs:Class>

 <rdfs:Class rdf:ID="Stadium">
  <rdfs:subClassOf rdf:resource="Entertainment"/>
 </rdfs:Class>

 <rdfs:Class rdf:ID="Outings_and_daytrips">
  <rdfs:subClassOf rdf:resource="Entertainment"/>
 </rdfs:Class>

 <rdfs:Class rdf:ID="Scenic_drive">
  <rdfs:subClassOf rdf:resource="Entertainment"/>
 </rdfs:Class>

 <rdfs:Class rdf:ID="Theatre">
  <rdfs:subClassOf rdf:resource="Entertainment"/>
 </rdfs:Class>

 <rdfs:Class rdf:ID="Water_fall">
  <rdfs:subClassOf rdf:resource="Entertainment"/>
 </rdfs:Class>

 <rdfs:Class rdf:ID="Zoo">
  <rdfs:subClassOf rdf:resource="Entertainment"/>
 </rdfs:Class>

 <rdfs:Class rdf:about="Historical_buildings">
    <rdfs:subClassOf rdf:resource="Buildings"/>
 </rdfs:Class>

 <rdfs:Class rdf:about="Modern_buildings">
    <rdfs:subClassOf rdf:resource="Buildings"/>
 </rdfs:Class>

 <rdfs:Class rdf:about="Castle">
    <rdfs:subClassOf rdf:resource="Historical_buildings"/>
 </rdfs:Class>

<rdfs:Class rdf:about="Fortress">
    <rdfs:subClassOf rdf:resource="Historical_buildings"/>
 </rdfs:Class>

<rdfs:Class rdf:about="Museums">
    <rdfs:subClassOf rdf:resource="Historical_buildings"/>
 </rdfs:Class>
```

```
 <rdfs:Class rdf:about="Local_museum">
    <rdfs:subClassOf rdf:resource="Museums"/>
 </rdfs:Class>

 <rdfs:Class rdf:about="National_museum">
    <rdfs:subClassOf rdf:resource="Museums"/>
 </rdfs:Class>

 <rdfs:Class rdf:about="Regional_museum">
    <rdfs:subClassOf rdf:resource="Museums"/>
 </rdfs:Class>

 <rdfs:Class rdf:about="Private_museum">
    <rdfs:subClassOf rdf:resource="Museums"/>
 </rdfs:Class>

 <rdfs:Class rdf:about="Monastery">
    <rdfs:subClassOf rdf:resource="Historical_buildings"/>
 </rdfs:Class>

<rdfs:Class rdf:about="Monument">
    <rdfs:subClassOf rdf:resource="Historical_buildings"/>
 </rdfs:Class>

 <rdfs:Class rdf:about="Palace">
    <rdfs:subClassOf rdf:resource="Historical_buildings"/>
 </rdfs:Class>

 <rdfs:Class rdf:about="Temple">
    <rdfs:subClassOf rdf:resource="Historical_buildings"/>
 </rdfs:Class>

 <rdfs:Class rdf:about="Temple_fortress">
    <rdfs:subClassOf rdf:resource="Historical_buildings"/>
 </rdfs:Class>

 <rdfs:Class rdf:about="National_library">
    <rdfs:subClassOf rdf:resource="Historical_buildings"/>
 </rdfs:Class>

 <rdfs:Class rdf:about="Industrial_buildings">
    <rdfs:subClassOf rdf:resource="Modern_buildings"/>
 </rdfs:Class>

 <rdfs:Class rdf:ID="Office_buildings">
  <rdfs:subClassOf rdf:resource="Modern_buildings"/>
 </rdfs:Class>

 <rdfs:Class rdf:ID="Modern_palace">
  <rdfs:subClassOf rdf:resource="Modern_buildings"/>
 </rdfs:Class>

 <rdfs:Class rdf:ID="Power_station">
  <rdfs:subClassOf rdf:resource="Modern_buildings"/>
 </rdfs:Class>
```

```
   <rdfs:Class rdf:ID="University">
    <rdfs:subClassOf rdf:resource="Modern_buildings"/>
  </rdfs:Class>
```

Accommodation RDFS Type Hierarchy

```
<rdfs:Class rdf:ID="Accommodation"/>

 <rdfs:Class rdf:about="Guest_house">
    <rdfs:subClassOf rdf:resource="Accommodation"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="Hotel">
    <rdfs:subClassOf rdf:resource="Accommodation"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="Apartment">
      <rdfs:subClassOf rdf:resource="Accommodation"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="Lodge">
      <rdfs:subClassOf rdf:resource="Accommodation"/>
  </rdfs:Class>

  <rdfs:Class rdf:ID="Resort">
    <rdfs:subClassOf rdf:resource="Accommodation"/>
  </rdfs:Class>
```

Region RDFS Type Hierarchy

```
<rdfs:Class rdf:ID="Country"/>
<rdfs:Class rdf:ID="Region"/>
<rdfs:Class rdf:ID="Province"/>
<rdfs:Class rdf:ID="Block"/>
<rdfs:Class rdf:ID="Subblock"/>

 <rdfs:Class rdf:about="City">
   <rdfs:subClassOf rdf:resource="Subblock"/>
 </rdfs:Class>

 <rdfs:Class rdf:about="Town">
    <rdfs:subClassOf rdf:resource="Subblock"/>
 </rdfs:Class>

  <rdfs:Class rdf:about="Village">
     <rdfs:subClassOf rdf:resource="Subblock"/>
  </rdfs:Class>

  <rdfs:Class rdf:about="Place">
    <rdfs:subClassOf rdf:resource="Subblock"/>
  </rdfs:Class>
 </rdf:RDF>
```

Vita

Candidate's full name:

Tshering Dema

University attended (with dates and degrees obtained):
Kanglung College,
University of Delhi, 2001-2004
Bachelor of Computer Science, 2004

University of New Brunswick, 2005-2008

Publications:

Harold Boley, Greg Sherman, Tshering Dema, Benjamin Larry Craig, Judy Zhao "Translating FOAF/RDF-like Profiles for an eTourism Use Case of OO jDREW", 5th Annual Research Exposition, Fredericton, New Brunswick, Canada, April 10, 2008.

Harold Boley, Benjamin Larry Craig, Judy Zhao, Greg Sherman, Tshering Dema "Combining Rules, Taxonomies and Probabilities in the Extended OO jDREW Rule Engine", 5th Annual Research Exposition, Fredericton, New Brunswick, Canada, April 10, 2008.