Inducing Queries from Examples of World Heritage

Jing Zhou
Yukun Zheng

Kgs. Lyngby 2003
IMM-THESIS-2003-57

Inducing Queries from Examples of World Heritage

Jing Zhou
Yukun Zheng

Kgs. Lyngby 2003

Technical University of Denmark
Informatics and Mathematical Modelling
Building 321, DK-2800 Lyngby, Denmark
Phone +45 45253351, Fax +45 45882673
reception@imm.dtu.dk
www.imm.dtu.dk

IMM-THESIS: ISSN 1601-233X

Content

List of Figures

1 Introduction

In this thesis, we propose a system that can computationally induce the user's interests on the basis of his/her browsing through the website of the World Heritage sites. A general approach of designing and implementing a query induction system will be presented, using the World Heritage sites as a case study.

This chapter describes the problem to be solved, the motivation and the goals and the related work of others that has inspired our thesis. Finally, we give a guide for reading this thesis.

1.1 Problem

With the advent of large catalogues and databases on the Net, it becomes a time-consuming job for the user to find the needed information. Some kind of easy-to-use retrieval and search interface is needed by the end-users. The most commonly used query technology is the keyword query. The keyword query is widely used because it is simple and the users are familiar with it. But the functionality of keyword query is limited for it is based on syntax matching. Sometimes, the keyword query gives the user a large amount of answers, without the information that the user is really looking for.

As most of the other websites currently, the website of the World Heritage sites also provides the keyword query. Problems arise because the World Heritage sites cover a large range of topics, which increases the difficulty for the user to find the needed information by the keyword query. For example, if a user is interested in the World Heritage sites with information about mammals, and input “mammal” as the keyword. The user can only find those sites whose texts include the word “mammal’ . The keyword query cannot find the sites that are related to the topic of mammals but without the word “mammal” in their texts.

If one site is about monkeys but without the word “mammal’ in its text, then the user can’t find it by the query on the keyword “mammal”.

In order to overcome the above problem, we propose a query induction system that can learn the users interests on the basis of his/her browsing through the website.

Considering the example above, if a user has visited sites on the topic of monkeys and whales, the query induction system could induce that the user is interested in the topic of mammals. The system will automatically suggest the user all the sites concerning the topic of mammals by the concept induction in an ontology rather than the syntax matching of the word “mammal’ . Therefore, the user could obtain the sites about other mammals, for example gorillas, without any further query.

Figure 1-1 illustrates the functionality of a query induction system. The topic of the mammals is induced because the user feels interested in the topics of monkeys and whales.

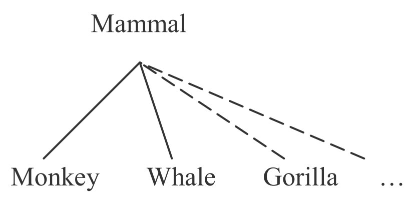


Figure 1-1: Induce the Topic of Mammals

The above example is quite simple. But, as mentioned before, the World Heritage sites are involved in many different subjects from nature science to culture knowledge. So developing the query induction system for the World Heritage sites is not only about system design but also about how to build up an ontology of the World Heritage sites.

1.2 Motivation and Goals

The query induction is a practical concept learning approach. For the websites that provide a lot of information with complicated concepts, for example the website of the World Heritage sites, the query induction is a good way of helping the users to find information that they are looking for.

We aim to study and extend the previous theories and methods concerning the query induction, and find a general way to apply them on a practical case. We consider a case study of World Heritage sites as an example to illustrate the methods and development steps of a query induction system. Once we have developed a query induction system for the World Heritage site, the methods and development steps can also be applied to the other domains.

The goals of this thesis are:

1. Study and extend the previous theories and methods concerning the query induction.

2. Use an E/R diagram to model the application domain for a query induction system, working on the World Heritage sites as a case study.

3. Use a generative ontology to model the application domain for a query induction system, working on the World Heritage sites as a case study.

4. Design the query induction process and system architecture for a query induction system, working on the World Heritage sites as a case study.

5. Adopt the suitable technologies to implement the query induction system; test and validate the query induction system after the implementation.

1.3 Related Work

Many websites are trying to suggest the user more information according to his/her interest. For example, some e-business websites suggest the user additional goods using the dynamic recommendation technology [10].

Although this technology is easy to implement, it does not follow a formal and systematic approach.

The query induction follows a more formal and systematic approach. The query induction is designed as a formal concept induction process as proposed in [9]. Lattice-structured conceptual models are used for concept induction [9].

Currently, there are many papers concerning ontology-based querying. In an ontology-based querying, ontology and generative ontology are used for concept induction.

A generative ontology extends an ontology which has the ISA relation as the backbone with some other relations. The suitable number of available relations may vary with different domains, but among the most domain models are WRT (With-respect-to), CHR (Characterized-by), CBY (Cause-by), TMP (Temporal), LOC (Location) [8].

A concept language ONTOLOG defines a set of semantic relations which can be used for "attribution" (feature-attachment) to form the compound concepts. The language ONTOLOG can be used to describe the concepts and the semantic relations in the generative ontology. The inclusion relation among the compound concepts is a knowledge based inclusion ISA_{KB} [1].

1.4 How to Read This Thesis

We organize this thesis by following the process of the software developing: analysis, design, implementation and test. Furthermore, in order to make the readers have a better understanding of this thesis, we also describe the theories and methods concerning the query theory and the ontology in some chapters.

In Chapter 2 **Domain Analysis of World Heritage**: We give a description of the World Heritage domain. The basic information about World Heritage sites is introduced. We analyse the important features of the World Heritage sites, which will contribute to the development of a query induction system later.

In Chapter 3 **E/R Model of World Heritage**: We use the E/R model as a semi-formal tool for conceptual modelling of the World Heritage domainThe readers will obtain more information of the application domain through the E/R model.

In Chapter 4 **Ontology and Generative Ontology**: First, we describe the basic concepts and theories about the ontology and the generative ontology. Then, the methods for building the generative ontology from the ER diagram will be discussed.

In Chapter 5 **Generative Ontology of World Heritage**: We present the generative ontology of the World Heritage. This chapter applies the theories and methods that are stated in Chapter 4 to the World Heritage domain. This chapter includes a lot of information of the World Heritage sites. Many Hasse diagrams are used to present the regions in the generative ontology of World Heritage.

In Chapter 6 **Design of Query Induction**: We present our design of the query induction. We describe the four steps of the query induction and several methods that are used to improve Concept Generalisation in detail. These designs can be applied to the different query induction applications. This chapter is an important part of this thesis.

In Chapter 7 **System Design**: We give a description of the three-tier client/server architecture of the query induction system. The designs corresponding to the different tiers of the query induction system of World Heritage are also described in this chapter. Especially the designs of query induction described in Chapter 6 are applied to the web server tier.

In Chapter 8 **System Implementation**: We describe the installation of the system. Some concise explanations of the important source codes, such as the Java classes, are given in this chapter. It will be easier for the reader to understand our implementation with these explanations

In Chapter 9 **System Test**: We describe the component test and integration test of the query induction system of World Heritage in this chapter. The readers can find screen shots of many interfaces in this chapter as results of the test.

In Chapter 10 **Conclusion**: We conclude on the achievements in this thesis. Some ideas for future improvements are also provided.

2 Domain Analysis of World Heritage

The purpose of this chapter is to obtain a good understanding of the World Heritage domain. This is done by first studying and analyzing the available information of the World Heritage sites. Then some prominent features of the World Heritage sites are chosen as the focuses for the following development of a query induction system.

2.1 Overview of the World Heritage Sites

World Heritage is the domain of this project. According to the list from World Heritage Committee 3rd of July 2003, there are 754 World Heritage sites located in 128 counties. The World Heritage Committee is an official organization, which defines the World Heritage sites and provides aid to conserve them. This is done to make sure that our future generations can inherit the treasures of the past.

The World Heritage site is something with outstanding universal value that we get from our earlier generations or from nature, for example a masterpiece of creative genius from the 16th century. Each site has a unique name, a picture, a brief description, the place where it is located and the year when it was inscribed by World Heritage Committee. Furthermore, a site must satisfy certain selection criteria. The criteria have evolved to match the evolution of the World Heritage concept itself. There are two kinds of selection criteria: natural criteria and cultural criteria[1].

Natural site: A site is considered a natural site when it meets one or more natural criteria. A natural site designates outstanding physical, biological, geological, or aesthetic features. A natural site might also be a habitat of threatened plants or animal species.

Cultural site: A site is considered a cultural site when it meets one or more cultural criteria. A cultural site is a monument, group of buildings or site of historical, aesthetic, archaeological, scientific, ethnological or anthropological value.

A site is considered a mixed site when it meets both some natural criteria and some cultural criteria.

Among all 754 sites, 149 are natural sites, 582 are cultural sites, and 23 are mixed sites. In order to illustrate the World Heritage sites, some sample sites are given as follows.

Here is a natural World Heritage site:

Sample1:

Fraser Island – Australian

Inscribed Year: 1992 Criteria: Natural (ii) (iii)

Brief Description: Fraser Island lies just off the east coast of Australia. At 122 km long, it is the largest sand island in the world. Majestic remnants of tall rainforest growing on sand and half the world's perched freshwater dune lakes are found inland from the beach. The combination of shifting sand dunes, tropical rainforests and lakes makes it an exceptional site.

The following are two cultural World Heritage sites:

Sample 2:

Amiens Cathedral - France

Inscribed Year: 1981 Criteria: Cultural (i) (ii)

[1] For detailed information about criteria, refer to: http://whc.unesco.org/nwhc/pages/doc/mainf3.htm

Brief Description: Amiens Cathedral, in the heart of Picardy, is one of the largest 'classic' Gothic churches of the 13th century. It is notable for the coherence of its plan, the beauty of its three-tier interior elevation and the particularly fine display of sculptures on the principal facade and in the south transept.

Sample 3:

Historic Monuments of Ancient Kyoto - Japan

Inscribed Year: 1994 Criteria: Cultural (ii) (iv)

Brief Description: Built in A.D. 794 on the model of the capitals of ancient China, Kyoto was the imperial capital of Japan from its foundation until the middle of the 19th century. As the centre of Japanese culture for more than 1,000 years, Kyoto illustrates the development of Japanese wooden architecture, particularly religious architecture, and the art of Japanese gardens, which has influenced landscape gardening the world over.

A sample of a mixed World Heritage site:

Sample 4:

Mount Tai – China

Inscribed Year: 1987 Criteria: C (i) (ii) (iii) (iv) (v) (vi); N (iii)

The sacred Mount Tai was the object of an imperial cult for nearly 2,000 years, and the artistic masterpieces found there are in perfect harmony with the natural landscape. It has always been a source of inspiration for Chinese artists and scholars and symbolizes ancient Chinese civilizations and beliefs.

2.2 Key Features of the World Heritage Sites

Category and *Geographical Location* are two features of the World Heritage sites that can be analysed and classified for the following development of a query induction system. Therefore, *Category* and *Geographical Location* are of special interests in our domain analysis.

2.2.1 Category

The *Category* of the sites is an important feature of the World Heritage sites. What kind of site is it? What is the research interest of it? Referring to the description of World Heritage sites and the criteria they meet, we can get the *Category* of the World Heritage sites.

The current website of the World Heritage provides the user a classification of sites by their categories as follows:

- Natural Sites
    - Fossil Sites
    - Biosphere Reserves
    - Ramsar Wetlands
    - Tropical Forests
    - Biogeographically Regions
- Cultural Sites
    - Hominid Sites
    - Industrial Sites
    - Cultural Landscapes
    - Rock-art Sites
    - Historic Cities and Towns

List 2-1: Classification of Sites by Categories from World Heritage Sites Website

The World Heritage sites are first divided into *Natural Sites* and *Cultural Sites*. And then more specific classifications by the categories are introduced under *Natural Sites* and *Cultural Sites*. For example, *Fossil Sites* is listed under *Natural Sites*.

The classification shown in List 2-1 is not complete, because it only covers some World Heritage sites, leaving many World Heritage sites uncovered. For example, there are six groups of sites below *Cultural sites*, which are *Hominid Sites*, *Industrial Sites*, *Cultural Landscapes*, *Rock-Art Sites*, and *Historic Cities and Towns*. *Amiens Cathedral* (sample 2) is a cultural site, but it can not be found in any of the above six groups.

In order to give a complete classification of *Category*, we referred the classification in List 2-1 and went through the descriptions and criteria of all World Heritage sites. Knowledge of many other fields was also taken into consideration, such as nature sciences, histories, and religions.

After we studied information of all sites and related subjects, we give a classification of *Category* as follows

- Natural Category
    - Natural Beauty
    - Topography
    - Geography
    - Biodiversity
    - Geology
    - Ecosystem
    - Biology
    - ‘Wintering’ Land
- Cultural Category
    - Hominid
    - Archaeology
    - Artistic Work
    - Historic City and Area
    - Construction
    - Cultural Landscape

List 2-2: Classification of Category of World Heritage Sites

The classification of *Category* in List 2-2 is complete. These categories can cover all the World Heritage sites.

Although some specific categories are listed under *Natural Category* and *Cultural Category*, the categories in List 2-2 still can not describe every World Heritage site explicitly. For example about one hundred sites belong to the category of *Historic City and Area*, and these sites can be classified into more specific categories. For example, *Historic Capital* is a more specificcategory, which is a sub-classification of the category *Historic City and Area*.

Let us take the four World Heritage sites listed above as examples to illustrate the specific categories, which can describe the World Heritage sites explicitly.

Fraser Island (sample 1) belongs to the categories: *Lake, Rain Forest*, and *Island*, which can be listed under *Geography* and *Geology*.

Amiens Cathedral (sample 2) belongs to the category: *Cathedral*, which can be listed under *Construction*.

Historic Monuments of Ancient Kyoto (sample 3) belongs to the category: *Historic Capital*, which can be listed under *Historic City and Area*.

Mount Tai (sample 4) is a mixed site, having both natural properties and cultural properties. Therefore the categories of it are: *Mountain*, *Artistic Work* and *Associative Cultural Landscape*. *Mountain* can be listed under

Topography and *Associative Cultural Landscape* can be listed under *Cultural Landscape*.

The specific categories of the four sample sites which were mentioned above can be listed under the categories that are more general them, shown in List 2-3. There are many specific categories and they will be identified in Chapter 5.

- Natural Category
    - Natural Beauty
    - Topography
        - Mountain
    - Geography
        - Lake
        - Rainforest
    - Biodiversity
    - Geology
        - Island
    - Ecosystem
    - Biology
    - 'Wintering' Land
- Cultural Category
    - Hominid
    - Archaeology
    - Artistic Work
    - Historic City and Area
        - Historic Capital
    - Construction
        - Cathedral
    - Cultural Landscape
        - Associative Cultural Landscape

List 2-3: Detailed Classification of Category of World Heritage Sites

2.2.2 Geographical Location

Geographical location of the World Heritage Sites is the other interesting feature that can be analyzed and classified.

Most sites are situated at a particular location such as a country. The scope of location can also be expanded to the composition of continent and continent.

For example *Amiens Cathedral* is located in *France*. *France* is a country located in the composition of continent *Western Europe*. And *Western Europe* is located in the continent *Europe*.

Some World Heritage sites cover a large area, which means they might be located in two or more countries. For example, the site *Caves of Aggtelek Karst and Slovak Karst*, is located in two countries *Hungary* and *Slovakia*.

Geographical Location is an interesting feature of the World Heritage sites. Some groups of sites, which are located close to each other, share some common traits. In natural sites, this phenomenon can be understood as: the evolution of earth and evolution of the living things in the earth have certain similarity within a geographical area. In cultural sites, this phenomenon can be explained as the cultural exchanges and influences within certain areas. For example, some cultural sites, which are close to each other, may have similar architectural styles and urban planning styles. Therefore, *Geographical Location* is a focus for the following development of a query induction system.

2.3 Conclusion

The domain analysis of World Heritage provides a study of the detailed information of the World Heritage sites.

Since in the following chapters we will further describe how to develop a query induction system based on the given domain analysis in this chapter, it is important that appropriate aspects from the domain are identified and analyzed. After studying the World Heritage sites, we choose *Category* and *Geographical Location* as important features of the World Heritage sites for the query induction system. In the later chapters, more analysis and modelling will be applied to *Category* and *Geographical location* of the World Heritage sites.

3 ER Modelof World Heritage

From the previous chapter, we have got a lot of information in the World Heritage domain. In order to understand the World Heritage domain more clearly, we classify the information into classes and relationships, and present the ER model of World Heritage in this chapter.

3.1 ER Model

The Entity Relationship Data Model (ER model) is a graph-oriented data presentation, which is popular in conceptual modelling and database design [3]. The ER model benefits from its pictorial presentation, using diagrams to model and represent the structure of data.

The components in ER models are [12]:

- *Entity Set* is a collection of *entities* with similar properties. *Entities* in an *Entity Set* are members of this *Entity Set*. *Entity Set* can be connected to other *Entity Set* by *Relationship*. *Entity Sets*are analogous to classes; *entities* are analogous to individuals.

- *Attributes* are values describing some properties of an *Entity*.

- *Relationships* are connections among two or more *Entity Sets*. A *Binary Relationship* connects between two *Entity Sets*. Additionally, an

Arrow represents the direction of a *Relationship* in ER model to reduce the conflicts.

In this project, we present the ER model using the elements as below:

- Entity Sets, represented by rectangles;
- Relationships, represented by diamonds with arrows presenting the directions;
- Connections between entity sets and relationships, represented by edges.

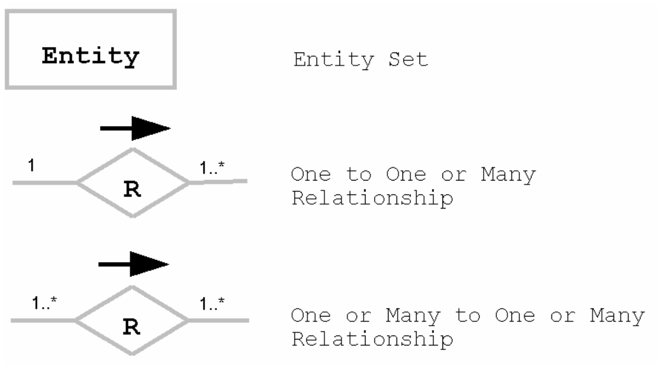

Entity
Entity Set
1
1..*
R
One to One or Many Relationship
1..*
1..*
R
One or Many to One or Many Relationship

Figure 3-1: Elements in the ER model

In many textbooks and papers, attributes are represented by ovals in the ER model. Because of the space limitation, we do not indicate the attributes in the diagram. Instead, they are explained in Section 3.3 with the explanation of entity sets.

3.2 Overview of the ER Model of World Heritage

Based on the domain analysis, we use the following figure to model the entity sets and relationships of the World Heritage domain. Many entity sets, relationships and attributes are needed to give a complete presentation. Here we only present the entity sets and relationships which are important to the following development of a query induction system.

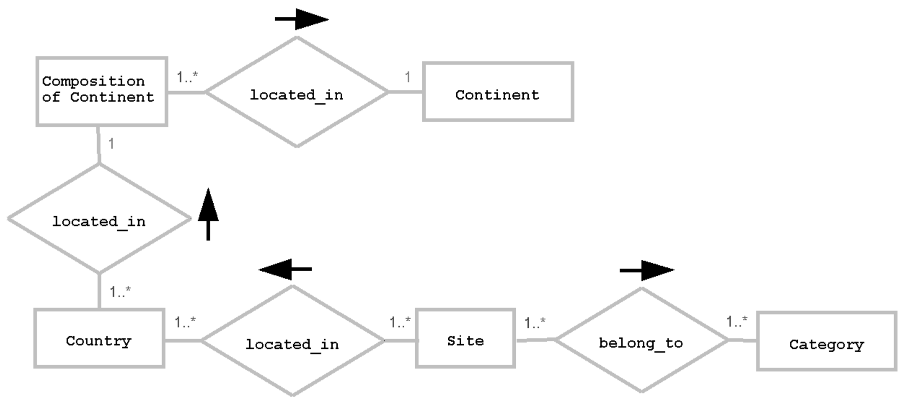


Figure 3-2: ER Model of World Heritage

Next, the detailed explanations of the entity sets, attributes and relationships will be given.

3.3 Explanation of the ER Model of World Heritage

3.3.1 Information about Site

The World Heritage domain consists of 754 World Heritage sites. The entity set *Site* stands for the World Heritage sites. The entity set *Site*has attributes *name* and *brief description*. The relationship between the entity set *Site*and entity set *Category* is *belong_to*. The relationship between the entity set *Site* and entity set *Country* is *located_in.*

The explanation of entity set *Site*

- ***Site***: is the entity set of the World Heritage sites, including its name and brief description. We take *Fraser Island* (sample 1 from Chapter 2) as an example to explain the entity set *Site*

The attributes of entity set *Site* are:

- ***name***: Every entity in the entity set *Site*has a unique name. We identify a World Heritage site by its name, for example *Fraser Island.*

- ***description***: Every entity in the entity set *Site*has a short text to describe its outstanding universal value. For example, the description of *Fraser Island* is "*Fraser Island lies just off the east coast of*

Australia. At 122 km long, it is the largest sand island in the world. Majestic remnants of tall rainforest growing on sand and half the world's perched freshwater dune lakes are found inland from the beach. The combination of shifting sand dunes, tropical rainforests and lakes makes it an exceptional site."

The relationships connecting entity set *Site*

- *Site* **belong_to** *Category*:

  Every entity in the entity set *Site* belongs to one or many categories. For example, *Fraser Island* belongs to the categories: *Rainforest*, *Lake*, and *Island*.

  Several entities in the entity set *Site*may belong to one same category. For example, there is another World Heritage site named *Macquarie Island* which also belongs to the category: *Island*.

  Therefore the multiplicity of this relationship is: *one or many* to *one or many*.

- *Site* **locate_in** *Country*:

  Every entity in the entity set *Site*is located in one or many counties. For example, *Fraser Island* is located in the country *Australia*. The site named *Caves of Aggtelek Karst and Slovak Karst* is located in both *Hungary* and *Slovakia*.

  Several entities in the entity set *Site* might be located in the same location. Same as Fraser Island, the *Macquarie Island* site is also located in *Australia*.

  Therefore the multiplicity of this relationship is: *one or many* to *one or many*.

3.3.2 Information about Category

From the domain analysis, we can see that the different category has different level according to how specific it is.

Natural Category, *Cultural Category* are two general categories, therefore they are in the highest level.

The categories in the higher level are more general than those in the lower level. The more specific categories are listed under *Natural Category*, *Cultural Category*. For example *Historic City and Area* is listed under *Cultural Category*.

If the categories in the lower level, they are more specific. For example *Historic Capital* is listed under *Historic City and Area*, and *Historic Capital* is more specific than *Historic City and Area*.

Therefore, the categories are classified into different levels as shown in List 2-3 in Chapter 2. We will give more study in the levels of categories in Chapter 4 and 5.

In the ER model of World Heritage, the entities in entity set *Category* are the specific categories of the World Heritage sites, because they can describe the sites most explicitly. Take the site *Historic Monuments of Ancient Kyoto* (sample 3 in Chapter 2) as an example; we define its category as *Historic Capital*, rather than *Historic City and Area* or *Cultural Category*. Therefore, we choose the specific categories as the entities describing the World Heritage sites in the database.

The explanation of entity set *Category*:

- ***Category***: a specific class or group in a classification system according to what kind of the World Heritage site it is, such as *Rainforest*, *Island*, and *Historic Capital*.

The attribute of entity set*Category* is:

- ***name***: Every entity in the entity set *Category* has a unique name. We identify a category by its name.

The relationship connecting to entity set Category:

- *Site* **belong_to** *Category*:

  It has been described in the explanation of entity set *Site*

3.3.3 Information about Geographical Location

Ordered by the geography scope, the entity set *Continent*, *Composition of Continent*, and *Country* are considered as one group of information about

Geographical Location of the World Heritage sites. Therefore, we introduce the concept *Geographical Location*, which includes the information about the entity sets *Continent*, *Composition of Continent*, *Country* and the relationships among them in ER model of World Heritage.

The entity sets and relations in *Geographical Location* as shown in Figure 3-2.

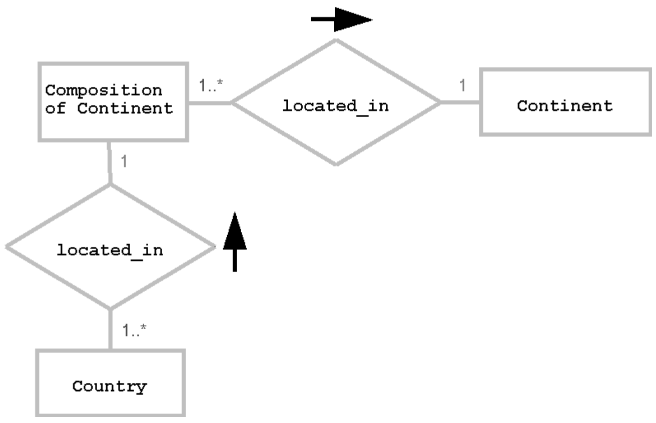


Figure 3-3: Entity Sets and Relationships in Geographical Location

The explanation of the entty sets with their attributes and relationships in the concept *Geographical Location*:

The explanation of entity set *Country*:

- ***Country***: geography and politics concept of country.

The attribute of entity set*Country* is:

- ***name***: Every entity in the entity set *Country* has a unique name. We identify a country by its name, for example *Denmark*.

The explanation of entity set *Composition of Continent*:

- ***Composition of Continent***: sub-area of a continent.

The attribute of entity set*Composition of Continent* is:

- ***name***: Every entity in the entity set *Composition of Continent* has a unique name. We identify a *composition of continent* by its name, for example *North Europe*.

The explanation of entity set *Continent*:

- ***Continent***: geography concept of continent, for example *Europe*.

The attribute of entity set*Co ntinent* is:

- ***name***: Every entity in the entity set *Continent* has a unique name. We identify a *continent* by its name, for example *Europe*.

The relationship connecting to entity set *Country*:

- *Site* **located_in** *Country*:

  It has been described in the explanation of entity set *Site*

- *Country* **located_in** *Composition of Continent*:

  One or many countries are located in one composition of Continent. For instance, *Denmark*, *Sweden*, *Norway*, and *Finland* are all located in *North Europe*.

  Therefore the multiplicity of this relationship is: *one or many* to *one*.

  There are some special cases that some countries are located in more than one continent. For example, *Russian* is located in *Eastern Europe* and *Northern Asia* according to its geographical scope. We simplify this relationship and classify *Russian* to *Eastern Europe* because its political and economical centres are in *Eastern Europe*.

- *Composition of Continent* ***located_in*** *Continent*:

  One or many compositions of continent are located in one continent. For instance, *North Europe*, *West Europe*, *South Europe*, *Middle Europe* and *East Europe* are all located in *Europe*.

  One composition of continent is only located in one continent. For instance, *North Europe* is only located in *Europe*, not located in any other continents.

  Therefore the multiplicity of this relationship is: *one or many* to *one*.

3.4 Conclusion

The ER model serves as a semi-formal tool for modelling the World Heritage domain. The domain of World Heritage was modelled by the ER diagram with some entity sets and the relationships among them. Through building the ER model for the World Heritage domain, we obtained much more information and understood the domain more clearly.

The ER model of World Heritage is also helpful for us to build the generative ontology of World Heritage in the later chapters, because the entity sets, entities and relationships in the ER model will becomethe concepts and the relations in the generative ontology. In the next two chapters, we will move on to build the generative ontology of World Heritage.

4 Ontology and Generative Ontology

In Chapter 3, we built the ER model of World Heritage. In this chapter, we will introduce the ontology and the generative ontology, which can also be used to model the application domain.

The advantage of using the ontology and the generative ontology to model the application domain is that they can be used for concept induction. The concept induction is a key function of a query induction system.

In the first part of this chapter, we will introduce the ontology, including its definition, elements, structure and the rules of concept induction in it. Then the generative ontology which is an extension of the ontology will be presented. Finally, the methods of building the generative ontology on top of the ER model will be discussed.

This chapter provides the ideas and methods to model the application domains of query induction by the ontology and generative ontology. These ideas and methods will be used to model the World Heritage domain in Chapter 5.

4.1 Ontology

4.1.1 What is an Ontology

The ontology plays a key role when developing a query induction system. The term “ontology” has been used by the artificial intelligence and knowledge representation community for a long time. And the term “ontology” is now becoming part of the standard terminology of a much wider community.

> ONTOLOGY
>
> 1. A systematic account of existence.
>
> 2. (From philosophy)An ontology is an explicit formal specification of how to represent the objects, concepts and other entities that are assumed to exist in some area of interest and the relationships that hold among them.
>
> 3. In extension, an ontology is a hierarchical structuring of knowledge about things by sub-categorizing them according to their essential (or at least relevant and/or cognitive) qualities.
>
> - HyperDictionary: www.hyperdicationary.com

4.1.2 Concepts and Relations in Ontology

An ontology defines a set of concepts and situates the concepts in a hierarchical structure according to how the concepts are related. Concepts and relations are basic elements in an ontology.

Concepts in an ontology comprise [8]:

- General concepts (classes), also called universal concepts or just universals.
- Individual concepts (individuals or particulars).

Relations in an ontology are the relations among concepts. The most widely used relation in an ontology is the ISA relation. The ISA relation can express the class inclusion and membership of an individual in a class.

An example of using ISA relation to express the class inclusion is shown in Figure 4-1. We use the arrows to show the direction of the ISA relation. For example, *Country* is a subclass of *Geographical Location*.

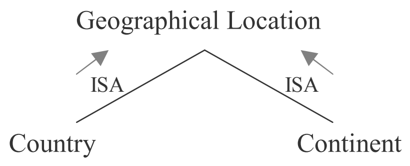


Figure 4-1: Class Inclusion Expressed by ISA Relation

An example of using ISA relation to express the individual membership in a class is shown in Figure 4-2. We use the arrows to show the direction of the ISA relation. For example, *France* is a member of the class *Country*.

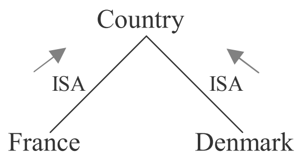


Figure 4-2: Individual Membership Expressed by ISA Relation

The relations in an ontology not only relate the concepts, they are also the partial orders that can order the concepts into a hierarchical levels within a Lattice skeleton.

4.1.3 Partial Order and Lattice

Before we discuss more about ontology, we will first give a description of Partial Order and Lattice because the properties of Partial Order and Lattice are important to an ontology. [5].

Partial Order

Definition: A binary relation '$\leq$ 'on a set $\boldsymbol{P}$ is a partial order on $\boldsymbol{P}$ if the following hold for all $x, y, z \in \boldsymbol{P}$:

- $x \leq x$ (reflexive)
- $x \leq y$ and $y \leq x$ implies $x = y$ (anti-symmetric)
- $x \leq y$ and $y \leq z$ implies $x \leq z$ (transitive)

ISA relation is a partial order, because it is reflexive, anti-symmetrical and transitive.

One of the nice features of partial orders is that they may be represented as directed graphs. Normally, partial orders of reasonable size may be drawn by means of the following procedure:

- If $x \le y$ then we place x below y and draw a line from x to y with the exceptions that
- Lines following from reflexivity are omitted
- Lines following from transitivity are omitted

Such diagrams are known as **Hasse diagrams** In a Hasse diagram, if there is a path from x to y in which x is strictly below y, then $x \le y$. The directions of the orders are already presented in a Hasse diagram, so no need to use the arrows to indicate the directions.

A Hasse diagram can handle basic ordering relations such as the ISA relation, classification or the Part-Of relation. A Hasse diagram can also handle some other binary relations as long as there is a relation who can order the concepts without introducing loops.

For example, we introduce a relation *LOC (located_in)* that can order the relations among the general concepts of *Country*, *Composition of continent* and *Continent*, as shown in Figure 4-3. For example: *France* is located in *Western Europe* and *Western Europe* is located in *Europe*.

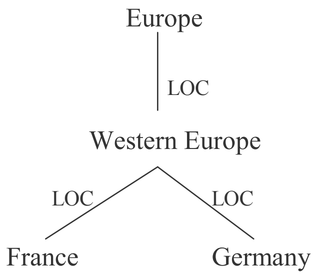


Figure 4-3: Hasse Diagram by the Partial Order - LOC

Lattice

Definition: Let ***P*** be a partially ordered set and let $a \in \boldsymbol{P}$. If $x \le a$ for all $x \in \boldsymbol{P}$ then a is the **greatest element** of ***P***, which is denoted by TOP. Let $b \in \boldsymbol{P}$. If $b \le x$ for all $x \in \boldsymbol{P}$ then b is the least element of ***P***, which is denoted by BOTTOM.

According to Figure 4-3, the TOP of the Hasse diagram is *Europe*. But there is no bottom. If we add an element, BOTTOM, and place BOTTOM below all the elements, then we obtain a BOTTOM of this Hasse diagram.

Definition: Let ***P*** be a partially ordered set and let $x, y \in \boldsymbol{P}$. An element $d \in \boldsymbol{P}$ is an **Upper Bound** of the pair (x, y) if $x \leq d$ and $y \leq d$. An element $c \in \boldsymbol{P}$ is the **Least Upper Bound**, notation $c = \sup (x, y)$, if c is an Upper Bound of (x, y) and for any other Upper Bound d of (x, y) we have that $c \leq d$.

In figure 4-3, sup (*France*, *Germany*) = *Western Europe*.

Definition: Let ***P*** be a partially ordered set and let $x, y \in \boldsymbol{P}$. An element $d \in \boldsymbol{P}$ is a **Lower Bound** of the pair (x, y) if $d \leq x$ and $d \leq y$. An element $c \in \boldsymbol{P}$ is the **Greatest Lower Bound**, notation $c = \inf (x, y)$, if c is a Lower Bound of (x, y) and for any other Lower Bound d of (x, y) we have that $d \leq c$.

Definition: Let ***L*** be a non-empty set with a partial order. If sup (x, y) and inf (x, y) exist for all x, y ∈ ***L*** then ***L*** is a **Lattice**.

The following figure is a classification presented by Lattice. The category of *Fossil- Hominid* covers those sites which both belong to the categories of *Hominid* and *Fossil*. A World Heritage site in *Australia* named *Willandra Lakes Region* belongs to the category of *Fossil-Hominid*, because this site has fossil of a series of lakes and sand formations and the evidence of human evolution.

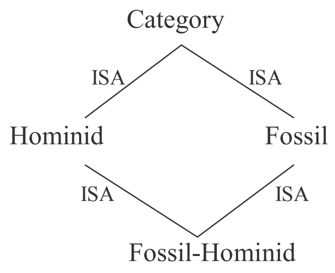


Figure 4-4: Classification Presented as Lattice

A bounded Lattice is a lattice with TOP and BOTTOM. Every **Finite Lattice** is a bounded lattice.

4.1.4 Structure of Ontology

Finite lattice, as defined in above, can be used as the skeleton of an ontology. An ontology can be formalised as lattice with two dimensions – Regions and Levels [8].

The TOP of an ontology is anything, which covers all the concepts in the ontology. The BOTTOM of an ontology is null, because the more specialised the concepts are, the more conditions they need to meet. There is no instance that can meet all the conditions. For example, in the World Heritage domain, there are no sites that are located in all the countries. Therefore, the BOTTOM of an ontology is null.

Regions of Ontology

Concepts are classified into groups according to their semantic relations, for example, all the concepts concerning foods can make a group of concepts about foods. An ontology may be divided into several regions corresponding to the groups of concepts.

In Figure 4-5, the diamond in the left gives a general idea of the regions of an ontology. And the diamond in the right is an example - the regions of the ontology of World Heritage. There are three regions: *Site, Category*, and *Geographical Location*.

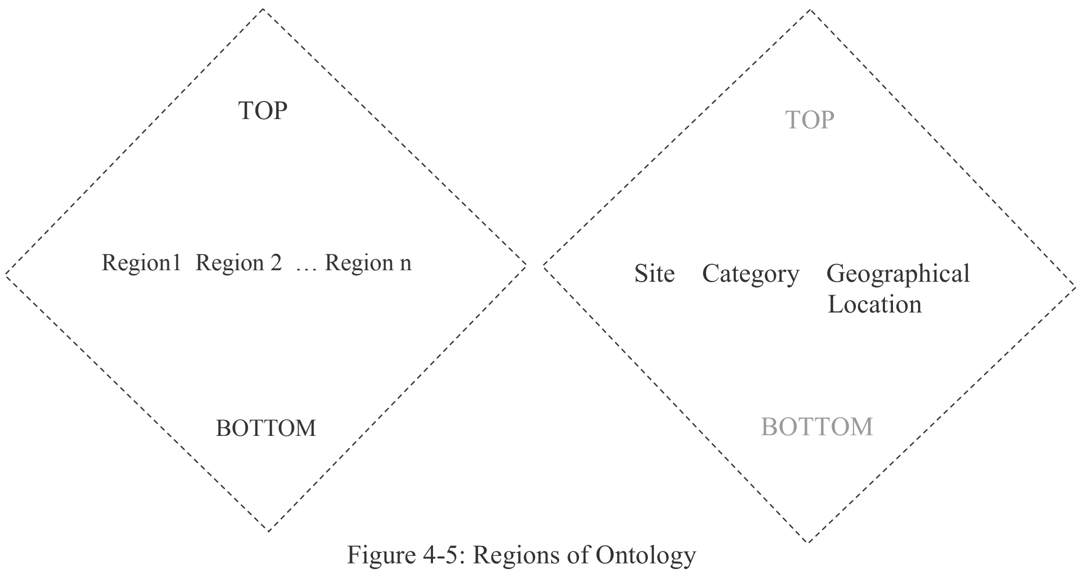

TOP
Region1 Region 2 … Region n
BOTTOM
TOP
Site Category Geographical Location
BOTTOM

Figure 4-5: Regions of Ontology

Levels of Ontology

An ontology may also be divided into several levels. The concepts in an ontology are arranged into hierarchical levels. Basically, the general concepts (classes) are situated in the general level. The less general concepts are situated in the medium level, for example the subclasses. The specialised concepts (individuals or particulars) are situated in the specialised level, for example the instances of the classes.

In order to situate the concepts into hierarchical levels, concepts needed to be ordered by the partial orders. When the concepts in one region of the ontology can be ordered by the partial orders, we can use a Hasse diagram to present the concepts and their relations.

For example, the ISA relation is a partial order that can order the concepts. The levels among the categories which have been mentioned in Chapter 2 and 3 are presented as a Hasse diagram with ISA relation in Figure 4-6.

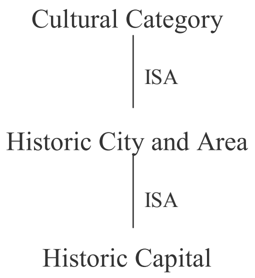


Figure 4-6: Concepts Situated in Hierarchical Levels

According to the property of partial order, the concepts become more and more specialised by moving downwards the Hasse diagram.

In Figure 4-6, the concept *Cultural Category* is a general concept. The concept *Historic City and Area* is a subclass of *Cultural Category* that are more specialised. The concept *Historic Capital* is a subclass of *Historic City and Area*, so the concept *Historic Capital* is the most specialised among the three concepts. Figure 4-7 illustrates the levels of an ontology.

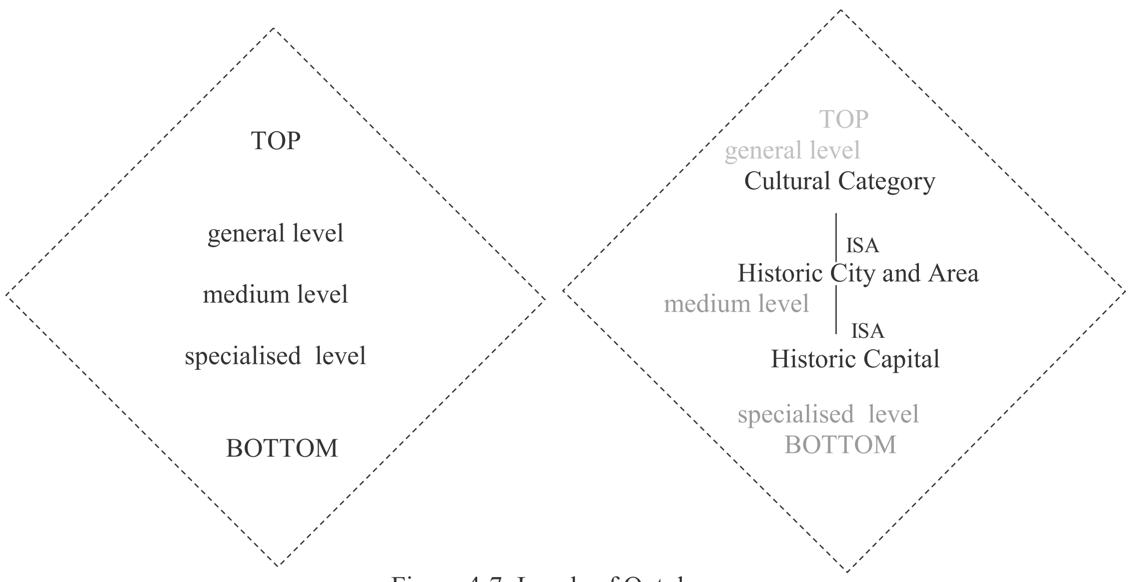


Figure 4-7: Levels of Ontology

4.1.5 Concept Induction in Ontology

The concept induction is a key function of a query induction system, which includes Concept Generalisation and Concept Specialisation. In this section, we will discuss how to induce concepts in an ontology.

The ER model is a widely used tool to model the application domain. But when modelling the application domain for a query induction system, concept induction is an important consideration, which is different from modelling the application domain for other systems. The advantage of using the ontology to model the application domain is that the concept induction can be achieved in the ontology. But the concept induction is harder to be achieved in the ER model.

Concept Ordering

The concept induction is based on the ordering of the concepts. Concepts in an ontology are situated into hierarchical levels: general level, medium level, and specialised level. The partial orders are used to order the concepts into hierarchical levels. So we can apply the properties of partial orders to order the concepts.

In an ontology, we can use the Hasse diagram to order the concepts. As discussed in 4.1.3, in a Hasse diagram, if there is a path from x to y in which x is strictly below y, then $x \leq y$. Therefore, if there is a path from concept x to concept y in which x is strictly below y in a Hasse diagram, then concept x is more specialised than concept y, written as $x \leq y$.

We define the concept ordering in an ontology as follows.

Definition: Assume a set of concepts $\boldsymbol{A}$, and the partial order '$\leq$' means 'more specialised than'. Then concept ordering in an ontology is recursively defined as follows:

- **Definition 1**: If $x, y \in \boldsymbol{A}$, and there is a path from x to y in which x is strictly below y, then $x \leq y$.

Because the partial orders have the property of transitivity, if concept x is more specialised than concept y, and concept y is more specialised than concept z, then concept x is also more specialised than concept z. This can be defined as:

- **Definition 2**: If $x, y, z \in \boldsymbol{A}$, and $x \leq y, y \leq z,$ then $x \leq z$.

Concept Generalisation and Concept Specialisation

After we have ordered the concepts by the partiral order, the concepts are arranged into hierarchical levels. It is easy to execute Concept Generalisation and Concept Specification in a Hasse diagram. The cocnepts become more and more specialised when moving downwards the Hasse diagram.

The generalisation of concept x is achieved by moving upwards the Hasse diagram. Generalisation of a set of concepts x_i ($i = 1, \ldots, n$) is achieved by moving upwards the Hasse diagram to find the concepts that are generalised than any x_i.

For example, we assume all the elements are concepts in the Hasse diagram shown in Figure 4-8, and the partial order is ISA.

Concept Generalisation of (D, E, F) is the set {B, A} and Concept Generalisation of (E, F) is the set {B, C, A}.

More considerations are needed in the process of Concept Generalisation in order to make the result more explicit. Although all the elements in {B, C, A} belong to Concept Generalisation (E, F), {B, C} is a sharper bound, because {B, C} are in the most specialised level among all the elements in {B, C, A}. Therefore among all the generalised concepts, we choose the concepts that are in the most specialised level as the 'Least Concept Generalisation'.

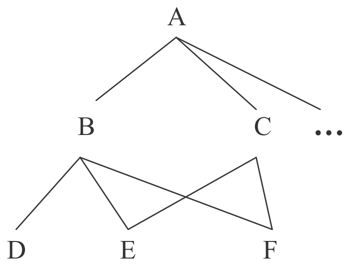


Figure 4-8: Concept Generalisation and Concept Specialisation

Similarly, the specialisation of concept x is achieved by moving downwards the ontology to find the concepts that more specialised than concept x. Specialisation of a set of concepts x_i ($i = 1, \ldots, n$) is achieved by moving downwards the Hasse diagram to find the concepts that are specialised than any x_i.

Concept Specialisation of (C) is the set $\{E, F\}$ and Concept Specialisation of (B, C) is the set $\{E\}$.

Although Concept Generalisation and Concept Specialisation have certain similarity with the theories of Upper Bound and Lower Bound, we can not use them directly.

First, the theories of Upper Bound and Lower Bound are applied to pairs of elements. But Concept Generalisation might be applied to different number of elements, from one to many.

Moreover, if we use the theories of Upper Bound to find the result of Concept Generalisation, we can only get $\{B, C, A\}$ as the result of Concept Generalisation of (E, F), however, A is a too general concepts which may reduce the accuracy of the query induction results. If we use the theories of Least Upper Bound to look for the result of Concept Generalisation of (E, F), we will fail to find any results because there are no Least Upper Bound of (E, F).

However, $\{B, C\}$ is an ideal result of Concept Generalisation, because they cover E and F, but still more specialised than A. Furthermore, B and C can be used for the following steps of the query induction, which we will describe in more detail in the later chapters.

According to the above approach, we give the rules of concept induction, including Concept Generalisation and Concept Specialisation. We give our own sketch instead of a formal definition.

Concept Generalisation and Least Concept Generalisation are extended from Upper Bound and Least Upper Bound in the Hasse diagram. And Concept Specialisation is extended from Lower Bound.

Assume a set of concepts $\boldsymbol{A}$, and the binary relation '$\leq$' means 'more specialised than'. The result of Concept Generalisation and the result of Concept Specialisation are defined as follows:

- **Rule 1**: Let x_i, cg, $lcg \in \boldsymbol{A}$, $i = 1, \ldots, n$

  $cg \in$ **Concept Generalisation** $(x_1, \ldots, x_n)$
  if for any x_i we have $x_i \leq cg$

  $lcg \in$ **Least Concept Generalisation** $(x_1, \ldots, x_n)$

if $lcg \in$ **Concept Generalisation** $(x_1, \ldots, x_n)$ and lcg is in the most specialised level among all the elements in **Concept Generalisation** $(x_1, \ldots, x_n)$

- **Rule 2:** Let xi, $cs \in \boldsymbol{A}$, $i = 1, \ldots, n$

  $cs \in$ **Concept Specialisation** $(x_1, \ldots, x_n)$.
  if for any x_i we have that $cs \le x_i$,

4.2 Generative Ontology

Here, we introduce an extended ontology - generative ontology. A generative ontology is extended with some other relations, having the ISA relation as the backbone. The suitable available relations may vary with different domains, but the important relations that will be presented in most domain models are WRT (with-respect-to), CHR (characterized-by), CBY (caused-by), TMP (temporal), LOC (location)[8].

With the introduction of the extended relations, a generative ontology provides more flexibility to model the application domain. The prominent characteristic of a generative ontology is that it can produce the compound concepts, by connecting the concepts from different regions with the extended relations.

In the following part of this section, we will present the difference between a generative ontology and an ontology, focusing on the extension. And we will introduce a language, ONTOLOG, which can be used to describe the compound concepts and relations in a generative ontology.

4.2.1 Concepts and Relations in Generative Ontology

We can make compound concepts in a generative ontology, by attaching some concepts to another concept with the relations between them.

For example, *Site that are located in France* can be considered as a compound concept in a generative ontology. *France* is the concept attaches to the concept *Site* by the relation *LOC (located_in)*.

In this thesis, we introduce two extended relations in the generative ontology. The first extended relation is *LOC (located_in)*, which is analogous to the relationship *located_in* in the ER model presented in Chapter 3. The second extended relation is *BET (belong_to)*, which is analogous to the relationship *belong_to* in theER model .

Some rules are needed to produce the terms of the compound concepts. Here, we introduce a language ONTOLOG [8], which can be used for describing compound concepts. The following three samples are illustrated to explain how to use ONTOLOG to express the compound concepts.

Sample 1:

A World Heritage site belongs to the category of Mountain.

Concept: *Site, Mountain*
Relation: *BET (belong_to)*
Concept *Mountain* is attached to the concept *Site*by relation *BET*.

The compound concept is expressed as:
Site [BET: Mountain]

Example 2:

A World Heritage site belongs to the category of Mountain and is located in China.

Concept: *Site, Mountain, China*
Relation: *BET (belong_to), LOC (located_in)*
Concept *Mountain* is attached to the concept *Site*by relation *BET*,
Concept *China* is attached to the concept *Site*by relation *LOC*.

The compound concept is expressed as:
Site [BET: Mountain, LOC: China]

Example 3:

A World Heritage site belongs to the category of *Mountain* and *Artistic work*.

Concept: *Site, Mountain, Artistic Work*
Relation: *BET (belong_to)*
Concept *Mountain* is attached to the concept *Site*by relation *BET*,
Concept *Artistic Work* is also attached to the concept *Site*by relation *BET*.

The compound concept is expressed as:
Site [BET: {Mountain, Artistic Work}]

In the following part of this thesis, we will use the definition as below to express the compound concepts. The definition is based on [1] with some modifications and additions.

Definition: Assume a set of concepts $\boldsymbol{A}$; and a set of extended relations $\boldsymbol{R}$, as indicated with $\boldsymbol{R}$ = {WRT, CHR, CBY, TMP, LOC, BET …}. Then the set of well-formed term $\boldsymbol{L}$ of the ONTOLOG language is recursively defined as follows:

- **Definition 1**: If $x \in \boldsymbol{A}$ then $x \in \boldsymbol{L}$
- **Definition 2**: If $x \in \boldsymbol{L}$, $r_i \in \boldsymbol{R}$ and $y_i \in \boldsymbol{L}$, $i = 1, \ldots, n$
  then $x\ [r_1{:}\ y_1, \ldots, r_n{:}\ y_n] \in \boldsymbol{L}$
- **Definition 3**: If $x \in \boldsymbol{L}$, $r_i \in \boldsymbol{R}$ and $y_{ij} \in \boldsymbol{L}$, $i = 1, \ldots, n, j = 1, \ldots, m$
  then $x\ [r_1{:}\{y_{11}, \ldots, y_{1m}\}, \ldots, r_n{:}\ \{y_{n1}, \ldots, y_{nm}\}] \in \boldsymbol{L}$

Example

The above definition can be applied to describe the World Heritage sites with the features of *Category* and *Geographical Location*. *Amiens Cathedral*, which is represented in Chapter 2 and 3, is used as an example:

Site: *Amiens Cathedral*
Category: *Cathedral*
Location: *France*

The concept set $\boldsymbol{A}$ of the above World Heritage site is:

A = {*Amiens Cathedral*, *Site*, *Cathedral*, *France*}

The relations $\boldsymbol{R}$ can be referred from the ER model of World Heritage in Chapter 3. The relationship between the entity set *Site* and the entity set *Category* is the relation *BET* (*belong_to*). The relation ship between the entity set *Site* and entity set *Country* is the relation *LOC* (*located_in*):

R = {*BET*, *LOC*}

The compound conceptin $\boldsymbol{L}$ is expressed as follows:

Site [BET: *Cathedral*, *LOC*: *France]*

Site [BET: *Cathedral*, *LOC*: *France]* expresses a subclass of the World Heritage sites that belong to the category of *Cathedral* and are located in *France*. *Amiens Cathedral* is an instance of this class.

The compound concept can be understood as a Lower Bound in a Hasse diagram. For example, *Site [LOC: France]* is a compound concept, which is both a site and an object located in France. We use the dashed line to present the extended relation *LOC* in the generative ontology, so *LOC: France* is a concept stands for all kinds of objects that are located in *France*. The concept *Site [LOC: France]* is a Lower Bound of concept *Site*and *LOC: France* as shown in Figure 4-9.

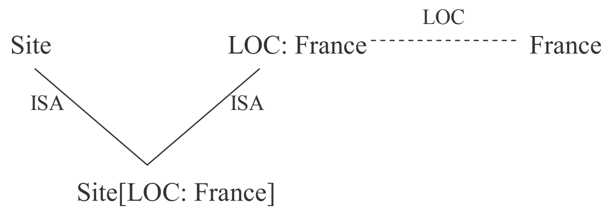


Figure 4-9: Compound Concept Site [LOC: France] as Lower Bound

In following part of this thesis, we omit the objects introduced by the extended relations, for example *LOC: France* in Figure 4-9. Instead, the compound concept *Site [LOC: France]* connects to the concept *Site*by the relation *ISA* and the concept *France* by the relation *LOC* directly as shown in Figure 4-10.

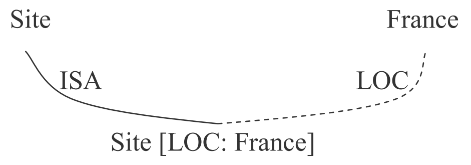


Figure 4-10: Compound Concept Site [LOC: France]

4.2.2 Structure of Generative Ontology

Same as the structure of an ontology, the structure of a generative ontology can be formalised in lattice skeleton with two dimensions – Regions and Levels. First, a generative ontology may be divided into regions corresponding to the groups of concepts. And the concepts in every region are situated into hierarchical levels according to the partial orders among them as we have discussed in Section 4.1.4.

Besides the regions and levels, we use the terms defined in Section 4.2.1 to connect the concepts from different regions and produce compound concepts. The concepts of some regions are attached to concepts of other regions by the dashed lines.

There are three regions in generative ontology of World Heritage: *Sites (the World Heritage sites)*, *Geographical Location*, and *Category*. The extended relations are: *BET (belong_to)*, and *LOC (located_in)*.

The dashed line expresses the concepts from regions *Geographical Location*, and *Category* are attached to the concepts *Site* and produce a compound concept *Site [BET: Cathedral, LOC: France]*.

In this way, *Site [BET*: *Cathedral*, *LOC*: *France]* as a subclass of the World Heritage sites with *Amiens Cathedral* as an instance can be presented as Figure 4-11.

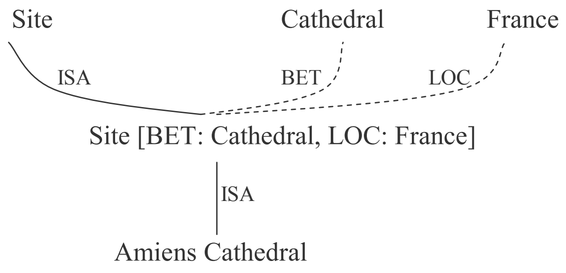


Figure 4-11: An Instance of Compound Concept

4.2.3 Concept Induction in Generative Ontology

A generative ontology is an extension of an ontology, therefore the concepts induction of an ontology as presented in Section 4.1.5 can also be applied to the concepts induction of a generative ontology.

Concept Ordering

Besides the ordering rules in an ontology, a concept becomes more specialised after some other concepts attached to it. For example, the compound concept *Site [BET: Cathedral, LOC: France]* is more specialised than the concept *Site* because it has the limitations from *Category* and *Geographical Location*.

Similarly, the compound concept *Site [BET: Mountain, Artistic Work]* is more specialised than the concept *Site [BET: Mountain]* because it belongs to the category *Mountain* and *Artistic Work.*

We give the definition of concept ordering in a generative ontology as follows. The definition is based on [1] with some modifications and additions.

Definition: Assume a set of concepts $\boldsymbol{A}$; and a set of extended relations $\boldsymbol{R}$, as indicated with $\boldsymbol{R}$ = {WRT, CHR, CBY, TMP, LOC, BET …}, and the binary relation '≤' means 'more specialised than'. Then concept ordering of a generative ontology is recursively defined as follows:

- **Definition 1**: If $x, y \in \boldsymbol{A}$, and there is a path from x to y in which x is strictly below y, then $x \leq y$.
- **Definition 2**: If $x, y, z \in \boldsymbol{A}$, and $x \leq y, y \leq z$, then $x \leq z$.
- **Definition 3**: If $x, y_i \in \boldsymbol{A}$, $r_i \in \boldsymbol{R}$, $i = 1, \ldots, n$, then $x\ [r_1: y_1, \ldots, r_n: y_n] \leq x$.
- **Definition 4**: If $x, y_{ij} \in \boldsymbol{A}$, $r_i \in \boldsymbol{R}$, $i = 1, \ldots, n, j = 1, \ldots, m$ then $x\ [r_1: \{y_{11}, \ldots, y_{1m}\}, \ldots, r_n: \{y_{n1}, \ldots, y_{\mathrm{nm}}\}] \leq x\ [r_1: y_1, \ldots, r_n: y_n\ ]$.

Concept Generalisation and Concept Specialisation

Concept Generlalisation and Concept Specialisation are excuted in each region seperately. Therefore, the concept induction Rule 1 and Rule 2 as defined in 4.1.5 can be also applied to a generative ontology.

4.3 Applying Generative Ontology to Application Domain

Besides ER diagram, the ontology and generative ontology can also be used to model the application domain. We will give a discussion how to use the generative ontology to model the application.

We can use the ER model to develop the generative ontology. The generative ontology is built on top of the ER model. The entity sets, entities of entity sets and relationships from the ER model can be shown in the generative ontology.

More precisely, as shown in figure 4-12, the generative ontology forms a specialization hierarchy of concepts.

The general level concepts in the generative ontology are analogous to the entity sets in the ER model; the specialised level concepts are the entities of entity sets which are kept in the database schemas. The medium level concepts are the entities of the entity sets or some newly introduced concepts to create the subclasses between the general level concepts and specialised level concepts.

The relationships from ER diagram can be interpreted as the extended relations in the generative ontology.

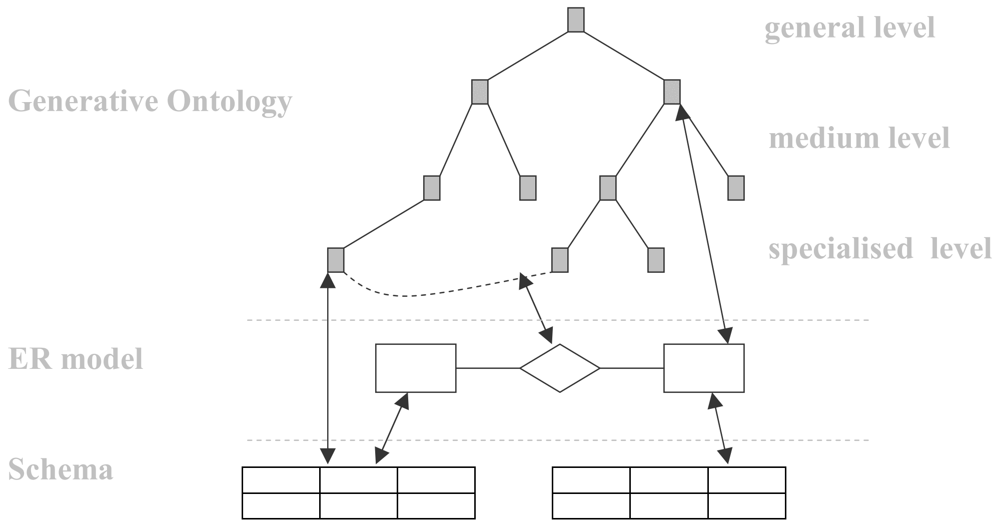


Figure 4-12: Connection between Generative Ontology and ER model

4.4 Conclusion

This chapter gives a description of the ontology and the generative ontology. Inspired by the previous theories and research concerning the topic of ontology and generative ontology, we apply and extend the theories and methods for the design of a query induction system.

Fragments of the generative ontology of World Heritage are presented as the examples in this chapter, but they are only fragments, not the complete generative ontology of World Heritage. In the next chapter, we will apply the ideas and methods which are discussed in this chapter to model the World Heritage domain.

5 Generative Ontology of World Heritage

In the previous chapter, we discussed the theories and methods that can be used to build up the generative ontology of an application domain. In this chapter, we will apply these theories and methods to build the generative ontology of World Heritage.

This chapter includes a lot of information of World Heritage. We will first give an overview so that the reader can understand the generative ontology of World Heritage quickly. More detailed information is presented in Section 5.2. If the readers are not interested in World Heritage, this part could be skipped.

5.1 Overview of the Generative Ontology of World Heritage

A generative ontology may be divided into several regions corresponding to the groups of concepts. There are three regions of the generative ontology of World Heritage: *Site*, *Category* , and *Geographical Location*.

The concepts in each region of a generative ontology are arranged into hierarchical levels. As we have described in Chapter 4, the general level concepts are analogous to the entity sets in the ER model; the specialised level concepts are the entities kept in the database schemas. The medium level concepts are the entities or some newly introduced concepts as subclasses between the general level concepts and specialised level concepts.

The above description is only a general sketch. Actually, it is very hard to define the clear boundaries among the general level, medium level and specialised level. But generally speaking, the concepts in a Hasse diagram become more and more specialised when moving downwards the Hasse diagram.

In the following sections, we will give a description of the concepts in the regions *Site*, *Category* and *Geographical Location* and how the concepts are situated into the hierarchical levels in each individual region.

5.1.1 Overview of the Region Site

There is an entity set *Site* in the ER model of World Heritage, so concept *Site* is the general level concept in the region *Site*. The World Heritage sites are defined as entities in the ER model. These entities make up the specialised level concepts in the region *Site*. We haven't added any medium level concepts in this region because the ISA relation expresses the individual membership, which is the partial order in the Hasse diagram.

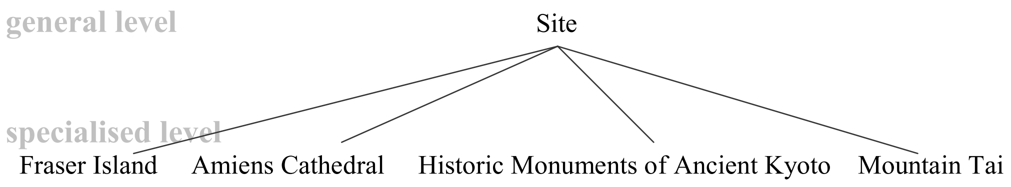


Figure 5-1: Hasse Diagram of Site

5.1.2 Overview of the Region Category

There is an entity set *Category* in the ER model of World Heritage, so the concept *Category* is a general level concept. Concept *Category* can be classified into *Natural Category* and *Cultural Category*. Although *Natural Category* and *Cultural Category* are not entity sets in the ER model, they are considered as the general level concepts in the Hasse diagram of region *Category*, because they cover a lot of sites. The general level concepts in the region *Category* are shown in Figure 5-2.

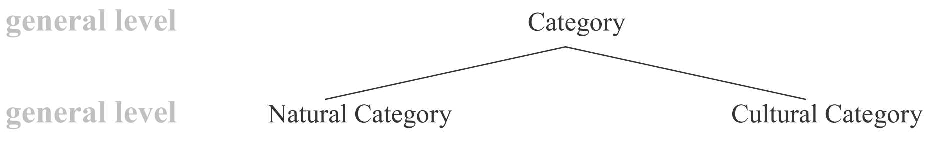


Figure 5-2: Hasse Diagram of Category – General Level Concepts

The partial order in Figure 5-2 is ISA, expressing the class inclusion.

More categories are introduced as the medium level concepts in region *Category*. For the need of the following development of the query induction system, concepts in several levels make up the medium level concepts.

For examples, *Category* and *Cultural Category* are general level concepts, and *Historic Capital* is a specialised level concept, *Historic City and Area* and *Politics Historic Centre* are both medium level concept, which are shown in Figure 5-3. The partial order in Figure 5-3 is ISA, expressing the class inclusion.

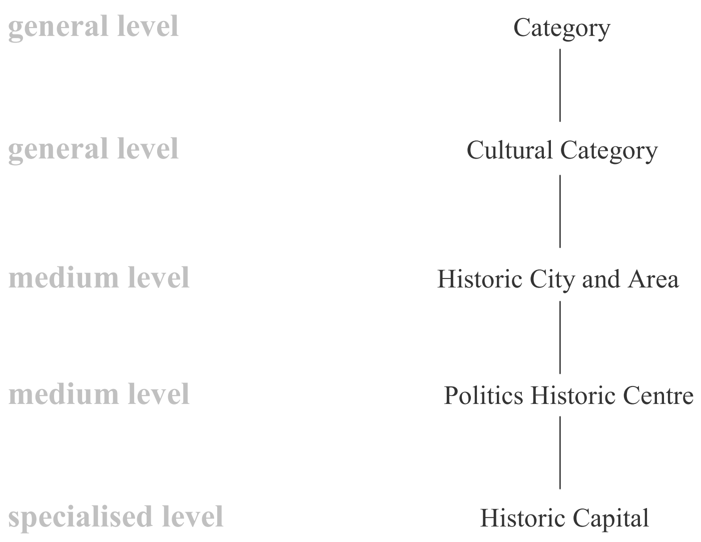


Figure 5-3: Levels in the Hasse Diagram

In the following part of this section, we will give a brief explanation of the medium level concepts in the region *Category*.

Concept *Natural Category* can be further classified. We go through the descriptions and criteria of all the sites that belong to the category *Natural Category*, and then classify them into different categories according to our knowledge of the nature science. These different categories are arranged in the medium level and specialised level according to how specialised the concepts are.

An overview of the medium level concepts under *Natural Category* is given in Figure 5-4. The partial order in Figure 5-4 is ISA, expressing the class inclusion. More details are supplemented in Section 5.2.1.

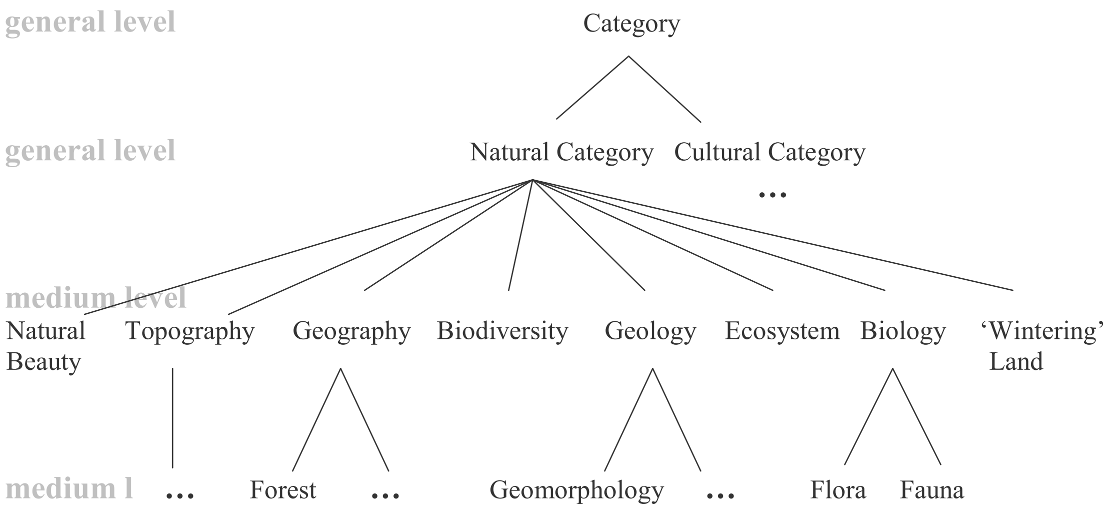


Figure 5-4: Hasse Diagram of Natural Category – Medium Level Concepts

An overview of the medium level concepts under *Cultural Category* is given in Figure 5-5.

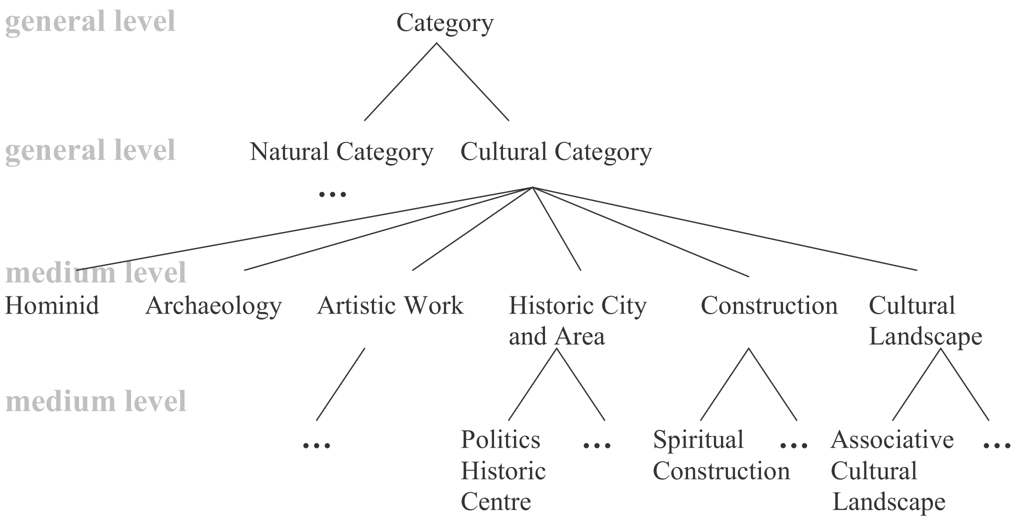


Figure 5-5: Hasse Diagram of Cultural Category – Medium Level Concepts

The partial order in Figure 5-5 is ISA, expressing the class inclusion.

As we mentioned in Chapter 3, the entities in the entity set *Category* are the specific categories that can describe the World Heritage sites as explicitly as possible. These categories are the specialised level concepts in the region *Category*, such as *Mount*, *Cathedral*, and *Historic Capital*, so these specific categories are under the medium level concepts. There are many specialised level concepts in the region *Category*, and we will situate all of them under the corresponding medium level concepts in Section 5.2.

5.1.3 Overview of the Region Geographical Location

Geographical Location is a concept including the entity sets *Continent*, *Composition of Continent* and *Country* in the ER model, so *Geographical Location* is a general level concept in the region.

The entities of the entity set *Country* can describe the locations of the World Heritage sites explicitly. Therefore, the entities of the entity set *Country* are the specialised level concepts in the region *Geographical Location*. The entities of entity sets *Continent* and *Composition of continent* are the medium level concepts. The Hasse diagram of region *Geographical Location* is shown in Figure 5-6. The partial order in Figure 5-6 is *LOC (located_in)* and ISA.

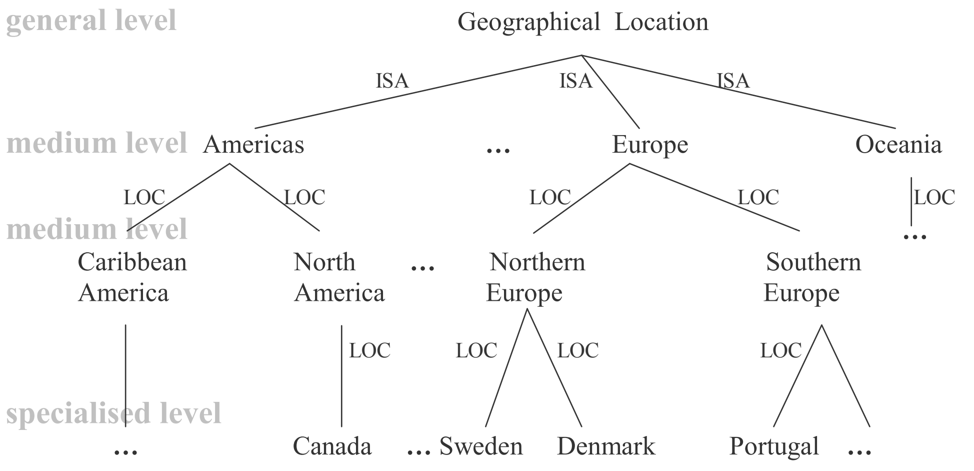


Figure 5-6: Hasse Diagram of Geographical Location

All the concepts in region *Geographical Location* are situated in a Hasse diagram structure. There are 230 countries, 21 compositions of continents and 5 continents in this region. We can not illustrate all of them. Figure 5-6 only expresses some sample concepts. However it is easy for the readers to understand the relations of them based on the common geographical knowledge.

The entities of entity sets *Country, Composition of Continent*, and *Continent* are the concepts in this region, but the entity sets are not the concepts. From the view of a generative ontology, it is not necessary to include these three entity sets in the ER model of World Heritage, and it is not necessary to keep their entities in the database.

Actually, we experiment with two ways of storing the Hasse diagrams. The concepts and relations in the Hasse diagram *Category* will be kept in the Java files and the concepts and the relations of Hasse diagram *Geographical Location* will be kept in the relational tables. Therefore, we create these 3 entity sets and keep the concepts as entities in the database.

It is convenient to keep the Hasse diagram of *Geographical Location* in the relational tables. There are many concepts in this Hasse diagram. Furthermore the relationships connecting these three entity sets are both *located_in* with the multiplicity *one or many* to *one*, which is very easy to implement by the relational database. The connection between the ER model and theHasse diagram is shown in Figure 5-7.

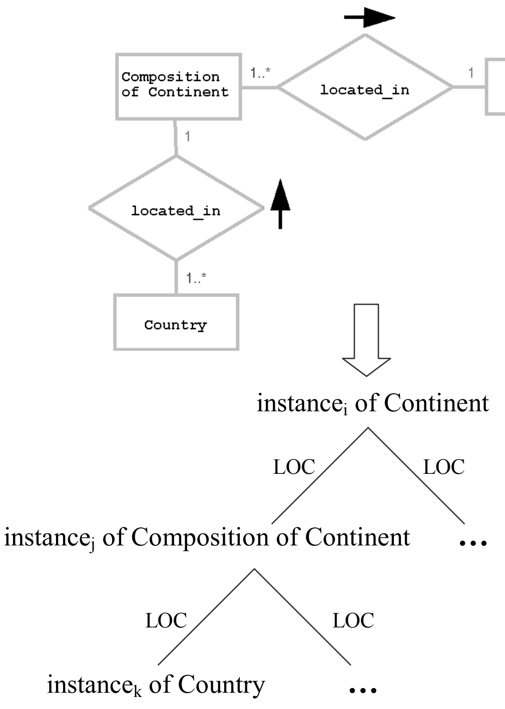


Figure 5-7: Connection between ER Model and Hasse Diagram

5.1.4 Structure of the Generative Ontology of World Heritage

The structure of the generative ontology of World Heritage will be given in this section. After we have explained the concepts in each region of the generative ontology of World Heritage separately, we will explain how to connect the concepts among the regions by the extended relations in the generative ontology.

A World Heritage site with the information of *Category* and *Geographical location* is expressed as a compound concept in the generative ontology. The extended relations in the generative ontology of World Heritage are *BET (belong_to)* and *LOC (located_in)*. The concepts in the region *Category* are attached to the concepts in the region *Site*by the relation *BET*. And the concepts in the region *Geographical Location* are attached to the concepts in the region *Site* by the relation *LOC*. Therefore, we know that the extended relations in the generative ontology of World Heritage connect the concepts from three different regions.

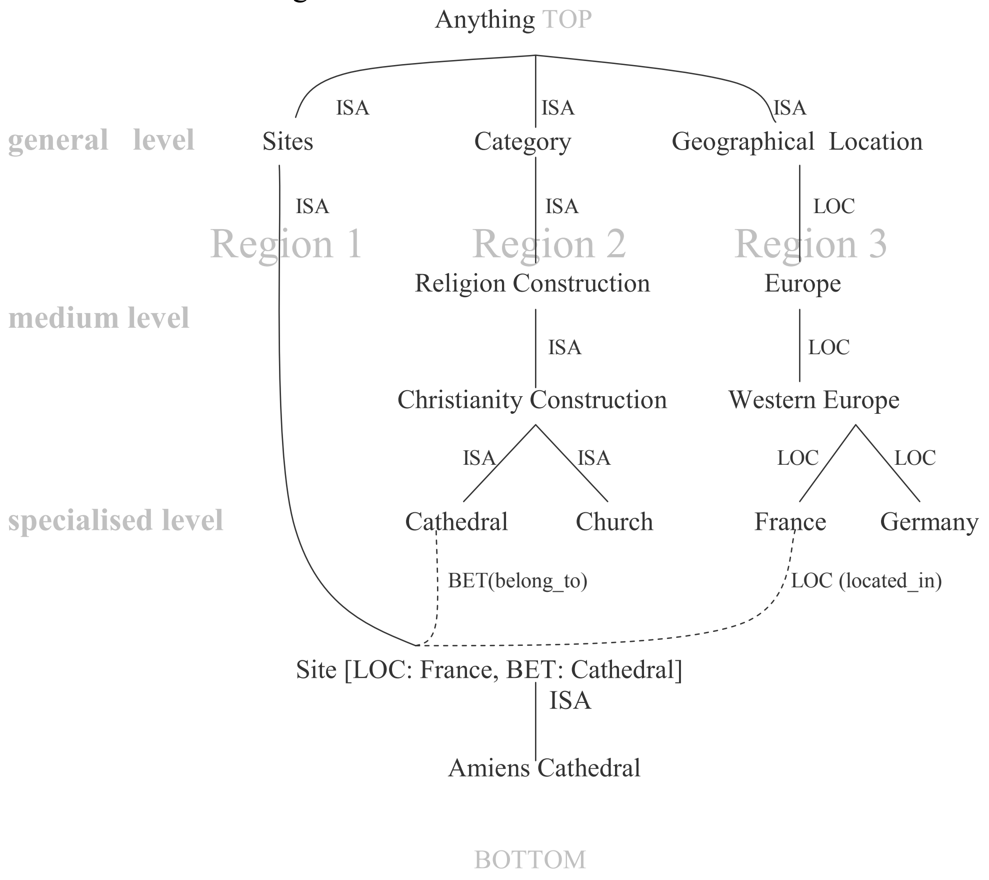


Figure 5-8: The Generative Ontology of World Heritage

Amiens Cathedral is situated under the compound concept *Site [BET: Cathedral*, *LOC: France]* as an instance in Figure 5-8.

5.2 Detailed Information of the Region Category

The previous section presented an overview of the generative ontology of World Heritage. The concepts in the three regions and the extended relations were explained.

In this section, we will give the detailed information of the region *Category*. It is an important part of this thesis to build up a Hasse diagram for the region *Category*. In order to classify the World Heritage sites into specific categories, about 130 concepts are introduced. All the concepts in the region *Category* are situated in the Hasse diagram structure with ISA as the partial order.

In this section, we will present all the medium level and specialised level concepts in the region *Category*.

5.2.1 Natural Category

The *Natural Category* is first classified into 8 sub-categories. These 8 concepts are the medium level concepts. Their definitions are as follows:

- ***Natural beauty***: The heritage sites that belong to this category have some outstanding universal values of natural beauties, for example an aesthetic importance or a spectacular natural view.

- ***Topography*** The heritage sites that belong to this category have special features of a place or district, especially the position of its rivers, mountains, etc.

- ***Geography***: The heritage sites that belong to this category relate to the science of the earth's surface. We classify this concept into many more specialised concepts, for example *Forest*. Many natural heritage sites belong to this category; especially there are many sites that belong to the category *Forest*

- ***Biodiversity***: The heritage sites that belong to this category contain the important and significant natural habitats with biological diversity.

- ***Geology***: The heritage sites that belong to this category could be outstanding examples representing major stages of the earth's history, including the record of life, significant on-going geological processes in the development of landforms, or significant geomorphic or physiographic features. There are many natural heritage sites belong to this category.

- ***Ecosystem***: The heritage sites that belong to this category could be outstanding examples representing significant on-going evolution, involving all the plants and living creatures in a particular area together with the physical environment.

- ***'Wintering' Land***: The heritage sites that belong to this category are inscribed as World Heritage because they are the areas where the migrating birds spend the winter.

- ***Biology***: The heritage sites that belong to this category are about the science that is concerned with the growth, development, and functioning of living things. Many natural sites belong to this category, which are for the purpose of protecting the endangered or threaten species or some kind of special species.

Under these 8 medium level concepts, there are more medium level and specialised level concepts, which describe the natural sites clearly. When we define the categories for every site, we try to choose the categories as specialised as possible to describe the site explicitly.

Here is a sample natural site:

> Dorset and East Devon Coast - United Kingdom
>
> Inscribed Year: 2001 Criteria: Natural (i)
>
> Brief Description: The cliff exposures along the Dorset and East Devon coast provide an almost continuous sequence of rock formations spanning the Mesozoic Era, or some 185 million years of the earth's history. The area's important fossil sites and classic coastal geomorphologic features have contributed to the study of earth sciences for over 300 years.

We choose two categories *Cliff* and *Geomorphology* to describe the above site, because they can describe the site more explicitly than the category *Geology*.

*Geomorpholog*y is a branch of *Geology*, which studies the characteristics, configuration and evolution of rocks and landforms. Therefore, it is a more specialised concept than *Geology*.

The complete Hasse diagram of *Natural Heritage* is as follows:

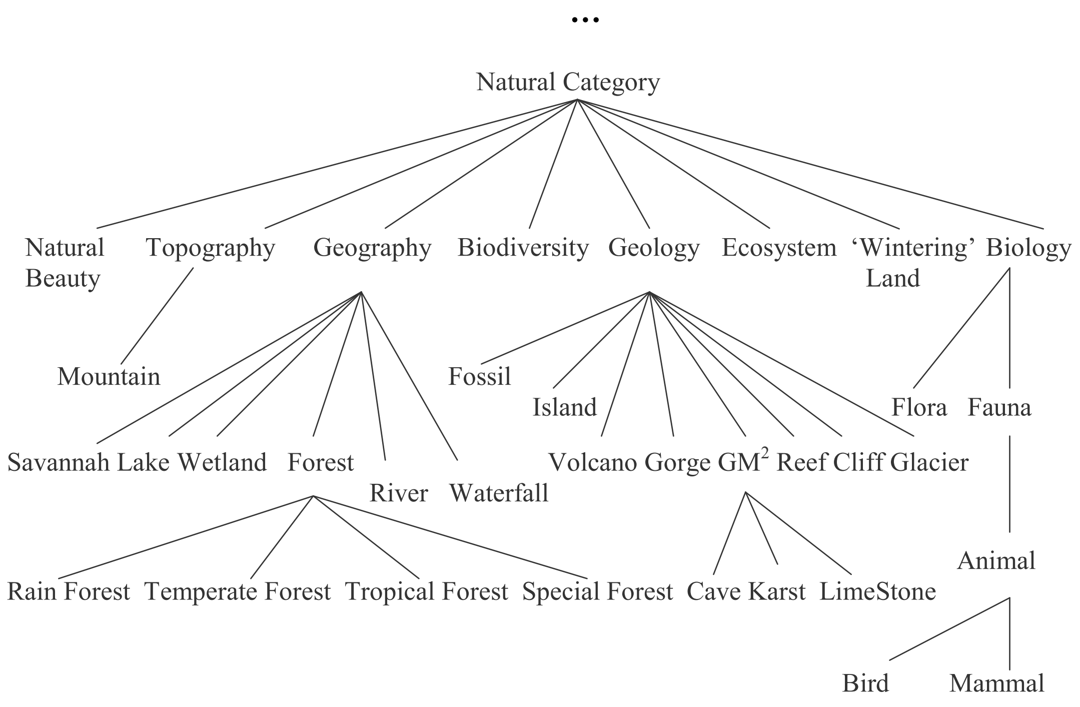


Figure 5-9: Hasse Diagram of Natural Category

5.2.2 Cultural Category

The *Cultural Category* is first classified into 6 sub-categories. These 6 concepts are medium level concepts. Their definitions are as follows:

- ***Hominid***: The heritage sites that belong to this category have fossil evidence of human origins collected from different habitats and samples from many countries.

- ***Archaeological***: The heritage sites that belong to this category have evidences, which can contribute to the study of ancient cultures, people and periods of history by scientific analysis of physical remains, especially those found in the ground.

- ***Artistic Work***: The heritage sites that belong to this category are works of monumental sculpture, painting or inscriptions which are of outstanding universal value from a historical or artistic view.

[2] Geomorphology

- ***Historic City and Area***: The heritage sites that belong to this category are some cities and areas, which represent the living presence of the past. These sites have the most tangible evidences of the wealth and diversities of cultural, religious and social activities.

- ***Construction***: The heritage sites that belong to this category are outstanding examples of a type of building, architectural ensemble or technological ensemble, such as temples, castles and railways.

- ***Cultural landscape***: The heritage sites that belong to this category are representative of the different regions of the world. Combined works of nature and humankind, they express long and intimate relationships between people and their natural environment, such as cultivated terraces on lofty mountains, gardens, and sacred places.

…

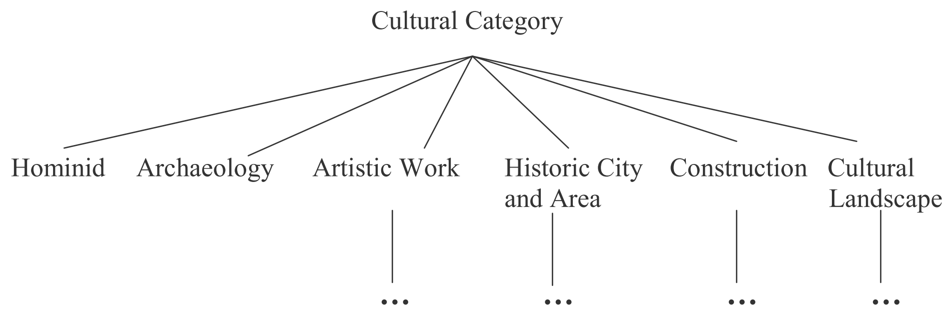


Figure 5 10: Hasse Diagram of Cultural Category

Under these 6 medium level concepts, there are more medium level and specialised level concepts. In the following part of this section, we will give an explanation of these concepts.

Because *Hominid* and *Archaeology* only cover two small groups of culture sites, no more specialised concepts are introduced under them.

There are more concepts under *Artistic Work*. Besides *Caving*, *Painting*, and *Mosaic*, *Rock Art* is an important category because over 99% of known prehistoric art is rock art, which recorded thousands of years of artistic creation and culture development.

The concepts under *Artistic Work* are shown in Figure 5-11.

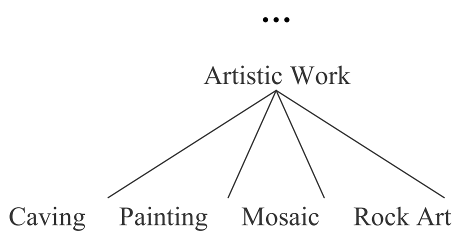


Figure 5-11: Hasse Diagram of Artistic Work

Historic City and Area, *Construction*, and *Cultural landscape* are three large groups of cultural sites. Therefore more concepts are introduced under them. Because of the limitation of the space, they can not be displayed in Figure 5-10. In the following part of this section, we will describe the concepts under the concepts *Historic City and Area*, *Construction*, and *Cultural landscape*. The Hasse diagrams in the following part of the section should be considered as part of the Hasse diagram in Figure 5-10.

Historic City and Area

Category *Historic City and Area* covers more than half of all the cultural sites. These sites focus on the outstanding roles in history, together with their significant constructions inside the area. They record the original appearances of the settlement of human beings and town planning.

Because *Historic City and Area* covers many World Heritage sites, it can not describe the sites explicitly. More specialised concepts are needed to describe the site more explicitly. We use the purpose of the sites to classify the site of category *Historic City and Area. Why did these sites develop to cities and areas with historic importance?* By answering this question, we can get the purpose of the sites. The possible purposes are: *Culture*, *Economy*, *Spirit*, *Military*, *Politics*, *and Residence*.

Therefore concepts *Culture Historic Centre*, *Economy Historic Centre*, *Spiritual Historic Centre*, *Military Historic Centre*, *Politics Historic Centre*, and *Residence Historic Centre* are introduced. Under each of them, more specialised concepts are situated according to the ISA relation. Here is a sample of *Historic City and Area* sites:

Island of Mozambique - Mozambique

Inscribed Year: 1991 Criteria: Cultural (iv) (vi)

Brief Description: The fortified city of Mozambique is located on this island, a former Portuguese trading-post on the route to India. Its remarkable

> architectural unity is due to the consistent use, since the 16th century, of the same building techniques, building materials and decorative principles.

The explicit categories of the above site can be: *Architecture Exhibitions*, *Fortified Town*, *Colony Settlement*, and *Trading Centre*, because this site has an architectural unity and it was a Portuguese colony for trading purpose before. These categories can describe this site more explicitly than *Historic City and Area*.

The complete Hasse diagram of *Historic City and Area* is divided into 4 smaller parts, presented from Figure 5-12 to Figure 5-15.

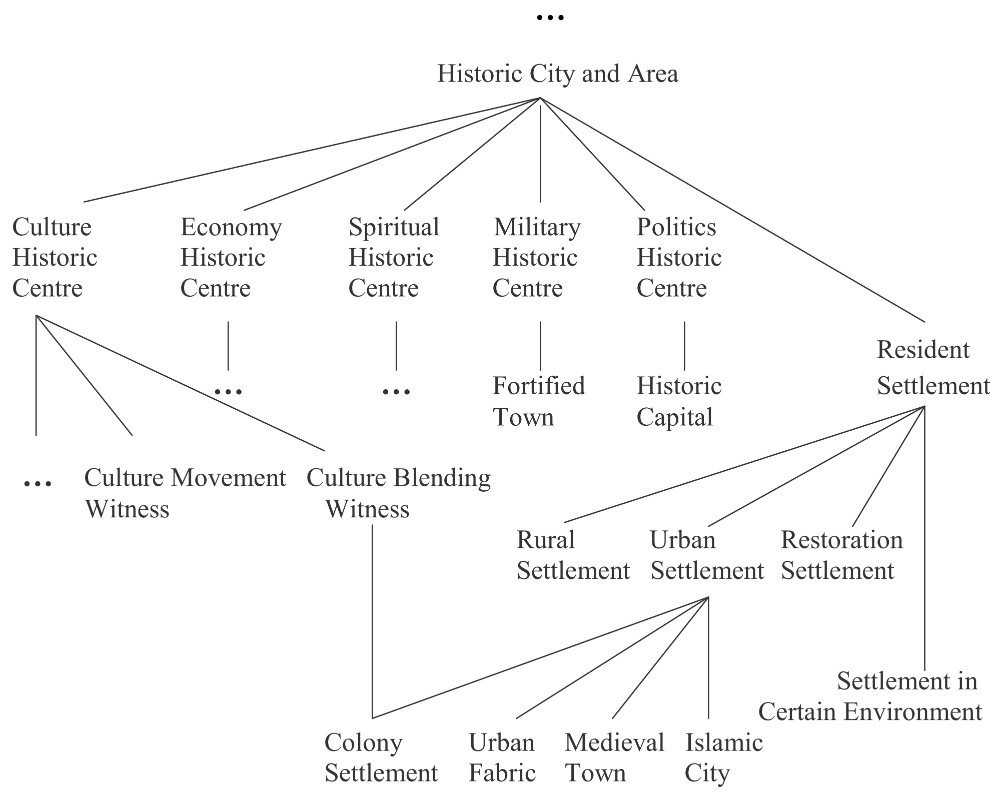


Figure 5-12: Hasse Diagram of Historic City and Area

The following figure should be placed under *Culture Historic Centre* in Figure 5-12.

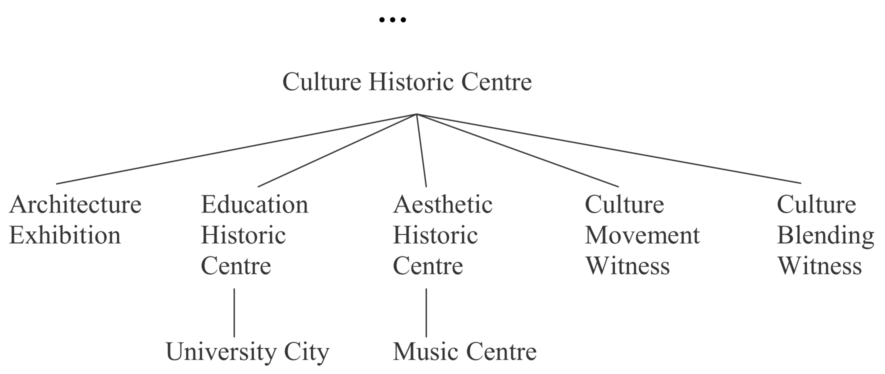


Figure 5-13: Hasse Diagram of Culture Historic Centre

The sites of category *Culture Historic Centre* may have special importance on the aspects of architecture, education, aesthetics, and so on.

For example, the site *Historic Centre of Vienna* in *Austria* has been universally acknowledged as the musical capital of Europe since the 16th century, so the site *Historic Centre of Vienna* belongs to the category *Music Centre*.

The category *Culture Movement Witness* covers the sites which have experienced the movement of different cultures during the history. For example, some towns in Europe have experienced the three key periods of European cultural and political development - the Middle Ages, the Baroque period, and the Gründerzeit.

Colony Settlement is a kind of *Culture Blending Witness*, because these sites witnessed the mixture of the local cultures and the cultures from the ruling countries. At the same time, *Colony Settlement* is also a kind of *Urban Settlement*. Therefore, the concept *Colony Settlement* is placed under both *Culture Blending Witness* and *Urban Settlement* in Figure 5-12.

The following figure should be placed under the concept *Economy Historic Centre* in Figure 5-12.

The sites of category *Economy Historic Centre* developed in history because of economical reasons. The economical reasons could be *Agriculture*, *Industry* or *Commerce*. The sites that belong to *Industry Historic Centre* make up a big group. Therefore, there are more concepts introduced under *Industry Historic Centre*.

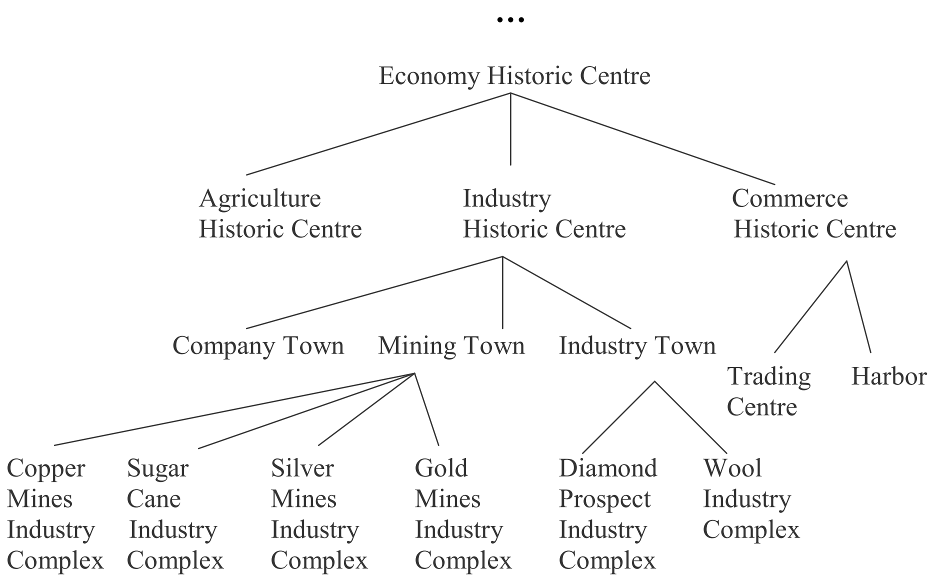

...
Economy Historic Centre
Agriculture Historic Centre
Industry Historic Centre
Commerce Historic Centre
Company Town
Mining Town
Industry Town
Trading Centre
Harbor
Copper Mines Industry Complex
Sugar Cane Industry Complex
Silver Mines Industry Complex
Gold Mines Industry Complex
Diamond Prospect Industry Complex
Wool Industry Complex

Figure 5-14: Hasse Diagram of Economy Historic Centre

The following figure is under *Spiritual Historic Centre* in Figure 5-12.

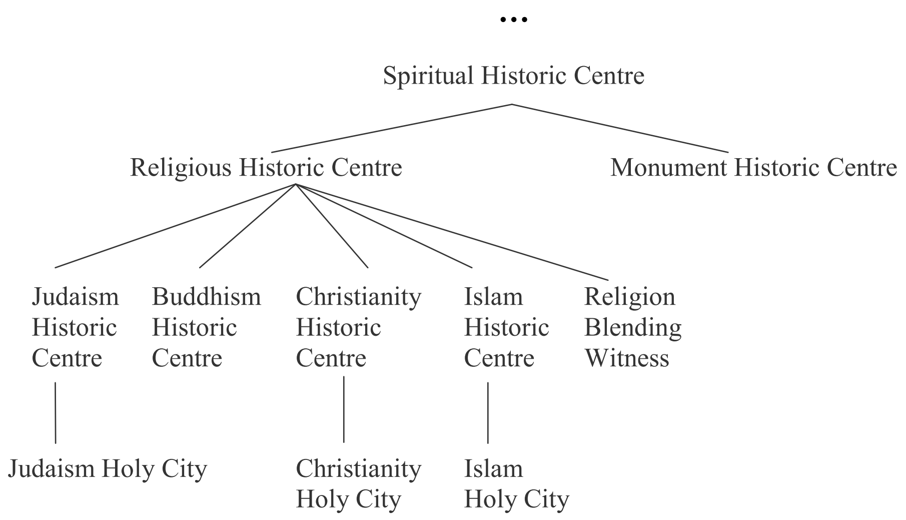


Figure 5-15: Hasse Diagram of Spiritual Historic Centre

Most of the sites of the category *Spiritual Historic Centre* have special importance concerning religion, such as *Buddhism*, *Christianity*, and *Islam*.

Construction

Many cultural sites belong to the category *Construction*. It could represent a masterpiece of human creative genius with outstanding historic values or architectural styles and technologies.

Similar with *Historic City and Area*, the purposes of the constructions can be used to sub-classify the category *Construction*. *What is the reason of building this construction? What is the usage of the construction?* For example, there are many cultural sites built for religious reasons, such as *Churches*, *Mosques*, and *Temples*.

Therefore we get the concepts *Culture Construction*, *Economy Construction*, *Spiritual Construction*, *Military Construction*, *Politics Construction*, and *Residence Construction*.

Here is a sample of a *Construction* site:

> La Lonja de la Seda of Valencia - Spain
>
> Inscribed Year: 1996 Criteria: Cultural (i) (iv)
>
> Brief Description: Built between 1482 and 1533, this group of buildings was originally used for trading silk (hence its name, the Silk Exchange) and it has always been a centre for commerce. It is a masterpiece of late Gothic architecture.

From the above description, we can see that the specialised category of this sample is *Commerce Construction*, because it was used for silk trading.

The complete Hasse diagram of *Construction* is divided into 4 smaller parts, presented in Figure 5-16 to Figure 5-19. The following figure should be placed under the concept *Construction* in Figure 5-10.

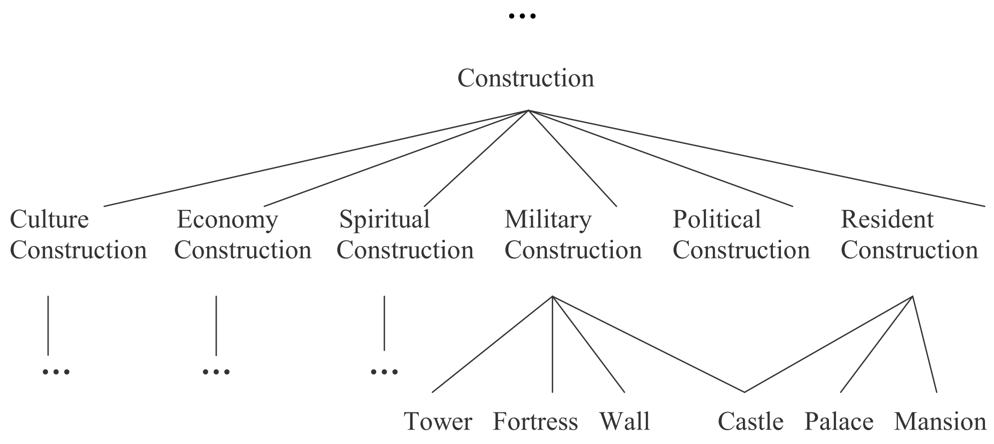


Figure 5-16: Hasse Diagram of Construction

The following figure should be placed under *Culture Construction* in Figure 5-16.

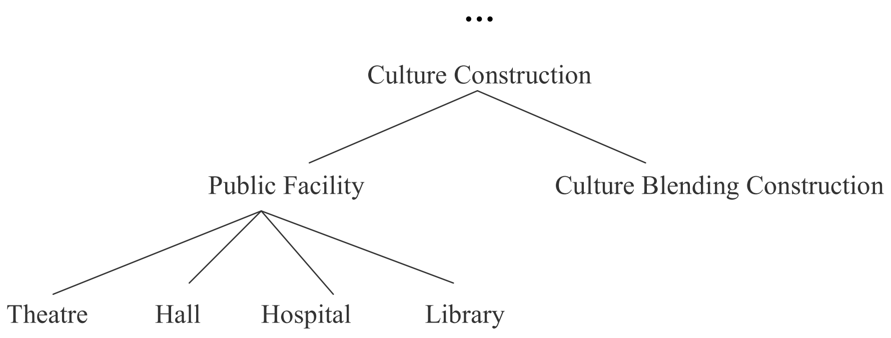


Figure 5-17: Hasse Diagram of Culture Construction

Culture Construction includes *Public Facility* and *Culture Blending Construction*. Some constructions mixed the architecture styles from different cultures. These sites belong to the category *Culture Blending Construction*.

The following figure should be placed under *Economy Construction* in Figure 5-16.

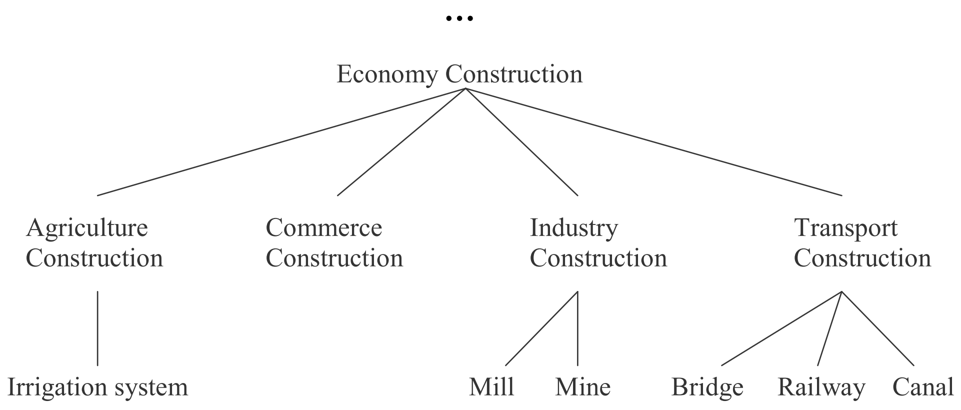


Figure 5-18: Hasse Diagram of Economy Construction

Some constructions are built for the purpose of *Economy*. They are classified into 4 groups: *Agriculture Construction*, *Industry Construction*, *Commerce Construction* and *Transport Construction*.

The following figure should be placed under *Spiritual Construction* in Figure 5-16.

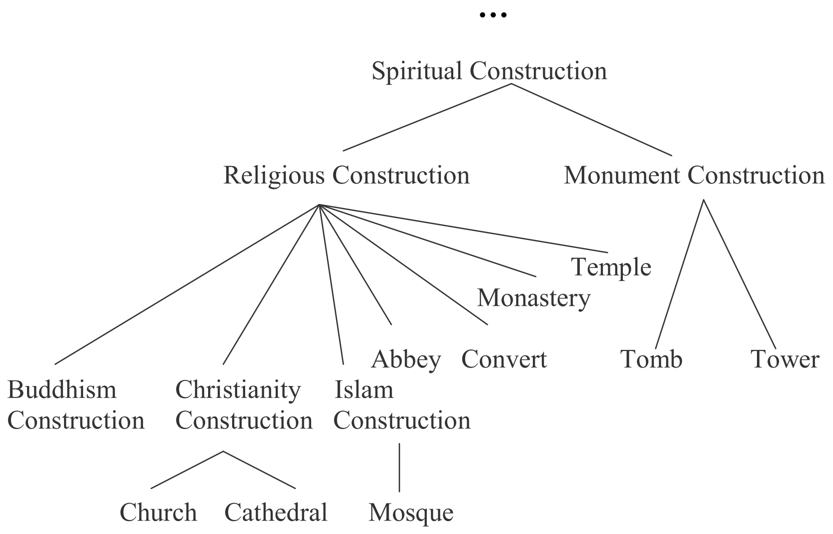


Figure 5-19: Hasse Diagram of Spiritual Construction

Many sites of category *Spiritual Construction* are religious buildings. Especially in Europe, churches and cathedral make up a big group of the World Heritage sites.

Monument Construction includes *Tomb* and *Tower* For example, the famous site, *Pyramid Fields from Giza to Dahshur,* is a group of tombs for ancient kings in Egypt. There are also some cultural sites belonging to the category *Tower*, which were built in commemoration of an event, achievement, or person.

Cultural Landscape

Referring to the classification from the World Heritage Committee; we sub-classify the category of *Cultural landscape* as follows.

- ***Clearly defined landscape***: The heritage sites belonging to this category include gardens, and parklands constructed for aesthetic reasons, which are often associated with religious or other monumental buildings.

- ***Organically evolved landscape***: The heritage sites belonging to this category result from an initial social, economic, administrative, and/or religious imperative. These sites have developed with the association of their natural environment.

- ***Associative cultural landscape***: The inclusion of such landscapes on the World Heritage List is justifiable by virtue of the religious, artistic or cultural associations of the natural elements rather than cultural elements.

There are some specialised level concepts under these 3 medium level concepts, which can describe the *Cultural Landscape* sites clearly.

Here is a sample of a *Cultural Landscape* site:

> Rice Terraces of the Philippine Cordilleras - Philippines
>
> Inscribed Year: 1995 Criteria: Cultural (iii) (iv) (v)
>
> Brief Description: For 2,000 years, the highrice fields of the Ifugao have followed the contours of the mountains. The fruit of knowledge handed down from one generation to the next, and the expression of sacred traditions and a delicate social balance, they have helped to create a landscape of great beauty that expresses the harmony between humankind and the environment.

The category of this sample site is *Agricultural Landscape*, which is a sub-category of *Associative Cultural Landscape*.

The following figure presents all the concepts under *Cultural Landscape*. It should be placed under the concept *Cultural Landscape* in Figure 5-10.

…

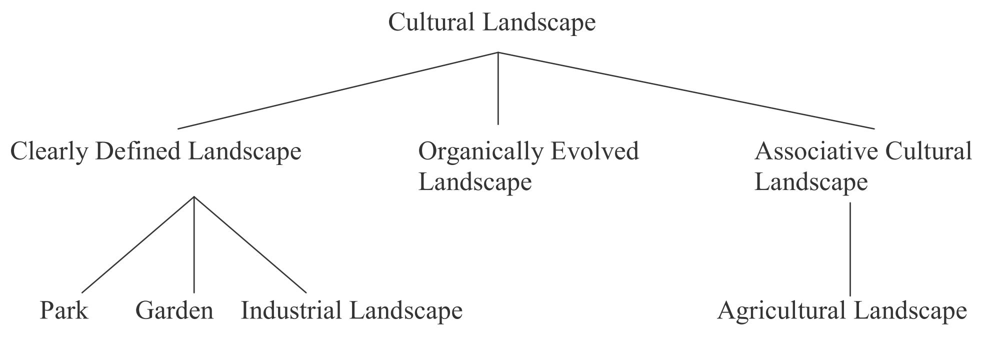

Cultural Landscape
Clearly Defined Landscape
Organically Evolved Landscape
Associative Cultural Landscape
Park
Garden
Industrial Landscape
Agricultural Landscape

Figure 5-20: Hasse Diagram of Cultural Landscape

It is necessary to be aware that there are no clear boundaries between the medium level and specialised level concepts. Although we always try to choose the more specialised concepts as the categories of the sites to describe them explicitly, some times we have to use medium level concepts as the categories of them because the information of the sites are limited. Here is an example:

> Shrines and Temples of Nikko - Japan
>
> Inscribed Year: 1999 Criteria: Natural (i) (iv) (vi)
>
> Brief Description: The shrines and temples of Nikko, together with their natural surroundings, have for centuries been a sacred site known for its architectural and decorative masterpieces. They are closely associated with the history of the Tokugawa Shoguns.

From the description of above site, we can classify it under the category *Religious Historic Centre*, but it is hard for us to find out which religion it is. The reason is that we are not familiar with Japanese culture and we can not get the sufficient information form the description. Therefore, we can not find the more specialised category for this site.

5.3 Conclusion

This chapter presented the generative ontology of World Heritage built by us, which will be used for concept induction in the following development of the query induction system.

We analyze and situate the concepts in the region *Category* according to information of the World Heritage sites and our knowledge of the natural science, history, religion and so on, but we are not experts in these fields. Furthermore, the information of the sites is not sufficient. Our references are the sites' criteria and descriptions that only have about 100 words for each site. Therefore, there may be some inadequacies in this generative ontology.

Regardless, this generative ontology can cover all the sites. It is also built from a view without expert knowledge, which may be easy for common users to understand. Furthermore, this generative ontology can be applied to the query induction system because the concepts of each region are organized by the partial orders.

6. Design of Query Induction

This chapter presents the design of query induction. First, we will present an overview of the approach of query induction. Then we describe the approach in steps to give a detailed explanation.

Afterwards the methods that can improve the performance of query induction will be discussed. These methods are: Concept Generalisation by Count; Similar Interests, Concept Induction by Recent Examples; and Concept Generalisation by Weight. These methods are our suggestions because we did not find any publication on the similar topics.

The rules of concept induction in a generative ontology described in Chapter 4 are the foundations for the design of the query induction.

6.1 Approach of Query Induction

6.1.1 Overview

When a user browses the web, the user will select the information that he/she feels interested in. To a query induction, the selected information is an example from the user. Because most of the dynamic websites keep the web information in a database, the examples selected by the user from the web are some entities from the database, which are under certain concepts in a generative ontology. Therefore, the example from the user can be situated under these concepts.

A query induction can adopt the rules of concept induction that have been discussed in Section 4.1.5 to execute Concept Generalisation and Concept Specialisation.

The query induction takes the concepts which cover the example as inputs to execute Concept Generalisation. The result of Concept Generalisation can reveal the user's interest, so it can be specialised to get more concepts which are within the user's interest. Because Concept Generalisation and Concept Specialisation are executed in each region separately, the result of Concept Specialisation from each region can be attached together to form compound concepts.

The induced compound concepts can be used as queries to retrieve all the other entities from the database. These entities are under the induced compound concept but not visited by the user. They are suggested to the user because they are close to the examples from the user.

The process of the query induction can be illustrated by the data flow as Figure 6-1. We use the dashed arrow to indicate the roles of the generative ontology and the database in the process of the query induction.

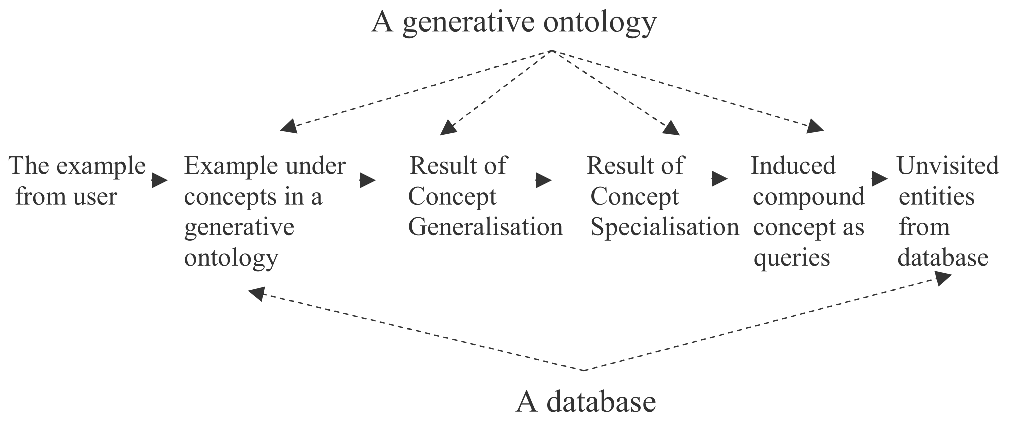

A generative ontology
The example from user
Example under concepts in a generative ontology
Result of Concept Generalisation
Result of Concept Specialisation
Induced compound concept as queries
Unvisited entities from database
A database

Figure 6-1: Data Flow of Query Induction

6.1.2 Steps of Query Induction

After we gave the overview of the query induction approach, we will give a detailed discussion of the process of the query induction. The steps of the query induction are listed as follows:

Step 1: The query induction finds the relative information of the example from the user, which are some entities from the database. According to these entities, the query induction can find the concepts that cover the example from the user in different regions of the generative ontology.

Step 2: Concept Generalisation of these concepts that cover the example from the user is executed inside each region separately according to concept induction Rule 1.

Step 3: The query induction takes the result of Concept Generalisation from Step 2 as the inputs to execute Concept Specialisation according to the concept induction Rule 2. Concept Specialisation is executed inside each region separately.

Step 4: The results of Concept Specialisation from different regions are attached together to form the induced compound concepts, which are used as queries to retrieve all the entities that the user has not visited from the database.

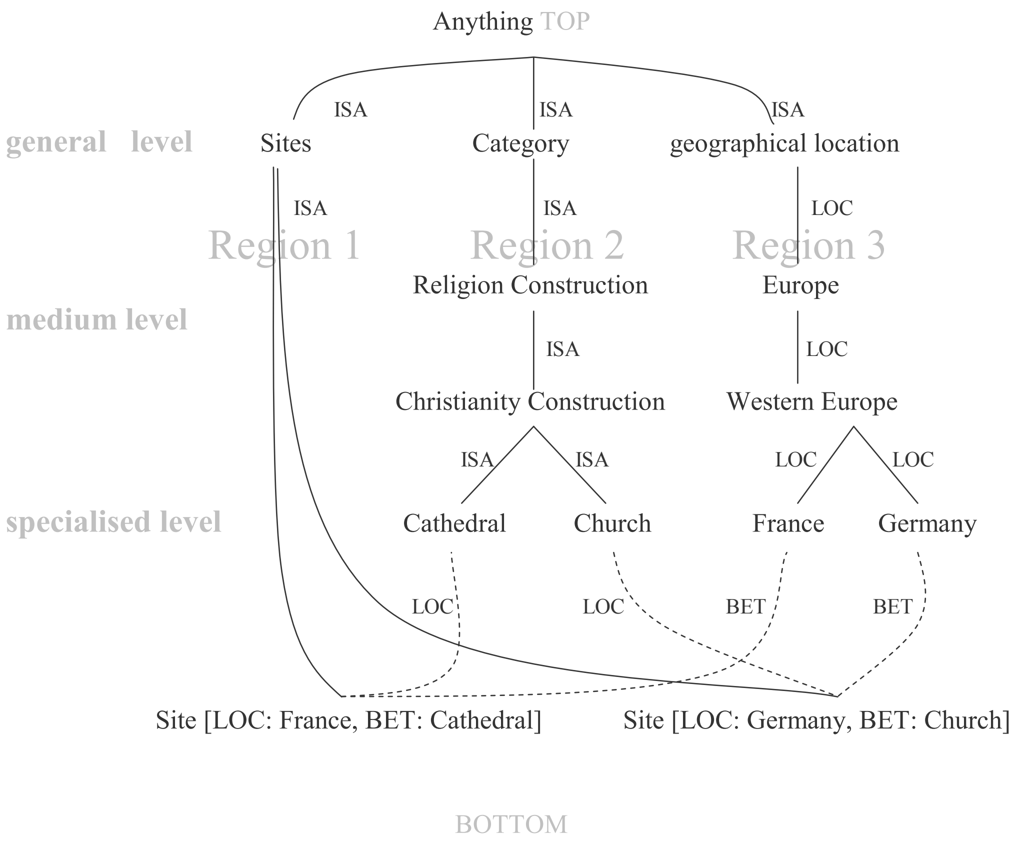


Figure 6-2: Steps of Query Induction – an Example

Here we use two sample sites and a part of the generative ontology of the World Heritage domain to illustrate the steps of a query induction. Figure 6-2 shows a part of the generative ontology. We assume *Amiens Cathedral* and *Pilgrimage Church of Wies* are two examples from the user.

Step 1 of the process of query induction

The query induction needs to search the database for the categories and countries of these two sites. Part of the ER diagram is shown in Figure 6-3.

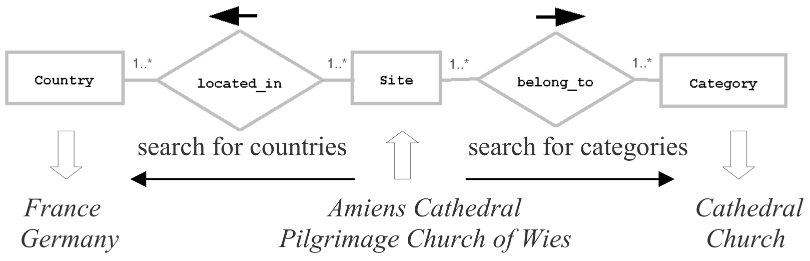


Figure 6-3: Search Categories and Countries in Database

The query induction searches the database and gets *Cathedral* from entity set *Category* and *France* from entity set *Country* for *Amiens Cathedral*. Similarly, the query induction gets *Church* and *Germany* for *Pilgrimage Church of Wies.*

Therefore the two examples from the user fall under the concepts *Cathedral*, *Church*, *France*, and *Germany* in the generative ontology.

Step 2 of the process of query induction

Then the query induction executes Concept Generalisation inside the regions *Category* and *Geographical Location* separately.

In the region *Category*, the result of Concept Generalisation of concepts *Cathedral* and *Church*is *Christianity Construction.*

In region *Geographical Location*, the result of Concept Generalisation of concepts *France* and *Germany* is *Western Europe*.

Step 3 of the process of query induction

Christianity Construction is the result of Concept Generalisation in the region *Category*. The query induction specialises the concept *Christianity Construction* by the concept induction Rule 2. According to the generative ontology, {*Christianity Monastery, Cathedral, Church ...*} = Concept Specialisation (*Christianity Construction*).

Western Europe is the result of Concept Generalisation in region *Geographical Location*. The query induction specialises the concept *Western Europe* by the concept induction Rule 2. According to the generative ontology, {*Holland*, *France*, *Germany …*} = Concept Specialisation (*Western Europe*).

Concept Specialisation (x_1, …, x_i) might be a large set includes all the concepts that are more specialised than all the elements in the set {x_1, …, x_i} in the generative ontology. Because of the limitation of the space, we only list some members of the set to illustrate the idea.

Step 4 of the process of query induction

In the above example, the result of Concept Specialisation from in *Category* and *Geographical Location* are attached to the concepts in the region *Site*to form the compound concepts. There might be many induced compound concepts, as long as the sites belong to the category *Religion Construction* and are located in *Western Europe*.

If there are concepts from *i* different regions, $i = 1 .. n$. The size of the result set of Concept Specialisation in each region is $size_i$. The number of the possible induced compound concepts is $size_1 \times size_2 \times \dots \times size_n$.

The induced compound concept can be expressed as a database query to retrieve all the other entities that are not visited by the user from the database. For example the query induction search for the entities of the compound concept *Site [BET: Christianity Monastery, LOC: Holland]* in the database, shown in Figure 6-4.

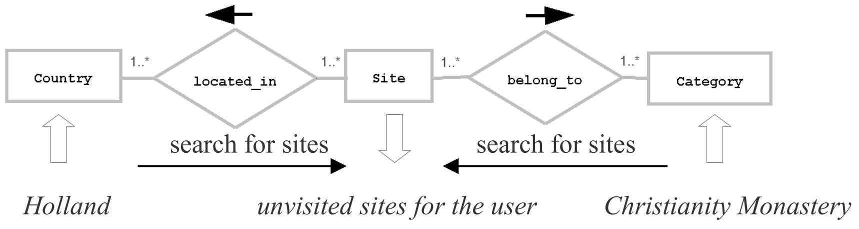


Figure 6-4: Search Sites in Database

6.1.3 Result of Concept Generalisation

The important consideration in a query induction is the result of Concept Generalisation. The result of Concept Generalisation in each region can reveal a certain aspect of the user's interests. For example, the result of Concept Generalisation in region *Category* can reveal the categories of the sites that the user feels interested in.

It is important to get the suitable result of Concept Generalised. In Figure 6-5, both the *Concept A* and the *Concept B* can cover the examples from the user. The more general concepts can cover more entities. A too general concept will be specialised to too many entities, so it is not the suitable result of Concept Generalisation.

Therefore, it is better to choose *Concept B* as the result of Concept Generalisation in Figure 6-5. *Entity 4* and *Entity 5* under *Concept B* can be used to form the compound concepts. And the reduced compound concepts can retrieve the unvisited entities for the user from the database. If we choose *Concept A*, it will include many entities under *Concept C*, and some of them are not so close to the examples from the user.

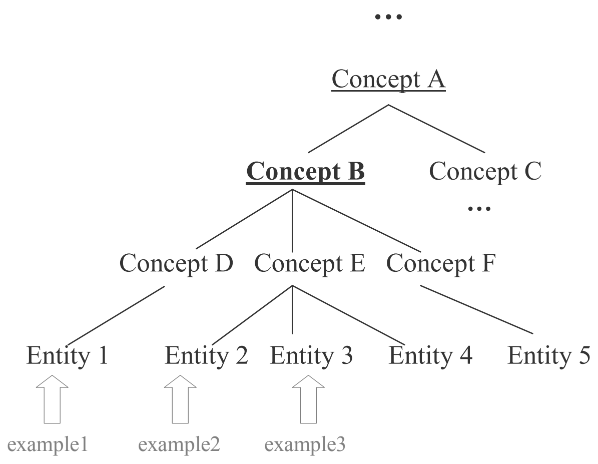


Figure 6-5: Result of Concept Generalisation

Among the four steps of the process of query induction that was described in the previous section, Step 2 is the step of Concept Generalisation process. In the following part of this chapter, we propose some methods and apply them in Step 2 to improve the result of Concept Generalisation.

6.2 Concept Generalisation by Count

An important consideration of a query induction is to find the suitable result of Concept Generalisation. We will propose a method, Concept Generalisation by Count, to produce more suitable result of Concept Generalisation.

First, we will explain the problem to be solved. Then we will explain the method in detail. Finally, we will discuss the improvements that this method can bring to the result of Concept Generalisation.

6.2.1 Problem

In the previous section, we have discussed how to execute Concept Generalisation according to concept induction Rule 1. The method introduced in this section, Concept Generalisation by Count, is based on concept induction Rule 1 with some modifications. It is designed to solve a practical problem in the query induction.

We use some World Heritage sites as examples to illustrate the problem to be solved by the method of Concept Generalisation by Count. There are many sites in the World Heritage domain. If a user visits many sites, the categories of these sites will fall into many different concepts in the region *Category* of the generative ontology of World Heritage.

First of all, if generalising a category to cover many different concepts in the Hasse diagram, the result of Concept Generalisation might often be the TOP.

If the result of Concept Generalisation is TOP, then the result of Concept Specialisation in Step 3 will be all the categories in the Hasse diagram. Obviously, TOP is not a good result because it is too general and cannot help the user find the information that he/she is looking for.

Generally, the problem to be solved by the method of Concept Generalisation by Count is:

- When the examples from the user fall under many concepts of the generative ontology, the result of Concept Generalisation according to concept induction Rule 1 is not very suitable because it is very possible for the TOP of the Hasse diagram to be the result of Concept Generalisation.

In order to solve this problem, we apply 'count' when executing Concept Generalisation. The method of Concept Generalisation by Count evolves from the concept induction Rule 1 with some modifications.

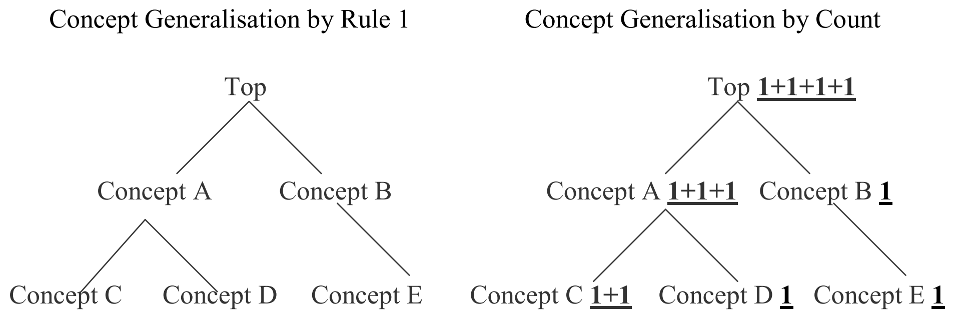
6.2.2 Concept Generalisation by Count

Every time an example from the user falls under a certain concept, the count of this concept is increased, and the counts of all the concepts which are more general are increased as well. Therefore, if one concept has a larger count, it is most likely the one the user feels interested in because the user has selected many examples under this concept. Therefore, the count of a concept can be expressed as:

count = the times that the concept covers the examples from the user

It is obvious that the more general concepts normally have larger counts and the TOP of a Hasse diagram always has the largest count. But as we have discussed in Section 6.1.3, a too general concept is not a good result of Concept Generalisation. Therefore, when the query induction chooses the result of Concept Generalisation, there are two factors taken into consideration. The first factor is the count of a concept. And the second factor is how general the concept is in the Hasse diagram.

The Concept Generalisation by Count method can solve the problem presented in previous section. Let us use an example to compare Concept Generalisation by concept induction Rule 1 and Concept Generalisation by Count.



1 example under *D*	1 example under *D*	*D*:1, *A*:1, TOP:1
2 examples under *C*	2 examples under *C*	*C*:2, *A*:3, TOP:3
A is the answer	**A is the answer**	
1 example under *E*	1 example under *E*	*E*:1, *B*:1, TOP:4
Top is the answer	**A is the answer**	

Figure 6-6: Concept Generalisation by Count - a Compare

The explanation for Concept Generalisation by Rule 1 is as follows:

Let us assume that one example from the user falls under *Concept D*; two examples fall under *Concept C*; one example falls under *Concept E*. Then the query induction result is TOP according to the concept induction Rule 1, notation: TOP = Least Concept Generalisation (*C*, *D*, *E*).

The explanation for Concept Generalisation by Count is as follows:

When one example from the user falls under *Concept D*, so the count of *Concept D* is increased by 1; and the counts of *Concept A* and TOP are also increased by 1. Similarly, when two examples fall under *Concept C*, the counts of *Concept C*, *Concept A and* TOP are increased by 2. When one example falls under *Concept E*, the counts of *Concept E*, *Concept B*, and TOP are increased by 1.

The next step is to compare the counts to choose the result of Concept Generalisation. We avoid choosing TOP as the result of Concept Generalisation, although it has the largest count. Instead, concept *A* can be chosen as the result, because it has the second largest count and it is more specialised than TOP.

Concept Generalisation by Count is an improvement of concept induction by Rule 1. It is also used in Step 2 of the query induction process. We conclude the steps of Concept Generalisation by Count and list them as follows. Because it evolves from Step 2 of the query induction process, we indicate them as Step 2a and Step 2b.

Assume a set of concepts $\boldsymbol{A}$. The steps of Concept Generalisation by Count are defined as follows:

- Step 2a: Let x, $cg_i \in \boldsymbol{A}$, $i = 1, \ldots, n$. Every time an example from the user falls under a certain concept x in a generative ontology, the count of the concept cg_i is increased by 1, if $cg_i \in$ **Concept Generalisation** (x).
- Step 2b: Choose the suitable result of Concept Generalisation according to the count of a concept and how general the concept is in the generative ontology. The TOP in a Hasse diagram is not a suitable result because it is too general, although it has the largest count.

6.2.3 Result of Concept Generalisation by Count

As we have mentioned, the result of Concept Generalisation is an important consideration in a query induction. Concept Generalisation by Count is designed to get a suitable result of Concept Generalisation. From the above example, we can see that this method can always get a result concept which can reveal the user's interest but not too general.

The way to select the result of Concept Generalisation by Count is flexible. It depends on the application domain of the query induction. If there are many entities under each concept in the generative ontology, it is better to choose the more specialised concept as the result. Otherwise a too general concept will bring too many results.

Therefore, the result of Concept Generalisation by Count can cover the most intensive examples from the user as shown in Figure 6-7.

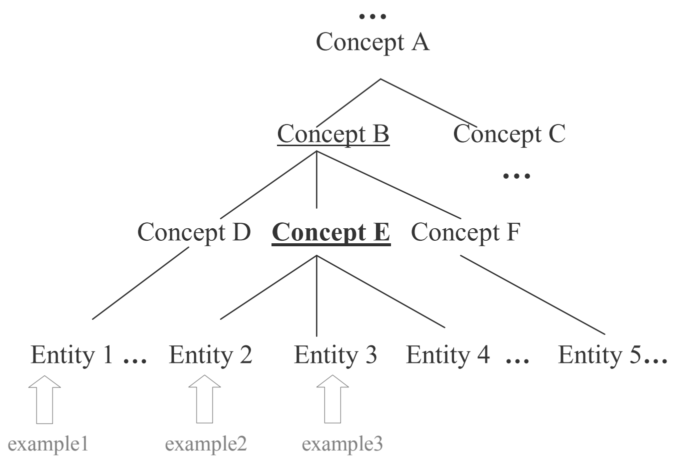


Figure 6-7: Result of Concept Generalisation by Count

If there are many entities under the concepts, the *Concept E* is more suitable to be the result of Concept Generalisation than *Concept B*, because it can cover most of the examples from the user and is more specialized than *Concept B*.

Furthermore, the method of Concept Generalisation by Count has the advantage of keeping the history of the result. The count of a certain concept is increased every time an example falls under this concept. The new count of a concept is based on the old count of this concept. Therefore, the result of Concept Generalisation by Count has the accumulative property.

In conclusion, the advantages of using Concept Generalisation by Count are:

- The query induction can avoid choosing the TOP of the Hasse diagram as the result. Instead, the query induction can choose the suitable result of Count Generalisation according to the count of the concept and how general the concept is.

- The new count of a concept is based on the old count of this concept. Therefore the history of the query induction results is kept. The result of the query induction is accumulative.

6.3 Similar Interests

Similar Interests are the interests that are not the user's interests exactly, but they are close to the user's interests.

In this section we will discuss why the method of Similar Interests is used and how to use this method to improve the query induction.

6.3.1 Problem

In the previous section, we have discussed the Concept Generalisation by Count, which is designed to solve a practical problem in the query induction. The method of Similar Interests also aims to solve a practical problem.

Take the World Heritage domain as an example. The category *Urban Settlement* covers 119 sites, while the category *Rural Settlement* only covers 5 sites.

First, it is hard for a user to find the sites of the category *Rural Settlement* because there are very few. On the contrary, it is very possible for the user to select the sites of the category *Urban Settlement* because of the big amount. Therefore, it is more possible for the concept *Urban Settlement* to be the result of Concept Generalisation.

Second, because the result of Concept Generalisation by Count has the property of accumulation, when the count of concept *Urban Settlement* has reached 6, the concept *Rural Settlement* can never be the result, because its count can never exceed 5. Instead, the concept *Urban Settlement* will be the result of Concept Generalisation repeatedly.

Generally, the problems to be solved by the method of Similar Interests are:

- It is hard for the user to find these entities that are small groups among all the web information.
- The concepts that cover many entities might be the result of Concept Generalisation more easily and repeatedly. Therefore the query induction might give the same results of query induction to the user repeatedly.

6.3.2 Similar Interests

It is boring for a user to get the same query induction results again and again. Furthermore, if the user visits more and more entities under certain concepts, there are less and less unvisited entities to be suggested later on. Therefore, it is reasonable and suggestive to give the user some entities, which are not in the user's interest exactly, but similar to the user's interest.

As we have discussed in Section 6.1.1, the result of Concept Generalisation can be considered as the user's interest. Therefore, the query induction can choose some other concepts that are similar to the result of Concept Generalisation. These concepts are considered to be the Similar Interests.

Considering the example above, if the concept *Urban Settlement* has been the result of Concept Generalisation for several times, the query induction will try to choose the concept *Rural Settlement* as the result of Concept Generalisation because the concept *Rural Settlement* is similar to the concept *Urban Settlement*.

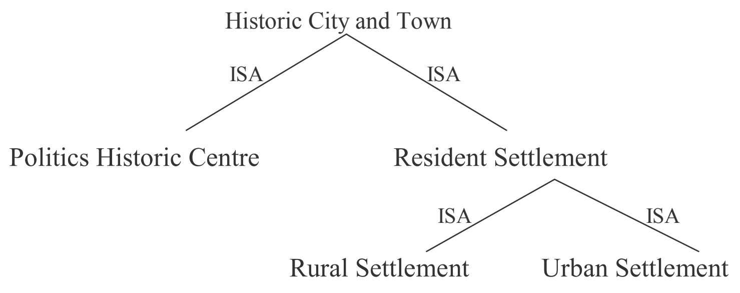


Figure 6-8: Similar Interests – an Example

If the concept *Urban Settlement* has been the result of Concept Generalisation for several times, we can find it is easy to get the similar interests of the concept *Urban Settlement,* which is *Rural Settlement,* because both of them are directly under *Resident Settlement*, as shown in Figure 6-8.

The method of Similar Interests is an improvement of Step 2 of the query induction process after Concept Generalisation by Count has been introduced, so we indicate this method as Step 2c.

Assume a set of concepts ***A***. The steps of Concept Generalisation by Count are defined as follows:

- Step 2c: Let r, x, $y_i \in \boldsymbol{A}$, $i = 1, \ldots, n$. If x has been the result of Concept Generalisation for several times, a query induction can choose y_i as the result of Concept Generalisation if there is a path from r to x in which x is strictly below r, and there is also a path from r to y_i in which y_i is strictly below r.

6.3.3 Result of Concept Generalisation by Similar Interests

When one concept has been the result of Concept Generalisation for several times, the query induction chooses some other concepts which are similar to this concept as the result. So the user can get various and suggestive query induction results.

In figure 6-9, when *Concept E* has been the result of Concept Generalisation for several times, the query induction system can suggest *Concept D* and *Concept F* to be the results of Concept Generalisation because they are similar to the concept *E.* Therefore the entities under *Concept D* and *Concept F* can be used to search for the unvisited entities in the database for the user.

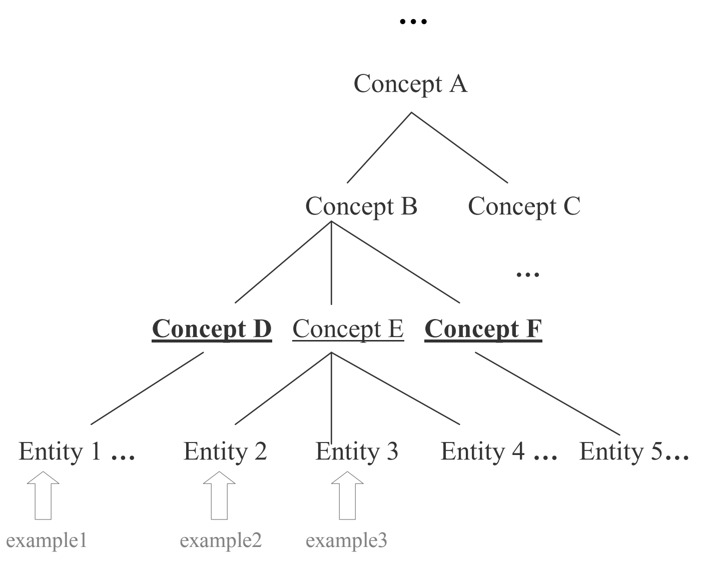


Figure 6-9: Result of Concept Generalisation by Similar Interests

In conclusion, there are two advantages of the method of Similar Interests:

- It provides the user some possibilities to find the entities that are small groups among all the web information, which might hard for the user to find.
- This method can give the user some unvisited entities similar to his/her interests, and avoid giving the user the same query induction results repeatedly.

6.4 Concept Generalisation by Recent Examples

In this section we will discuss the method of Concept Generalisation by Recent Examples. This method is introduced for the query induction to catch the change of the user's interests quickly.

6.4.1 Problem

The accumulation of the result of Concept Generalisation by Count improves the performance of the query induction, but it also brings the problem that the query induction is not sensitive to the change of the user's interest.

For example, when the user has selected 5 different World Heritage sites of the category *Urban Settlement*, the concept *Urban Settlement* is the result of Concept Generalisation. If the user changes his/her interest from the category *Urban Settlement* to the category *Forest* The concept *Forest* will be the result of Concept Generalisation only when its count exceeds the count of concept *Urban Settlement*. So the query induction can not get the concept *Forest* as the result until the user has selected 6 sites of the category *Forest*.

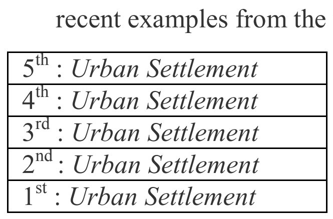
When the user's interest has changed, it takes the query induction, which adopts the method of Concept Generalisation by Count, a long time to catch the user's new interest. Generally, the problem to be solved by the method of Concept Induction by Recent Examples is:

- The accumulative property of Concept Generalisation by Count reduces the sensitivity to the change of the user's interest. So it takes a long time for the query induction to catch the user's new interest.

6.4.2 Concept Generalisation by Recent Examples

In order to catch the new interest of the user, we introduce the method of Concept Generalisation by Recent Examples. This method executes the query induction based on the recent examples from the user. The recent examples from the user can reveal the new interest of the user, so the query induction can catch the user's new interest quickly.

Considering the example above, we assume that the query induction is based on 5 most recent examples from the user. When the user selects 5 World Heritage sites of the category *Urban Settlement,* the query induction is based on these 5 sites. The result of Concept Generalisation in the region *Category* is the concept *Urban Settlement* because its count is 5, shown in Figure 6-10.

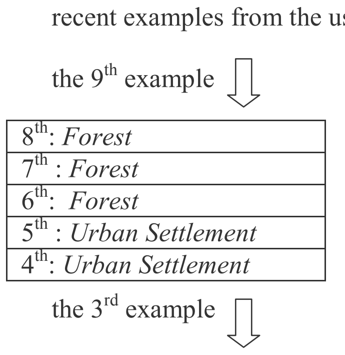


1. The first five World Heritage sites are of the Category *Urban Settlement*.

Figure 6-10: Urban Settlement is the Result of Concept Generalisation

When the user changes his/her interest to the category *Forest* and begins to select the sites of category *Forest*. Every time when the user selects an example, the query induction will clear the oldest example, so that the query induction is always based on the 5 most recent examples.

Shown in Figure 6-11, after the user selected 3 sites of the category *Forest*, the count of concept *Urban Settlement* is decreased to 2, and the count of concept *Forest* is increased to 3. Therefore, the result of Concept Generalisation based on 5 most recent examples is *Forest* because the concept *Forest* has a larger count than the concept *Urban Settlement*.



2. When the user selected 3 World Heritage sites of the category *Forest*, the concept *Forest* is the result of Concept Generalisation.

Figure 6-11: Forest is the Result of Concept Generalisation

From the problem description in Section 6.4.1, we know that the query induction dose not change the result of Concept Generalisation to *Forest* until the user has selected 6 sites of the category *Forest* using the method of Concept Generalisation by Count.

Using the method Concept Generalisation by Recent Examples, after the user selected 3 sites of *Forest*, the query induction can change the result of Concept Generalisation to the user's new interest. Therefore, the query induction can catch the user's new interest more quickly.

In Figure 6-10 and Figure 6-11, we use 5 most recent examples from the user to execute Concept Generalisation. The choice of the number is depend on the application domain and the system. One consideration is to avoid the 'Tie' situation, which happens when there are some concepts with same counts. We choose the odd number to avoid the 'Tie' situation, but it will still happen in the case shown in Figure 6-12. Both *Forest* and *Urban Settlement* will be chosen as the result of Concept Generalisation.

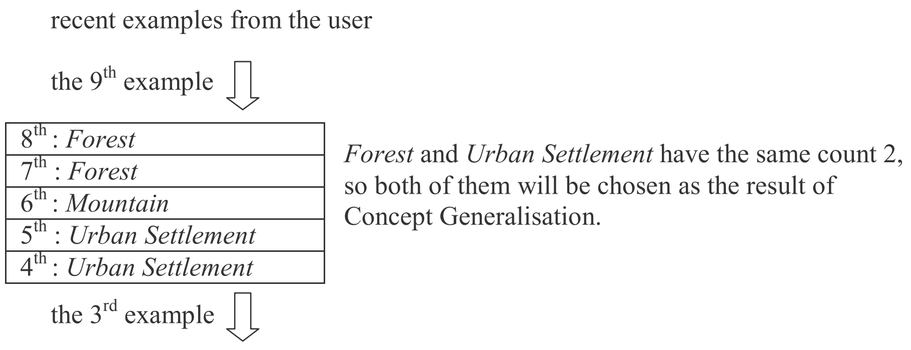

recent examples from the user
the 9th example
8th : *Forest*
7th : *Forest*
6th : *Mountain*
5th : *Urban Settlement*
4th : *Urban Settlement*
the 3rd example
Forest and *Urban Settlement* have the same count 2, so both of them will be chosen as the result of Concept Generalisation.

Figure 6-12: 'Tie' Situation

Concept Generalisation by Recent Examples is an improvement of Step 2 of the query induction process after Concept Generalisation by Count and Similar Interests have been introduced. Therefore we indicate this method as Step 2d.

- Step 2d: Every time there is a new example from the user, the query induction clears the oldest example, so the query induction is always based on some most recent examples.

6.4.3 Result of Concept Generalisation by Recent Examples

The result of Concept Generalisation by Recent Examples can reveal the user's new interest, because the query induction is based on the recent examples from the user.

The advantage of the method of Concept Generalisation by Recent Examples is:

- The query induction, which adopts the method of Concept Generalisation by Recent Examples, can easily catch the user's new interests. Instead of keeping all the history, the query induction is only based on the most recent examples from the user.

6.5 Concept Generalisation by Weight

We introduce the unit weights for the concepts to achieve Concept Generalisation by Weight. A weighed query induction allows the user assign higher unit weights to the concepts that they feel interested in. So it is more possible for concepts with higher unit weights to be the query induction result.

6.5.1 Problem

The methods discussed in the previous several sections consider two factors when executing the Concept Generalisation. The first factor is the times that the certain concept covers the examples from the user. The second factor is how specialised the concept is in the Hasse diagram.

Although these designs improve the performance of the Concept Generalisation, there are still two practical problems needed to be solved.

First, when a user selects some examples by mistake, the count of the concepts that cover the examples will be increased, although in fact the selected examples are not in the user's interest.

This situation is possible to happen in the World Heritage domain because some names of the sites can not express the contents of the sites. Therefore, a user might select some sites by mistake, which are not what he/she is looking for. However, the counts of the concepts, which cover the selected sites by mistake, are increased anyway.

The second problem can also be illustrated by the examples from World Heritage domain. There are many sites that belong to two or more categories. Let us assume that one site belongs to the two categories *Historic Capital* and *Cathedral*. A user selects this site, because he/she feels interested in *Historic Capital*. However, the counts of both concepts are increased although *Cathedral* is not the user's interest.

Therefore, the problem that needs to be solved by Concept Generalisation by Weight is:

- In some cases, the counts of some concepts are increased by mistake.

6.5.2 Concept Generalisation by Weight

In order to improve the accuracy of the query induction results, we let the user assign the unit weights to the concepts according to their interests.

This method considers three factors during the execution of Concept Generalisation. The first factor is the times that the concept covers the examples from the user. The second factor is how specialised the concept is. And the third factor is the pre-assigned unit weights of the concepts. The first and second factors are same as before, while the third factor is newly introduced by Concept Generalisation by Weight.

The first factor and third factor compose the total weight of a concept, which can be calculated by:

$$\text{Weight} = \text{Count} \times \text{Unit Weight}$$

To achieve Concept Generalisation by weight, some modifications are needed during the Concept Generalisation. Every time an example from user falls under a certain concept, the weight of this concept is increased by its unit weight. Therefore, if one concept has higher weight, it is very possible that the user feels more interested in this concept.

We choose some concepts from the region *Category* in the generative ontology of World Heritage as examples. The chosen concepts are {*Historic City and Area*, *Cultural Landscape*, *Construction*}.

Let us assume that the unit weight is in the scope of 0 ~ 1. If a user feels very interested in the sites of the category *Historic City and Area*, the user can assign the unit weight of *Historic City and Area* to 1.0. The category *Culture Landscape* is acceptable to the user, so its unit weight is 0.5. The user dislikes the category *Construction*, so its unit weight is 0. The steps of the Concept Generalisation by Weight are illustrated in Figure 6-13.

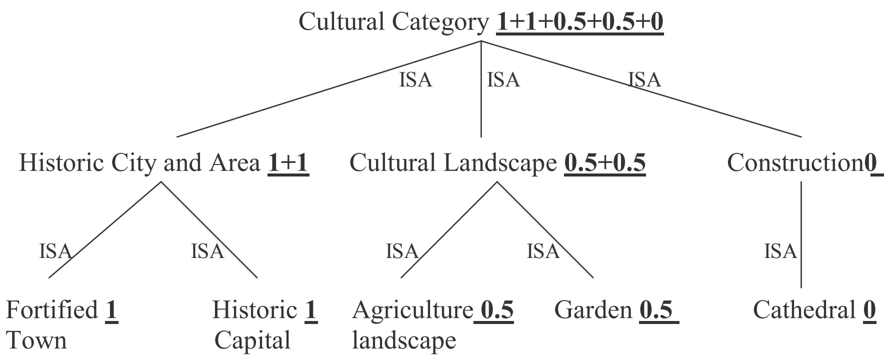

Cultural Category 1+1+0.5+0.5+0
ISA
ISA
ISA
Historic City and Area 1+1
Cultural Landscape 0.5+0.5
Construction0
ISA
ISA
ISA
ISA
ISA
Fortified 1 Town
Historic 1 Capital
Agriculture 0.5 landscape
Garden 0.5
Cathedral 0

Figure 6-13: Concept Generalisation by Weight – an Example

We assume that the concepts that are more specialised than *Historic City and Area* all have the unit weight 1.0, because the unit weight of *Historic City and Area* is 1.0,

So when there is an example fall under *Fortified Town* and *Historic Capital*, the weights of them are all increased by 1.0. And the weight of all the concepts that are more general than them are all increased by 1.0; therefore the weight of concept *Historic City and Town* is 2.0.

We assume that the concepts that are more specialised than *Cultural Landscape* all have the unit weight 0.5, because the unit weight of *Cultural Landscape* is 0.5.

So when there is an example fall under *Agriculture landscape* and *Garden*, the weights of them are increased by 0.5. And the weight of all the concepts that are more general than them are all increased by 0.5; therefore the weight of concept *Culture Landscape* is 1.0.

We assume that the concepts that are more specialised than *Construction* are all weighed 0, because the unit weight of *Construction* is 0.

So when there an example falls under *Cathedral*, its weight is increased by 0. And the weights of all the concepts that are more general than *Cathedral* are all increased by 0.

From Figure 6-13, we can see the resulting weight of each concept. We avoid choosing *Cultural Category* as the result of Concept Generalisation although it has the highest weight, because it is too general. Instead, *Historic City and Area* has the second highest weight and is more specialised, so it can be chosen as the result of Concept Generalisation.

Although *Cultural Landscape* has the same count as *Historic City and Area*, it has lower weight because its unit weight is lower. This result fulfils the user's requirement, because the user assigned a higher unit weight to the concept *Historic City and Area*, because he/she feels more interested in it.

Although the concept *Construction* was counted once, its weight is 0, because its unit weight is 0, which means the user isn't interested in it, so *Construction* has no weight.

The steps of Concept Generalisation by Weight are listed as follows. Because it evolves from Step 2 of the query induction approach, we indicate them as Step 2a', Step 2b'and Step 2c'.

- Step 2a': A user assigns the unit weights to the concepts according to his/her interests. The concepts, which are more interesting to the user, are assigned with higher unit weights.

- Step 2b': Let x, $cg_i \in \boldsymbol{A}$, $j = 1, \ldots, n$. Every time an example from the user falls under a certain concept x in a generative ontology, the weight of the concept cg_i is increased by its unit weight, if $cg_i \in$ **Concept Generalisation** (x).

- Step 2c': Choose the suitable result of Concept Generalisation according to the weight of a concept and how generalised the concept is in the generative ontology. The concept with the largest weight in a Hasse diagram is normally not a suitable result of Concept Generalisation because it is too general.

6.5.3 Result of Concept Generalisation by Weight

The result of Concept Generalisation by weight can be closer to the user's interest, because the user can assign higher unit weight to the concept that he/she feels interested in. The advantage of the method of Concept Generalisation by Weight is:

- The method of Concept Generalisation by Weight considers both the count and pre-assigned unit weights during the execution of Concept Generalisation. Therefore this method can improve the accuracy of the query induction results.

In the above example, we assume that the concepts, which are more specialised than *Historic City and Area*, have the same unit weight with *Historic City and Area*. Actually, every concept can have its unit weight. It is not necessary for the concepts to have the same unit weights as the more generalised concepts.

The other consideration is what should be done if the user forgets to assign the unit weights to some concepts. One possible solution is to set the default unit weight as 0.5, which is in the middle of the scope. By this way, the query induction will still weigh these concepts during execution. But they are not the most attractive to the user obviously, so the default unit weight should not be 1.0.

The method of Concept Generalisation by Weight can be used in different applications. For example, if a user has registered some interests in one website, the query induction system of this website can put larger unit weights to the concepts according the user's interests and remember the user's interests. Therefore, the website can provide better service to the user.

6.6 Conclusion

This chapter described a general approach of the query induction using the World Heritage domain as a case study. This approach might be applied to different domains.

First we designed the process of a query induction as 4 steps listed in Section 6.1. Referring to the previous theories, we concluded that Concept Generalisation and Concept Specialisation are the two main steps of the query induction. Then, inspired by the works done in Chapter 3 and 4, we identified the roles of the database and the generative ontology during the process of query induction, the roles of the database and the generative ontology are also included in the process of query induction

Furthermore, we analysed the problems which might interfere with the performance of the query induction and gave our suggestions to solve these problems. The suggested methods are: Concept Generalisation by Count; Similar Interests, Concept Induction by Recent Examples; and Concept Generalisation by Weight. These methods can improve the performance of the query induction with higher accuracy and suggestive results.

In the next chapter, we will apply these approaches and methods to build the query induction system of World Heritage.

7 System Design

In this chapter, we will introduce the architecture and the technologies applied on a query induction system. Based on the ER model and the generative ontology of World Heritage, we will design a query induction system of World Heritage. It provides the user the function of inducing queries from examples of World Heritage.

We will also present the detail designs corresponding to the different tiers in the query induction system of World Heritage. The designs of query induction described in the previous chapter are adopted in the Web Server tier.

7.1 System Architecture

The query induction system can be designed as a three-tier client/server system. The three-tier client/server architecture is the most popular type of n-tier client/server architecture which evolves from the two-tier architecture. The three-tier architecture emerges to overcome the limitations of the two-tier architecture. It separates application components into three logical tiers: the User Interface tier, the Business Logic tier, and the Database Access tier. In a three-tier client/server system, the User Interface tier communicates only with the Business Logic tier, never directly with the Database Access tier. The Business Logic tier is the middle tier and communicates both with the User Interface tier and the Database Access tier.

For the query induction system, the User Interface tier could be a browser. The user can see the interface in this tier. The Business Logic tier could be a Web Server. The query induction application could be executed here. The Database Access tier could be a Database Server. A database that stores the data of the query induction domain is in this tier.

Usually, the three-tier client/server system will be built up from the lower tier to the upper tier. So we will describe the query induction system of World Heritage in the same way. For each tier, there are many kinds of technologies that could be applied. For example, for the Web Server tier, there are many dynamic web page generation technologies such as ASP, PHP and JSP. In the following, we only present the technologies that are adopted by us in the query induction system of World Heritage.

The architecture of the query induction system of World Heritage is presented in the below figure. Here from lower tier to upper tier, we called the three-tiers respectively as Database Server tier, Web Server tier and User Interface tier.

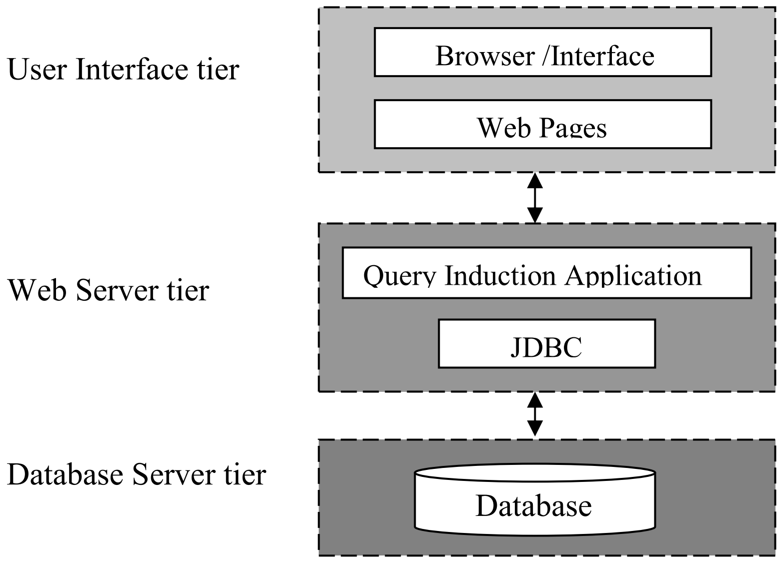


Figure 7-1: The System Architecture

The Database Server tier is the Database Server of a query induction system. For the system, it is a server only, in that it only responds to the requests from the Web Server tier. For the query induction system of World Heritage, we use the MySQL Database Server[3]. The MySQL Database Server is the most popular open source database management system, recognized for its speed and reliability.

The Web Server tier is the Web Server of the query induction system. From the view of the whole system, it acts as both a server and a client. It is a server relative to the User Interface tier because it processes its requests, and it is a client to the Database Server tier because it sends requests to it. The query induction application is implemented in this tier.

[3] www.MySQL.com

We adopt the Java Server Pages (JSP) technology for the implementation of the query induction system of World Heritage to process the users' requests. JSP technology is one kind of Dynamic Web Page Generation Technology [2]. The reasons we chose the JSP technology are:

- It allows web developers and designers to rapidly develop and easily maintain, information-rich and dynamic web pages that leverage existing business systems.

- As part of the Java family, it enables rapid development of web-based applications that are platform independent.

- It separates the User Interface from content generation, enabling designers to change the overall page layout without altering the underlying algorithms and classes.

In addition, we apply the JDBC[4] technology to implement the connection with the database of the World Heritage sites. JDBC technology is an API (Application Interface) that provides universal data access from the Java programming language [6]. Since JDBC also comes from the Java family, the codes of JDBC can be embedded in the JSP files to execute the queries in the database.

With a JDBC technology-enabled driver, the system can easily connect all the data even in a heterogeneous environment. We use MySQL Connector/J[5] as the Java driver because it is a connector between Java applications and the MySQL Database Server.

The User Interface tier is the browser of the users. It is a client only, in that it only sends requests to the Web Server tier. The users can browse the static web pages written in some kind of Markup Language such as HTML and XML. The dynamic web pages generated by the JSP programs can also be browsed by the users.

In the query induction system of World Heritage, we write some pages in HTML because the content of them are static. Most pages involved in the query induction are dynamically generated by the corresponding JSP programs.

---

[4] JDBC: Java DataBase Connectivity

[5] MySQL Connector/J is a native Java driver that converts JDBC calls into the network protocol used by the MySQL database.

The three-tier architecture is used when an efficient distributed client/server design is needed because it can provide increased performance, flexibility, maintainability, reusability, and scalability compared to the two-tier architecture. It can also hide the complexity of distributed processing from the user. For the query induction system of World Heritage, the user could be anyone with interest in the common World Heritage. Many users could visit the system at the same time. The Web Server can accept the requests from the User Interface tier. And then it runs the query induction application and connects with the Database Server tier in order to produce the query induction result. Afterwards it returns the induction result to the users. With the Web Server tier, the performance and flexibility of the query induction system will be increased. And since the users can't connect directly to the database, the data security is also increased.

In order to make the query induction system function well, the different designs are needed by the different tiers of the system. They are database design, the query induction design for an application domain and the interface design. For the different application domains of the query induction system, these designs could be different. In the following sections, we only present the designs in the different tiers for the query induction system of World Heritage.

7.2 Design of the Database Server Tier

Corresponding to the role of the database in a three-tier system architecture, we need to design a database to store the data for the system. A lot of data about the World Heritage sites are involved in the query indcuction system. Here we use a relational database to store the data. The goal of the database design is to generate a set of schemas that allow us to

- Store information without unnecessary redundancy.
- Support the query induction.

We design and create the schemas according to the ER model of World Heritage that was presented in Chapter 3.

site (siteid, site_name, site_briefdesc)

The schema *Site*stores the entities of the entity set *Site*, together with their attributes' values. Here is a sample site *Mountain Tai*.

siteid	site_name	site_briefdesc
725	Mount Tai	The sacred Mount Tai was the object of an imperial cult for nearly 2,000 years, and the artistic masterpieces found there are in perfect harmony with the natural landscape. It has always been a source of inspiration for Chinese artists and scholars and symbolizes ancient Chinese civilizations and beliefs.

BETcategory (siteid, category)

Schema *BETcategory* stores relationship *belong_to* between entity sets *Site* and *Category*, whose multiplicity is *one or many* to *one or many*. We use the name of the category in the schema *BETcategory*, so we do not use the extra schema for the entity set *Category*.

Natural sites will have one or more natural categories. Cultural sites will have one or more cultural categories. Mixed sites have both natural categories and cultural categories. For example *Mount Tai* is a mixed site that has one natural category *Mountai*n, and two cultural categories *Artistic Work* and *Associative Cultural Landscape*.

siteid	category
725	Artistic Work
725	Associative Cultural Landscape
725	Mountain

country (countryid, compositionid, country_name)

The schema *country* stores the entities of the entity set *Country* together with their attributes' values and relationship *belong_to* between entity sets *Country* and *Composition*, whose multiplicity is *one or many* to *one*.

countryid	Compositionid	country_name
109	10	China

LOCcountry (siteid, countryid)

The schema *LOCcountry* stores relationship *located_in* between the entity sets *Site* and *Country*, whose multiplicity is *one or many* to *one or many*. The sample site *Mount Tai* is only located in the country *China*.

siteid	Countryid
725	109

composition (compositionid, continentid, composition_name)

The schema *composition* stores the entities of the entity set *Composition of Continent* together with their attributes values and the relationship *belong_to* between entity sets *Composition of Continent* and *Continent*, whose multiplicity is *one or many* to *one*.

compositionid	**continentid**	**composition_name**
10	3	Eastern Asia

continent (continentid, continent_name)

The schema *continent* stores the entities of the entity set *Continent* together with their attributes' values.

continentid	**continent_name**
3	Asia

To input all the data for the above schemas of the database could be hard work because there are 754 World Heritage sites and a lot of data for each site. Fortunately, we obtained about 30 tables of the current database of the World Heritage sites from VRheritage Organization[6]. We choose some data from their tables, which will be used for the query induction, and copy the data to our own database.

7.3 Design of the Web Server Tier

Corresponding to the role of the Web Server tier the system architecture, the data structure and the design of query induction are needed by the query induction application in this tier. The data structure design for the query induction application is very important. A good data structure will make the query induction easy to implement and improve the efficiency. The general ideas of the query induction design are introduced in Chapter 6. We will apply them in the query induction system of World Heritage. First we will design the data structure. Then some methods for implementing the concept induction in the query induction system of World Heritage will be presented.

[6] For more information, refer to http://vrheritage.org/vrorg/

7.3.1 Data Structure

From Chapter 4 and 5, we know that the query induction of World Heritage is based on the concept induction in each region of the generative ontology of World Heritage separately. And the concepts situated in each region form a Hasse diagram structure. According to the design of the concept induction mentioned in the previous chapter, first we need to design a data structure for the concepts in the generative ontology. Second, we need to design a data structure to present the Hasse diagram structure of each region in the generative ontology of World Heritage.

Data Structure of Concept

The concepts, which are situated in the regions of the generative ontology of World Heritage, have their unique names. Here the name is represented by the concept itself. And each concept has its own path. Here we quote the definition of the path in a Hasse diagram. The path of one concept is a list of concepts ordered in the partial order. But the path does not include the TOP of the Hasse diagram. The reason is that the TOP concept will always have the largest count. In order to meet the needs of Concept Generalisation by Count and the other Concept Generalisation methods, each concept should also have a counter. It is used to count the times of a certain concept covering the examples from the user. Due to the path of the concept, it is easy to count the generalised concepts that are in the path. According to how specialised the concepts are, they could be in the different levels. The level of concepts directly below the TOP of the Hasse diagram is defined as 1. The levels increase along the Hasse diagram in the opposite direction of the partial order. Therefore, we design the data structure of the concept as below:

Concept: [conceptname, path, level, count]

There are four fields for each concept. For example, concept *Church*can be expressed as [Church, [Church, Christianity Construction, Religion Construction], 3, 0]. Its path is a list of concepts from itself to the *Construction* but not including *Construction* because *Construction* is the TOP of this Hasse diagram. Its level is 3. Initially its count is 0.

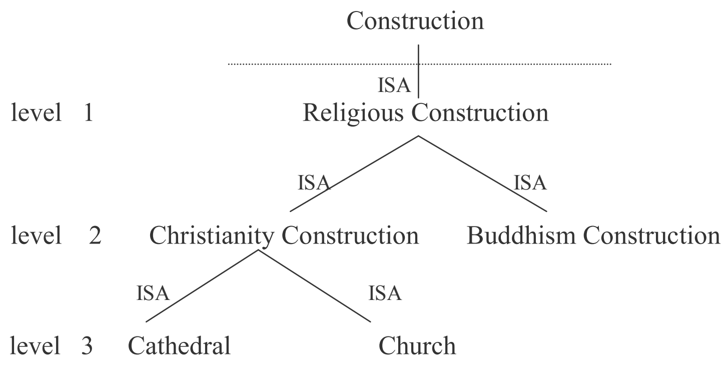


Figure 7-2: Data Structure of Concept – an Example

In the method of Concept Generalisation by Weight, the concepts need to have the unit weights according to the different degrees of the users' interests in them. So we define the data structure of the concept with weight as following:

Wconcept: [conceptname, path, level, weight, unit weight]

The *weight* field of a concept is decided by the times that it cover the examples from the user and its own unit weight. For example, if the concept *Church* covered 3 examples from the user and its *unit weight* field is 0.6, then the value of the *weight* field is 1.8.

There are several advantages of this data structure for Concept Generalisation. First, once an example of the user falls under a certain concept, it is easy to find all the more generalised concepts because they are on its path.

Second, through the *count* field of one concept, it is easy to count the times the concept cover the examples from the user. Every time when the concept covers one example, its count is increased. Accordingly the counts of the more generalised concepts on its path are also increased by 1. Instead, in Concept Generalisation by Weight, the weights of all the concepts on the path will be increased by their own unit weights instead of 1.

Third, the *level* field can be used to select a more suitable result of Concept Generalisation. If the system considers giving the more suitable concepts as the result of Concept Generalisation, the concept with the largest count or weight is not always chosen as the result of Concept Generalisation. Through comparing the levels of the different concepts, the system can know which one is more generalised or which one is more specialised. Then the system considers the count (or weight) and level of the concepts together and chooses one or several suitable concepts as the result of Concept Generalisation.

For the Similar Interests method, the *level* field can be used to find the similar interests as the new result of Concept Generalisation.

Data Structure of Region

In the generative ontology of World Heritage, the data structure of each region is a Hasse diagram, for example the region *Category*. Beside the concept set from one region, a name and the size of the concept set could be defined in the data structure. In the query induction, it should be avoided that the most generalised concept becomes the result of Concept Generalisation. So the TOP of the Hasse diagram is not included in the concept set. Since we have clearly expressed the positions and the relations among the concepts by the field *level* and *path* in the data structure of concept, they are not repeated in the data structure of region. The data structure of each region is defined as below:

Region:

```
{
  name,
  the number of concepts,
  [
    [concept1name, path, level, count],
    [concept2name, path, level, count],
    [concept3name, path, level, count]
    …
  ]
}
```

We give an example to illustrate the part of the Hasse diagram of region *Category* using the data structure defined above.

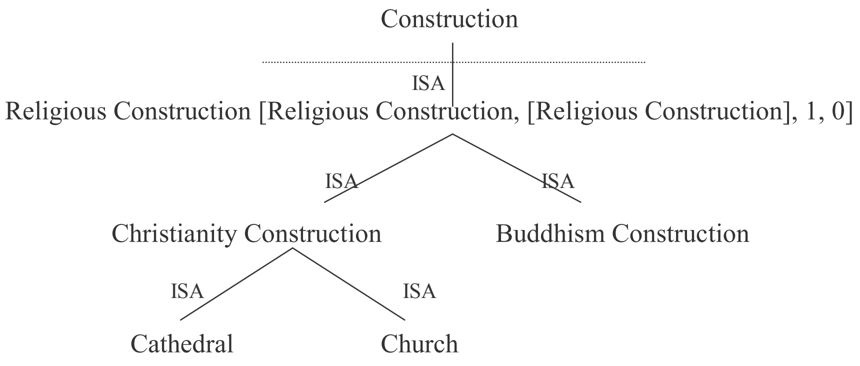

Construction
ISA
Religious Construction [Religious Construction, [Religious Construction], 1, 0]
ISA
ISA
Christianity Construction
Buddhism Construction
ISA
ISA
Cathedral
Church

Figure 7-3: Date Structure of Hasse Diagram – an Example

The above Hasse diagram could be expressed as:

```
{
  name: Construction,
  the number of the concepts: 6
  [
    [Religious Construction, [Religion Construction], 1, 0],
    [Buddhism Construction, [Buddhism Construction, Religious Construction], 2, 0],
    [Christianity Construction, [Christianity Construction, Religious Construction], 2, 0],
    [Cathedral, [Cathedral, Christianity Construction, Religious Construction], 3, 0],
    [Church, [Church, Christianity Construction, Religious Construction], 3, 0]
  ]
}
```

It is easy to achieve both Concept Generalisation by Count and Concept Generalisation by Weight with the above data structure. For example, if the user visits one site about *Church*, the concept can be found in the concept set. From its path, the more generalised concepts are found. And then their *count* fields are increased or their *weight* fields are increased according to which method of Concept Generalisation is adopted. Using the Concept Generalisation by Count method, the result is:

```
{
  name: Construction,
  the number of the concepts: 6
  [
    [Religious Construction, [Religious Construction], 1, 1],
    [Buddhism Construction, [Buddhism Construction, Religious Construction], 2, 0],
    [Christianity Construction, [Christianity Construction, Religious Construction], 2, 1],
    [Cathedral, [Cathedral, Christianity Construction, Religious Construction], 3, 0],
    [Church, [Church, Christianity Construction, Religious Construction], 3, 1]
  ]
}
```

Using the Concept Generalisation by Weight method, if we assume all the unit weights of these concepts in the figure 7-3 are 0.6, then the result is

```
{
  name:Construction,
  the number of the concepts:6
  [
    [Religious Construction, [Religious Construction], 1, 0.6],
    [Buddhism Construction, [Buddhism Construction, Religious Construction], 2, 0],
    [Christianity Construction, [Christianity Construction, Religious Construction], 2, 0.6],
    [Cathedral, [Cathedral, Christianity Construction, Religious Construction], 3, 0],
    [Church, [Church, Christianity Construction, Religious Construction], 3, 0.6]
  ]
}
```

7.3.2 The Query Induction Application of World Heritage

Based on the data structures of the concepts and the regions of generative ontology, and referring to the design of query induction described in Chapter 6, we will present the specific methods which are used in the query induction application of the system. In order to be read easily, we also describe the corresponding methods in pseudo-code which is composed of the natural language and the structure language.

All the below methods are based on the four steps of query induction that were described in detail in Chapter 6. The data structure design is also applied in the methods. All the following pseudo-code uses the example of the query induction in the region *Category* of the generative ontology of World Heritage. For the region *Geographical Location*, the process is similar. Here we will not repeat it.

Our design follows the data-flow in the induction query system of World Heritage. The process is shown in the following figure:

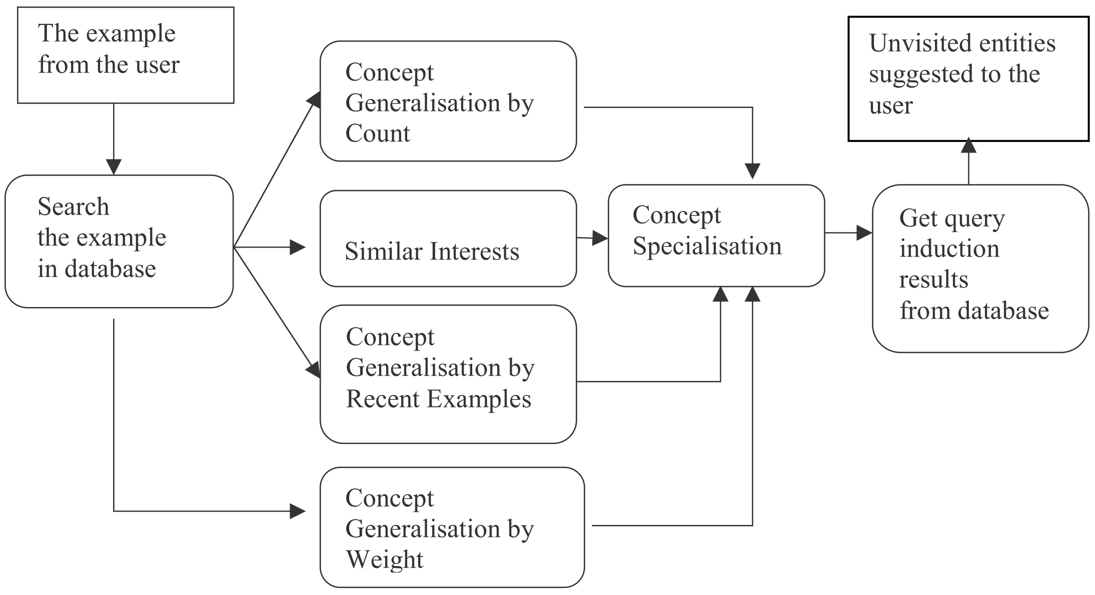

The example from the user
Search the example in database
Concept Generalisation by Count
Similar Interests
Concept Generalisation by Recent Examples
Concept Generalisation by Weight
Concept Specialisation
Get query induction results from database
Unvisited entities suggested to the user

Figure 7-4: Process in the Query Induction System of World Heritage

Step 1: The **Search** method is used to find the categories and countries of the example form the user in the database. If the user visits the site *Fraser Island*, the system will connect with the database and obtain the *Category* of this site: *Rainforest, Island, Lake* and the *Country* of this site: *Australia.*

CCPTset represents a set of the categories of the example site. The categories are retrieved from the database.

```
Input (example from user)

Open a connection with database

if (site in database = example )
   then
      CCPTset.add (the categories of the site)

Return CCPTset
```

Step 2: The **Concept Generalisation** method is used to execute Concept Generalisation of the concepts that cover the examples from the users.

Here we adopt the method of Concept Generalisation by Count, and the other methods will be explained later in this section.

The system tries to find the categories from Step 1 in the Hasse diagram of the region *Category* in the generative ontology of World Heritage (OWHset) (see Figure 5-8). If one concept is found, all the *count* fields of the concepts on its path will be increased according to Step 2a of Concept Generalisation by Count presented in section 6.2.2. For the Concept Generalisation by Weight, the *weight* fields of concepts will be increased by their unit weights.

OWHCset represents a set of concepts in the *Category* region of the generative ontology of World Heritage.

```
Input (CCPTset)

for (int i ; i< OWHCset.size; i++)
{
   for (int j ;j<CCPTset.size;j++)
   {
     if (OWHCset(i)=CCPTset(j))
        then
        {
          Get the concepts from OWHCset
          Get the paths of these concepts
        }
     all the concepts on the paths Do count++
   }
}
```

In order to get more suitable result, the result of Concept Generalisation could be decided by the field *count* and *level* of the concepts. The way to select the result of Concept Generalisation is flexible. We try to select the more specialised concept as the results in order to avoid inducing too many sites for the users. If the result of Concept Generalisation is more specialised, the result of Concept Specialisation in the next step will be more explicit. As a result, the induced sites are also closer to the users' interests.

After counting the concepts in the generative ontology, we stated the select method to choose the result of Concept Generalisation. If there is only one concept *A* with the largest count, then the system compares the *count* fields of the concepts that are in the GCSset. GCSset is the result set of Concept Specialisation (*A*) but not including *A*. At last, select these concepts in GCSset that have the largest count as the result of Concept Generalisation.

GCSset represents a set of concepts that are more specialised than the concept with the largest count. It does not include the concept with the largest count.
CGset represents the concept set of the result of Concept Generalisation.

```
if (only A.count is the largest)
  then
  {
    for(int i ; i< GCS.  size; i++)
      {
       if  ( (GCS.get(i)).count is the largest in  GCS set
         then CGset.add(GCS.get(i))
      }
    Return CGset
  }
  else
 {
    CGset.add(A)
    Return CGset
  }
```

If there are several concepts with the largest count or weight, the concepts with the lowest level will be selected as the result of Concept Generalisation. The reason is thatthey are the most specialised concepts.

```
If (several concepts have the largest count)
  then
  {
    Compare concept.level
    Get the concepts with the largest level
    CGset.add(the concepts with the largest level)
    Return CGset
  }
```

Step 3: The **Concept Specialisation** method is used to find the concepts that are more specialised than the concepts which are the result of Concept Generalisation.

According to the path definition of the data structure of concept, the path is a list of concepts from the concept itself to the TOP (not including the TOP) in the Hasse Diagram. In order to find the more specialised concepts, the system tries to find the concepts whose paths contain the concepts in the CGset (the concept set of the result of Concept Generalisation).

CSset represents the result set of Concept Specialisation.

```
Input (CGset)

for  (int i ; i< OWHCset.size; i++)
{
  for  (int j; j< CGset.size; j++)
  {
    if (OWHCset(i).path contains CGset(j))
    then
      CSset.add (OWHCset(i))
  }
}
Return CSset
```

Step 4: The **Get Query Induction Results** mehod is used to find the entities in the database that conform to the compound concepts which are composed by the results of Concept Specialisation from each region of the generative ontology.

CSset is the result of Concept Specialisation of the region *Category* from the above steps. Assume that we also get LSset as result of Concept Specialisation of the region *Geographical Location* by the similar process as listed above. According to the generative ontology of World Heritage, the concepts in CSset of *Category* and the concepts in LSset of *Geographical Location* will be attached to *Site* and form a set of the compound concepts, CCset. The system will query the database by the compound concepts and return Siteset which is a set of the World Heritage sites as the results of query induction. In other words, the elements in Siteset are the unvisited entities for the user, which are induced from the user's examples.

```
Input (CSset,LSset)

for (int i; i<CCset.size;i++)
{
  Query database by CCset(i)
 }

Return Siteset
```

In Chapter 6, we presented some improvements of Concept Generalisation, which are Similar Interests, Concept Generalisation by Recent Examples and

Concept Generalisation by Weight. In the following, we will describe the design of these methods.

The **Similar Interests** method is used to find new similar concepts when the result of Concept Generalisation has been repeated in a number of times.

In order to know if the result of Concept Generalisation is repeated, the system keeps the result of Concept Generalisation of every example from the user. Here we defined a variable named *repeat times* to record the repeat time of the result of Concept Generalisation. It is initialized to 0. The system always compares the current result with the previous result, if they are the same, the *repeat times* variable will be incremented. When the *repeat times* exceeds a fixed value, the system will take some new concepts, which are similar to the original result of Concept Generalisation, as the new result of Concept Generalisation. According to definition of the Hasse diagram, we know that two concepts are similar if they are strictly under one same concept in the Hasse diagram. For the query induction system of World Heritage, once the *repeat times* is more than 5, the system will provide the similar interests for the users.

CGSset represents a set of the old CGset. CGSset is used to keep the result of Concept Generalisation of every example of the user.
SICGset represents a set of the similar concepts which is the result of Concept Generalisation.

```
Input (CGset)

repeat times=0

if (repeat times < 5)
  then
  {
    Compare the CGset with CGSset.lastelement
    if  true
      then
        repeat times +1
    execute Concept Generalisation as Step2
    return CGset
  }
  else
  {
    if (repeat times = 5)
      then
        for (int i ;i<CGset.size();i++)
        {
          for  (int j; j< OWHCset.size; j++)
          {
            if (CGset(i).level = OWHCset(j).level) And
              (CGset(i).path and OWHCset(j).path
                contains (CGset(i).path.size-1) common  concepts)
                then
                  SICGset.add(OWHCset(j))
            }
        }

    repeat times = 0
    Return SICGset
```

The **Concept Generalisation by Recent Examples** method is also an improvement of Concept Generalisation algorithm. It executes Concept Generalisation based on the recent visiting history of the user.

Our method focuses on the latest 9 sites visited by the user and clear the other old records. That means when the user visits the 10th site, the system will clear the result of Concept Generalisation of the 1st site he/she visited.

For the first 9 sites, the system will execute Concept Generalisation as Step 2 and increase the visiting times. System executes Concept Generalisation every time when it got an example from the user, so there are 9 sets of result of Concept Generalisation for the 9 visited sites. These result sets are kept in the NCGSset.

NCGset represents a set of the CGset, it always keeps the categories of the latest 9 sites visited by theusers.
OCGset represents a set of the old result of Concept Generalisation.
visit times represent how many sites have been visited by the user.

```
Input (CGset)

visit times = 0
NCGNset.add(CGset)
execute Concept Generalisation as Step2
visit times++
```

When the user selects the 10th sites and the system adds the 10th result of Concept Generalisation to the NCGSset, the result of Concept Generalisation of the 1st site are move out and saved in the OCGset. Then the system will find the corresponding concepts in the generative ontology of World Heritage and subtract 1 from their counts. That means the history of the 1st site is erased from the system.

```
if (visit times > =9)
  then
  {
    OCGset.add(CGNset.firstelement)
    remove CGNset.firstelement
   }
```

```
for (int i ;i<OCGset.size();i++)
{
  for  (int j; j< OWHCset.size; j++)
  {
    if  (OCGset(i) = OWHCset(j))
       then
          OWHCset(j).count-1
  }
}

 execute Concept Generalisation as Step2
 get a new CGset
 Return the new CGset
```

The **Concept Generalisation by Weight** method for the query induction system of World Heritage is not presented in here because it is similar to Concept Generalisation by Count. By using Concept Generalisation by Weight, the users can assign the unit weights to the concepts according to the different degrees of the user's interest in the concepts. And then the concepts that cover the examples from the user can be counted by the concepts' unit weights instead of 1.

The way to assign the unit weights to concepts is flexible. In the query induction system of World Heritage, we only ask the users to assign the unit weights for some medium level concepts in the region *Category*, and the concepts that are more specialised than them will get the same unit weight. The unit weight is the real number form 0 to 1. The larger the value of the unit weight is, the more interested the user is in the concepts.

We come up with all the above methods according to the design of query induction and the design of data structure. Some other algorithms are used. For example, the sort algorithm [4] is used to find the concepts with the largest count and which concept with the lowest level.

In addition, the above description in pseudo-code mainly is used to induce the queries in the region *Category* of the generative ontology of World Heritage. The query induction in *Geographical Location* region is similar. The differences are the concepts and the partial order in the Hasse diagram.

With all these pseudo-codes, it is easy to implement all the designs of query induction in the high level program language like Java. The details about the implementations will be introduced in the next chapter.

7.3.3 The Transactions in the Web Server Tier

In this section, we will describe the system how to deal with the transactions in the Web Server tier by using the design of query induction. As mentioned before, the query induction system of World Heritage could be visited by several users at the same time. So when the different users submit their requests to the Web Server, the Web Server will initialize the different application instances for them. The initialized instances include the instances of the Hasse diagram with the concepts and relations from each region. All the variables that are used in query induction application is also included.

Meanwhile, the Web Server will send a request to the Database Server. And then a connection is opened for each user. For every user and every site he/she visited, the Web Server will keep the result of Concept Generalisation, visiting times and the repeat times of the same result of Concept Generalisation. Then the system will execute Concept Specialisation and find the query induction result from the database. At last, the system will return the web pages with the unvisited sites to the users as the responses to the requests of the users.

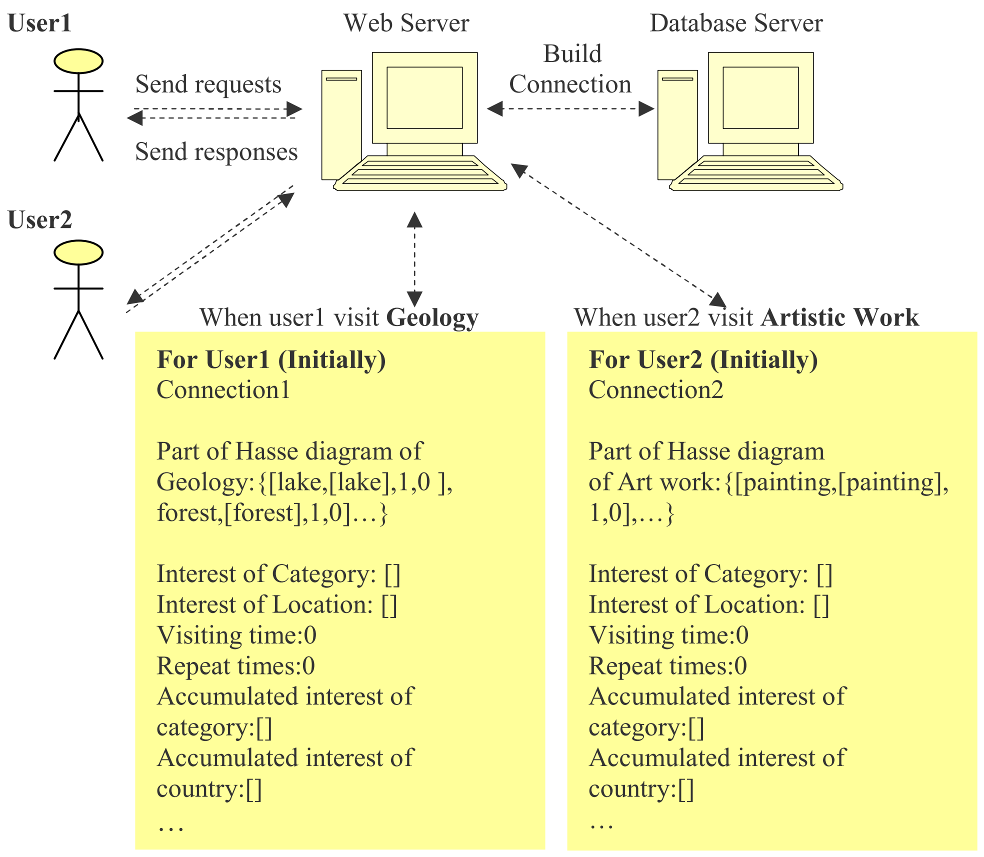

User1
Web Server
Database Server
Send requests
Send responses
Build Connection
User2
When user1 visit **Geology**
When user2 visit **Artistic Work**
For User1 (Initially)
Connection1
Part of Hasse diagram of Geology:{[lake,[lake],1,0], forest,[forest],1,0]…}
Interest of Category: []
Interest of Location: []
Visiting time:0
Repeat times:0
Accumulated interest of category:[]
Accumulated interest of country:[]
…
For User2 (Initially)
Connection2
Part of Hasse diagram of Art work:{[painting,[painting], 1,0],…}
Interest of Category: []
Interest of Location: []
Visiting time:0
Repeat times:0
Accumulated interest of category:[]
Accumulated interest of country:[]
…

Figure 7-5: Transactions in the Web Server Tier

In the Figure 7-5, when User1 begins to visit the sites of category *Geology*, the system will initialize the instance of the Hasse diagram of *Geology*, and the instances of all the variables that are used in the query induction for User1. Every time the user selects one site, the system will give him/her a dynamically generated web page with the information of the selected site and some unvisited sites as the query induction results.

If User1 stops visiting the sites of category *Geology* and begins to visit the sites of category *Historic City and Area*, the system would initialize the Hasse diagram of *Historic City and Area*. And all the history records of category *Geology* would be cleared. The Transactions in the Web Server tier is similar for User2. The difference is initialization of the instance of different Hasse diagram. In the next section, we will give more explanation of the initialization of the Hasse diagram, according to the different selection from the user.

The above serial transactions will be handled by the Web Server via running the application programs of query induction. For the query induction system of World Heritage, the application programs are written in Java. And then the Java classes and the JDBC code used to connect with database will be imported in some JSP files. These JSP files will be used to dynamically generate the web pages for the users. In next section, the design of the User Interface tier will be presented. The specific implementation of the query induction application programs will be described in Chapter 8.

7.4 Design of the User Interface Tier

Corresponding to the User Interface tier of the system architecture, the interface design is needed by the query induction system.

In order to let the readers easily know what the interfaces will look like, we use Figure 7-6 to show the contents of the pages. The interfaces of the Web portals consist of static HTML pages and HTML pages dynamically generated by JSP. All the underlined words will be the links on the web pages. We will give some explanations of the web pages as follows.

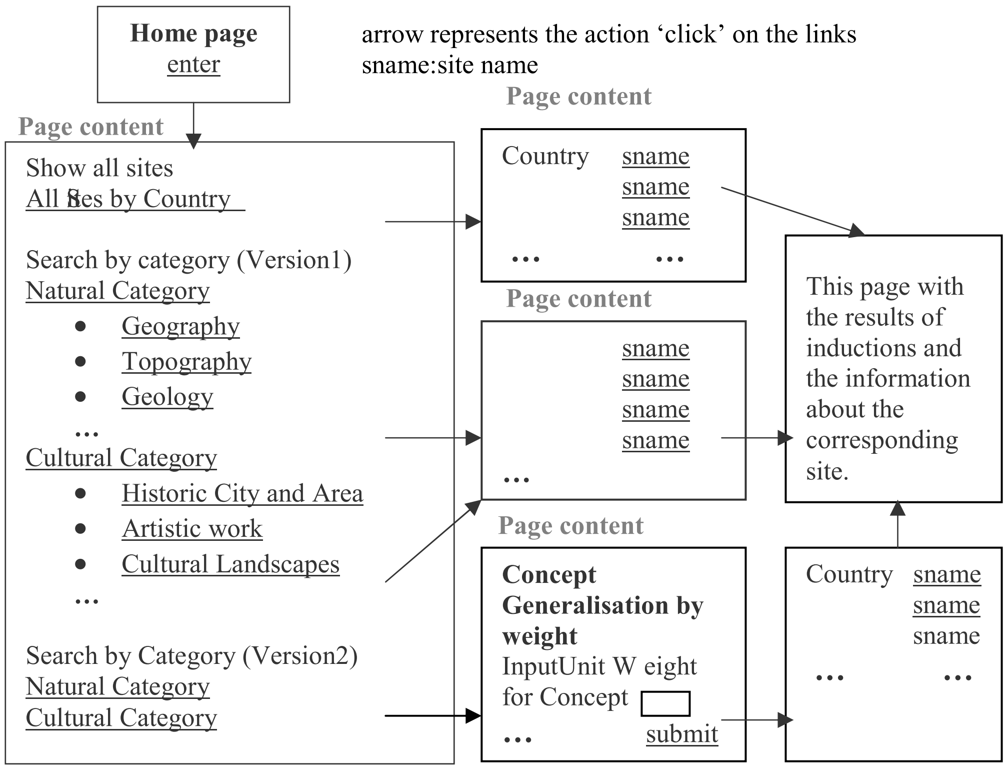

Home page
enter
arrow represents the action 'click' on the links
sname:site name
Page content
Show all sites
All Sites by Country
Search by category (Version1)
Natural Category
Geography
Topography
Geology
Cultural Category
Historic City and Area
Artistic work
Cultural Landscapes
Search by Category (Version2)
Natural Category
Cultural Category
Page content
Country sname sname sname
Page content
sname sname sname sname
Page content
Concept Generalisation by weight
InputUnit W eight for Concept
submit
This page with the results of inductions and the information about the corresponding site.
Country sname sname sname

Figure 7-6: Design of the User Interface Tier

The Web home page is a static HTML page which describes this project and the results produced in the process of the query induction.

An entrance should exist on the homepage which could link to the interface that provides the user different choices of query induction.

<u>All Sites by Country</u>: All sites are listed by the country in the alphabetic order. System will initialize the whole Hasse diagrams of the region *Category* and *Geographical Location* for the user. So the concept inductions are executed among all the concepts in the region *Category* and *Geographical Location*, based on the method of Concept Generalisation by Count. We listed the sites by their countries because many sites will be shown.

<u>Natural Category</u> (version 1): All natural sites are listed by the country in the alphabetic order. System will initialize the Hasse diagrams of *Natural Category* in the region *Category* and the whole Hasse diagram of *Geographical Location* for the user. So the concept inductions are executed among all concepts of *Natural Category* and all the concepts in region

Geographical Location, based on the method of Concept Generalisation by Count.

The link Cultural Category (version 1) is similar to the link Natural Category (version 1). All cultural sites are listed by the country in the alphabetic order. The concept inductions are executed among all concepts of *Cultural Category* and all the concepts in region *Geographical Location*.

Geography: All natural sites of the category *Geography* are listed by the alphabetic order without countries. System will initialize the Hasse diagrams of *Geography* in the region *Category*, so the concept inductions are executed among all concepts of *Geography*, based on the method of Concept Generalisation by Count.

Similar with Geography, the link Historic City and Area lead to execute the concept induction in part of the whole Hasse diagrams of region *Category*, the Hasse diagram of *Historic City and Area*.

Natural Category (version 2): The user will enter a static HTML page to assign the unit weights according to his/her interest. After the user submits the assignment, all natural sites are listed by the country in the alphabetic order. Similar to version 2, the concept inductions are executed among all concepts of *Natural Category* and all the concepts in region *Geographical Location*. Different from version 1, the concept induction is based on the method of Concept Generalisation by Weight in version 2.

The link Cultural Category (version 2) is similar to the link Natural Category (version 2). The difference is that it shows cultural sites and the concept induction is executed on the concepts of *Cultural Category*.

sname: This link is the name of the World Heritage site, for example Fraser Island. When the users click this link, the system will generate dynamically the pages with information of the selected site and the induction results.

Here we only described the design of these interfaces, in the following chapters, we will present the complete interfaces corresponding to the different HTML and JSP files.

7.5 Conclusion

We described three-tier client/server architecture of the query induction system in this chapter. The technologies adopted by the query induction system of World Heritage are also presented. Corresponding to the three-tiers of the query induction system of World Heritage, the different designs are described in details, especially the induction query design in the Web Server tier.

The current data structure of the Hasse diagrams and the methods of Concept Generalisation and Concept Specialisation can present the steps of the query induction described in Chapter 6. More efficient implementation can be achieved by using the data structure of *Directed Acyclic Graph (DAG)* and the graph algorithm of *Depth-First Search* [4]. Although the Hasse diagrams in the generative ontology of World Heritage are very close to the data structure of *Tree*, the data structure of *Directed Acyclic Graph* can be applied to more general cases.

In the next chapter, we will present the implementation of the query induction system of World Heritage.

8 System Implementation

In this chapter, the implementation of the query induction system of World Heritage will be discussed. In order to make the readers easily understand our implementation, we give the detailed explanations of the central part of the implementation. All the source code can be found in the appendix.

8.1 Installation

In order to build the three-tier client/server architecture and implement the query induction system of World Heritage, we downloaded and installed the following programs:

```
Apache: Apache-1.3.27.tar.gz
Java Servlet Container: Jakarata-tomcat-3.3.1.tar.gz
MySQL Database: MySQL 4.0
Java Program-Compiling Environment: J2sdk1-4-1- 02-fcs-Linux-i586.rpm
Database Connection: Mysql- connector-java-3.0.7-stable.tar.gz
```

First, we make Apache and the Java Servlet Container Tomcat work together as our web server. Secondly, the MySQL 4.0 database server is installed. Third, we obtain the Java application program-compiling environment and the connection between the database server and web server by the installation of the last two files.

In order to implement the three-tier client/server architecture in network environment and improve the efficiency of the Web Server and the Database Server, we installed the MySQL Database Server and the Web Server on the different computers. We also installed the MySQL Control Center in these two computers, the computer without installing the MySQL Database Server can use its local Control Center to access remotely the Database Server in the other computer through the Mysql- connector. For our prototype system, the IP address of Database Server is 130.225.76.177. The IP address of Web Server is 130.225.76.156. Through the HTTP protocol, the port 8080 of the Web Server can be visited because we assigned it as the Tomcat home directory.

We suppose the users of the query induction system of World Heritage have their own computers with a web browser installed. And they can access our local network. After all the installations and tests, both the Web Server and the Database Server work well. Now we have a three-tier distributed system. Since we installed the system in the Linux operating system, the prototype of the query induction system of World Heritage will run in the Linux platform. The remaining work is to implement the application of the query induction in JDBC and JSP.

The below figure is about the files scattered in the dictionary of the Web Server:

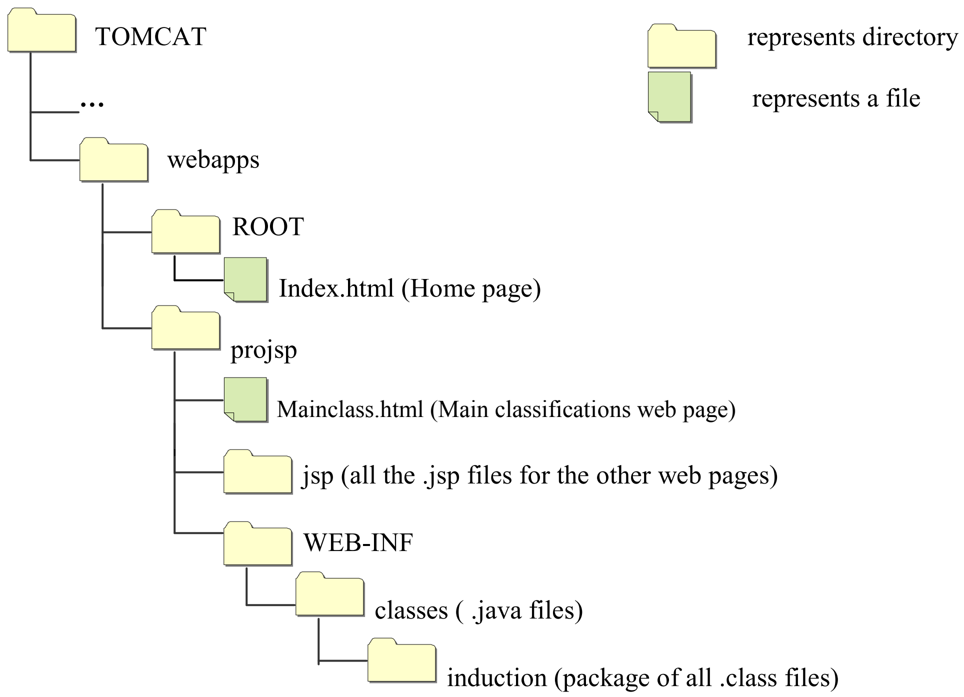

TOMCAT
...
webapps
ROOT
Index.html (Home page)
projsp
Mainclass.html (Main classifications web page)
jsp (all the .jsp files for the other web pages)
WEB-INF
classes (.java files)
induction (package of all .class files)
represents directory
represents a file

Figure 8-1: The Directory Structure of the Web Server

8.2 Explanations of the Source Code

The query induction system can be implemented by different technologies and markup languages. In the implementation of the query induction system of World Heritage, we mainly used the JSP, JDBC technologies and HTML. So we have three kinds of source code: JSP, Java and HTML. We use the JDBC in the Java files and pack all the Java classes into a package. Then in the JSP files, we import the package. We use HTML to describe the static web pages. In the following sections, we will explain the central part of the implementation. We apply the data structures and methods described in Chapter 7 to implement the query induction system.

8.2.1 Java Files

We defined the Java classes in order to implement the query induction. We defined two different classes for the concepts of the region *Category* and *Geographical Location* in the generative ontology of World Heritage. They are the class *Concept* and *Position* separately. We did not define the attributes of level and path in the class *Position* because this kind of information of the concepts in the region of *Geographical Location* is saved in the relational database of World Heritage. For the *Category* region, we implemented them in the Java class *Concept*. The instances of class *Position* could be one of the three kinds of geographical location: country, composition of continent and continent. So we have to discern them in the class *AllLocations* in order to execute Concept Generalisation on the region *Geographical Location*. We defined three vectors for the three kinds of positions. Some methods of Concept Generalisation require the system to keep the old result of Concept Generalisation. Class *Interest* was defined to keep all the old results of Concept Generalisation and other relative variables such as *repeat times* and *visiting times*. In addition, for Concept Generalisation by Weight, we had to redefine some classes because the data type of weight was changed to float type in Java. We will explain the main attributes and methods for every Java class.

The class ***Connect*** implements the connection between the Database Server and Web Server. There are four attributes in it*: username, password, url* and *con*. Through the constructor of this class, these four attributes are initialized. Four methods exist in this class.

```
class Connect
  {
     String userName;
     String password;
     String url;
     Connection con;
  }
```

- *getAttribute ( )* is used to get the concepts about *Category* or *Geographical Locations* of one site.

- *getAquery ( )* is used to get an correct query after knowing the users' interest about the *Geographical Location*.
- *getCporReg ( )* is used to submit a query about the composition of continent or continent of one site to database and return the corresponding results.
- *getIndOfresult ( )* is used to hand in the query to the database according to the result of Concept Specialisation. All the induction results will be returned by this method.

The class ***Concept*** contains the concepts of the generative ontology of World Heritage. There are four attributes of this class: *conceptname*, *level, path* and *count*. The data structure of *Concept* has been described in the data structure design (Section 7.3.1). We do not repeat it here.

```
class Concept
  {
     String  conceptname;
     int level;
     Vector path = new Vector();
     Int count;
  }
```

- *conceptname* is the concept in the generative ontology of World Heritage.
- *Getlevel( )* can obtain the level of a certain concept.

The class ***Category*** implements the region *Category* of the generative ontology of World Heritage. It has the same data structure as the *Region* we described in the Section 7.3.1. There are three attributes of the class: *categoryname, conceptnum* and *concepts.* The attribute *concepts* of the class *Category* is a vector of the objects of the class *Concept.* Some of the methods we described in the previous chapter are implemented in this class.

```
class Category
  {
     String categoryname;
     int conceptnum;
     Vector concepts = new Vector ();
  }
```

- *categoryname* represents the most generalised concept in the generative ontology of World Heritage.
- *conceptnum* is the total number of the concepts.
- *concepts* is a list of all the concepts in this Category.

- *initConcepts( )* can initialize *concepts* field.
- *sort( )* is used to sort an array.
- *getpath( )* can obtain the path of a concept.
- *Countcategory( )* is used to count the times that a certain concept covers the examples from the user.
- *Getinterest( )* is used to obtain the results of Concept Generalisation, which are the interests of the user in some concepts.
- *getRelativeInterest( )* is used to get the similiar interest when the same induction result is repeated more than 5 times.
- *Getinduce( )* is used to get the induction results by using the results of Concept Specialisation.

Class ***Position*** implements the region *Geographical Location* of the generative ontology of World Heritage. Its object could be one of country, composition of continent and continent. There are not any methods in this class. We define the class *Position* different from the class *Concept.* Since the information about the level or path of the concepts in *Geographical Location* has already been saved in the database, we don't need to redefine them in the class *Posistion.*

```
class Position
  {
     String Psname;
     int count;
  }
```

- *Psname* is the name of the Position.
- *count* is the times when the corresponding Position cover the examples from the users.

Class ***AllLocations*** is about the locations that cover the examples from the users. There are three attributes: *vCountry*, *vComposition*, *vGeography.* Each of them is a list of the objects of class *Posistion.* As mentioned before, the object of *Position* could be one of country, composition of continent and continent.

```
class AllLocations
  {
     Vector vCountry ;
     Vector vComposition;
     Vector vGeography;
  }
```

- *vCountry* represents the list of the countries (objects of class *Position*) which cover the examples from the users.

- *vComposition* is the list of the compositions of continent (objects of class *Position*) which cover the examples from the users.
- *vGeography* is the list of the continents (objects of class *Position*) which cover the examples from the users.

- *AddorCountP( )* is used to add the new *Position* or count the *Position* already existed in the corresponding *Position* Vector *vCountry*, *vComposition* or *vGeography*.
- *countPosition( )* is used to save the counts of the *Position* in an array.
- *getMostP( )* is used to obtain the *Position* which cover the examples from the users at most.
- *getInstofP( )* is used to get the result of Concept Generalisation on the *Geographical Location*. It is the user's interest of *Geographical Location*.

Class ***Interest*** is used to keep all kinds of the interests of the user during he/she visited the World Heritage site one by one, such as the interest of *Category* and the interest of *Geographical Location*. It is needed by the Similar Interest method and Concept Induction by Recent Examples method. The repeat times of the same result of Concept Generalisation and the visiting times of the users are also kept in two attributes of this class.

```
class Interest
  {
     int visitNum;
     int repNum;
     Vector interestC;
     Vector interestL;
     Vector Accinterest;
     Vector AccinterestC;
     Vector AccinterestCp;
     Vector AccinterestG;
  }
```

- *visitNum* is the number of sites that the users have visited before.
- *repNum* represents the number of same result of Concept Generalisation according to the visiting history of the users.
- *interestC* is the result of Concept Generalisation of *Category*, the interest of *Category.*
- *interestL* is the result of Concept Generalisation of *Geographical Location*, the interest of *Geographical Location*. It could be country list, composition of continent list or continent list.
- *Accinterest* is a list of the interests of *Category.*

- *AccinterestC, AccinterestCp* and *AccinterestG*, respectively, represent the list of the interests of country, composition of continent and continent.

- *getoutofOldinst( )* is used to clear the oldest result of Concept Generalisation and keep it in a Vector when the users have visited more than 10 sites.
- *GetRepeatTimes( )* is used to justify if the same induction result is repeated more than 5 times.

For Concept Generalisation by Weight, we define the other four classes: *Wconcept*, *Wcategory*, *Wpostion* and *WAllLocations*.

Class ***Wconcept*** is similar to the class *concept.* But it has the two different attributes: *weight* and *unitweight.* The *unitweight* of a concept is assigned by the users. The *weight* is the accumulated result of the *unitweight* of the corresponding concept. For *Wcategory* class, we only need to increase or decrease the value of the *weight* by the *unitweight* of the concept. So we have to change the data type into float type.

```
class Wconcept
  {
     String  conceptname;
     int level;
     Vector path = new Vector();
     float  weight;
     float unitweight;
  }
```

- *weight* is the result of accumulated *unitweight.*
- *unitweight* is the unit weight of the concept. It could be assigned by the users.

```
class Wposition
  {
     String Psname;
     float count;
  }
```

We need to define the class *WAllLoctions*, because we have to change the *Postion* objects into the *Wpostion* objects. And for the *Wpostion,* we have to change the data type of attribute *count* to float type according to the need of Concept Generalisation by Weight.

8.2.2 HTML Files and JSP Files

In this section, we will explain the HTML file in brief because it is easy to be read. We will focus on the explanation of the main JSP files of the query induction and the connection among them. The corresponding web pages of HTML and JSP files are not presented here. All the User Interfaces will be shown in the next chapter.

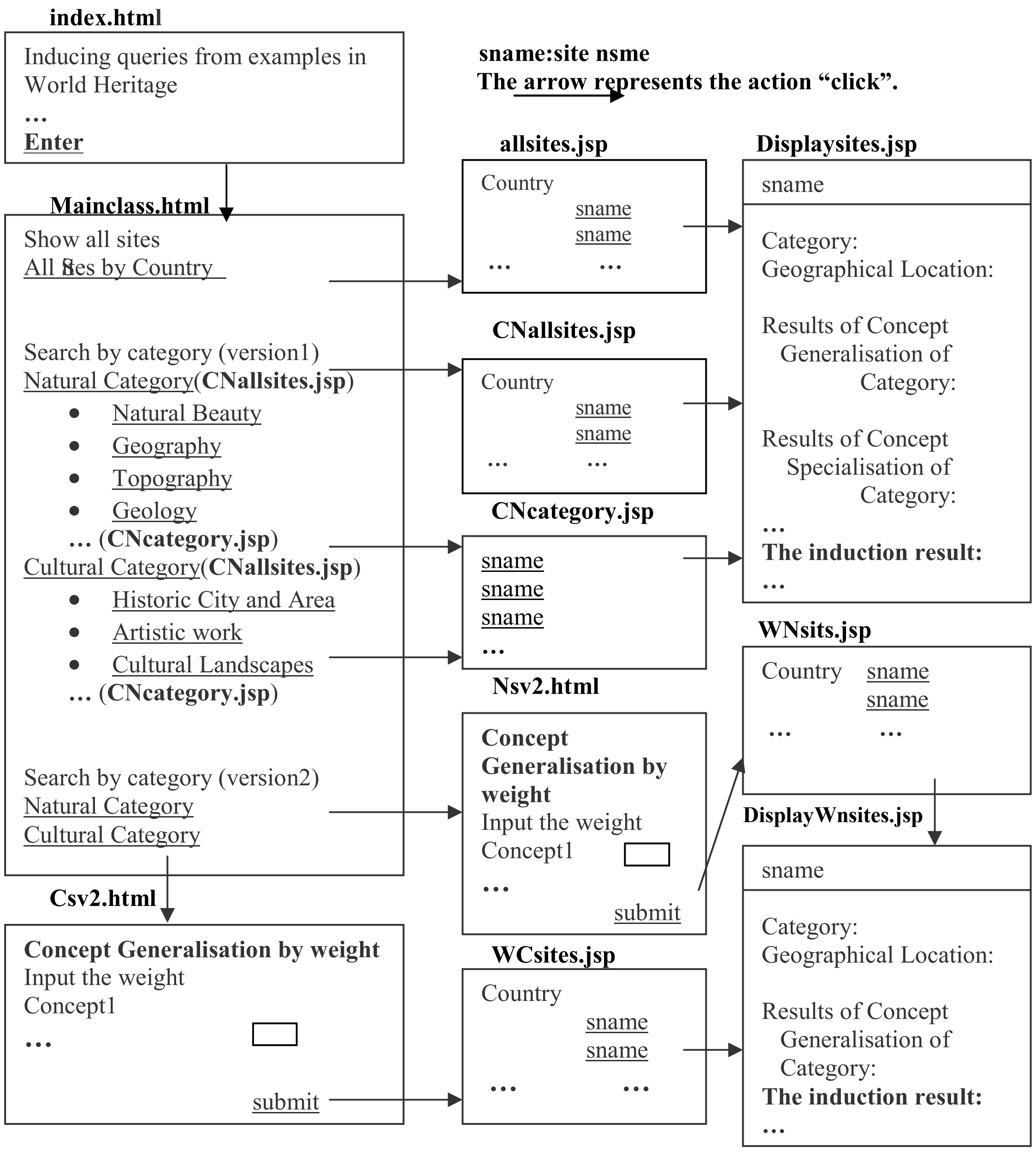


Figure 8-2: HTML and JSP Files Distribution

HTML file is only used to implement the static pages. JSP files import the Java classes described in the previous section. Then it dynamically generates the parts of the page by executing the embedded Java code and statically generates the rest of page using the HTML code. According to the interface design described in Chapter 7, we write the Page content and arrange the results produced in the process of the query induction on the suitable places of the pages using HTML. And we use the hyper-links to connect the different web pages.

We use Figure 8-2 to show how the HTML and JSP files are distributed in the system. All the HTML and JSP files in the query induction system of World Heritage are presented in the Figure 8-2. The connections among the HTML and JSP files are also shown. The arrow in the Figure 8-2 can represent the action of clicking the link. All the underlined words represent the links on the web pages. The readers can refer to the interfaces which are pasted in the next chapter to understand these web pages.

index.html is the home page of our system. We mainly give a description of this system on this page.

Mianclass.html is a web page about the main classifications of the World Heritage sites. From the link *"Enter"* on the home page, the user can enter the page of the main classifications of the World Heritage sites. Most of the links on this web page are JSP files because the system needs to retrieve the data from the database server and initialize the Hasse Diagram in the *Category* region. But this page itself is a HTML file. The two versions of *Search by Category* are presented on this page. They respectively use the Concept Generalisation by Count method and the Concept Generalisation by Weight method when the system make the induction.

allsites.jsp is called by the system when the user clicked the link "*All Sites by Country"* . The whole Hasse diagram of World Heritage will be initialized in it. It generates a web page with all sites in the different countries.

CNallsites.jsp is linked by the link *"Natural Category"* or *"Cultural Category"* under the *Search by Category (Version1).* According to the different parameter *subcategoryid* from the user requests, *CNallsites.jsp* can generate the web pages with all the natural sites or all the cultural sites respectively. And the Hasse diagram of *Natural Category* or *Cultural Category* will be initialized according to the different requests.

CNcategory.jsp is called by the system once the user clicks the links under the *Natural Category* or *Cultural Category* in the *Search by Category (Version1),* for example *"Natural Beauty"*. In this JSP file, the system gets the parameters*:* *CategoryName*, *Nodenum* and *Subcid* from the request of the user. And the corresponding Hasse Diagram of the category will be initialized. The user can see all the sites which belong to this category on the web page generated by this JSP file. Every site name will also be a link on the web pages generated from the *CNcategory.jsp*. Furthermore, if the user starts visiting the sites, *DisplySites.jsp* will be called by the system.

Nsv2.html and ***Csv2.html***are the HTML pag es which are linked by the link *"Natural Category"* or *"Cultural Category"* under the *Search by Category (Version2)* on the page of *Mainclass.html.* On these two pages, the Concept Generalisation by Weight method is introduced to the users and the important concepts about the natural category or cultural category are listed. The users can input the unit weights for concepts depending on their interests.

Displaysites.jsp is very important for the query induction because all the query induction methods are called by it and the induction results of sites are produced by it. In other words, all kinds of methods of the Java classes will be called by *Displaysites.jsp*, and the user will get a web page dynamically generated by the *Displaysites.jsp*, which contains all the useful information of the corresponding site and the corresponding induction result. We will explain the connection among *Displaysites.jsp* and the other JSP codes in detail.

Except for the links under *Search by category (version2)*, all the other links in the *Mainclass.html* page are the JSP files. They could be *allsites.jsp*, *CNallsites.jsp* or *CNcategory.jsp*. For these three JSP files, a package named *induction* is imported. The *induction* package included all the .class files that are produced by compiling the .java files. An individual session[7] will be generated when one of them is called. The corresponding instance of class *Category, AllLocations* and *Interest* will also be initialized in these three JSP files. Through the links on the web pages generated by these JSP files, *Displaysites.jsp* will be linked. *Displaysites.jsp* also imports the package *induction* in order to call all the methods in the Java classes. Via the session, the initialized Java instances are passed to the *Displaysites.jsp.* Then in the *Displaysites.jsp,* all the methods from the Java classes will be called. Then these methods can implement the query induction from the user's examples of World Heritage.

The below figure presents the connection among the other JSP codes and *Displaysites.jsp*. The Java methods called by *Displaysites.jsp* and the executions of them also are shown in the Figure 8-3.

[7] Session is a hidden object of JSP, an instance of the class javax.servlet.http.HttpSeession. Trough setting the session attribute in the page as "true", the session can be built.

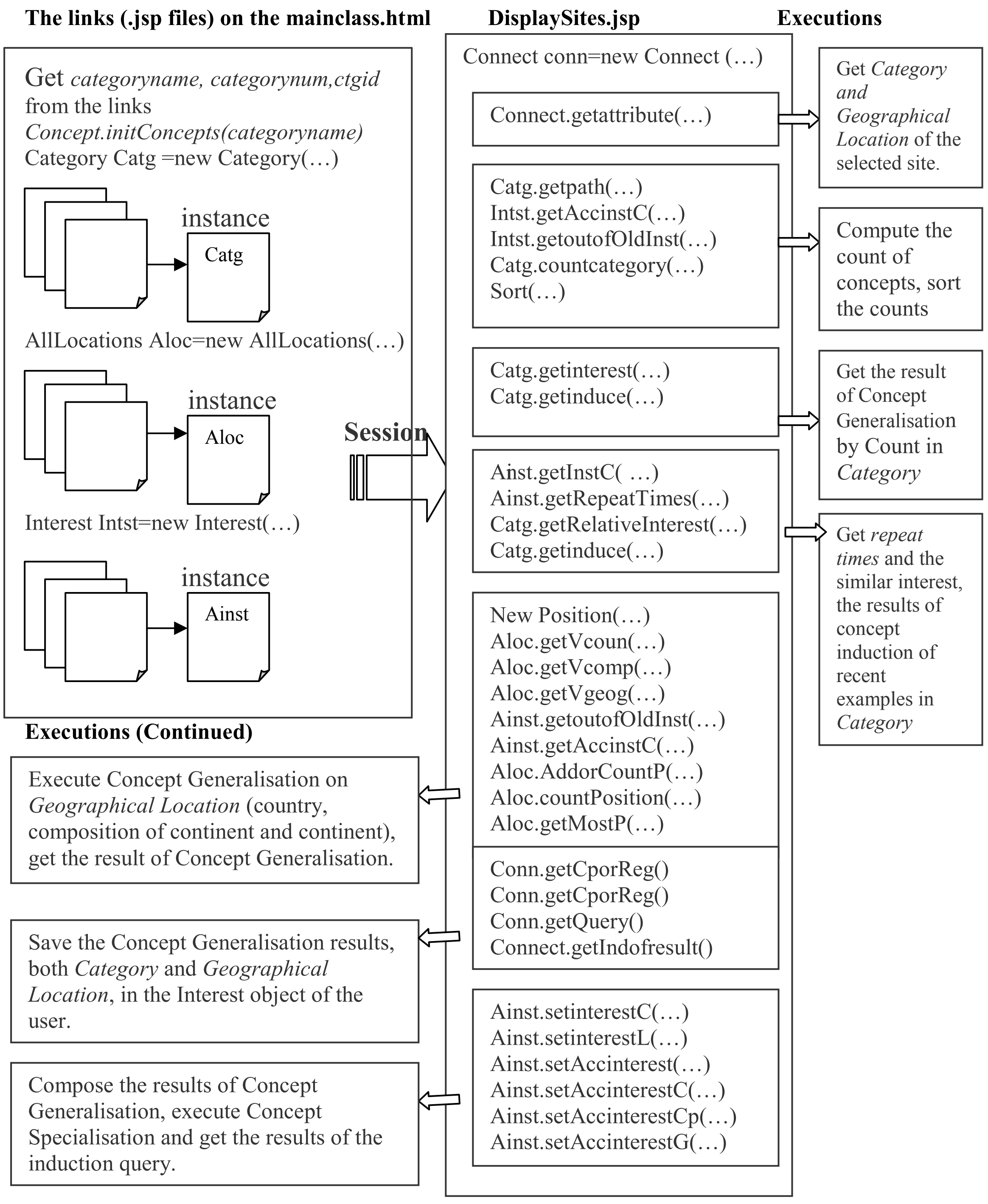

The links (.jsp files) on the mainclass.html
DisplaySites.jsp
Executions
Get *categoryname, categorynum,ctgid* from the links
Concept.initConcepts(categoryname)
Category Catg =new Category(…)
instance
Catg
AllLocations Aloc=new AllLocations(…)
instance
Aloc
Interest Intst=new Interest(…)
instance
Ainst
Session
Connect conn=new Connect (…)
Connect.getattribute(…)
Catg.getpath(…)
Intst.getAccinstC(…)
Intst.getoutofOldInst(…)
Catg.countcategory(…)
Sort(…)
Catg.getinterest(…)
Catg.getinduce(…)
Ainst.getInstC(…)
Ainst.getRepeatTimes(…)
Catg.getRelativeInterest(…)
Catg.getinduce(…)
New Position(…)
Aloc.getVcoun(…)
Aloc.getVcomp(…)
Aloc.getVgeog(…)
Ainst.getoutofOldInst(…)
Ainst.getAccinstC(…)
Aloc.AddorCountP(…)
Aloc.countPosition(…)
Aloc.getMostP(…)
Conn.getCporReg()
Conn.getCporReg()
Conn.getQuery()
Connect.getIndofresult()
Ainst.setinterestC(…)
Ainst.setinterestL(…)
Ainst.setAccinterest(…)
Ainst.setAccinterestC(…)
Ainst.setAccinterestCp(…)
Ainst.setAccinterestG(…)
Get *Category and Geographical Location* of the selected site.
Compute the count of concepts, sort the counts
Get the result of Concept Generalisation by Count in *Category*
Get *repeat times* and the similar interest, the results of concept induction of recent examples in *Category*
Executions (Continued)
Execute Concept Generalisation on *Geographical Location* (country, composition of continent and continent), get the result of Concept Generalisation.
Save the Concept Generalisation results, both *Category* and *Geographical Location*, in the Interest object of the user.
Compose the results of Concept Generalisation, execute Concept Specialisation and get the results of the induction query.

Figure 8-3: The Executions of Displaysites.jsp

In the Figure 8-3, an instance of *Category*: *Catg,* an instance of *AllLocations***:** *Aloc* and an instance of *Interest*: *AInst* are generated in *allsites.jsp*, *CNallsites.jsp* or *CNcategory.jsp*. Using the method of session: *session.setAttribute ()*, these three instances are included in the session. In *Displaysites.jsp*, *session.getAttribute()* method is used to get these three instance. We have not written the specific instructions in the figure. We only expressed the pass process of the instances by session. Through the session, the system can server the multi-users at the same time because the different session will be built for the different users. After the three instances are obtained by *Displaysites.jsp*, the corresponding Java methods are called by it. We describe the executions of these methods in the Figure 8-3.

Because of the limitation of space, we have to omit the parameters of the methods in the above figure. But the order of executions obeys the steps of query induction.

WNsites.jsp **and** ***WCsites.jsp*** are linked by submitting the form on the *NSv2.html* or *Csv2.html* separately. A list of the World Heritage sites ordered by the countries is generated by these two JSP files. The sites names are links again, which can link to *DisplayNWsites.jsp.*

DisplayNWsites**.jsp** is similar to *Displaysites.jsp*. And for the Version2, the relationships between the *WNsites.jsp* (or *WCsites.jsp*) and *DisplayNWsites.jsp* and the execution of *DisplayNWsites.jsp* are very similar to the Version1. We do not explain them again.

8.3 Conclusion

In this chapter, we introduced the system implementation and explained the central part in detail. Here we have not included the parameters of the methods in the Java classes, but the reader can refer to the source code in the appendix. After we implemented the query induction system of World Heritage, we will test whether the system can work well and whether the applications of query induction are correct. The process of the test will be presented in the next chapter.

9 System Test

In this chapter, we will present the system test after the query induction system of World Heritage is implemented. The system test is divided into two parts: Component Test and Integration Test. Here we mainly focus on testing the induction query applications because we have already tested the Web Server and the Database Server when we installed them.

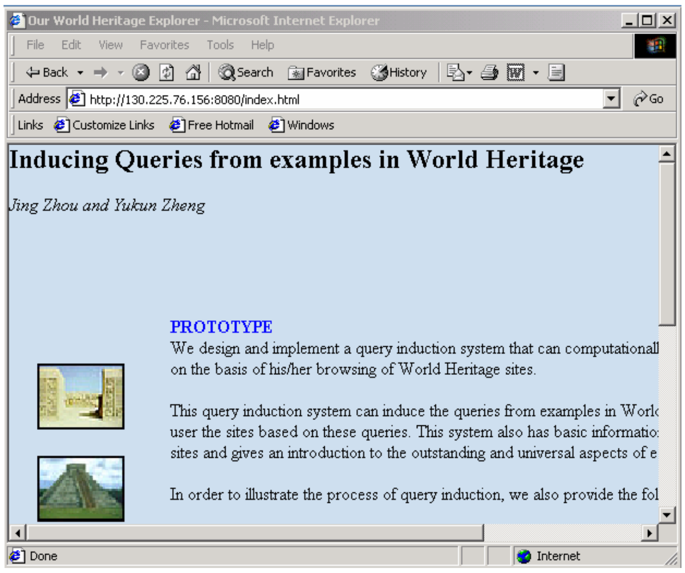
9.1 Component Test

During the implementation of the system, we tested the different source codes respectively. For HTML files, we used the internet browser to test them. For all Java source files, first we compiled them in the J2SDK environment. After they were successfully compiled, we got the corresponding *.class* flies. We continued to check all the methods of these classes by calling them in a main class. After we ensured all the methods execute the correct logic of the query induction, we packed all the Java classes into a Java package named *induction*. This package is imported in the JSP files which are used to implement the query induction. In fact, the Java source codes and JSP files could be tested at the same time because JSP files are the HTML files which are embedded with Java code. We can use the out.println() method of JSP to print the results of executing Java code on the web pages. So we can know the execution of the Java code. However, debugging the JSP programs is more difficult than the Java programs. So testing the Java code first can make the test of JSP files easily. For the JSP files, we only need to use the internet browser to check the content of the page generated by the JSP files since we have tested the Java code before. In order to make the users know the process of the query induction clearly, we still output the results produced in the process of the query induction on the web pages in our system.

9.1.1 Test the HTML Files

1. **Test *index.html.***

According to the source code of *index.html*, the system should show the users the description of the query induction system of World Heritage and the explanation of the results which are produced by the query induction on this page. There is an *"Enter"* link on it. Through the web browser, we can see this page looks like below:


Our World Heritage Explorer - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://130.225.76.156:8080/index.html
Go
Links Customize Links Free Hotmail Windows
Done
Internet

Inducing Queries from examples in World Heritage

Jing Zhou and Yukun Zheng

PROTOTYPE
We design and implement a query induction system that can computationall
on the basis of his/her browsing of World Heritage sites.

This query induction system can induce the queries from examples in Worl
user the sites based on these queries. This system also has basic informatio
sites and gives an introduction to the outstanding and universal aspects of e

In order to illustrate the process of query induction, we also provide the fol

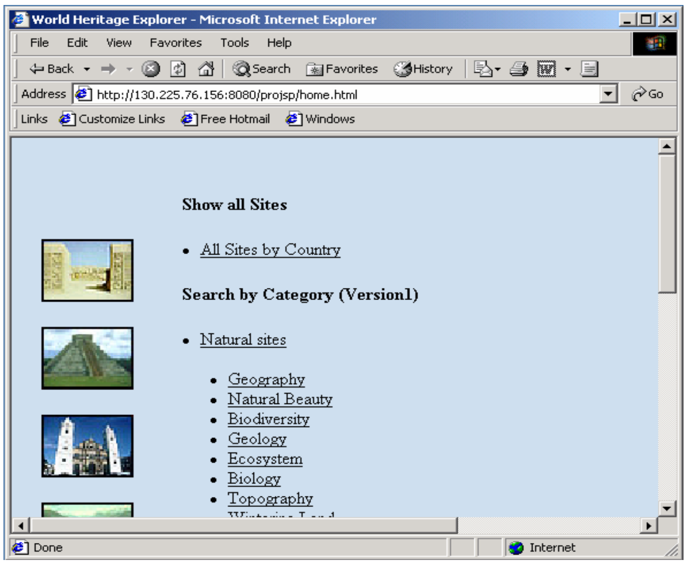
Figure 9-1: Page of index.html

2. **Test *Mainclass.html*.**

According to the source code of Mainclass.html, the system should show the users three groups:

- Search all sites. There is a link following it, which is "*All Sites by country*".
- Search by Category (Version1). There are the different classifications of the Category of World Heritage in Natural sites and Cultural sites. And every classification should be a link on this page. In this version, the Concept Generalisation by Count method is adopted in the query induction.
- Search by Category (Version 2). There are only two links, *"Natural Category"* and *"Cultural Category"*. In this version, the Concept Generalisation by Weight method is adopted in the query induction.

Through the web browser, the page is shown as following:


World Heritage Explorer - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://130.225.76.156:8080/projsp/home.html
Go
Links Customize Links Free Hotmail Windows
Done
Internet

Show all Sites

- All Sites by Country

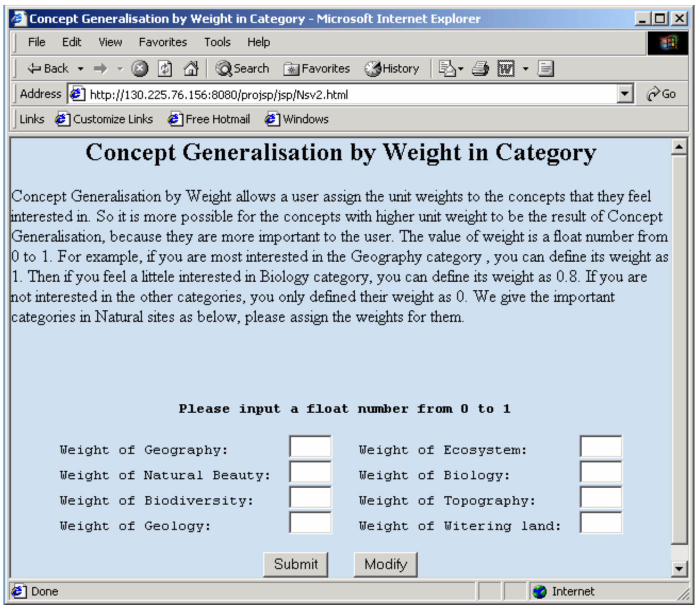
Search by Category (Version1)

- Natural sites
  - Geography
  - Natural Beauty
  - Biodiversity
  - Geology
  - Ecosystem
  - Biology
  - Topography

Figure 9-2: Page of Mainclass.html

3. **Test *Nsv2.html* and *Csv2.html.***

Concept Generalisation by Weight method is introduced on the pages of *Nsv2.html* and *Csv2.html*. These two HTML files show a form for the users. The users can fill in them depending on their interests of the different categories. On the page of *Nsv2.html*, 8 important concepts in the natural category are listed in the form. On the page of *Csv2.html*, 6 important concepts in the cultural category are listed in the form. Click the *"Submit"* button, all the unit weights assigned by the users will be passed to *WNsite.jsp* or *WCsites.jsp*. Using the *"Modify"* button, the users can modify their assignments of the unit weights. Through the web browser, we can see the page of *Nsv2.html* likes below (the page of *Csv2.html* is similar; we do not show it here).


Concept Generalisation by Weight in Category - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://130.225.76.156:8080/projsp/jsp/Nsv2.html
Links Customize Links Free Hotmail Windows
Submit
Modify
Done
Internet

Concept Generalisation by Weight in Category

Concept Generalisation by Weight allows a user assign the unit weights to the concepts that they feel interested in. So it is more possible for the concepts with higher unit weight to be the result of Concept Generalisation, because they are more important to the user. The value of weight is a float number from 0 to 1. For example, if you are most interested in the Geography category , you can define its weight as 1. Then if you feel a littele interested in Biology category, you can define its weight as 0.8. If you are not interested in the other categories, you only defined their weight as 0. We give the important categories in Natural sites as below, please assign the weights for them.

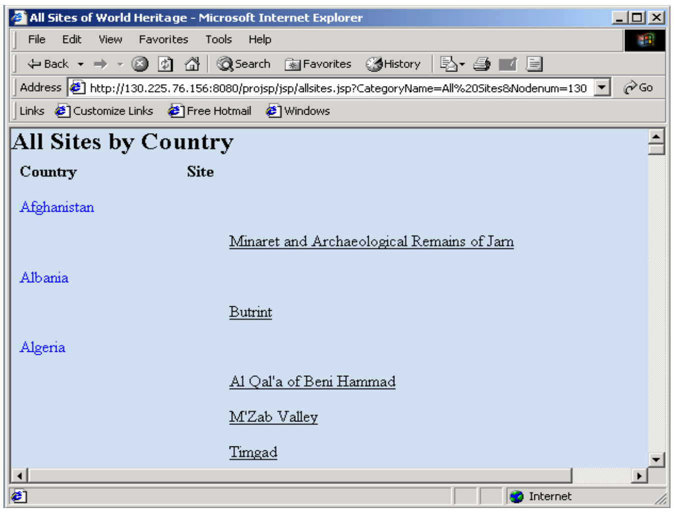
Please input a float number from 0 to 1

Weight of Geography:		Weight of Ecosystem:	
Weight of Natural Beauty:		Weight of Biology:	
Weight of Biodiversity:		Weight of Topography:	
Weight of Geology:		Weight of Witering land:	

Figure 9-3: Page of Nsv2.html

9.1.2 Test the Java Files and JSP Files

1. **Test *allsites.jsp*.**


All Sites of World Heritage - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://130.225.76.156:8080/projsp/jsp/allsites.jsp?CategoryName=All%20Sites&Nodenum=130
Links Customize Links Free Hotmail Windows
Internet

All Sites by Country

Country	Site
Afghanistan	
	Minaret and Archaeological Remains of Jam
Albania	
	Butrint
Algeria	
	Al Qal'a of Beni Hammad
	M'Zab Valley
	Timgad

Figure 9-4: Page Generated by allsites.jsp

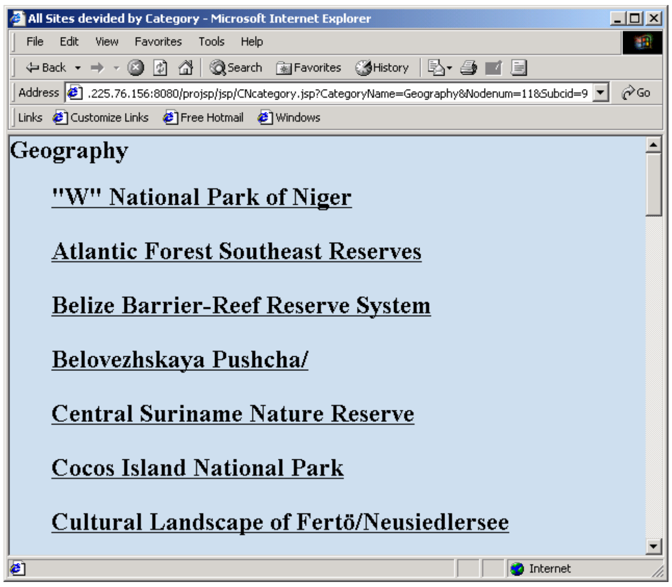
The system is linked to *allsites.jsp* by the link *"All Sites by Country"* on the *Maiclass.html* page. All the Java classes and the Hasse diagram of the generative ontology of World Heritage are initialized in this file. And the session will be built for pass the corresponding instances of Java classes to the *Displaysites.jsp*. In the page generated by the *allsite.jsp,* the countries are ordered by the alphabetically on the left side, and the corresponding World Heritage sites are listed on the right side. Through the web browse, the users will see the page of *allsites.jsp* as above.

2. **Test *CNallsites.jsp***.

CNallsites.jsp is very similar to the *allsites.jsp*. The system is linked to it by the links of *"Natural Category"* or *"Cultural Category"* under *Search by Category (Version1)*. Accordingly, a list of all the natural sites with their countries can be shown from the link of *"Natural Category"*. A list of all the cultural sites with their countries can be shown from the link of "*Cultural Category"*. The page generated by the *CNallsites.jsp* looks just like the Figure 9-4, here we will not show it.

3. **Test *CNcategory.jsp*.**

The system is linked to *CNcategory.jsp* by all links of the categories under *Natural Category* or *Cultural Category* such as *Geography, Historic City and Area,* and *Construction*. The Hasse diagram of the corresponding category will be initialized in *CNcategory.jsp*. And the session will be built in order to pass the corresponding Java instances to the *Displaysites.jsp*. On the page generated by the *CNcategory.jsp*, all the sites that belong to the entry category will be listed.


All Sites devided by Category - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address
Go
Links Customize Links Free Hotmail Windows
Internet

.225.76.156:8080/projsp/jsp/CNcategory.jsp?CategoryName=Geography&Nodenum=11&Subcid=9

Geography

"W" National Park of Niger

Atlantic Forest Southeast Reserves

Belize Barrier-Reef Reserve System

Belovezhskaya Pushcha/

Central Suriname Nature Reserve

Cocos Island National Park

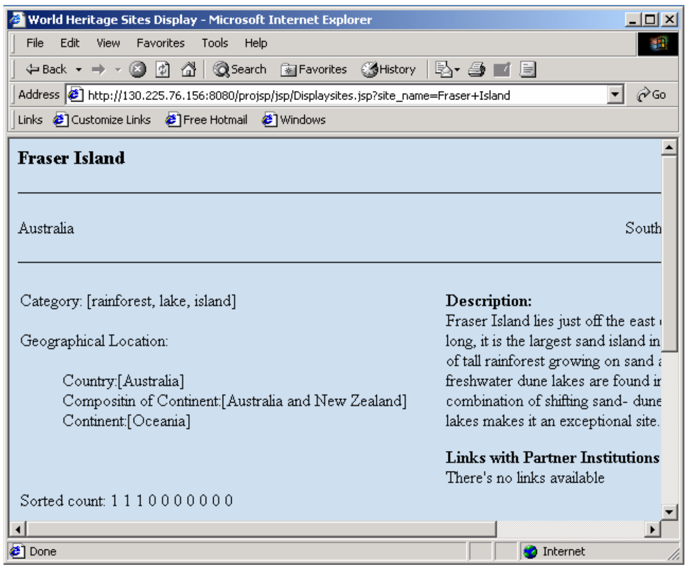
Cultural Landscape of Fertő/Neusiedlersee

Figure 9-5: Page Generated by CNcategory.jsp for Geography

4. **Test *Displaysites.jsp***

The system is linked to *Displaysites.jsp* by the links of sites on the pages generated by the *allsites.jsp*, *CNallsites.jsp* and *CNcategory.jsp* In this file, the instance of Java classes from the *allsites.jsp*, *CNallsites.jsp* or *CNcategory.jsp* will be obtained via the session. And all the methods of Java classes will be called. If they can't work as we supposed, the content of its corresponding page will be wrong or the system will give the error messages if the file failed in compiling. Even if the content of the page can be presented, we still need to check if the logic errors exist in these methods although we have checked the logic of Java codes before. For example, we can check further if the results produced in the process of the query induction make sense. If we can avoid all the mistakes above, the users should see the information of the selected site and the induction results. The induction results are a list of the sites recommended by the system. In order to avoid repeating, here we only test the page content. The description of testing the results produced in the process of query induction will place in the corresponding part of Integration Test.

On the page generated by the *Displaysites.jsp,* all kinds of results produced in the process of the query induction are listed on the left side, and the description of the site is listed on the right side. On the up side of the page, the site name and country will be shown. Because of the limitation of the page size, we show the page generated by *Displaysites.jsp* for the site *"Fraser Island"* in two figures.


World Heritage Sites Display - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address
Go
Links Customize Links Free Hotmail Windows
Done
Internet

http://130.225.76.156:8080/projsp/jsp/Displaysites.jsp?site_name=Fraser+Island

Fraser Island

Australia South

Category: [rainforest, lake, island]

Geographical Location:

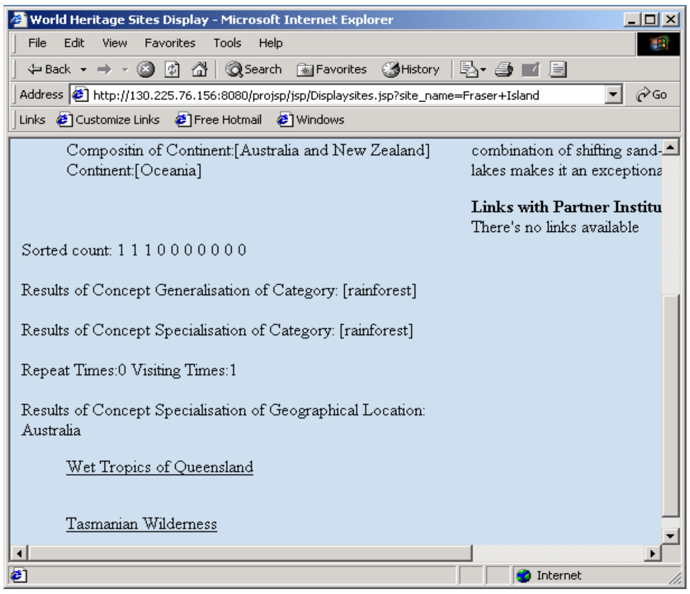
Country:[Australia]
Compositin of Continent:[Australia and New Zealand]
Continent:[Oceania]

Sorted count: 1 1 1 0 0 0 0 0 0 0

Description:
Fraser Island lies just off the east
long, it is the largest sand island in
of tall rainforest growing on sand
freshwater dune lakes are found ir
combination of shifting sand- dune
lakes makes it an exceptional site.

Links with Partner Institutions
There's no links available

Figure 9-6: First Part of the Page Generated by Displaysites.jsp


World Heritage Sites Display - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://130.225.76.156:8080/projsp/jsp/Displaysites.jsp?site_name=Fraser+Island
Go
Links Customize Links Free Hotmail Windows
Internet

Compositin of Continent:[Australia and New Zealand]
Continent:[Oceania]

combination of shifting sand-
lakes makes it an exceptiona

Links with Partner Institu
There's no links available

Sorted count: 1 1 1 0 0 0 0 0 0 0

Results of Concept Generalisation of Category: [rainforest]

Results of Concept Specialisation of Category: [rainforest]

Repeat Times:0 Visiting Times:1

Results of Concept Specialisation of Geographical Location:
Australia

Wet Tropics of Queensland

Tasmanian Wilderness

Figure 9-7: Second Part of the Page Generated by Displaysites.jsp

5. **Test *WNsites.jsp* and *WCsites.jsp***

The system is linked to these two files by handing in the forms on the pages of *Nsv2.html* and *Csv2.html* separately. When the users submit the form in *Nsv2.html* or *Csv2.html*, the unit weights assigned by the users are passed to the *WNsites.jsp* or *WCsites.jsp*. And the pages generated by *WNsites.jsp* or *WCsites.jsp* respectively will be shown to the users. These two pages look just like the page generated by *allsites.jsp* (see Figure9-3). The Hasse diagrams of the corresponding *Natural Category* or *Cultural Category* are initialized in these two files. And the sessions will be built for passing the corresponding parameters to the *DisplayWNsites.jsp*.

6. **Test *DisplayWNsites.jsp***

The functions of *DisplayWNsites.jsp* and the web page generated by it are quite similar to the *Displaysites.jsp*. Through running this file, the induction result of Concept Generalisation by Weight can be obtained.

Although we have not pasted all the interfaces of HTML or generated by the JSP files here, we have tested all of them. And results of Component Test indicate these individual parts of the system work well.

9.2 Integration Test

After the component test, we know all the components can work well. Then we will do the Integration Test in order to find whether all the components can work together well. We select some special cases to test the system. We will first test the transfers among the links and then test the results produced by all kinds of methods of induction query, such as Similar Interest and Concept Induction by Recent Examples. The process of the integration test will present in the following four test branches. In order to save the space, we will only show the web pages dynamically generated by the *Displaysites.jsp* or *DisplayWnsites.jsp.* For the other web pages, the reader can refer to the web pages shown in the previous section.

9.2.1 First Branch of Integration Test

In this branch, we will test the "*Enter*" link, "*All sites by country*" link and the links on the page generated by the *allsites.jsp* and the induction results on the page dynamically generated by the *Displaysites.jsp*

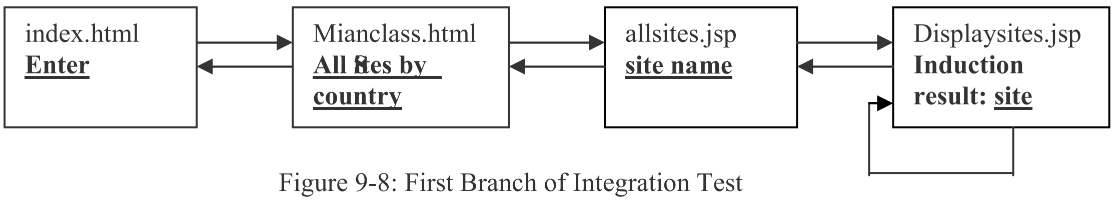
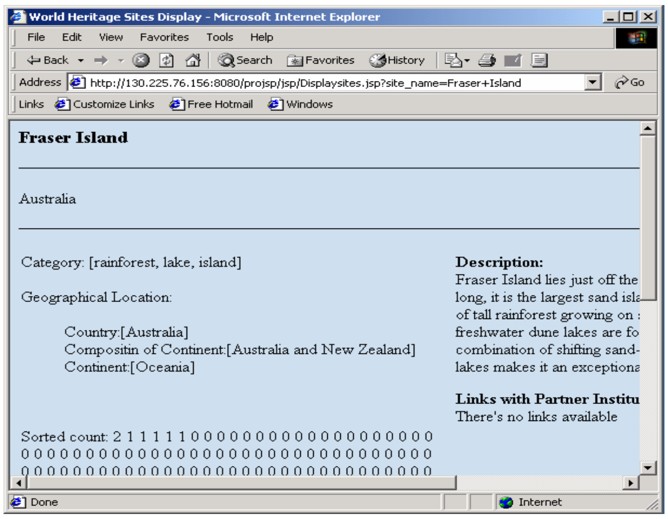


Figure 9-8: First Branch of Integration Test

When we input *http://130.225.76.156: 8080/* in *IE* web browser, we see the page of *index.html* (Figure 9-1). Then we click the *Enter* link, the page of *Mainclass.html* (Figure 9-2) is shown to us. Then we click the link of *All Sites by Country*, we see a page generated by the *allsites.jsp*(see Figure 9-3). We select a site named *Fresher Island* which is a link on the page generated by the *allsites.jsp* (Figure 9-4). Then we get the following web page dynamically generated by the *Displaysites.jsp.* These two figures look same as Figure 9-6 and 9-7 because all of them are the web pages about the site *Fraser Island*. The following figures are obtained by the link of *"Fraser Island"* on the page generated by *allsites.jsp*.


World Heritage Sites Display - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://130.225.76.156:8080/projsp/jsp/Displaysites.jsp?site_name=Fraser+Island
Go
Links Customize Links Free Hotmail Windows
Done
Internet

Fraser Island

Australia

Category: [rainforest, lake, island]

Geographical Location:

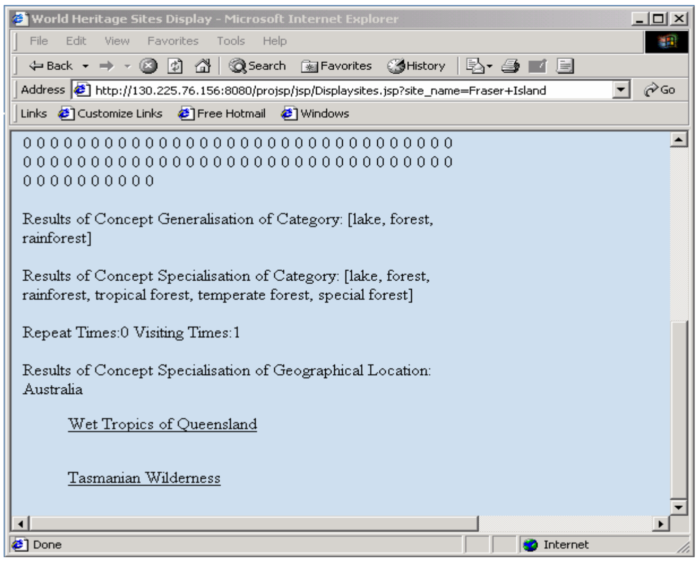
Country:[Australia]
Compositin of Continent:[Australia and New Zealand]
Continent:[Oceania]

Sorted count: 2 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Description:
Fraser Island lies just off the
long, it is the largest sand isla
of tall rainforest growing on
freshwater dune lakes are fo
combination of shifting sand-
lakes makes it an exceptiona

Links with Partner Institu
There's no links available

Figure 9-9: First Part of the Page Generated by Displaysites.jsp


World Heritage Sites Display - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://130.225.76.156:8080/projsp/jsp/Displaysites.jsp?site_name=Fraser+Island
Go
Links Customize Links Free Hotmail Windows
Done
Internet

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0

Results of Concept Generalisation of Category: [lake, forest, rainforest]

Results of Concept Specialisation of Category: [lake, forest, rainforest, tropical forest, temperate forest, special forest]

Repeat Times:0 Visiting Times:1

Results of Concept Specialisation of Geographical Location: Australia

Wet Tropics of Queensland

Tasmanian Wilderness

Figure 9-10: Second Part of the Page Generated by Displaysites.jsp

On the up part of this page (Figure 9-9), we can see the site name: *Fraser Island*, the country of this site: *Australia*. On the right part of this page, we see a simple description of the *Fraser Island.* These data are obtained from the relational database of World Heritage by the system. On the left part of this page, there is a list of the results which are produced during the process of query induction. We have already explained every item on the home page *index.html*. Let us check the correctness of them now.

- ✓ Category: [rainforest, lake, island]
- ✓ Geographical Location:

Country: Australia
Composition of Continent: Australia and New Zealand
Continent: Oceania

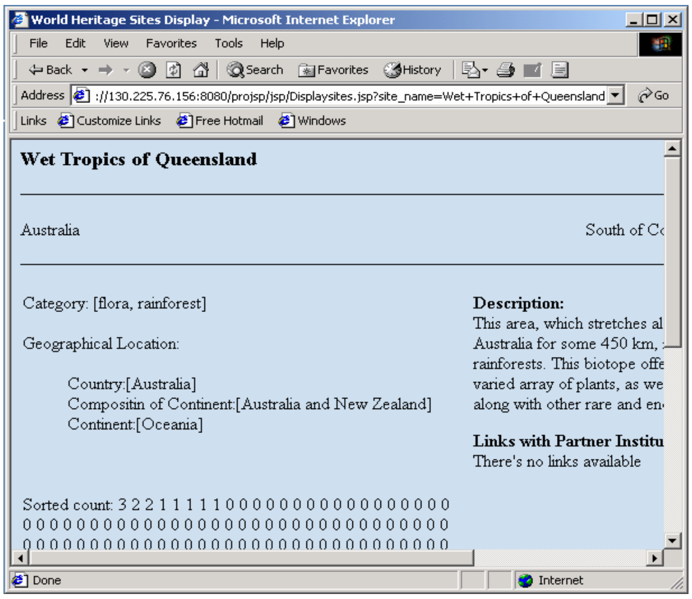
- ✓ Sorted Count: 21111100000…
- ✓ Results of Concept Generalisation of Category: [lake, forest, rainforest]
- ✓ Results of Concept Specialisation of Category: [lake, forest, rainforest, tropical forest, temperate forest, special forest]
- ✓ Repeat Times: 0 Visiting Times: 1
- ✓ Results of Concept Generalisation of Geographical Location: [Australia]
- ✓ Results of Induction:

  Wet Tropics of Queensland
  Tasmanian Wilderness

According to the generative ontology of World Heritage and the data obtained from the relational database of World Heritage, we know the *Fraser Island* site belongs to the categories: *rainforest, lake* and *island*. And it is located in Australia. According the geographical knowledge, we know Australia is located in Australia and New Zealand, and Australia and New Zealand is a part of Australia continent. Since the whole Hasse Diagram of the generative ontology of World Heritage is initialized in *allsites.jsp,* 130 concepts' *count* field will be used to execute Concept Generalisation by using the Concept Generalisation by Count method. Refer to the methods we have designed, we know the largest *count* belong to the *Geography* concept, which is 2. The *count* of concepts: *rainforest, forest, lake, island, Geology* are 1. Since the *rainforest, forest, lake* are the specialised concepts under the *Geography* concept, they become the result of Concept Generalisation. Furthermore, the result of Concept Specialisation is *[lake, forest, rainforest, tropical forest, temperate forest, special forest]*. In the concepts of *Geographical Location, Australia* is the most specialised concept, so it becomes the result of Concept Specialisation of *Geographical Location*. Therefore the induction query is the site which is located in *Australia* and it belongs to the category: *lake, forest, rainforest, tropical forest, temperate forest or special forest.* Using this query, we get two recommended sites: *Wet Tropics of Queensland* and *Tasmanian Wilderness*.

If we are really interested in the result of query induction above, we can continue to visit the recommended sites: *Wet Tropics of Queensland* and *Tasmanian Wilderness*. We select the site: *Wet Tropics of Queensland,* then we can see a new page which is also dynamically generated by the *Diaplaysites.jsp*. The page of the site *Wet Tropics of Queensland* is shown in the following figures. We also check the correctness of the results from the process of the induction query. They are correct.

We also tested some other sites on the page generated by the *allsites.jsp*, we found the query induction can be executed correctly. And we can return to the home page from the page generated by the *Displaysites.jsp*. So the results of the test indicate the system can execute this branch correctly.


World Heritage Sites Display - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address ://130.225.76.156:8080/projsp/jsp/Displaysites.jsp?site_name=Wet+Tropics+of+Queensland
Go
Links Customize Links Free Hotmail Windows
Done
Internet

Wet Tropics of Queensland

Australia South of Co

Category: [flora, rainforest]

Geographical Location:

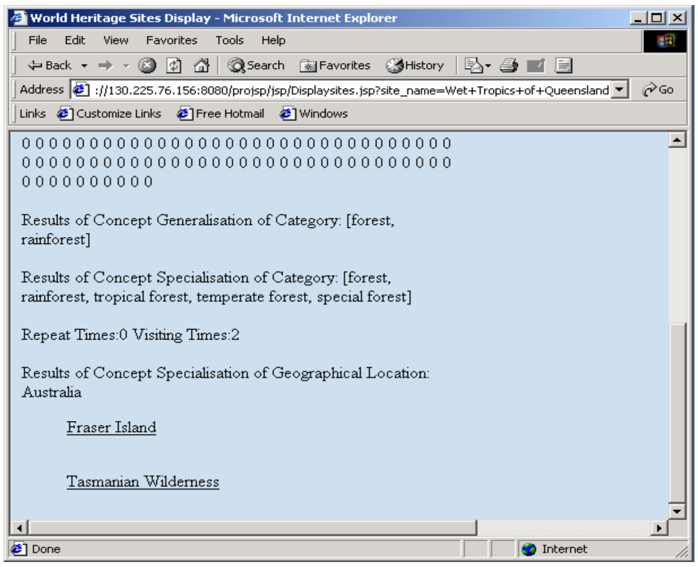
Country:[Australia]
Compositin of Continent:[Australia and New Zealand]
Continent:[Oceania]

Description:
This area, which stretches al
Australia for some 450 km,
rainforests. This biotope offe
varied array of plants, as we
along with other rare and en

Links with Partner Institu
There's no links available

Sorted count: 3 2 2 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Figure 9-11: First Part of Page of Wet Tropics of Queensland


World Heritage Sites Display - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address ://130.225.76.156:8080/projsp/jsp/Displaysites.jsp?site_name=Wet+Tropics+of+Queensland
Go
Links Customize Links Free Hotmail Windows
Done
Internet

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0

Results of Concept Generalisation of Category: [forest, rainforest]

Results of Concept Specialisation of Category: [forest, rainforest, tropical forest, temperate forest, special forest]

Repeat Times:0 Visiting Times:2

Results of Concept Specialisation of Geographical Location: Australia

Fraser Island

Tasmanian Wilderness

Figure 9-12: Second Part of Page of Wet Tropics of Queensland

9.2.2 Second Branch of Integration Test

In this breach of test, we omit the test of the link form the *index.html* to the *Mainclass.html* because we have tested it in the last branch. We will test the link of *"Natural sites",* the links on the page generated by *CNcategory.jsp* and the induction results on the page dynamically generated by the *Displaysites.jsp.* Here we also test the *Similar Interest* method of Concept Generalisation.

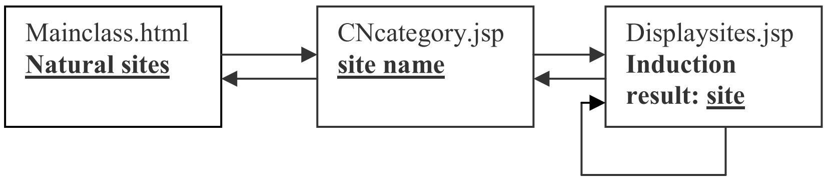
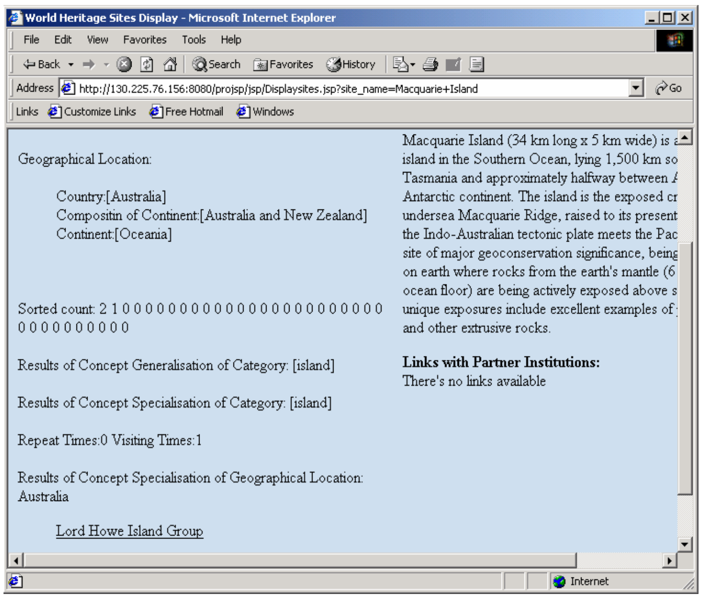


Figure 9-13: Second Branch of Integration Test

We click the link "*Natural Category"* on the page of Maiclass.html, and then the page generated by the *CNcategory.jsp* is shown to us. From the Similar Interest method, we know the system will recommend some similar interests to us when we get the same result of Concept Generalisation more than 5 times. We try to visit some sites about the category *island* in order to test the *Similar Interest* method of Concept Generalisation.

We visited the following sites: *Macquarie Island, Lord Howe Island group, Fraser Island, Cocos Island National Park,* and *Galapagos Islands.* When we visit the 6th *Isole Eolie* site, the *Similar Interest* method of Concept Generalisation will called. Then we will check the result of Concept Generalisation, the result of Concept Specialisation, and the induction results. For the first five sites, the result of Concept Generalisation is same, which is *island*. So the result of Concept Specialisation is also same because the concept *island* is a specialised concept in the region *Category* of the generative ontology of World Heritage. Since three of the sites are located in *Australia*, so it is the result Concept Specialisation of *Geographical Location.* Therefore the induction result of the first 5 sites is the island in the Australia. And the island as the result of Concept Generalisation gets 5 times. When we visit the 6th site, the *Similar Interest* method is called, we obtain the similar categories of *island*: *fossil, volcano, geomorphology, reef, cliff, glaciers*. We present the first site and last two sites' web pages generated by the *Displaysites.jsp* as below.


World Heritage Sites Display - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://130.225.76.156:8080/projsp/jsp/Displaysites.jsp?site_name=Macquarie+Island
Go
Links Customize Links Free Hotmail Windows
Internet

Geographical Location:

Country:[Australia]
Compositin of Continent:[Australia and New Zealand]
Continent:[Oceania]

Sorted count: 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0

Results of Concept Generalisation of Category: [island]

Results of Concept Specialisation of Category: [island]

Repeat Times:0 Visiting Times:1

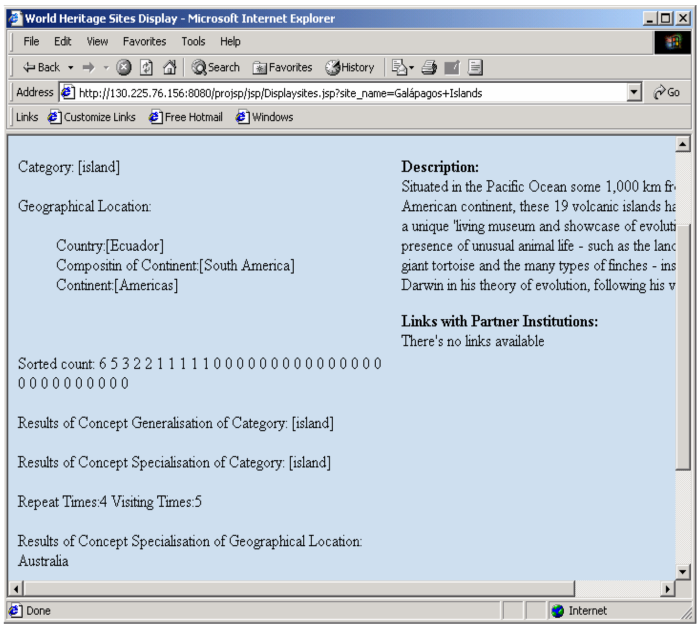
Results of Concept Specialisation of Geographical Location:
Australia

Lord Howe Island Group

Macquarie Island (34 km long x 5 km wide) is a
island in the Southern Ocean, lying 1,500 km so
Tasmania and approximately halfway between
Antarctic continent. The island is the exposed cr
undersea Macquarie Ridge, raised to its present
the Indo-Australian tectonic plate meets the Pac
site of major geoconservation significance, being
on earth where rocks from the earth's mantle (6
ocean floor) are being actively exposed above s
unique exposures include excellent examples of
and other extrusive rocks.

Links with Partner Institutions:
There's no links available

Figure 9-14: Page of Macquarie Island


World Heritage Sites Display - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://130.225.76.156:8080/projsp/jsp/Displaysites.jsp?site_name=Galápagos+Islands
Go
Links Customize Links Free Hotmail Windows
Done
Internet

Category: [island]

Geographical Location:

Country:[Ecuador]
Compositin of Continent:[South America]
Continent:[Americas]

Sorted count: 6 5 3 2 2 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0

Results of Concept Generalisation of Category: [island]

Results of Concept Specialisation of Category: [island]

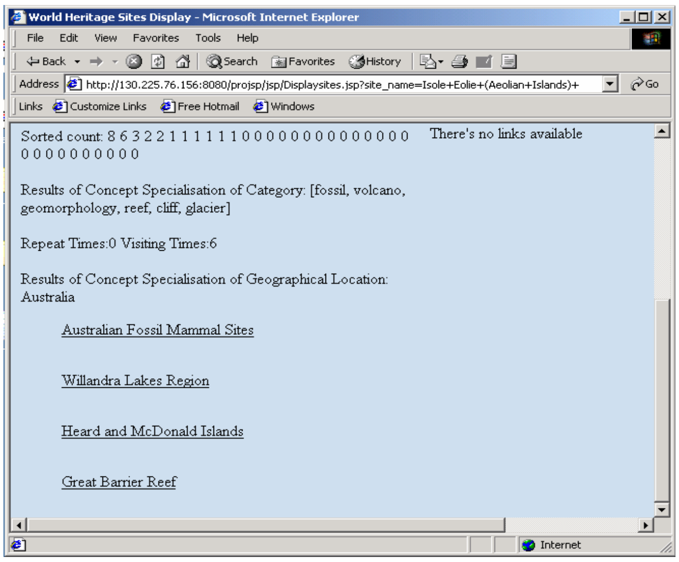
Repeat Times:4 Visiting Times:5

Results of Concept Specialisation of Geographical Location:
Australia

Description:
Situated in the Pacific Ocean some 1,000 km fr
American continent, these 19 volcanic islands ha
a unique 'living museum and showcase of evoluti
presence of unusual animal life - such as the lanc
giant tortoise and the many types of finches - ins
Darwin in his theory of evolution, following his v

Links with Partner Institutions:
There's no links available

Figure 9-15: Page of Galapagos Islands



World Heritage Sites Display - Microsoft Internet Explorer

Address http://130.225.76.156:8080/projsp/jsp/Displaysites.jsp?site_name=Isole+Eolie+(Aeolian+Islands)+

```
Sorted count: 8 6 3 2 2 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0    There's no links available
0 0 0 0 0 0 0 0 0 0

Results of Concept Specialisation of Category: [fossil, volcano,
geomorphology, reef, cliff, glacier]

Repeat Times:0 Visiting Times:6

Results of Concept Specialisation of Geographical Location:
Australia

    Australian Fossil Mammal Sites

    Willandra Lakes Region

    Heard and McDonald Islands

    Great Barrier Reef
```

File Edit View Favorites Tools Help
Back Search Favorites History
Links Customize Links Free Hotmail Windows
Go
Internet

Figure 9-16: Page of Isole Eolie

Check and compare the results which are produced in the process of the query induction, and refer to the generative ontology of World Heritage, we find the *Similar Interest* Method works well.

We also test the link "*Cultural Category"* under *Search by Category (Version1)* in the same way. We also test the *Similar Interest* method by visiting the other sites. The results of this branch of integration test indicate the system can execute this part of the query induction application correctly.

9.2.3 Third Branch of Integration Test

In this breach of test, we also omit the test of the link form the *index.html* to the *Mainclass.html*. In the page of *Maincalss.html*, all the links under the *Natural Category* and *Cultural Category* can link to the page generated by the *CNcategory.jsp*. Here we will test the link of the category *Historic City and Area* on the page of *Mainclass.html* and the links on the page generated by *CNcategory.jsp*. And we also will test the method: *Concept Induction by Recent Examples* in this branch.

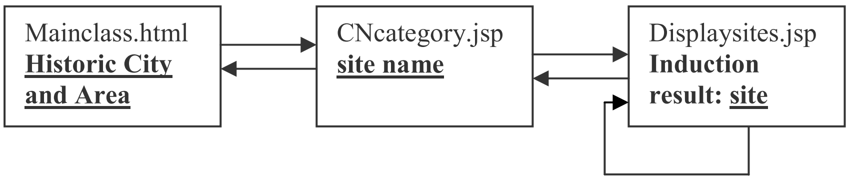
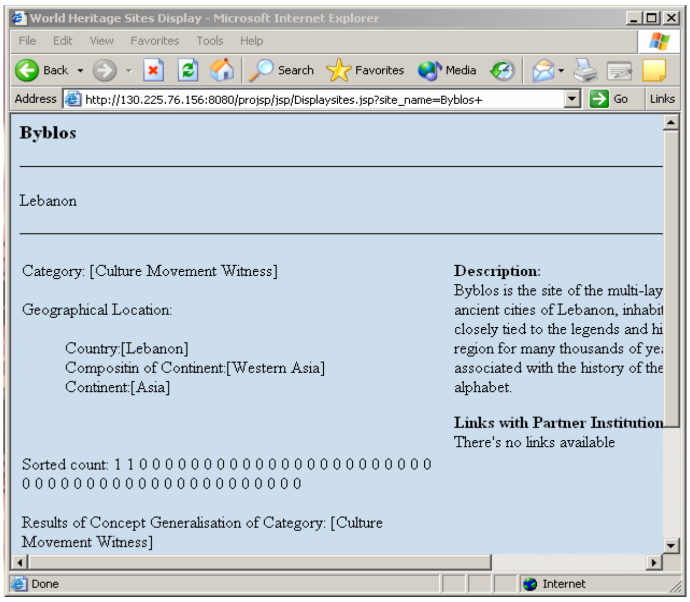


Figure 9-17: Third Branch of Integration Test

We clicked the link *"Historic City and Area"* on the page of *Mainclass.html,* and then we see a web page generated by *CNcategory.jsp.* On this page, all the sites belong to the category *Historic City and Area* are listed. Then we choose the site *"Byblos"*. The below figure is the web page generated by *Displaysites.jsp* for this site. We also test the other links under the *Natural Category* and *Cultural Category* in *Search by Category (Version1)*, such as *Geography*, *Geology* and *Construction*. All of them can be executed in the correct way.


World Heritage Sites Display - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://130.225.76.156:8080/projsp/jsp/Displaysites.jsp?site_name=Byblos+
Go Links
Done
Internet

Byblos

Lebanon

Category: [Culture Movement Witness]

Geographical Location:

Country:[Lebanon]
Compositin of Continent:[Western Asia]
Continent:[Asia]

Sorted count: 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Results of Concept Generalisation of Category: [Culture Movement Witness]

Description:
Byblos is the site of the multi-lay
ancient cities of Lebanon, inhabi
closely tied to the legends and hi
region for many thousands of ye
associated with the history of the
alphabet.

Links with Partner Institution
There's no links available

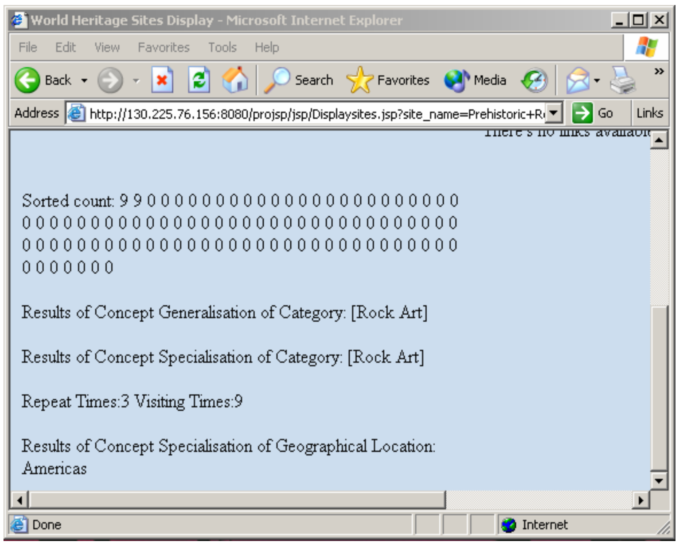
Figure 9-18: Page of Byblos

In order to test the *Concept Induction by Recent Examples* method, we have to visit 9 sites that they belong to the same category. Then we changed our interest to another category. According to this method, the system always executes induction based on the recent 9 examples from us. So when we visit the 5th site that belongs to another category, the system already knows that we have changed our interest and it uses the new examples to make induction.

We clicked the link *"Cultural Category"* and began to visit some sites that belong to the Rock-Art category.

1. *Rapa Nui National Park*
2. *The Dazu Rock Carvings*
3. *San Agustin*
4. *Decorated Grottoes of the Vézère Valley*
5. *Archaeological Park*
6. *Rock Drawings in Valcamonica*

7. *Rock-Art Sites of Tadrart Acacus*
8. *Rock Drawings of Alta*
9. *Prehistoric Rock Art Sites in the Côa Valley*


World Heritage Sites Display - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://130.225.76.156:8080/projsp/jsp/Displaysites.jsp?site_name=Prehistoric+R
Go Links
Done
Internet

There's no links availab

```
Sorted count: 9 9 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0
```

Results of Concept Generalisation of Category: [Rock Art]

Results of Concept Specialisation of Category: [Rock Art]

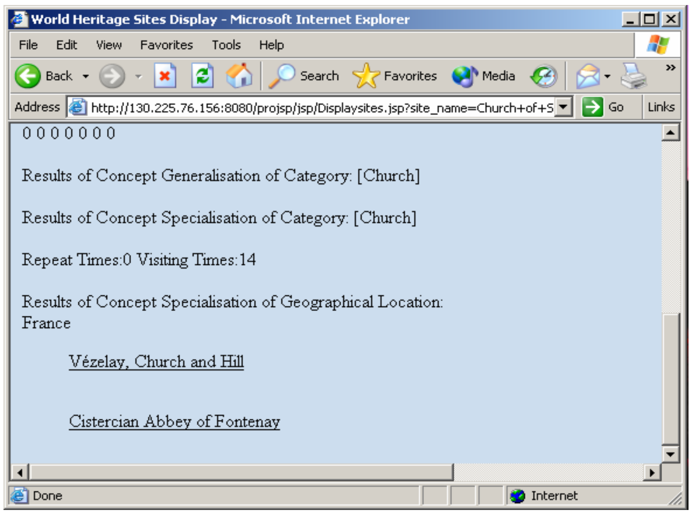
Repeat Times:3 Visiting Times:9

Results of Concept Specialisation of Geographical Location: Americas

Figure 9 19: Page of Prehistoric Rock Art Sites in the Côa Valley

Then we changed our interest to the category *Church* We begin to visit the sites about *Church*

1. *Rock-hewn Churches of Lalibela*
2. *Petäjävesi Old Church*
3. *Vézelay, Church and Hill*
4. *Cistercian Abbey of Fontenay*
5. *Church of Saint-Savin sur Gartempe*


World Heritage Sites Display - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://130.225.76.156:8080/projsp/jsp/Displaysites.jsp?site_name=Church+of+S
Go Links
Done
Internet

```
0 0 0 0 0 0 0
```

Results of Concept Generalisation of Category: [Church]

Results of Concept Specialisation of Category: [Church]

Repeat Times:0 Visiting Times:14

Results of Concept Specialisation of Geographical Location: France

Vézelay, Church and Hill

Cistercian Abbey of Fontenay

Figure 9-20: Page of Church of Saint-Savin sur Gartempe

When we visit the 5th site about *Church Church of Saint-Savin sur Gartempe,* we find the system has already caught the change of our interests, and the result of Concept Generalisation is changed to *Church* (Figure 9-20). According to the *Concept Induction by Recent Examples* method, we know the system executes the induction on the 9 recent examples from the users. When the 5th site about *Church*is visited, that means there were 5 examples about church in the 9 recent examples from user, the concept *Church* became the result of Concept Generalisation. Therefore, we know the *Concept Induction by Recent Examples* method is implemented correctly.

We also test the *Concept Induction by Recent Examples* method for other sites, the results also are acceptable. All the results of this branch of the test indicate the system can execute this part of the query induction application well.

9.2.4 Forth Branch of Integration Test

In this branch of the test, we will test the link *"Natural Category"* under *Search by Category (Version2)*, the link *"Submit"* on the page of *Nsv2.html*, the links on the pages generated by the *WNsites.jsp* and the induction results produced by the Concept Generalisation by Weight on the pages generated by *DisplayWNsites.jsp*.

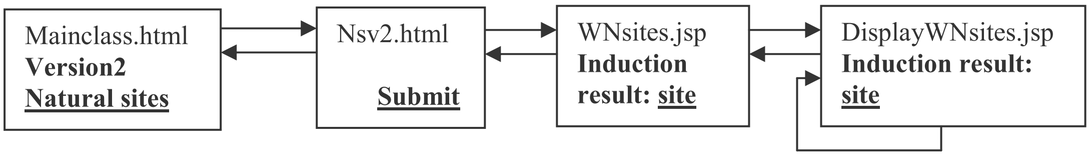
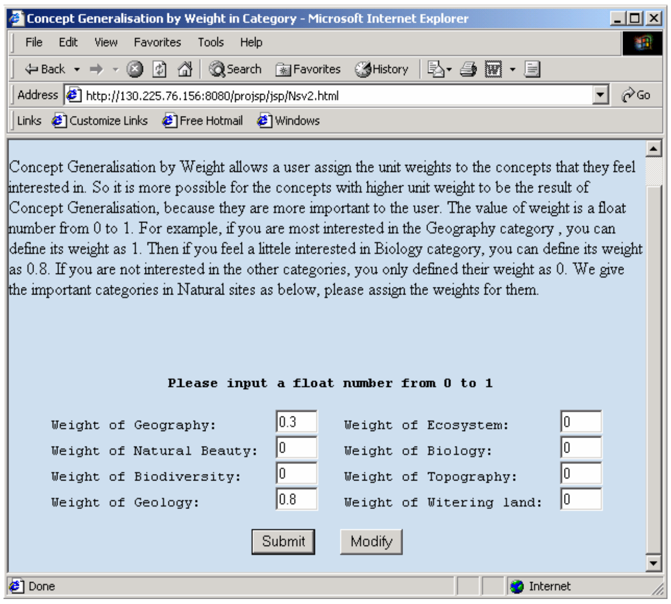


Figure 9-21: Forth Branch of Integration Test

We click the link "*Natural category"* under *Search by Category (Version2)*, the page of *Nsv2.html* is shown to us. The Concept Generalisation by Weight method is introduced on this page. There is a form of the important categories of the Natural Category, and then we assign the unit weights to them as shown in the below figure.


Concept Generalisation by Weight in Category - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address http://130.225.76.156:8080/projsp/jsp/Nsv2.html
Submit
Modify
Done
Internet

Concept Generalisation by Weight allows a user assign the unit weights to the concepts that they feel interested in. So it is more possible for the concepts with higher unit weight to be the result of Concept Generalisation, because they are more important to the user. The value of weight is a float number from 0 to 1. For example, if you are most interested in the Geography category , you can define its weight as 1. Then if you feel a littele interested in Biology category, you can define its weight as 0.8. If you are not interested in the other categories, you only defined their weight as 0. We give the important categories in Natural sites as below, please assign the weights for them.

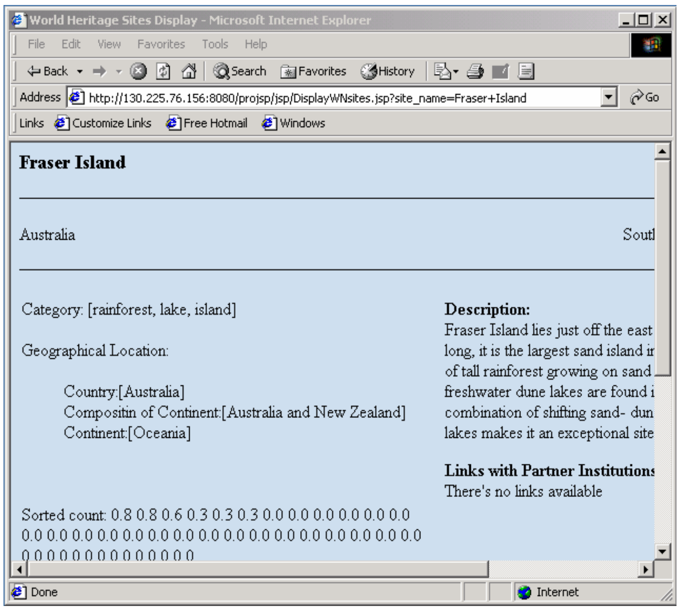
Please input a float number from 0 to 1

Weight of Geography:	0.3	Weight of Ecosystem:	0
Weight of Natural Beauty:	0	Weight of Biology:	0
Weight of Biodiversity:	0	Weight of Topography:	0
Weight of Geology:	0.8	Weight of Witering land:	0

Figure 9-22: Assign the Unit Weights to Natural Categories

Suppose we are more interested in *Geology* category, we assign 0.8 as its unit weight. And we feel a little interested in *Geography* we assign 0.3 as its unit weight. For the other categories, we are not interested at all so that we assign 0 as their unit weights. Then we submit the form. Then we see the page generated by *WNsite.jsp*. We continue to visit the sites on this page. We still choose the *"Fraser Island"* site. From the Figure 9-9 and Figure 9-10, we know the result of Concept Generalisation by Count is *[lake, forest, rainforest]*. These three concepts belong to the *Geography* category and they are more specialised than *Geography*. Different from the result of Concept Generalisation by Count, we can see that the result of Concept Generalisation by Weight is *[island]* (Figure 9-23 and Figure 9-24). Obviously we get a different result. The reason why the result is changed is that we assign the larger weight for *Geology* concepts. From the comparison above, we know the Concept Generalisation by Weight can be executed correctly.

We also tested the link "Cultural Category" in the same way. And for the link "*Cultural Category"*, the system can be linked to the page of *Csv2.html*. After the weights of some cultural categories are assigned and submitted, the system is linked to the page generated by *WCsites.jsp.* All the links on this page can also work well. So the results of this branch of integration test indicate the system can execute this part of the query induction application very well.


World Heritage Sites Display - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://130.225.76.156:8080/projsp/jsp/DisplayWNsites.jsp?site_name=Fraser+Island
Go
Links Customize Links Free Hotmail Windows
Done
Internet

Fraser Island

Australia Sout

Category: [rainforest, lake, island]

Geographical Location:

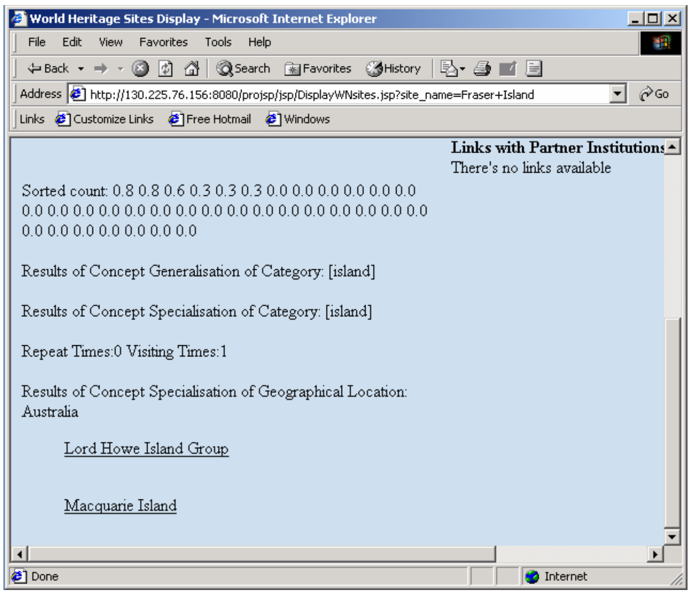
Country:[Australia]
Compositin of Continent:[Australia and New Zealand]
Continent:[Oceania]

Description:
Fraser Island lies just off the east
long, it is the largest sand island ir
of tall rainforest growing on sand
freshwater dune lakes are found i
combination of shifting sand- dun
lakes makes it an exceptional site

Links with Partner Institutions
There's no links available

Sorted count: 0.8 0.8 0.6 0.3 0.3 0.3 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0 0 0 0 0 0 0 0 0 0 0 0 0 0

Figure 9-23: First Part of Page of Fraser Island


World Heritage Sites Display - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://130.225.76.156:8080/projsp/jsp/DisplayWNsites.jsp?site_name=Fraser+Island
Go
Links Customize Links Free Hotmail Windows
Done
Internet

Links with Partner Institutions
There's no links available

Sorted count: 0.8 0.8 0.6 0.3 0.3 0.3 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0

Results of Concept Generalisation of Category: [island]

Results of Concept Specialisation of Category: [island]

Repeat Times:0 Visiting Times:1

Results of Concept Specialisation of Geographical Location:
Australia

Lord Howe Island Group

Macquarie Island

Figure 9-24: Second Part of Page of Fraser Island

9.3 Conclusion

Based on the results of component test and integration test, we can conclude the query induction system of World Heritage can work well. Since the query induction system of World Heritage is a multi-users system, we also test the system by 2-5 persons concurrently. The system can correctly execute all the induction query application programs and give the reasonable induction results to the every user. After the system test, we know we have successfully designed and implemented an induction query system of World Heritage. In the next chapter, we will conclude the whole thesis and present our achievements in this project and the future improvements.

10 Conclusion

In this thesis, we have described the theories, designs and implementations of a query induction system in detail. In this chapter we will conclude on the achievements of this thesis and provide recommendations for future work.

10.1 Achievements

The goal of this thesis was to show that the theories available in the research field of query induction were in fact applicable in the World Heritage domain. In the following part, we will discuss to what degree we have succeeded in constructing a feasible and efficient query induction system.

The work carried out in this thesis can be divided into three parts.

The first part was to give a theoretical foundation of query induction. We studied previous theories and methods concerning partial order, Hasse diagram, Lattice, ontology and generative ontology. We specialised them to develop a generative ontology from the ER model for an application domain.

The second part was based on the theoretical foundation. This part was to provide a framework for the implementation of the query induction system. The work done in this part was to design the query induction process, design the database and user interface for a query induction system, and choose the proper software for the system.

The final part was to implement a query induction system that can induce the user's interests on the basis of his/her browsing through the website of the World Heritage sites.

The achievements of this thesis are summarized as follows:

1. In Chapter 4, we described the previous theories and methods concerning partial order, Hasse diagram, Lattice, ontology and generative ontology. We extended these theories and methods with the rules of concept induction in an ontology or a generative ontology. The rules of concept induction include Concept Generalisation and Concept Specialisation and we adopted them in the query induction system of World Heritage.

2. In Chapter 3, 4 and 5, we presented the general idea of how to develop a generative ontology for an application domain, while applying this general idea to the World Heritage sites as a case study. We described how to analyse and situate the concepts in the generative ontology from the entities and entity sets in the ER model. We also described how to analyse and situate the relations in the generative ontology from the relationships in the ER model.

3. In Chapter 6, we explained the design of the query induction. The steps of query induction were carefully designed and described, including Concept Generalisation and Concept Specialisation. The roles of the database and the generative ontology during the query induction were also included in the steps.

   Furthermore, we presented our suggestions of improving the performance of the query induction. The method Concept Generalisation by Count and Concept Generalisation by Weight was designed to improve the accuracy of the query induction. Some other aspects were also considered. The method Similar Interests was designed to give the user various and suggestive results. And the method Concept Generalisation by Recent Examples was designed to catch the change of the user's interests quickly.

4. In Chapter 7, we gave a detailed explanation of the approach of designing a query induction system, including the three-tier client/server system architecture design, the database design, the query induction application design, and the user interface design.

5. In Chapter 8 and 9, we implemented the query induction system that can induce queries from examples of World Heritage. The system was tested to function well according to the system design.

Our prototype system can provide the user the information about the World Heritage sites as a normal website. Furthermore, the system can induce queries based on the selected sites from the users, and suggest the user some unvisited sites retrieved from the database by the induced queries.

We only designed the basic and functional interfaces in the current system, because our main goal of this thesis is to find a general approach of designing and implementing a query induction system. Moreover, the available source information of the World Heritage sites is limited. For example, we did not get the pictures of the corresponding sites so we can't show them in the interfaces. However, the current interfaces present the results step by step to illustrate the process of query induction.

Our prototype system works on the Linux platform. The users can access the Web Server from different platforms such as Linux, UNIX and Windows. It is a little slow when the system retrieves larger amounts of data from the database. That is acceptable because we install the Web Sever and the Database Sever on normal personal computers. If the system was implemented on high performance servers, the speed would improve. The prototype system can be visited by multi-users concurrently. We have tested that the system works well when 2-5 persons are visiting it at the same time. Due to the limitations of time and resources, we haven't had the chance to test our query induction system of World Heritage by more than 10 users concurrently.

10.2 Future Work

Several future extensions will improve the query induction system. Due to the time constrains, we give some recommendations of the future work with some outlining explanations.

1. More user requirements

The current query induction system of World Heritage focuses on illustrating the general theories and methods of developing a query induction system. This system fulfils the overall goals that we mentioned in Chapter 1. But it does not specify the user's requirements in detail. More work can be carried out to collect and analyse the users' requirements carefully.

User requirements are important for the query induction system. If we know the views of the users about the application domain, we can improve the system to provide more accurate and satisfying results of query induction for the users. The user requirements can be used in the ontology development and the system design.

For example, in the E-Business domain, the query induction systems need to consider the age, gender, income and shopping habit of the users. If the user has the tendency to buy cheap products, the goods on sale or with lower price can be assigned with higher unit weights by the system (according to the Concept Generalisation by Weight in Section 6.5.2) so that they have higher priorities to be suggested to the user.

2. Further development of the generative ontology of World Heritage

The current generative ontology of World Heritage has three regions: *Site*, *Category* and *Geographical Location*. More regions can be built for the query induction. For example, for the sites of category *Construction*, we can build a region concerning the style of the constructions and execute the query induction in it.

The World Heritage domain is hard to model by the generative ontology, because the domain extends into many specialised fields. We found it difficult to fully understand and analyse these specialised fields, for example natural science. Expert knowledge and more information of the sites will be helpful to further develop the generative ontology of the World Heritage.

Considering the user requirements, the generative ontology of World Heritage can be developed according to the age, gender, background, and browsing purpose of the users. The current generative ontology may be too complicated for kids, so more simple objects instead of terminologies as *Hominid* are needed when developing the generative ontology for kids.

3. Easier maintenance of the Hasse diagrams

The concepts in the generative ontology are structured as the Hasse diagrams in the regions. It often happens that some concepts need to be changed, added or deleted from the Hasse diagrams, so easy maintenance of the Hasse diagrams is important to a query induction system.

We experimented with two ways of storing the Hasse diagrams in the generative ontology. The Hasse diagram of region *Category* is kept in Java files using the data structure we predefined, and the Hasse diagram of region *Geographical Location* is kept in the relational database.

Many concepts in the generative ontology are the entities kept in the database already. It is a good idea to design some extra relational tables to keep the Hasse diagram in the database too, because it is easier to create and update the Hasse diagrams by this method.

We use Figure 10-1 to illustrate the ideas.

Table1: Entity Set - Category

Military Construction
Resident Construction
Fortress
Castle
Mansion

Hasse diagram of *Construction*

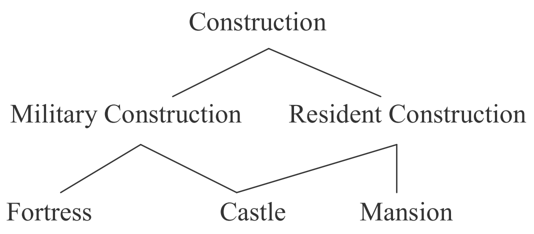


Table 2: Hasse diagram in a relational table

Construction	Military Construction
Construction	Resident Construction
Military Construction	Fortress
Military Construction	Castle
Resident Construction	Castle
Resident Construction	Mansion

Figure 10-1: Hasse Diagram in Relational Table

According to the ER diagram, Table 1 keeps the entities of the entity set *Category*. These entities can describe the categories of the World Heritage sites.

As we presented in Section 4.3, the Hasse diagram of *Construction* are built on the Table 1. So the entities in Table 1 are the concepts in the medium level and the specialised level of the Hasse diagram. *Construction* is a general level concept introduced in the Hasse diagram, so it is not an entity in Table 1.

Table 2 is created to keep the concepts and their relations in the Hasse diagram. The left column of Table 2 is the concept itself and the right column is the concept just below it according to the partial order in the Hasse diagram.

Because part of the data in Table 2 come from Table 1, like *Military Construction* in Figure 10-1, we can set them to be updated automatically when they are changed in Table 1, which is easy to achieve by foreign key constraints using common relational database software. By this method, it is easier to create and update the Hasse diagrams in the generative ontology.

The problem of keeping Hasse diagrams in relational tables is that the execution will slow down if the Web Server needs to connect the Database Server to get the information of the Hasse diagrams frequently. This problem is not difficult to solve. For example, in the current query induction system of World Heritage, the query induction application is written in Java, so we can define a method to transfer the Hasse diagrams in the relational table into instances in Java files when the system begins to execute the query induction.

Furthermore, if there are many concepts, it is important to have tools to show the structure of the generative ontology pictorially. We are not very satisfied with dividing the whole Hasse diagram in region *Category* into parts, but it seems to be the only choice for us to show it in this document because of the limitation of the space.

4. More efficient implementation of the Hasse diagrams

The concept induction is executed in the Hasse diagram, so a good data structure of the Hasse diagrams can make the query induction system easy to implement and improve the speed of the system.

The current data structure of the Hasse diagrams can be improved by using the data structure of *Directed Acyclic Graph (DAG)*, because the Hasse diagrams are directed by the partial orders, and have no loops. We can use the graph algorithm of *Depth-First Search* to find the level and the path for every concept in the Hasse diagrams, so the algorithm *Depth-First Search* can be used when designing the methods of Concept Generalisation and Concept Specialisation in the query induction application [4].

More measurements should be used to test the running time and the needed resources when executing the query induction, for example the memory. With these precise test results, we can compare the different ways of implementing the Hasse diagrams and choose the most efficient one.

5. More interaction with the user

The query induction system is a knowledge based system. Although it has artificial intelligence, the interaction with the user can be used to improve the accuracy.

The method of Concept Generalisation by Weight is an attempt. In the current query induction system, we let the user assign the unit weights to some concepts that cover big groups of sites before the user begins to visit the sites. By this method, the query induction will take the interests of the individual user into consideration and improve the accuracy.

The first problem is that part of the generative ontology is shown to the users, because we show some concepts to the users directly. However, the generative ontology should play its role behind. Furthermore, these concepts might be too abstract for the user when the user hasn't visited any sites. Therefore, it might be not so easy for the user to assign the unit weights to these concepts.

More thoughtful design will improve the system. For example, we can let the user value the site after he/she has visited it.

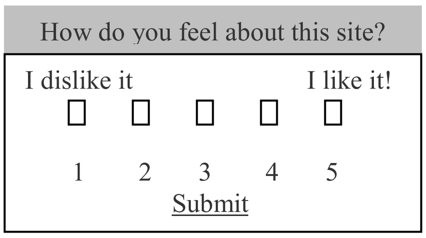


Figure 10-2: Value the World Heritage Site

If the user likes the site, he/she can choose button '5', so the system can find the concepts that cover this site and assign its unit weight with 1.0. Assume that the scope of the unit weight of the concept is 0 ~ 1 as the current system, then '1' stands for unit weight 0; '2' stands for unit weight 0.25; '3' stands for unit weight 0.5; '4' stands for unit weight 0.75 and '5' stands for unit weight 1.0.

The above description is only a general idea; more particular cases should be studied. For example, if the user likes one site and dislikes the other, but these two sites belong to the same concept, then how to assign the unit weight to this concept?

In this thesis, we provided a general approach of designing and implementing a query induction system and took a case study of the World Heritage sites to illustrate the methods and development steps.

A query induction is not complicated to implement. The main part of a query induction system is to model the application domain by the ontology that is suitable to execute concept induction. The application of the query induction system can be implemented in high level programming languages with the Dynamic Web Page Generation Technologies.

The query induction system can provide the convenience for the users. For example, induction query systems can be widely used in the E-Business field. The domain of E-Business is suitable to model by ontologies or generative ontologies, because the specifications of goods are normally uniformed, such as price, size, and category. With the query induction, the E-Business applications can easily find the commodities that the users are interested in so that more personal services can be provided for the users.

Therefore, from the work carried out in this thesis, we feel that the query induction system could have a wide area of application fields.

Bibliography

[1] Troels Andreasen, Henrik Bulskov, and Rasmus Knappe
On ontology-based querying
In Heiner Stuckenschmidt (Eds.):
18th International Joint Conference on Artificial Intelligence,
Ontologies and Distributed Systems, 2003, Workshop Program

[2] Karl Avedal, Danny Ayers and Timothy Briggs.
Professional JSP
Wrox Press Ltd. 2001

[3] Hans Bruun and Jørgen Fischer Nilsson
Entity – Relationship Models as Grammars and Lattices– a Foundational View

[4] Thomas H.Cormen, Charles E.Leiserson and Ronald L.Rivest
Introductions to Algorithms
The MIT Press, 1998

[5] B.A. Davey and H.A Priestley
Introduction to Lattices and Order
Cambridge University Press, 1990

[6] Cay s.Horstmann and Gary Cornell.
Core Java 2 Volume II: Advanced Features
Prentice Hall 2000

[7] Henrik Legind Larsen and Jørgen Fischer Nilsson
Fuzzy Querying in a Concept Object Algebraic Data Model
In H. Christiansen, T. Andreasen, H.L. Larsen (eds.)
Flexible Query Answering Systems, Kluwer Academic Publishers, 1997

[8] Jørgen Fischer Nilsson
A Logico-Algebraic Framework for Ontologies ONTOLOG
In Jensen & Skadhauge (eds.) 2001

[9] Jørgen Fischer Nilsson and Hele-Mai Haav
Inducing queries from examples as concept formation
In: Proc. of the 8th European-Japanese Conference on Information
Modelling and Knowledge Bases, 1998

[10] Matthew Reynolds
Beginning E-Commerce
Wrox Press Ltd. 2000

[11] Ian Sommerville
Software Engineering (Sixth Edition)
Addison-Wesley, 2001

[12] Jeffrey D. Ullman and Jennifer Widom
A First Course in Database Systems
Prentice Hall, 1997

[Apache]
The home page of Apache
www.apache.org

[JSP]
The page of JSP on the Sun Java web site
www.java.sun.com/products/JSP

[JDBC]
The page of JDBC on the Sun Java web site
www.java.sun.com/products /JDBC

[MySQL]
The home page of MySQL
www.mysql.com

[Tomcat]
The home page of Tomcat
http://jakarta.apache.org/tomcat/index.html

[Unesco]
The official web site of Unesco World Heritage Committee
http:// whc.unesco.org

[VR-Heritage]
The web site of VR-Heritage organization
www.vrheritage.org

[VR-Heritage Prototype]
Prototype of the VR-Heritage web site
http://world-heritage-explorer.org/engine/explorer

Appendix A

Java source code

**********Connect.java**********

```
package induction;

import java.sql.*;
import java.util.*;
import java.io.*;

public class Connect
  {
    String userName;
    String password;
    String url;
    Connection con;

public Connect()

  {
    try
      {
       userName= "kun";
       password="some_pass";
       url= "jdbc:mysql://130.225.76.177/exwh";
       Class.forName ("com.mysql.jdbc.Driver").newInstance ();
       con= DriverManager.getConnection (userName, url, password);
      }
    catch(Exception e){}
  }

public Connection getCon()
  {
    return con;
  }

public static Vector getCporReg(String q1,Connection conn,String ctg)

  {
    Vector vcr = new Vector(); /*vector of composition or region*/
    try
      {
      Class.forName ("com.mysql.jdbc.Driver").newInstance ();
      System.out.println ("Database connection established");
      }
    catch (Exception e)
      {
         System.err.println ("Cannot connect to database server");
         }
       finally
         {
          if (conn != null)
            {
            try
              {
               Statement stmt = conn.createStatement();
              ResultSet rs = stmt.executeQuery(q1);

              while (rs.next())
                {
                String cr =rs.getString(ctg);
                vcr.add(cr);
                }

              System.out.println(vcr);
              rs.close();

              }
            catch (Exception e){};
            }
         }
         return vcr;
       }
    }

public static Vector getAttribute ( String sn,String q1,Connection conn,String atr)
    {
       Vector va = new Vector(); /*vector of attributes*/
       try
         {
          Class.forName ("com.mysql.jdbc.Driver").newInstance ();
          System.out.println ("Database connection established");
         }
       catch (Exception e)
         {
         System.err.println ("Cannot connect to database server");
         }
       finally
         {
         if (conn != null)
           {
           try
             {
              PreparedStatement ps1 = conn.prepareStatement(q1);
              ps1.setString(1,sn);
             ResultSet rs1 = ps1.executeQuery();
              System.out.println("****Get the category of Sites****");

             while (rs1.next())
               {
```

```
              String atb =rs1.getString(atr);
              va.add(atb);
             }

            System.out.println(va);
            rs1.close();
           }
         catch (Exception e){};
         }
      }
    return va;
  }

public static String getAQuery(Vector vInt, Vector vG, Vector vCP)
  {
    String query ="";
    for (int i = 0; i <vInt.size() ; i++)
      {
         if (vG.contains((String)vInt.get(i)))
            {
            query = "select distinct site_name from site, cncategory, continent, composition,
            country, LOCcountry where site.siteid = cncategory.siteid and site.siteid =
            LOCcountry.siteid and LOCcountry.countryid = country.countryid and
            composition. compositionid=country.compositionid  and
            composition.continentid= continent.continentid and cncategory.cncategory = ? and
            continet.continent_name = ? and site.site_name!=?";
            }
         else
            if (vCP.contains((String)vInt.get(i)))
            {
            query = "select distinct site_name from site, cncategory, composition, country,
            LOCcountry where site.siteid=cncategory.siteid and site.siteid =
            LOCcountry.siteid and LOCcountry .countryid = country.countryid and
            composition.compositionid = country.compositionid and cncategory.cncategory =
            ? and composition.composition_name = ? and site.site_name!=?";
            }
           else
           {
            query = "select distinct site_name from site, cncategory,country,LOCcountry
            where site.siteid=cncategory.siteid and site.siteid = LOCcountry.siteid and
            LOCcountry.countryid  = country.countryid and cncategory.cncategory=? and
            country.country=? and site.site_name!=?";
            }
      }
      return query;
  }

public static Vector getIndOfresult(Vector vi,Connection conn,String q2,String pos,String sN)
  {
    Vector vir = new Vector();
    try
      {
      PreparedStatement ps2 = conn.prepareStatement(q2);
```

```
        Vector vind = new Vector();

        for (int i = 0; i <vi.size() ; i++)
          {
          ps2.setString(1,(String)vi.get(i));
          ps2.setString(2,pos);
          ps2.setString(3,sN);
          ResultSet rs2 = ps2.executeQuery();

          while (rs2.next())
             {
             String snm =rs2.getString("site_name");
             if(!vir.contains(snm))
               {
               vir.add(snm);
               }
             }
          rs2.close();
          }
        }
        catch (Exception e){};
      return vir;
  }
  }
```

**********Concept.java**********

```
package induction;

import java.sql.*;
import java.util.*;
import java.io.*;

public class Concept
   {
      int level;
      String conceptname;
      Vector path = new Vector();
      int count;

public  Concept(String nn,int lev,Vector v,int vts)
   {
       level=lev;
       conceptname=nn;
       path =v;
       count=vts;
   }

 public static int[ ] getlevel(Vector vcon)

   {
      int [ ] lev = new int[vcon.size()];
```

```
       for (int i = 0; i < lev.length; i++)
         {
         lev[i]=  ((Concept)vcon.get(i)).level;
         }
       return lev;

  }
  }
```

**********Category.java***********

```
package induction;

import java.sql.*;
import java.util.*;
import java.io.*;

public class Category
   {
     String categoryname;
     int conceptnum;
     Vector concepts = new Vector();

public Category(String cgn, int cgnum,Vector vcpt)
    {
         categoryname=cgn;
         conceptnum=cgnum;
         concepts=vcpt;
  }

public static Vector initConcepts(String cgn){

    Vector vp0 = new Vector();
    Vector vp1 = new Vector();
    Vector vp2 = new Vector();
    Vector vp3 = new Vector();
    Vector vp4 = new Vector();
    Vector vp5 = new Vector();
    Vector vp6 = new Vector();
    Vector vp7 = new Vector();
    Vector vp8 = new Vector();
    Vector vp9 = new Vector();
    Vector vp10 = new Vector();
    Vector vp11 = new Vector();
    Vector vp12 = new Vector();
    Vector vp13 = new Vector();
    Vector vp14 = new Vector();
    Vector vp15 = new Vector();
    Vector vp16 = new Vector();
    Vector vp17 = new Vector();
    Vector vp18 = new Vector();
    Vector vp19 = new Vector();
Vector vp20 = new Vector();
Vector vp21 = new Vector();
Vector vp22 = new Vector();
Vector vp23 = new Vector();
Vector vp24 = new Vector();
Vector vp25 = new Vector();
Vector vp26 = new Vector();
Vector vp27 = new Vector();
Vector vp28 = new Vector();
Vector vp29 = new Vector();
Vector vp30 = new Vector();
Vector vp31 = new Vector();
Vector vp32 = new Vector();
Vector vp33 = new Vector();
Vector vp34 = new Vector();
Vector vp35 = new Vector();
Vector vp36 = new Vector();
Vector vp37 = new Vector();
Vector vp38 = new Vector();
Vector vp39 = new Vector();
Vector vp40 = new Vector();
Vector vp41 = new Vector();
Vector vp42 = new Vector();
Vector vp43 = new Vector();
Vector vp44 = new Vector();
Vector vp45 = new Vector();
Vector vp46 = new Vector();
Vector vp47 = new Vector();
Vector vp48 = new Vector();
Vector vp49 = new Vector();
Vector vp50 = new Vector();
Vector vp51 = new Vector();
Vector vp52 = new Vector();
Vector vp53 = new Vector();
Vector vp54 = new Vector();
Vector vp55 = new Vector();
Vector vp56 = new Vector();
Vector vp57 = new Vector();
Vector vp58 = new Vector();
Vector vp59 = new Vector();
Vector vp60 = new Vector();
Vector vp61 = new Vector();
Vector vp62 = new Vector();
Vector vp63 = new Vector();
Vector vp64 = new Vector();
Vector vp65 = new Vector();
Vector vp66 = new Vector();
Vector vp67 = new Vector();
Vector vp68 = new Vector();
Vector vp69 = new Vector();
Vector vp70 = new Vector();
Vector vp71 = new Vector();
Vector vp72 = new Vector();
Vector vp73 = new Vector();
Vector vp74 = new Vector();
Vector vp75 = new Vector();
Vector vp76 = new Vector();
Vector vp77 = new Vector();
Vector vp78 = new Vector();
Vector vp79 = new Vector();
Vector vp80 = new Vector();
Vector vp81 = new Vector();
Vector vp82 = new Vector();
Vector vp83 = new Vector();
Vector vp84 = new Vector();
Vector vp85 = new Vector();
Vector vp86 = new Vector();
Vector vp87 = new Vector();
Vector vp88 = new Vector();
Vector vp89 = new Vector();
Vector vp90 = new Vector();
Vector vp91 = new Vector();
Vector vp92 = new Vector();
Vector vp93 = new Vector();
Vector vp94 = new Vector();
Vector vp95 = new Vector();
Vector vp96 = new Vector();
Vector vp97 = new Vector();
Vector vp98 = new Vector();
Vector vp99 = new Vector();
Vector vp100 = new Vector();
Vector vp101 = new Vector();
Vector vp102 = new Vector();
Vector vp103 = new Vector();
Vector vp104 = new Vector();
Vector vp105 = new Vector();
Vector vp106 = new Vector();
Vector vp107 = new Vector();
Vector vp108 = new Vector();
Vector vp109 = new Vector();
Vector vp110 = new Vector();
Vector vp111 = new Vector();
Vector vp112 = new Vector();
Vector vp113 = new Vector();
Vector vp114 = new Vector();
Vector vp115 = new Vector();
Vector vp116 = new Vector();
Vector vp117 = new Vector();
Vector vp118 = new Vector();
Vector vp119 = new Vector();
Vector vp120 = new Vector();
Vector vp121 = new Vector();
Vector vp122 = new Vector();
Vector vp123 = new Vector();
Vector vp124 = new Vector();
Vector vp125 = new Vector();
```

```
 Vector vp126 = new Vector();
 Vector vp127 = new Vector();
 Vector vp128 = new Vector();
 Vector vp129 = new Vector();
 Vector vp130 = new Vector();
 Vector vp131 = new Vector();
 Vector vl1 = new Vector();

if ("Natural Sites".equals(cgn))
    {
     vp1.add("natural beauty");
     vp2.add("topography");
     vp3.add("geography");
     vp4.add("biodiversity");
     vp5.add("geology");
     vp6.add("ecosystem");
     vp7.add("biology");
     vp8.add("wintering Land");

     vp9.add("mountain");
     vp9.add("topgraphy");

     vp10.add("savannah");
     vp10.add("geography");

      vp11.add("lake");
      vp11.add("geography");

    vp12.add("wetland");
     vp12.add("geography");

     vp13.add("waterfall");
     vp13.add("geography");

     vp14.add("river");
     vp14.add("geography");

     vp15.add("forest");
     vp15.add("geography");

     vp16.add("fossil");
     vp16.add("geology");

     vp17.add("volcano");
     vp17.add("geology");

     vp18.add("george");
     vp18.add("geology");

     vp19.add("geomorphology");
     vp19.add("geology");

     vp20.add("reef");
```

```
vp20.add("geology");

vp21.add("cliff");
vp21.add("geology");

vp22.add("glacier");
vp22.add("geology");

vp23.add("island");
vp23.add("geology");

vp24.add("fauna");
vp24.add("biology");

vp25.add("flora");
vp25.add("biology");

 vp26.add("rainforest");
 vp26.add("forest");
 vp26.add("geography");

 vp27.add("tropical forest");
 vp27.add("forest");
 vp27.add("geography");

 vp28.add("temperate forest");
 vp28.add("forest");
 vp28.add("geography");

 vp29.add("special forest");
 vp29.add("forest");
 vp29.add("geography");

 vp30.add("cave");
 vp30.add("gemorphology");
 vp30.add("geology");

 vp31.add("karst");
 vp31.add("gemorphology");
 vp31.add("geology");

 vp32.add("limestone");
 vp32.add("gemorphology");
 vp32.add("geology");

 vp33.add("birds");
 vp33.add("fauna");
 vp33.add("biology");

 vp34.add("animal");
 vp34.add("fauna");
 vp34.add("biology");
 vp35.add("mammal");
 vp35.add("animal");
 vp35.add("fauna");
 vp35.add("biology");

 Concept node1 = new Concept("natural beauty", 1,vp1,0);
 Concept node2 = new Concept("topography", 1,vp2,0);
 Concept node3 = new Concept("geography", 1,vp3,0);
 Concept node4 = new Concept("biodiversity", 1, vp4,0);
 Concept node5 = new Concept("geology", 1, vp5,0);
 Concept node6 = new Concept("ecosystem", 1, vp6,0);
 Concept node7 = new Concept("biology", 1, vp7,0);
 Concept node8 = new Concept("wintering land", 1, vp8,0);
 Concept node9 = new Concept("mountain", 2, vp9,0);
 Concept node10 = new Concept("savannah", 2,vp10,0);
 Concept node11 = new Concept("lake", 2,vp11,0);
 Concept node12 = new Concept("wetland", 2,vp12,0);
 Concept node13 = new Concept("waterfall", 2, vp13,0);
 Concept node14 = new Concept("river", 2, vp14,0);
 Concept node15 = new Concept("forest", 2, vp15,0);
 Concept node16 = new Concept("fossil", 2, vp16,0);
 Concept node17 = new Concept("volcano", 2, vp17,0);
 Concept node18 = new Concept("gorge", 2, vp18,0);
 Concept node19 = new Concept("geomorphology", 2, vp19,0);
 Concept node20 = new Concept("reef", 2, vp20,0);
 Concept node21 = new Concept("cliff", 2,vp21,0);
 Concept node22 = new Concept("glacier", 2,vp22,0);
 Concept node23 = new Concept("island", 2,vp23,0);
 Concept node24 = new Concept("fauna", 2, vp24,0);
 Concept node25 = new Concept("flora", 2, vp25,0);
 Concept node26 = new Concept("rainforest", 3, vp26,0);
 Concept node27 = new Concept("tropical forest", 3, vp27,0);
 Concept node28 = new Concept("temperate forest", 3, vp28,0);
 Concept node29 = new Concept("special forest", 3, vp29,0);
 Concept node30 = new Concept("cave", 3, vp30,0);
 Concept node31 = new Concept("karst", 3, vp31,0);
 Concept node32 = new Concept("limestone", 3, vp32,0);
 Concept node33 = new Concept("birds", 3, vp33,0);
 Concept node34 = new Concept("animal", 3, vp34,0);
 Concept node35 = new Concept("mammal", 4, vp34,0);

vl1.add(node1);
vl1.add(node2);
vl1.add(node3);
vl1.add(node4);
vl1.add(node5);
vl1.add(node6);
vl1.add(node7);
vl1.add(node8);
vl1.add(node9);
vl1.add(node10);
vl1.add(node11);
vl1.add(node12);
```

```
vl1.add(node13);
vl1.add(node14);
vl1.add(node15);
vl1.add(node16);
vl1.add(node17);
vl1.add(node18);
vl1.add(node19);
vl1.add(node20);
vl1.add(node21);
vl1.add(node22);
vl1.add(node23);
vl1.add(node24);
vl1.add(node25);
vl1.add(node26);
vl1.add(node27);
vl1.add(node28);
vl1.add(node29);
vl1.add(node30);
vl1.add(node31);
vl1.add(node32);
vl1.add(node33);
vl1.add(node34);
vl1.add(node35);
}
else if ("Geography".equals(cgn))
{
vp1.add("savannah");
vp1.add("geography");

vp2.add("lake");
vp2.add("geography");

vp3.add("wetland");
vp3.add("geography");

vp4.add("waterfall");
vp4.add("geography");

vp10.add("forest");
vp10.add("geography");

 vp5.add("river");
 vp5.add("geography");

 vp6.add("rainforest");
 vp6.add("forest");
 vp6.add("geography");

 vp7.add("tropical forest");
 vp7.add("forest");
 vp7.add("geography");

 vp8.add("temperate forest");
 vp8.add("forest");
 vp8.add("geography");

 vp9.add("special forest");
 vp9.add("forest");
 vp9.add("geography");

 Concept node1 = new Concept("savannah", 1,vp1,0);
 Concept node2 = new Concept("lake", 1,vp2,0);
 Concept node3 = new Concept("wetland", 1,vp3,0);
 Concept node4 = new Concept("waterfall", 1, vp4,0);
 Concept node5 = new Concept("river", 1, vp5,0);
 Concept node10 = new Concept("forest", 1, vp10,0);
 Concept node6 = new Concept("rainforest", 2, vp6,0);
 Concept node7 = new Concept("tropical forest", 2, vp7,0);
 Concept node8 = new Concept("temperate forest", 2, vp8,0);
 Concept node9 = new Concept("special forest", 2, vp9,0);

vl1.add(node1);
vl1.add(node2);
vl1.add(node3);
vl1.add(node4);
vl1.add(node5);
vl1.add(node10);
vl1.add(node6);
vl1.add(node7);
vl1.add(node8);
vl1.add(node9);
}
else if ("Topography".equals(cgn))
{
 vp0.add("topography");

 vp1.add("mountain");
 vp1.add("topography");

 Concept node0 = new Concept("topography", 1, vp0,0);
 Concept node1 = new Concept("mountain", 2,vp1,0);

 vl1.add(node0);
 vl1.add(node1);
}
 else if ("Biology".equals(cgn))
{
vp0.add("biology");

 vp1.add("flora");
 vp1.add("biology");

 vp2.add("fauna");
 vp2.add("biology");

 vp3.add("bird");
 vp3.add("fauna");
 vp3.add("biology");

 vp4.add("animal");
 vp4.add("fauna");
 vp4.add("biology");

 vp5.add("mammal");
 vp5.add("animal");
 vp5.add("fauna");
 vp5.add("biology");

 vp6.add("reptile");
 vp6.add("animal");
 vp6.add("fauna");
 vp6.add("biology");

 vp7.add("amphibians");
 vp6.add("animal");
 vp6.add("fauna");
 vp7.add("biology");

 Concept node0 = new Concept("biology", 1, vp0,0);
 Concept node1 = new Concept("flora", 2,vp1,0);
 Concept node2 = new Concept("fauna", 2,vp2,0);
 Concept node3 = new Concept("bird", 3,vp3,0);
 Concept node4 = new Concept("animal", 3, vp4,0);
 Concept node5 = new Concept("mammal", 4, vp5,0);
 Concept node6 = new Concept("reptile", 4, vp6,0);
 Concept node7 = new Concept("amphibians", 4, vp7,0);

vl1.add(node0);
vl1.add(node1);
vl1.add(node2);
vl1.add(node3);
vl1.add(node4);
vl1.add(node5);
vl1.add(node6);
vl1.add(node7);
}
else if ("Geology".equals(cgn))
{
vp0.add("geology");

 vp1.add("fossil");
 vp1.add("geology");

 vp2.add("volcano");
 vp2.add("geology");

 vp3.add("reef");
 vp3.add("geology");
 vp4.add("glacier");
 vp4.add("geology");

 vp5.add("island");
 vp5.add("geology");

vp6.add("gorge");
vp6.add("geology");

vp7.add("geomorphology");
vp7.add("geology");

vp8.add("cave");
vp8.add("geomorphology");
vp8.add("geology");

vp9.add("karst");
vp9.add("geomorphology");
vp9.add("geology");

vp10.add("limestone");
vp10.add("geomorphology");
vp10.add("geology");

Concept node0 = new Concept("geology", 1, vp0,0);
Concept node1 = new Concept("fossil", 2,vp1,0);
Concept node2 = new Concept("volcano", 2,vp2,0);
Concept node3 = new Concept("reef", 2,vp3,0);
Concept node4 = new Concept("glacier", 2, vp4,0);
Concept node5 = new Concept("island", 2, vp5,0);
Concept node6 = new Concept("gorge", 2, vp10,0);
Concept node7 = new Concept("geomorphology", 2, vp6,0);
Concept node8 = new Concept("cave", 3, vp7,0);
Concept node9 = new Concept("karst", 3, vp8,0);
Concept node10 = new Concept("limestone", 3, vp9,0);

vl1.add(node0);
vl1.add(node1);
vl1.add(node2);
vl1.add(node3);
vl1.add(node4);
vl1.add(node5);
vl1.add(node6);
vl1.add(node7);
vl1.add(node8);
vl1.add(node9);
vl1.add(node10);
}
else if("Historic Cities and Areas".equals(cgn))
{
 vp1.add("Economy Historic centre");  // depth 1
 vp2.add("Culture Historic Centre");
 vp3.add("Spiritual Historic Centre");
```

```
vp4.add("Military Historic Centre");
vp5.add("Politics Historic Centre");
vp6.add("Resident Settlement");

vp7.add("Agriculture Centre");
vp7.add("Economy Historic centre");

vp8.add("Industry Historic Centre");
vp8.add("Economy Historic centre");

vp9.add("Commerce Historic Centre");
vp9.add("Economy Historic centre");

vp10.add("Mining Town");
vp10.add("Industry Historic Centre");
vp10.add("Economy Historic centre");

vp11.add("Company Town"); // not needed
vp11.add("Industry Historic Centre");
vp11.add("Economy Historic centre");

vp12.add("Industry Town");
vp12.add("Industry Historic Centre");
vp12.add("Economy Historic centre");

vp13.add("Wool Industry Complex");
vp13.add("Industry Town");
vp13.add("Industry Historic Centre");
vp13.add("Economy Historic centre");

vp14.add("Copper Mines Industry Complex");
vp14.add("Mining Town");
vp14.add("Industry Historic Centre");
vp14.add("Economy Historic centre");

vp15.add("Sugar Cane Industry Complex");
vp15.add("Industry Town");
vp15.add("Industry Historic Centre");
vp15.add("Economy Historic centre");

vp16.add("Silver Mines Industry Complex");
vp16.add("Mining Town");
vp16.add("Industry Historic Centre");
vp16.add("Economy Historic centre");

vp17.add("Gold Mines Industry Complex");
vp17.add("Mining Town");
vp17.add("Industry Historic Centre");
vp17.add("Economy Historic centre");

vp18.add("Diamond Prospect Industry Complex");
vp18.add("Mining Town");
vp18.add("Industry Historic Centre");
```

```
vp18.add("Economy Historic centre");

vp19.add("Trading Centre");
vp19.add("Commerce Historic Centre");
vp19.add("Economy Historic centre");

vp20.add("Harbor");
vp20.add("Commerce Historic Centre");
vp20.add("Economy Historic centre");

vp21.add("Culture Movement Witness");
vp21.add("Culture Historic Centre");

vp22.add("Culture Blending Witness");
vp22.add("Culture Historic Centre");

vp23.add("Colonial Settlement");
vp23.add("Culture Blending Witness");
vp23.add("Culture Historic Centre");

vp24.add("Education Historic Centre");
vp24.add("Culture Historic Centre");

vp25.add("Aesthetic Historic Centre");
vp25.add("Culture Historic Centre");

vp26.add("Architecture Centre");
vp26.add("Culture Historic Centre");

vp27.add("University City");
vp27.add("Education Historic Centre");
vp27.add("Culture Historic Centre");

vp28.add("Music Centre");
vp28.add("Aesthetic Historic Centre");
vp28.add("Culture Historic Centre");

vp29.add("Monument Historic Centre");
vp29.add("Spiritual Historic Centre");

vp30.add("Religious Historic Centre");
vp30.add("Spiritual Historic Centre");


vp31.add("Judaism Historic Centre");
vp31.add("Religious Historic Centre");
vp31.add("Spiritual Historic Centre");

vp32.add("Buddhism Historic Centre");
vp32.add("Religious Historic Centre");
vp32.add("Spiritual Historic Centre");

vp33.add("Christianity Historic Centre");
```

```
vp33.add("Religious Historic Centre");
vp33.add("Spiritual Historic Centre");

vp34.add("Islam Historic Centre");
vp34.add("Religious Historic Centre");
vp34.add("Spiritual Historic Centre");

vp35.add("Religious Blending Witness");
vp35.add("Religious Historic Centre");
vp35.add("Spiritual Historic Centre");

vp36.add("Judaism Holy City");
vp36.add("Judaism Historic Centre");
vp36.add("Religious Historic Centre");
vp36.add("Spiritual Historic Centre");

vp37.add("Christianity Holy City");
vp37.add("Christianity Historic Centre");
vp37.add("Religious Historic Centre");
vp37.add("Spiritual Historic Centre");

vp38.add("Islam Holy City");
vp38.add("Islam Historic Centre");
vp38.add("Religious Historic Centre");
vp38.add("Spiritual Historic Centre");

vp39.add("Fortified Town");
vp39.add("Military Historic Centre");

vp40.add("Historic Capital");
vp40.add("Politics Historic Centre");

vp41.add("Rural Settlement");
vp41.add("Resident Settlement");

vp42.add("Urban Settlement");
vp42.add("Resident Settlement");

vp43.add("Settlement in Certain Environment");
vp43.add("Urban Settlement");
vp43.add("Resident Settlement");

vp44.add("Settlement Restoration");
vp44.add("Urban Settlement");
vp44.add("Resident Settlement");

vp45.add("Urban Fabric");
vp45.add("Urban Settlement");
vp45.add("Resident Settlement");

vp46.add("Medieval Town");
vp46.add("Urban Settlement");
vp46.add("Resident Settlement");
```

```
vp47.add("Islamic City");
vp47.add("Urban Settlement");
vp47.add("Resident Settlement");

/* the second path for Colonial Settlement*/
vp23.add("Urban Settlement");
vp23.add("Resident Settlement");

 Concept node1 = new Concept("Economy Historic Centre", 1,vp1,0);
 Concept node2 = new Concept("Culture Historic Centre", 1,vp2,0);
 Concept node3 = new Concept("Spiritual Historic Centre", 1,vp3,0);
 Concept node4 = new Concept("Military Historic Centre", 1, vp4,0);
 Concept node5 = new Concept("Politics Historic Centre", 1, vp5,0);
 Concept node6 = new Concept("Resident Settlement", 1, vp6,0);
 Concept node7 = new Concept("Agriculture Centre", 2, vp7,0);
 Concept node8 = new Concept("Industry Historic Centre", 2, vp8,0);
Concept node9 = new Concept("Commerce Historic Centre", 2, vp9,0);
 Concept node10 = new Concept("Mining Town", 3, vp10,0);
 Concept node11 = new Concept("Company Town", 3, vp11,0);
 Concept node12 = new Concept("Industry Town", 3, vp12, 0);
 Concept node13 = new Concept("Wool Industry Complex", 4, vp13, 0);
 Concept node14 = new Concept("Copper Mines Industry Complex", 4, vp14, 0);
 Concept node15 = new Concept("Sugar Cane Industry Complex", 4, vp15, 0);
 Concept node16 = new Concept("Silver Mines Industry Complex", 4, vp16, 0);
 Concept node17 = new Concept("Gold Mines Industry Complex", 4, vp17, 0);
 Concept node18 = new Concept("Diamond Prospect Industry Complex", 4, vp18, 0);
 Concept node19 = new Concept("Trading Centre", 3, vp19, 0);
 Concept node20 = new Concept("Harbor", 3, vp20, 0);
 Concept node21 = new Concept("Culture Movement Witness", 2, vp21, 0);
 Concept node22 = new Concept("Culture Blending Witness", 2, vp22, 0);
 Concept node23 = new Concept("Colonial Settlement", 3, vp23, 0);
 Concept node24 = new Concept("Education Historic Centre", 2, vp24, 0);
 Concept node25 = new Concept("Aesthetics Historic Centre", 2, vp25, 0);
 Concept node26 = new Concept("Architecture Centre", 2, vp26, 0);
 Concept node27 = new Concept("University City", 3, vp27, 0);
 Concept node28 = new Concept("Music Centre", 3, vp28, 0);
 Concept node29 = new Concept("Monument Historic Centre", 2, vp29, 0);
 Concept node30 = new Concept("Religious Historic Centre", 2, vp30, 0);
 Concept node31 = new Concept("Judaism Historic Centre", 3, vp31, 0);
 Concept node32 = new Concept("Buddhism Historic Centre", 3, vp32, 0);
 Concept node33 = new Concept("Christianity Historic Centre", 3, vp33, 0);
 Concept node34 = new Concept("Islam Historic Centre", 3, vp34, 0);
 Concept node35 = new Concept("Religious Blending Witness", 3, vp35, 0);
 Concept node36 = new Concept("Judaism Holy City", 4, vp36, 0);
 Concept node37 = new Concept("Christianity Holy City", 4, vp37, 0);
 Concept node38 = new Concept("Islam Holy City", 4, vp38, 0);
 Concept node39 = new Concept("Fortified Town", 2, vp39, 0);
 Concept node40 = new Concept("Historic Capital", 2, vp40, 0);
 Concept node41 = new Concept("Rural Settlement", 2, vp41, 0);
 Concept node42 = new Concept("Urban Settlement", 2, vp42, 0);
 Concept node43 = new Concept("Settlement in Certain Environment", 3, vp43, 0);
 Concept node44 = new Concept("Settlement Restoration", 3, vp44, 0);
Concept node45 = new Concept("Urban Fabric", 3, vp45, 0);
Concept node46 = new Concept("Medieval Town", 3, vp46, 0);
Concept node47 = new Concept("Islamic City", 3, vp47, 0);


vl1.add(node1);
vl1.add(node2);
vl1.add(node3);
vl1.add(node4);
vl1.add(node5);
vl1.add(node6);
vl1.add(node7);
vl1.add(node8);
vl1.add(node9);
vl1.add(node10);
vl1.add(node11);
vl1.add(node12);
vl1.add(node13);
vl1.add(node14);
vl1.add(node15);
vl1.add(node16);
vl1.add(node17);
vl1.add(node18);
vl1.add(node19);
vl1.add(node20);
vl1.add(node21);
vl1.add(node22);
vl1.add(node23);
vl1.add(node24);
vl1.add(node25);
vl1.add(node26);
vl1.add(node27);
vl1.add(node28);
vl1.add(node29);
vl1.add(node30);
vl1.add(node31);
vl1.add(node32);
vl1.add(node33);
vl1.add(node34);
vl1.add(node35);
vl1.add(node36);
vl1.add(node37);
vl1.add(node38);
vl1.add(node39);
vl1.add(node40);
vl1.add(node41);
vl1.add(node42);
vl1.add(node43);
vl1.add(node44);
vl1.add(node45);
vl1.add(node46);
vl1.add(node47);
}
```

```
else if("Artistic Work".equals(cgn))
{
vp1.add("Caving");
vp2.add("Painting");
vp3.add("Mosaic");
vp4.add("Rock Art");

Concept node1 = new Concept("Caving", 1,vp1,0);
Concept node2 = new Concept("Painting", 1,vp2,0);
Concept node3 = new Concept("Mosaic", 1,vp3,0);
Concept node4 = new Concept("Rock Art", 1, vp4,0);

vl1.add(node1);
vl1.add(node2);
vl1.add(node3);
vl1.add(node4);
}
else if("Culture Landscape".equals(cgn))
{
vp1.add("Clearly Defined Culture Landscape Designed and Created by Man");
vp2.add("Organically Evolved Landscape");
vp3.add("Associative Culture Landscape");

vp4.add("park");
vp4.add("Clearly Defined Culture Landscape Designed and Created by Man");

vp5.add("Garden");
vp5.add("Clearly Defined Culture Landscape Designed and Created by Man");

vp6.add("Industry Landscape");
vp6.add("Clearly Defined Culture Landscape Designed and Created by Man");

vp7.add("Agriculture Landscape");
vp7.add("Associative Culture Landscape");

Concept node1 = new Concept("Clearly Defined Culture Landscape Designed and
Created by Man",    1,vp1,0);
Concept node2 = new Concept("Organically Evolved Landscape", 1,vp2,0);
Concept node3 = new Concept("Associative Culture Landscape", 1,vp3,0);
Concept node4 = new Concept("Park", 2, vp4,0);
Concept node5 = new Concept("Garden", 2,vp1,0);
Concept node6 = new Concept("Industry Landscape", 2,vp2,0);
Concept node7 = new Concept("Agriculture Landscape", 2, vp4,0);

vl1.add(node1);
vl1.add(node2);
vl1.add(node3);
vl1.add(node4);
vl1.add(node5);
vl1.add(node6);
vl1.add(node7);
}
else if("Construction".equals(cgn))
```

```
{
 vp1.add("Economy Construction");
vp2.add("Cultural Construction");
vp3.add("Spiritual Construction");
vp4.add("Military Construction");
vp5.add("Politics Construction");
vp6.add("Resident Construction");

vp7.add("Agriculture Construction");
vp7.add("Economy Construction");

vp8.add("Commerce Construction");
vp8.add("Economy Construction");

vp9.add("Industry Construction");
vp9.add("Economy Construction");

vp10.add("Transport Construction");
vp10.add("Economy Construction");

vp11.add("Public Facility");
vp11.add("Culture Construction");

vp12.add("Culture Blending Construction");
vp12.add("Culture Construction");

vp13.add("Tower");
vp13.add("Military Construction");
vp13.add("Monument Construction");
vp13.add("Spiritual Construction");

vp14.add("Fortress");
vp14.add("Military Construction");

vp15.add("Wall");
vp15.add("Military Construction");

vp16.add("Castle");
vp16.add("Military Construction");
vp16.add("Resident Construction");

vp17.add("Palace");
vp17.add("Resident Construction");

vp18.add("Mansion");
vp18.add("Resident Construction");

vp19.add("Religious Construction");
vp19.add("Spiritual Construction");

vp20.add("Monument Construction");
vp20.add("Spiritual construction");
```

```
vp21.add("Buddhism Construction");
vp21.add("Religious Construction");
vp21.add("Spiritual Construction");

vp22.add("Christianity Construction");
vp22.add("Religious Construction");
vp22.add("Spiritual Construction");

vp23.add("Islam Construction");
vp23.add("Religious Construction");
vp23.add("Spiritual Construction");

vp24.add("Tomb");
vp24.add("Monument Construction");
vp24.add("Spiritual Construction");

vp25.add("Temple");
                    vp25.add("Religious Construction");
vp25.add("Spiritual Construction");

vp26.add("Monastery");
                    vp26.add("Religious Construction");
vp26.add("Spiritual Construction");

vp27.add("Abbey");
                    vp27.add("Religious Construction");
vp27.add("Spiritual Construction");

vp32.add("Convent"); // added late
vp32.add("Religious Construction");
vp32.add("Spiritual Construction");

vp28.add("Church");
vp28.add("Christianity Construction");
vp28.add("Religious Construction");
vp28.add("Spiritual Construction");

vp29.add("Cathedral");
vp29.add("Christianity Construction");
vp29.add("Religious Construction");
vp29.add("Spiritual Construction");

vp30.add("Mosque");
vp30.add("Islam Construction");
vp30.add("Religious Construction");
vp30.add("Spiritual Construction");

vp31.add("Garden-Tomb");
vp31.add("Tomb");
vp31.add("Monument Construction");
vp31.add("Spiritual Construction");

Concept node1 = new Concept("Economy Construction", 1,vp1,0);
Concept node2 = new Concept("Culture Construction", 1,vp2,0);
Concept node3 = new Concept("Spiritual Construction", 1,vp3,0);
Concept node4 = new Concept("Military Construction", 1,vp4,0);
Concept node5 = new Concept("Politics Construction", 1,vp5,0);
Concept node6 = new Concept("Resident Construction", 1,vp6,0);

Concept node7 = new Concept("Agriculture Construction", 2,vp7,0);
Concept node8 = new Concept("Commerce Construction", 2,vp8,0);
Concept node9 = new Concept("Industry Construction", 2, vp9, 0);
Concept node10 = new Concept("Transport Construction", 2, vp10, 0);

Concept node11 = new Concept("Public Facility", 2,vp11,0);
Concept node12 = new Concept("Culture Blending Construction", 2,vp12,0);

Concept node13 = new Concept("Tower", 2,vp13,0);
Concept node14 = new Concept("Fortress", 2,vp14,0);
Concept node15 = new Concept("Wall", 2,vp15,0);
Concept node16 = new Concept("Castle", 2,vp16,0);
Concept node17 = new Concept("Palace", 2,vp17,0);
Concept node18 = new Concept("Mansion", 2,vp18,0);

Concept node19 = new Concept("Religious Construction", 2,vp19,0);
Concept node20 = new Concept("Monument Construction", 2,vp20,0);

Concept node21 = new Concept("Buddhism Construction", 3,vp21,0);
Concept node22 = new Concept("Christianity Construction", 3,vp22,0);
Concept node23 = new Concept("Islam Construction", 3,vp23,0);
Concept node24 = new Concept("Tomb", 3,vp24,0);

Concept node25 = new Concept("Temple", 3,vp25,0);
Concept node26 = new Concept("Monastery", 3,vp26,0);
Concept node27 = new Concept("Abbey", 3,vp27,0);
Concept node32 = new Concept("Convent", 3, vp32, 0);

Concept node28 = new Concept("Church", 4,vp28,0);
Concept node29 = new Concept("Cathedral", 4,vp29,0);
Concept node30 = new Concept("Mosque", 4,vp30,0);
Concept node31 = new Concept("Garden-Tomb", 4,vp31,0);

vl1.add(node1);
vl1.add(node2);
vl1.add(node3);
vl1.add(node4);
vl1.add(node5);
vl1.add(node6);
vl1.add(node7);
vl1.add(node8);
vl1.add(node9);
vl1.add(node10);
vl1.add(node11);
vl1.add(node12);
vl1.add(node13);
vl1.add(node14);
```

```
vl1.add(node15);
vl1.add(node16);
vl1.add(node17);
vl1.add(node18);
vl1.add(node19);
vl1.add(node20);
vl1.add(node21);
vl1.add(node22);
vl1.add(node23);
vl1.add(node24);
vl1.add(node25);
vl1.add(node26);
vl1.add(node27);
vl1.add(node28);
vl1.add(node29);
vl1.add(node30);
vl1.add(node31);
vl1.add(node32);
}
else if ("Cultural Sites".equals(cgn))
{
 vp1.add("Historic Cities and Areas");
 vp2.add("Construction");
 vp3.add("Artistic Work");
 vp4.add("Hominid Sites");
 vp5.add("Archaeological Sites");
 vp6.add("Culture Landscape");

 vp7.add("Economy Historic centre");
 vp7.add("Historic Cities and Areas");

vp8.add("Culture Historic Centre");
 vp8.add("Historic Cities and Areas");

vp9.add("Spiritual Historic Centre");
 vp9.add("Historic Cities and Areas");

 vp10.add("Military Historic Centre");
 vp10.add("Historic Cities and Areas");

 vp11.add("Politics Historic Centre");
 vp11.add("Historic Cities and Areas");

 vp12.add("Resident Settlement");
 vp12.add("Historic Cities and Areas");

 vp13.add("Economy Construction");
 vp13.add("Construction");

vp14.add("Cultural Construction");
 vp14.add("Construction");

vp15.add("Spiritual Construction");
```

```
 vp15.add("Construction");

vp16.add("Military Construction");
 vp16.add("Construction");

vp17.add("Politics Construction");
 vp17.add("Construction");

vp18.add("Resident Construction");
 vp18.add("Construction");

 vp19.add("Caving");
 vp19.add("Artistic Work");

 vp20.add("Painting");
 vp20.add("Artistic Work");

 vp21.add("Mosaic");
 vp21.add("Artistic Work");

 vp22.add("Clearly Defined Culture Landscape Designed and Created by Man");
 vp22.add("Culture Landscape");

 vp23.add("Organically Evolved Landscape");
 vp23.add("Culture Landscape");

 vp24.add("Associative Culture Landscape");
 vp24.add("Culture Landscape");

 vp25.add("Agriculture Historic Centre");
vp25.add("Economy Historic centre");
vp25.add("Historic Cities and Areas");

vp26.add("Industry Historic Centre");
vp26.add("Economy Historic centre");
vp25.add("Historic Cities and Areas");

vp27.add("Commerce Historic Centre");
vp27.add("Economy Historic centre");
vp27.add("Historic Cities and Areas");

vp28.add("Mining Town");
vp28.add("Industry Historic Centre");
vp28.add("Economy Historic centre");
vp28.add("Historic Cities and Areas");

vp29.add("Company Town");
vp29.add("Industry Historic Centre");
vp29.add("Economy Historic centre");
vp29.add("Historic Cities and Areas");

vp30.add("Industry Town");
vp30.add("Commerce Historic Centre");
```

```
vp30.add("Economy Historic centre");
vp30.add("Historic Cities and Areas");

vp31.add("Wool Industry Complex");
vp31.add("Industry Town");
vp31.add("Industry Historic Centre");
vp31.add("Economy Historic centre");
vp31.add("Historic Cities and Areas");

vp32.add("Copper Mines Industry Complex");
vp32.add("Mining Town");
vp32.add("Industry Historic Centre");
vp32.add("Economy Historic centre");
vp32.add("Historic Cities and Areas");

vp33.add("Sugar Cane Industry Complex");
vp31.add("Industry Town");
vp33.add("Industry Historic Centre");
vp33.add("Economy Historic centre");
vp33.add("Historic Cities and Areas");

vp34.add("Silver Mines Industry Complex");
vp34.add("Mining Town");
vp34.add("Industry Historic Centre");
vp34.add("Economy Historic centre");
vp34.add("Historic Cities and Areas");

vp35.add("Gold Mines Industry Complex");
vp35.add("Mining Town");
vp35.add("Industry Historic Centre");
vp35.add("Economy Historic centre");
vp35.add("Historic Cities and Areas");

vp36.add("Diamond Prospect Industry Complex");
vp36.add("Mining Town");
vp36.add("Industry Historic Centre");
vp36.add("Economy Historic centre");
vp36.add("Historic Cities and Areas");

vp37.add("Trading Centre");
vp37.add("Commerce Historic Centre");
vp37.add("Economy Historic centre");
vp37.add("Historic Cities and Areas");


vp38.add("Harbor");
vp38.add("Commerce Historic Centre");
vp38.add("Economy Historic centre");
vp38.add("Historic Cities and Areas");

vp39.add("Culture Movement Witness");
vp39.add("Culture Historic Centre");
 vp39.add("Historic Cities and Areas");
```

```
vp40.add("Culture Blending Witness");
vp40.add("Culture Historic Centre");
vp40.add("Historic Cities and Areas");

vp41.add("Colonial Settlement"); /*two pathes */
vp41.add("Culture Blending Witness");
vp41.add("Culture Historic Centre");
vp41.add("Urban Settlement");
vp41.add("Resident Settlement");
vp41.add("Historic Cities and Areas");

vp42.add("Education Historic Centre");
vp42.add("Culture Historic Centre");
vp42.add("Historic Cities and Areas");

vp43.add("Aesthetic Historic Centre");
vp43.add("Culture Historic Centre");
vp43.add("Historic Cities and Areas");

vp44.add("Architecture Centre");
vp44.add("Culture Historic Centre");
vp44.add("Historic Cities and Areas");

vp45.add("University City");
vp45.add("Education Historic Centre");
vp45.add("Culture Historic Centre");
vp45.add("Historic Cities and Areas");

vp46.add("Music Centre");
vp46.add("Aesthetic Historic Centre");
vp46.add("Culture Historic Centre");
vp46.add("Historic Cities and Areas");


vp47.add("Monument Historic Centre");
vp47.add("Spiritual Historic Centre");
 vp47.add("Historic Cities and Areas");

vp48.add("Religious Historic Centre");
vp48.add("Spiritual Historic Centre");
 vp48.add("Historic Cities and Areas");

vp49.add("Judaism Historic Centre");
vp49.add("Religious Historic Centre");
vp49.add("Spiritual Historic Centre");
 vp49.add("Historic Cities and Areas");

vp50.add("Buddhism Historic Centre");
vp50.add("Religious Historic Centre");
vp50.add("Spiritual Historic Centre");
vp50.add("Historic Cities and Areas");
```

```
vp51.add("Christianity Historic Centre");
vp51.add("Religious Historic Centre");
vp51.add("Spiritual Historic Centre");
vp51.add("Historic Cities and Areas");

vp52.add("Islam Historic Centre");
vp52.add("Religious Historic Centre");
vp52.add("Spiritual Historic Centre");
vp52.add("Historic Cities and Areas");

vp53.add("Religious Blending Witness");
vp53.add("Religious Historic Centre");
vp53.add("Spiritual Historic Centre");
vp53.add("Historic Cities and Areas");

vp54.add("Judaism Holy City");
vp54.add("Judaism Historic Centre");
vp54.add("Religious Historic Centre");
vp54.add("Spiritual Historic Centre");
vp54.add("Historic Cities and Areas");

vp55.add("Christianity Holy City");
vp55.add("Christianity Historic Centre");
vp55.add("Religious Historic Centre");
vp55.add("Spiritual Historic Centre");
vp55.add("Historic Cities and Areas");

vp56.add("Islam Holy City");
vp56.add("Islam Historic Centre");
vp56.add("Religion Historic Centre");
vp56.add("Spirit Historic Centre");
vp56.add("Historic Cities and Areas");

vp57.add("Fortified Town");
vp57.add("Military Historic Centre");
vp57.add("Historic Cities and Areas");


vp58.add("Historic Capital");
vp58.add("Politics Historic Centre");
vp58.add("Historic Cities and Areas");


vp59.add("Rural Settlement");
vp59.add("Resident Settlement");
vp59.add("Historic Cities and Areas");

vp60.add("Urban Settlement");
vp60 .add("Resident Settlement");
vp60.add("Historic Cities and Areas");


vp61.add("Settlement in Certain Environment");
```

```
vp61.add("Urban Settlement");
vp61.add("Resident Settlement");
vp61.add("Historic Cities and Areas");

vp95.add("Settlement Restoration");
vp95.add("Urban Settlement");
vp95.add("Resident Settlement");
vp95.add("Historic Cities and Areas");


vp62.add("Urban Fabric");
vp62.add("Urban Settlement");
vp62.add("Resident Settlement");
vp62.add("Historic Cities and Areas");

vp63.add("Medieval Town");
vp63.add("Urban Settlement");
vp63.add("Resident Settlement");
vp63.add("Historic Cities and Areas");

vp64.add("Islamic City");
vp64.add("Urban Settlement");
vp64.add("Resident Settlement");
vp64.add("Historic Cities and Areas");

 vp65.add("Rock Art");
 vp65.add("Artistic Work");

vp66.add("park");
 vp66.add("Clearly Defined Culture Landscape Designed and Created by Man");
 vp66.add("Culture Landscape");

 vp67.add("Garden");
 vp67.add("Clearly Defined Culture Landscape Designed and Created by Man");
 vp67.add("Culture Landscape");

 vp68.add("Industry Landscape");
 vp68.add("Clearly Defined Culture Landscape Designed and Created by Man");
 vp68.add("Culture Landscape");

 vp69.add("Agriculture Landscape");
 vp69.add("Associative Culture Landscape");
 vp69.add("Culture Landscape");

 vp70.add("Agriculture Construction");
vp70.add("Economy Construction");
vp70.add("Construction");

vp71.add("Commerce Construction");
vp71.add("Economy Construction");
vp71.add("Construction");

vp72.add("Industry Construction");
```

```
vp72.add("Economy Construction");
vp72.add("Construction");

vp73.add("Transport Construction");
vp73.add("Economy Construction");
 vp73.add("Construction");

vp74.add("Public Facility");
vp74.add("Culture Construction");
  vp74.add("Construction");

vp75.add("Culture Blending Construction");
vp75.add("Culture Construction");
 vp75.add("Construction");

vp76.add("Tower");
vp76.add("Military Construction");
vp76.add("Monument Construction");
vp76.add("Spiritual Construction");
vp76.add("Construction");

vp77.add("Fortress");
vp77.add("Military Construction");
vp77.add("Construction");

vp78.add("Wall");
vp78.add("Military Construction");
  vp78.add("Construction");

vp79.add("Castle");
vp79.add("Military Construction");
vp79.add("Resident Construction");
 vp79.add("Construction");

vp80.add("Palace");
vp80.add("Resident Construction");
 vp80.add("Construction");

vp81.add("Mansion");
vp81.add("Resident Construction");
vp81.add("Construction");

vp82.add("Religious Construction");
vp82.add("Spiritual Construction");
vp82.add("Construction");

vp83.add("Monument Construction");
vp83.add("Spiritual construction");
vp83.add("Construction");

vp84.add("Buddhism Construction");
vp84.add("Religious Construction");
vp84.add("Spirit Construction");
```

```
vp84.add("Construction");

vp85.add("Christianity Construction");
vp85.add("Religious Construction");
vp85.add("Spiritual Construction");
vp85.add("Construction");

vp86.add("Islam Construction");
vp86.add("Religious Construction");
vp86.add("Spiritual Construction");
vp86.add("Construction");

vp87.add("Tomb");
vp87.add("Monument Construction");
vp87.add("Spiritual Construction");
vp87.add("Construction");

vp88.add("Temple");
vp88.add("Religious Construction");
vp88.add("Spiritual Construction");
vp88.add("Construction");

vp89.add("Monastery");
vp89.add("Religious Construction");
vp89.add("Spiritual Construction");
vp89.add("Construction");

vp96.add("Convent"); // added late
vp96.add("Religious Construction");
vp96.add("Spiritual Construction");
vp96.add("Construction");


vp90.add("Abbey");
vp90.add("Religious Construction");
vp90.add("Spiritual Construction");
vp90.add("Construction");

vp91.add("Church");
vp91.add("Christianity Construction");
vp91.add("Religious Construction");
vp91.add("Spiritual Construction");
vp91.add("Construction");

vp92.add("Cathedral");
vp92.add("Christianity Construction");
vp92.add("Religious Construction");
vp92.add("Spiritual Construction");
vp92.add("Construction");

vp93.add("Mosque");
vp93.add("Islam Construction");
vp93.add("Religious Construction");
```

```
        vp93.add("Spiritual Construction");
        vp93.add("Construction");

        vp94.add("Garden-Tomb");
        vp94.add("Tomb");
        vp94.add("Monument Construction");
        vp94.add("Spiritual Construction");
        vp94.add("Construction");

        Concept node1 = new Concept("Historic Cities and Areas", 1,vp1,0);
        Concept node2 = new Concept("Construction", 1,vp2,0);
        Concept node3 = new Concept("Artistic Work", 1,vp3,0);
        Concept node4 = new Concept("Hominid Sites", 1,vp4,0);
        Concept node5 = new Concept("Archaeological", 1,vp5,0);
        Concept node6 = new Concept("Culture Landscape", 1,vp6,0);

        Concept node7 = new Concept("Economy Historic Centre", 2,vp7,0);
        Concept node8 = new Concept("Culture Historic Centre", 2,vp8,0);
        Concept node9 = new Concept("Spiritual Historic Centre", 2,vp9,0);
        Concept node10 = new Concept("Military Historic Centre", 2, vp10,0);
        Concept node11 = new Concept("Politics Historic Centre", 2, vp11,0);
        Concept node12 = new Concept("Resident Settlement", 2, vp12,0);

        Concept node13 = new Concept("Economy Construction", 2,vp13,0);
        Concept node14 = new Concept("Culture Construction", 2,vp14,0);
        Concept node15 = new Concept("Spiritual Construction", 2,vp15,0);
        Concept node16 = new Concept("Military Construction", 2,vp16,0);
        Concept node17 = new Concept("Politics Construction", 2,vp17,0);
        Concept node18 = new Concept("Resident Construction", 2,vp18,0);

        Concept node19 = new Concept("Caving", 2,vp19,0);
        Concept node20 = new Concept("Painting", 2,vp20,0);
        Concept node21 = new Concept("Mosaic", 2,vp21,0);

        Concept node22 = new Concept("Clearly Defined Culture Landscape Designed and
Created by Man", 2,vp22,0);
        Concept node23 = new Concept("Organically Evolved Landscape", 2,vp23,0);
        Concept node24 = new Concept("Associative Culture Landscape", 2,vp24,0);


        Concept node25 = new Concept("Agriculture Historic Centre", 3, vp25,0);
        Concept node26 = new Concept("Industry Historic Centre", 3, vp26,0);
        Concept node27 = new Concept("Commerce Historic Centre", 3, vp27,0);
        Concept node28 = new Concept("Mining Town", 4, vp28,0);
        Concept node29 = new Concept("Company Town", 4, vp29,0);
        Concept node30 = new Concept("Industry Town", 4, vp30, 0);
        Concept node31 = new Concept("Wool Industry Complex", 5, vp31, 0);
        Concept node32 = new Concept("Copper Mines Industry Complex", 5, vp32, 0);
        Concept node33 = new Concept("Sugar Cane Industry Complex", 5, vp33, 0);
        Concept node34 = new Concept("Silver Mines Industry Complex", 5, vp34, 0);
        Concept node35 = new Concept("Gold Mines Industry Complex", 5, vp35, 0);
        Concept node36 = new Concept("Diamond Prospect Industry Complex", 5, vp36, 0);
        Concept node37 = new Concept("Trading Centre", 4, vp37, 0);
Concept node38 = new Concept("Harbor", 4, vp38, 0);
Concept node39 = new Concept("Culture Movement Witness", 3, vp39, 0);
Concept node40 = new Concept("Culture Blending Witness", 3, vp40, 0);
Concept node41 = new Concept("Colonial Settlement", 4, vp41, 0);
Concept node42 = new Concept("Education Historic Centre", 3, vp42, 0);
Concept node43 = new Concept("Aesthetics Historic Centre", 3, vp43, 0);
Concept node44 = new Concept("Architecture Centre", 3, vp44, 0);
Concept node45 = new Concept("University City", 4, vp45, 0);
Concept node46 = new Concept("Music Centre", 4, vp46, 0);
Concept node47 = new Concept("Monument Historic Centre", 3, vp47, 0);
Concept node48 = new Concept("Religious Historic Centre", 3, vp48, 0);
Concept node49 = new Concept("Judaism Historic Centre", 4, vp49, 0);
Concept node50 = new Concept("Buddhism Historic Centre", 4, vp50, 0);
Concept node51 = new Concept("Christianity Historic Centre", 4, vp51, 0);
Concept node52 = new Concept("Islam Historic Centre", 4, vp52, 0);
Concept node53 = new Concept("Religious Blending Witness", 4, vp53, 0);
Concept node54 = new Concept("Judaism Holy City", 5, vp54, 0);
Concept node55 = new Concept("Christianity Holy City", 5, vp55, 0);
Concept node56 = new Concept("Islam Holy City", 5, vp56, 0);
Concept node57 = new Concept("Fortified Town", 3, vp57, 0);
Concept node58 = new Concept("Historic Capital", 3, vp58, 0);
Concept node59 = new Concept("Rural Settlement", 3, vp59, 0);
Concept node60 = new Concept("Urban Settlement", 3, vp60, 0);
Concept node61 = new Concept("Settlement in Certain Environment", 4, vp61, 0);
Concept node95 = new Concept("Settlement Restoration", 4, vp95, 0);
Concept node62 = new Concept("Urban Fabric", 4, vp62, 0);
Concept node63 = new Concept("Medieval Town", 4, vp63, 0);
Concept node64 = new Concept("Islamic City", 4, vp64, 0);
Concept node65 = new Concept("Rock Art", 2, vp65,0);
Concept node66 = new Concept("Park", 3, vp66,0);
Concept node67 = new Concept("Garden", 3,vp67,0);
Concept node68 = new Concept("Industry Landscape", 3,vp68,0);
Concept node69 = new Concept("Agriculture Landscape", 3, vp69,0);
Concept node70 = new Concept("Agriculture Construction", 3,vp70,0);
Concept node71 = new Concept("Commerce Construction", 3,vp71,0);
Concept node72 = new Concept("Industry Construction", 3, vp72, 0);
Concept node73 = new Concept("Transport Construction", 3, vp73, 0);
Concept node74 = new Concept("Public Facility", 3,vp74,0);
Concept node75 = new Concept("Culture Blending Construction", 3,vp75,0);
Concept node76 = new Concept("Tower", 3,vp76,0);
Concept node77 = new Concept("Fortress", 3,vp77,0);
Concept node78 = new Concept("Wall", 3,vp78,0);
Concept node79 = new Concept("Castle", 3,vp79,0);
Concept node80 = new Concept("Palace", 3,vp80,0);
Concept node81 = new Concept("Mansion", 3,vp81,0);

Concept node82 = new Concept("Religious Construction", 3,vp82,0);
Concept node83 = new Concept("Monument", 3,vp83,0);
Concept node84 = new Concept("Buddhism Construction", 4,vp84,0);
Concept node85 = new Concept("Christianity Construction", 4,vp85,0);
Concept node86 = new Concept("Islam Construction", 4,vp86,0);
Concept node87 = new Concept("Tomb", 4,vp87,0);
Concept node88 = new Concept("Temple", 4,vp88,0);
Concept node89 = new Concept("Monastery", 4,vp89,0);
Concept node90 = new Concept("Abbey", 4,vp90,0);
Concept node91 = new Concept("Church", 5,vp91,0);
Concept node92 = new Concept("Cathedral", 5,vp92,0);
Concept node93 = new Concept("Mosque", 5,vp93,0);
Concept node94 = new Concept("Garden-Tomb", 5,vp94,0);
Concept node96 = new Concept("Convent", 4, vp96, 0);

vl1.add(node1);
vl1.add(node2);
vl1.add(node3);
vl1.add(node4);
vl1.add(node5);
vl1.add(node6);
vl1.add(node7);
vl1.add(node8);
vl1.add(node9);
vl1.add(node10);
vl1.add(node11);
vl1.add(node12);
vl1.add(node13);
vl1.add(node14);
vl1.add(node15);
vl1.add(node16);
vl1.add(node17);
vl1.add(node18);
vl1.add(node19);
vl1.add(node20);
vl1.add(node21);
vl1.add(node22);
vl1.add(node23);
vl1.add(node24);
vl1.add(node25);
vl1.add(node26);
vl1.add(node27);
vl1.add(node28);
vl1.add(node29);
vl1.add(node30);
vl1.add(node31);
vl1.add(node32);
vl1.add(node33);
vl1.add(node34);
vl1.add(node35);
vl1.add(node36);
vl1.add(node37);
vl1.add(node38);
vl1.add(node39);
vl1.add(node40);
vl1.add(node41);
vl1.add(node42);
vl1.add(node43);
vl1.add(node44);
vl1.add(node45);
```

```
vl1.add(node46);
vl1.add(node47);
 vl1.add(node48);
vl1.add(node49);
vl1.add(node50);
vl1.add(node51);
vl1.add(node52);
vl1.add(node53);
vl1.add(node54);
vl1.add(node55);
vl1.add(node56);
vl1.add(node57);
vl1.add(node58);
vl1.add(node59);
vl1.add(node60);
vl1.add(node61);
 vl1.add(node95);
vl1.add(node62);
vl1.add(node63);
vl1.add(node64);
vl1.add(node65);
vl1.add(node66);
vl1.add(node67);
vl1.add(node68);
vl1.add(node69);
vl1.add(node70);
vl1.add(node71);
vl1.add(node72);
vl1.add(node73);
vl1.add(node74);
vl1.add(node75);
vl1.add(node76);
vl1.add(node77);
vl1.add(node78);
vl1.add(node79);
vl1.add(node80);
vl1.add(node81);
vl1.add(node82);
vl1.add(node83);
vl1.add(node84);
vl1.add(node85);
vl1.add(node86);
vl1.add(node87);
vl1.add(node88);
vl1.add(node89);
vl1.add(node90);
vl1.add(node91);
vl1.add(node92);
vl1.add(node93);
vl1.add(node94);
vl1.add(node96);

}
```

```
else if ("All Sites".equals(cgn))
{

 vp1.add("Historic Cities and Areas");
vp2.add("Construction");
vp3.add("Artistic Work");
 vp4.add("Hominid Sites");
 vp5.add("Archaeological Sites");
 vp6.add("Culture Landscape");

 vp7.add("Economy Historic centre");
 vp7.add("Historic Cities and Areas");

  vp8.add("Culture Historic Centre");
 vp8.add("Historic Cities and Areas");

  vp9.add("Spiritual Historic Centre");
 vp9.add("Historic Cities and Areas");

 vp10.add("Military Historic Centre");
 vp10.add("Historic Cities and Areas");

 vp11.add("Politics Historic Centre");
 vp11.add("Historic Cities and Areas");

 vp12.add("Resident Settlement");
 vp12.add("Historic Cities and Areas");

 vp13.add("Economy Construction");
 vp13.add("Construction");

vp14.add("Cultural Construction");
 vp14.add("Construction");

vp15.add("Spiritual Construction");
 vp15.add("Construction");

vp16.add("Military Construction");
 vp16.add("Construction");

  vp17.add("Politics Construction");
 vp17.add("Construction");

vp18.add("Resident Construction");
 vp18.add("Construction");

 vp19.add("Caving");
 vp19.add("Artistic Work");

 vp20.add("Painting");
 vp20.add("Artistic Work");

 vp21.add("Mosaic");
 vp21.add("Artistic Work");

 vp22.add("Clearly Defined Culture Landscape Designed and Created by Man");
 vp22.add("Culture Landscape");

 vp23.add("Organically Evolved Landscape");
 vp23.add("Culture Landscape");

 vp24.add("Associative Culture Landscape");
 vp24.add("Culture Landscape");

 vp25.add("Agriculture Historic Centre");
vp25.add("Economy Historic centre");
vp25.add("Historic Cities and Areas");

vp26.add("Industry Historic Centre");
vp26.add("Economy Historic centre");
vp25.add("Historic Cities and Areas");

vp27.add("Commerce Historic Centre");
vp27.add("Economy Historic centre");
vp27.add("Historic Cities and Areas");

vp28.add("Mining Town");
vp28.add("Industry Historic Centre");
vp28.add("Economy Historic centre");
vp28.add("Historic Cities and Areas");

vp29.add("Company Town");
vp29.add("Industry Historic Centre");
vp29.add("Economy Historic centre");
vp29.add("Historic Cities and Areas");

vp30.add("Industry Town");
vp30.add("Commerce Historic Centre");
vp30.add("Economy Historic centre");
vp30.add("Historic Cities and Areas");

vp31.add("Wool Industry Complex");
vp31.add("Industry Town");
vp31.add("Industry Historic Centre");
vp31.add("Economy Historic centre");
vp31.add("Historic Cities and Areas");

vp32.add("Copper Mines Industry Complex");
vp32.add("Mining Town");
vp32.add("Industry Historic Centre");
vp32.add("Economy Historic centre");
vp32.add("Historic Cities and Areas");

vp33.add("Sugar Cane Industry Complex");
vp31.add("Industry Town");
vp33.add("Industry Historic Centre");
```

```
vp33.add("Economy Historic centre");
vp33.add("Historic Cities and Areas");

vp34.add("Silver Mines Industry Complex");
vp34.add("Mining Town");
vp34.add("Industry Historic Centre");
vp34.add("Economy Historic centre");
vp34.add("Historic Cities and Areas");

vp35.add("Gold Mines Industry Complex");
vp35.add("Mining Town");
vp35.add("Industry Historic Centre");
vp35.add("Economy Historic centre");
vp35.add("Historic Cities and Areas");

vp36.add("Diamond Prospect Industry Complex");
vp36.add("Mining Town");
vp36.add("Industry Historic Centre");
vp36.add("Economy Historic centre");
vp36.add("Historic Cities and Areas");

vp37.add("Trading Centre");
vp37.add("Commerce Historic Centre");
vp37.add("Economy Historic centre");
vp37.add("Historic Cities and Areas");


vp38.add("Harbor");
vp38.add("Commerce Historic Centre");
vp38.add("Economy Historic centre");
vp38.add("Historic Cities and Areas");

vp39.add("Culture Movement Witness");
vp39.add("Culture Historic Centre");
 vp39.add("Historic Cities and Areas");

vp40.add("Culture Blending Witness");
vp40.add("Culture Historic Centre");
vp40.add("Historic Cities and Areas");

vp41.add("Colonial Settlement"); /*two pathes */
vp41.add("Culture Blending Witness");
vp41.add("Culture Historic Centre");
vp41.add("Urban Settlement");
vp41.add("Resident Settlement");
vp41.add("Historic Cities and Areas");

vp42.add("Education Historic Centre");
vp42.add("Culture Historic Centre");
vp42.add("Historic Cities and Areas");

vp43.add("Aesthetic Historic Centre");
vp43.add("Culture Historic Centre");
```

```
vp43.add("Historic Cities and Areas");

vp44.add("Architecture Centre");
vp44.add("Culture Historic Centre");
vp44.add("Historic Cities and Areas");

vp45.add("University City");
vp45.add("Education Historic Centre");
vp45.add("Culture Historic Centre");
vp45.add("Historic Cities and Areas");

vp46.add("Music Centre");
vp46.add("Aesthetic Historic Centre");
vp46.add("Culture Historic Centre");
vp46.add("Historic Cities and Areas");


vp47.add("Monument Historic Centre");
vp47.add("Spiritual Historic Centre");
vp47.add("Historic Cities and Areas");

vp48.add("Religious Historic Centre");
vp48.add("Spiritual Historic Centre");
vp48.add("Historic Cities and Areas");

vp49.add("Judaism Historic Centre");
vp49.add("Religious Historic Centre");
vp49.add("Spiritual Historic Centre");
vp49.add("Historic Cities and Areas");

vp50.add("Buddhism Historic Centre");
vp50.add("Religious Historic Centre");
vp50.add("Spiritual Historic Centre");
vp50.add("Historic Cities and Areas");

vp51.add("Christianity Historic Centre");
vp51.add("Religious Historic Centre");
vp51.add("Spiritual Historic Centre");
vp51.add("Historic Cities and Areas");

vp52.add("Islam Historic Centre");
vp52.add("Religious Historic Centre");
vp52.add("Spiritual Historic Centre");
vp52.add("Historic Cities and Areas");

vp53.add("Religious Blending Witness");
vp53.add("Religious Historic Centre");
vp53.add("Spiritual Historic Centre");
vp53.add("Historic Cities and Areas");

vp54.add("Judaism Holy City");
vp54.add("Judaism Historic Centre");
vp54.add("Religious Historic Centre");
```

```
vp54.add("Spiritual Historic Centre");
 vp54.add("Historic Cities and Areas");

vp55.add("Christianity Holy City");
vp55.add("Christianity Historic Centre");
vp55.add("Religious Historic Centre");
vp55.add("Spiritual Historic Centre");
vp55.add("Historic Cities and Areas");

vp56.add("Islam Holy City");
vp56.add("Islam Historic Centre");
vp56.add("Religion Historic Centre");
vp56.add("Spirit Historic Centre");
 vp56.add("Historic Cities and Areas");

vp57.add("Fortified Town");
vp57.add("Military Historic Centre");
 vp57.add("Historic Cities and Areas");


vp58.add("Historic Capital");
 vp58.add("Politics Historic Centre");
 vp58.add("Historic Cities and Areas");


vp59.add("Rural Settlement");
vp59.add("Resident Settlement");
 vp59.add("Historic Cities and Areas");

vp60.add("Urban Settlement");
vp60 .add("Resident Settlement");
 vp60.add("Historic Cities and Areas");


vp61.add("Settlement in Certain Environment");
vp61.add("Urban Settlement");
vp61.add("Resident Settlement");
 vp61.add("Historic Cities and Areas");

 vp95.add("Settlement Restoration");
vp95.add("Urban Settlement");
vp95.add("Resident Settlement");
 vp95.add("Historic Cities and Areas");


vp62.add("Urban Fabric");
vp62.add("Urban Settlement");
vp62.add("Resident Settlement");
 vp62.add("Historic Cities and Areas");

vp63.add("Medieval Town");
vp63.add("Urban Settlement");
vp63.add("Resident Settlement");
```

```
 vp63.add("Historic Cities and Areas");

vp64.add("Islamic City");
vp64.add("Urban Settlement");
vp64.add("Resident Settlement");
 vp64.add("Historic Cities and Areas");

 vp65.add("Rock Art");
 vp65.add("Artistic Work");

vp66.add("park");
 vp66.add("Clearly Defined Culture Landscape Designed and Created by Man");
 vp66.add("Culture Landscape");

 vp67.add("Garden");
 vp67.add("Clearly Defined Culture Landscape Designed and Created by Man");
 vp67.add("Culture Landscape");
vp68.add("Industry Landscape");
vp68.add("Clearly Defined Culture Landscape Designed and Created by Man");
vp68.add("Culture Landscape");

vp69.add("Agriculture Landscape");
vp69.add("Associative Culture Landscape");
vp69.add("Culture Landscape");

vp70.add("Agriculture Construction");
vp70.add("Economy Construction");
vp70.add("Construction");

vp71.add("Commerce Construction");
vp71.add("Economy Construction");
vp71.add("Construction");

vp72.add("Industry Construction");
vp72.add("Economy Construction");
vp72.add("Construction");

vp73.add("Transport Construction");
vp73.add("Economy Construction");
 vp73.add("Construction");

vp74.add("Public Facility");
vp74.add("Culture Construction");
 vp74.add("Construction");

vp75.add("Culture Blending Construction");
vp75.add("Culture Construction");
 vp75.add("Construction");

vp76.add("Tower");
vp76.add("Military Construction");
vp76.add("Monument Construction");
vp76.add("Spiritual Construction");
```

```
vp76.add("Construction");

vp77.add("Fortress");
vp77.add("Military Construction");
vp77.add("Construction");

vp78.add("Wall");
vp78.add("Military Construction");
 vp78.add("Construction");

vp79.add("Castle");
vp79.add("Military Construction");
vp79.add("Resident Construction");
 vp79.add("Construction");

vp80.add("Palace");
vp80.add("Resident Construction");
 vp80.add("Construction");

vp81.add("Mansion");
vp81.add("Resident Construction");
 vp81.add("Construction");

vp82.add("Religious Construction");
vp82.add("Spiritual Construction");
 vp82.add("Construction");

vp83.add("Monument Construction");
vp83.add("Spiritual construction");
 vp83.add("Construction");

vp84.add("Buddhism Construction");
vp84.add("Religious Construction");
vp84.add("Spirit Construction");
 vp84.add("Construction");

vp85.add("Christianity Construction");
vp85.add("Religious Construction");
vp85.add("Spiritual Construction");
 vp85.add("Construction");

vp86.add("Islam Construction");
vp86.add("Religious Construction");
vp86.add("Spiritual Construction");
 vp86.add("Construction");

vp87.add("Tomb");
vp87.add("Monument Construction");
vp87.add("Spiritual Construction");
 vp87.add("Construction");

vp88.add("Temple");
 vp88.add("Religious Construction");
```

```
vp88.add("Spiritual Construction");
 vp88.add("Construction");

vp89.add("Monastery");
vp89.add("Religious Construction");
vp89.add("Spiritual Construction");
vp89.add("Construction");

 vp131.add("Convent");
vp131.add("Religious Construction");
vp131.add("Spiritual Construction");
 vp131.add("Construction");


vp90.add("Abbey");
vp90.add("Religious Construction");
vp90.add("Spiritual Construction");
 vp90.add("Construction");

vp91.add("Church");
vp91.add("Christianity Construction");
vp91.add("Religious Construction");
vp91.add("Spiritual Construction");
vp91.add("Construction");

vp92.add("Cathedral");
vp92.add("Christianity Construction");
vp92.add("Religious Construction");
vp92.add("Spiritual Construction");
vp92.add("Construction");

vp93.add("Mosque");
vp93.add("Islam Construction");
vp93.add("Religious Construction");
vp93.add("Spiritual Construction");
 vp93.add("Construction");

vp94.add("Garden-Tomb");
vp94.add("Tomb");
vp94.add("Monument Construction");
vp94.add("Spiritual Construction");
vp94.add("Construction");


 vp96.add("natural beauty");
 vp97.add("topography");
 vp98.add("geography");
 vp99.add("biodiversity");
 vp100.add("geology");
 vp101.add("ecosystem");
 vp102.add("biology");
 vp103.add("wintering Land");
```

```
vp104.add("mountain");
vp104.add("topgraphy");

vp105.add("savannah");
vp105.add("geography");

vp106.add("lake");
vp106.add("geography");

vp107.add("wetland");
vp107.add("geography");

vp108.add("waterfall");
vp108.add("geography");

vp109.add("river");
vp109.add("geography");

vp110.add("forest");
vp110.add("geography");

vp111.add("fossil");
vp111.add("geology");

vp112.add("volcano");
vp112.add("geology");

vp113.add("george");
vp113.add("geology");

vp114.add("geomorphology");
vp114.add("geology");

vp115.add("reef");
vp115.add("geology");

vp116.add("cliff");
vp116.add("geology");

vp117.add("glacier");
vp117.add("geology");

vp118.add("island");
vp118.add("geology");

vp119.add("fauna");
vp119.add("biology");

vp120.add("flora");
vp120.add("biology");

vp121.add("rainforest");
vp121.add("forest");
```

```
 vp121.add("geography");

 vp122.add("tropical forest");
 vp122.add("forest");
 vp122.add("geography");

 vp123.add("temperate forest");
 vp123.add("forest");
 vp123.add("geography");

 vp124.add("special forest");
 vp124.add("forest");
 vp124.add("geography");

 vp125.add("cave");
 vp125.add("gemorphology");
 vp125.add("geology");

 vp126.add("karst");
 vp126.add("gemorphology");
 vp126.add("geology");

 vp127.add("limestone");
 vp127.add("gemorphology");
 vp127.add("geology");

 vp128.add("birds");
 vp128.add("fauna");
 vp128.add("biology");

 vp129.add("animal");
 vp129.add("fauna");
 vp129.add("biology");

 vp130.add("mammal");
 vp130.add("animal");
 vp130.add("fauna");
 vp130.add("biology");

Concept node1 = new Concept("Historic Cities and Areas", 1,vp1,0);
Concept node2 = new Concept("Construction", 1,vp2,0);
Concept node3 = new Concept("Artistic Work", 1,vp3,0);
Concept node4 = new Concept("Hominid Sites", 1,vp4,0);
Concept node5 = new Concept("Archaeological", 1,vp5,0);
Concept node6 = new Concept("Culture Landscape", 1,vp6,0);

Concept node7 = new Concept("Economy Historic Centre", 2,vp7,0);
Concept node8 = new Concept("Culture Historic Centre", 2,vp8,0);
Concept node9 = new Concept("Spiritual Historic Centre", 2,vp9,0);
Concept node10 = new Concept("Military Historic Centre", 2, vp10,0);
Concept node11 = new Concept("Politics Historic Centre", 2, vp11,0);
Concept node12 = new Concept("Resident Settlement", 2, vp12,0);
Concept node13 = new Concept("Economy Construction", 2,vp13,0);
Concept node14 = new Concept("Culture Construction", 2,vp14,0);
Concept node15 = new Concept("Spiritual Construction", 2,vp15,0);
Concept node16 = new Concept("Military Construction", 2,vp16,0);
Concept node17 = new Concept("Politics Construction", 2,vp17,0);
Concept node18 = new Concept("Resident Construction", 2,vp18,0);

Concept node19 = new Concept("Caving", 2,vp19,0);
Concept node20 = new Concept("Painting", 2,vp20,0);
Concept node21 = new Concept("Mosaic", 2,vp21,0);

Concept node22 = new Concept("Clearly Defined Culture Landscape Designed and
Created by Man", 2,vp22,0);
Concept node23 = new Concept("Organically Evolved Landscape", 2,vp23,0);
Concept node24 = new Concept("Associative Culture Landscape", 2,vp24,0);


Concept node25 = new Concept("Agriculture Historic Centre", 3, vp25,0);
Concept node26 = new Concept("Industry Historic Centre", 3, vp26,0);
Concept node27 = new Concept("Commerce Historic Centre", 3, vp27,0);
Concept node28 = new Concept("Mining Town", 4, vp28,0);
Concept node29 = new Concept("Company Town", 4, vp29,0);
Concept node30 = new Concept("Industry Town", 4, vp30, 0);
Concept node31 = new Concept("Wool Industry Complex", 5, vp31, 0);
Concept node32 = new Concept("Copper Mines Industry Complex", 5, vp32, 0);
Concept node33 = new Concept("Sugar Cane Industry Complex", 5, vp33, 0);
Concept node34 = new Concept("Silver Mines Industry Complex", 5, vp34, 0);
Concept node35 = new Concept("Gold Mines Industry Complex", 5, vp35, 0);
Concept node36 = new Concept("Diamond Prospect Industry Complex", 5, vp36, 0);
Concept node37 = new Concept("Trading Centre", 4, vp37, 0);
Concept node38 = new Concept("Harbor", 4, vp38, 0);
Concept node39 = new Concept("Culture Movement Witness", 3, vp39, 0);
Concept node40 = new Concept("Culture Blending Witness", 3, vp40, 0);
Concept node41 = new Concept("Colonial Settlement", 4, vp41, 0);
Concept node42 = new Concept("Education Historic Centre", 3, vp42, 0);
Concept node43 = new Concept("Aesthetics Historic Centre", 3, vp43, 0);
Concept node44 = new Concept("Architecture Centre", 3, vp44, 0);
Concept node45 = new Concept("University City", 4, vp45, 0);
Concept node46 = new Concept("Music Centre", 4, vp46, 0);
Concept node47 = new Concept("Monument Historic Centre", 3, vp47, 0);
Concept node48 = new Concept("Religious Historic Centre", 3, vp48, 0);
Concept node49 = new Concept("Judaism Historic Centre", 4, vp49, 0);
Concept node50 = new Concept("Buddhism Historic Centre", 4, vp50, 0);
Concept node51 = new Concept("Christianity Historic Centre", 4, vp51, 0);
Concept node52 = new Concept("Islam Historic Centre", 4, vp52, 0);
Concept node53 = new Concept("Religious Blending Witness", 4, vp53, 0);
Concept node54 = new Concept("Judaism Holy City", 5, vp54, 0);
Concept node55 = new Concept("Christianity Holy City", 5, vp55, 0);
Concept node56 = new Concept("Islam Holy City", 5, vp56, 0);
Concept node57 = new Concept("Fortified Town", 3, vp57, 0);
Concept node58 = new Concept("Historic Capital", 3, vp58, 0);
Concept node59 = new Concept("Rural Settlement", 3, vp59, 0);
Concept node60 = new Concept("Urban Settlement", 3, vp60, 0);
Concept node61 = new Concept("Settlement in Certain Environment", 4, vp61, 0);
Concept node95 = new Concept("Settlement Restoration", 4, vp95, 0);
Concept node62 = new Concept("Urban Fabric", 4, vp62, 0);
Concept node63 = new Concept("Medieval Town", 4, vp63, 0);
Concept node64 = new Concept("Islamic City", 4, vp64, 0);

Concept node65 = new Concept("Rock Art", 2, vp65,0);

Concept node66 = new Concept("Park", 3, vp66,0);
Concept node67 = new Concept("Garden", 3,vp67,0);
Concept node68 = new Concept("Industry Landscape", 3,vp68,0);
Concept node69 = new Concept("Agriculture Landscape", 3, vp69,0);

Concept node70 = new Concept("Agriculture Construction", 3,vp70,0);
Concept node71 = new Concept("Commerce Construction", 3,vp71,0);
Concept node72 = new Concept("Industry Construction", 3, vp72, 0);
Concept node73 = new Concept("Transport Construction", 3, vp73, 0);

Concept node74 = new Concept("Public Facility", 3,vp74,0);
Concept node75 = new Concept("Culture Blending Construction", 3,vp75,0);

Concept node76 = new Concept("Tower", 3,vp76,0);
Concept node77 = new Concept("Fortress", 3,vp77,0);
Concept node78 = new Concept("Wall", 3,vp78,0);
Concept node79 = new Concept("Castle", 3,vp79,0);
Concept node80 = new Concept("Palace", 3,vp80,0);
Concept node81 = new Concept("Mansion", 3,vp81,0);

Concept node82 = new Concept("Religious Construction", 3,vp82,0);
Concept node83 = new Concept("Monument", 3,vp83,0);

Concept node84 = new Concept("Buddhism Construction", 4,vp84,0);
Concept node85 = new Concept("Christianity Construction", 4,vp85,0);
Concept node86 = new Concept("Islam Construction", 4,vp86,0);
Concept node87 = new Concept("Tomb", 4,vp87,0);

Concept node88 = new Concept("Temple", 4, vp88,0);
Concept node89 = new Concept("Monastery", 4,vp89,0);
Concept node90 = new Concept("Abbey", 4,vp90,0);
Concept node91 = new Concept("Church", 5,vp91,0);
Concept node92 = new Concept("Cathedral", 5,vp92,0);
Concept node93 = new Concept("Mosque", 5,vp93,0);
Concept node94 = new Concept("Garden-Tomb", 5,vp94,0);

Concept node96 = new Concept("natural beauty", 2,vp96,0);
Concept node97 = new Concept("topography", 2,vp97,0);
Concept node98 = new Concept("geography", 2,vp98,0);
Concept node99 = new Concept("biodiversity", 2, vp99,0);
Concept node100 = new Concept("geology", 2, vp100,0);
Concept node101 = new Concept("ecosystem", 2, vp101,0);
Concept node102 = new Concept("biology", 2, vp102,0);
Concept node103 = new Concept("wintering land", 2, vp103,0);
Concept node104 = new Concept("mountain", 3, vp104,0);
```

```
 Concept node105 = new Concept("savannah", 3,vp105,0);
 Concept node106 = new Concept("lake", 3,vp106,0);
 Concept node107 = new Concept("wetland", 3,vp107,0);
 Concept node108 = new Concept("waterfall", 3, vp108,0);
 Concept node109 = new Concept("river", 3, vp109,0);
 Concept node110 = new Concept("forest", 3, vp110,0);
 Concept node111 = new Concept("fossil", 3, vp111,0);
 Concept node112 = new Concept("volcano", 3, vp112,0);
 Concept node113 = new Concept("gorge", 3, vp113,0);
 Concept node114 = new Concept("geomorphology", 3, vp114,0);
 Concept node115 = new Concept("reef", 3, vp115,0);
 Concept node116 = new Concept("cliff", 3,vp116,0);
 Concept node117 = new Concept("glacier", 3,vp117,0);
 Concept node118 = new Concept("island", 3,vp118,0);
 Concept node119 = new Concept("fauna", 3, vp119,0);
 Concept node120 = new Concept("flora", 3, vp120,0);
 Concept node121 = new Concept("rainforest", 4, vp121,0);
 Concept node122 = new Concept("tropical forest", 4, vp122,0);
 Concept node123 = new Concept("temperate forest", 4, vp123,0);
 Concept node124 = new Concept("special forest", 4, vp124,0);
 Concept node125 = new Concept("cave", 4, vp125,0);
 Concept node126 = new Concept("karst", 4, vp126,0);
 Concept node127 = new Concept("limestone", 4, vp127,0);
 Concept node128 = new Concept("birds", 4, vp128,0);
 Concept node129 = new Concept("animal", 4, vp129,0);
 Concept node130 = new Concept("mammal", 5, vp130,0);
Concept node131 = new Concept("Convent", 4, vp131, 0); // added late

vl1.add(node1);
vl1.add(node2);
vl1.add(node3);
vl1.add(node4);
vl1.add(node5);
vl1.add(node6);
vl1.add(node7);
vl1.add(node8);
vl1.add(node9);
vl1.add(node10);
vl1.add(node11);
vl1.add(node12);
vl1.add(node13);
vl1.add(node14);
vl1.add(node15);
vl1.add(node16);
vl1.add(node17);
vl1.add(node18);
vl1.add(node19);
vl1.add(node20);
vl1.add(node21);
vl1.add(node22);
vl1.add(node23);
vl1.add(node24);
vl1.add(node25);
```

```
vl1.add(node26);
vl1.add(node27);
vl1.add(node28);
vl1.add(node29);
vl1.add(node30);
vl1.add(node31);
vl1.add(node32);
vl1.add(node33);
vl1.add(node34);
vl1.add(node35);
vl1.add(node36);
vl1.add(node37);
vl1.add(node38);
vl1.add(node39);
vl1.add(node40);
vl1.add(node41);
vl1.add(node42);
vl1.add(node43);
vl1.add(node44);
vl1.add(node45);
vl1.add(node46);
vl1.add(node47);
 vl1.add(node48);
vl1.add(node49);
vl1.add(node50);
vl1.add(node51);
vl1.add(node52);
vl1.add(node53);
vl1.add(node54);
vl1.add(node55);
vl1.add(node56);
vl1.add(node57);
vl1.add(node58);
vl1.add(node59);
vl1.add(node60);
vl1.add(node61);
vl1.add(node95);
vl1.add(node62);
vl1.add(node63);
vl1.add(node64);
vl1.add(node65);
vl1.add(node66);
vl1.add(node67);
vl1.add(node68);
vl1.add(node69);
vl1.add(node70);
vl1.add(node71);
vl1.add(node72);
vl1.add(node73);
vl1.add(node74);
vl1.add(node75);
vl1.add(node76);
vl1.add(node77);
```

```
vl1.add(node78);
vl1.add(node79);
vl1.add(node80);
vl1.add(node81);
vl1.add(node82);
vl1.add(node83);
vl1.add(node84);
vl1.add(node85);
vl1.add(node86);
vl1.add(node87);
vl1.add(node88);
vl1.add(node89);
vl1.add(node90);
vl1.add(node91);
vl1.add(node92);
vl1.add(node93);
vl1.add(node94);
vl1.add(node96);
vl1.add(node97);
vl1.add(node98);
vl1.add(node99);
vl1.add(node100);
vl1.add(node101);
vl1.add(node102);
vl1.add(node103);
vl1.add(node104);
vl1.add(node105);
vl1.add(node106);
vl1.add(node107);
vl1.add(node108);
vl1.add(node109);
vl1.add(node110);
vl1.add(node111);
vl1.add(node112);
vl1.add(node113);
vl1.add(node114);
vl1.add(node115);
vl1.add(node116);
vl1.add(node117);
vl1.add(node118);
vl1.add(node119);
vl1.add(node120);
vl1.add(node121);
vl1.add(node122);
vl1.add(node123);
vl1.add(node124);
vl1.add(node125);
vl1.add(node126);
vl1.add(node127);
vl1.add(node128);
vl1.add(node129);
vl1.add(node130);
vl1.add(node131);
```

```
      }
   return vl1;
   }

public Vector getConcepts()
   {
   return concepts;
   }


public int[ ] sort(int[ ] instArr)
   {
      int pass,temp;
      for(pass=0;pass<instArr.length;pass++)
        {
         for (int i=0;i<instArr.length-pass-1;i++)
            {
            if (instArr[i]<instArr[i+1])
               {
                temp=instArr[i];
               instArr[i]=instArr[i+1];
               instArr[i+1]=temp;
               }
            }
         }
      return instArr;
    }

public  Vector getinterest(Vector vnode,int[ ] cInst)
   {
      Vector vinst= new Vector();
      Vector vminst= new Vector();
      Vector interest = new Vector();

      for (int j=0; j<vnode.size();j++)
        {
         if(((Concept)vnode.get(j)).count==cInst[0]&&cInst[0]>=1)
           {
           vinst.add((Concept)vnode.get(j));
           }
   }

      int [] vlev= Concept.getlevel(vinst);

      if (vinst.size()==1)
        {
        for (int j=0; j<vnode.size();j++)
           {
            if(((Concept)vnode.get(j)).count==cInst[1] && cInst[1]>=1)
               {
                vminst.add((Concept)vnode.get(j));
               }
           }
        for (int i=0; i<vminst.size();i++)
          {
          if
(((Vector)((Concept)vminst.get(i)).path).containsAll((Collection)(((Concept)vinst.get(0)).path
)))
              {
              interest.add(((Concept)vminst.get(i)).conceptname);
              }
          }

        if(interest.isEmpty())
              {
              interest.add(((Concept)vinst.get(0)).conceptname);
              }
        }
      else if (!vinst.isEmpty())
        {
        for (int i = 0; i< vlev.length; i++)
          {
           System.out.println(vlev[i]);
           }
        sort(vlev);
        for (int j=0; j< vinst.size();j++)
          {
           if(((Concept)vinst.get(j)).level==vlev[0])
              {
              interest.add(((Concept)vinst.get(j)).conceptname);
              }
          }
        }
      return(interest);
  }

 public  Vector getRelativeInterest(Vector vnode,Vector vi)
  {
      Vector vnInst= new Vector();
      Vector vnb= new Vector();

      for (int i=0; i<vi.size();i++)
         {
         for(int j=0; j<vnode.size();j++)
             {
             if ((((Concept)vnode.get(j)).conceptname).equals(vi.get(i)))
                 {
                 vnInst.add((Concept)vnode.get(j));
                 }
             }
         }
      for (int i=0; i<vnInst.size();i++)
         {
         for(int j=0; j<vnode.size();j++)
           {
            if
((((Concept)vnode.get(j)).level==((Concept)vnInst.get(i)).level)&&(((Concept)vnode.get(j)).c
onceptname!=((Concept)vnInst.get(i)).conceptname)&&(!(compare(((Concept)vnode.get(j)).p
ath,((Concept)vnInst.get(i)).path)).isEmpty()))
              {
              vnb.add(((Concept)vnode.get(j)).conceptname);
              }
            }
        }
      return vnb;
    }

 public  Vector compare(Vector v1, Vector v2)
   {
      Vector vr = new Vector();

      if (!v2.isEmpty())
         {
         for(int i=0;i<v2.size();i++)
            {
            if (v1.contains(v2.get(i)))
               {
               vr.add(v2.get(i));
               }
            }
         }
      return vr;
    }


public  Vector getinduce(Vector vp,Vector vnode)
   {
      Vector indResult = new Vector();
      for (int i = 0; i<vnode.size() ; i++)
         {
         for (int j = 0; j <vp.size() ; j++)
           {
            if (((Vector)(((Concept)vnode.get(i)).path)).contains(vp.get(j)))
              { if (!indResult.contains(((Vector)(((Concept)vnode.get(i)).path)).firstElement()))
                 {
                 indResult.add(((Vector)(((Concept)vnode.get(i)).path)).firstElement());
                 }
              }
           }
         }
      return (indResult);
   }

 public  Vector getpath(Vector vc,Vector vnode)
   {
      Vector vpath = new Vector();

      for (int j=0; j<vnode.size();j++)
```

```
            {
            if (vc.contains(((Concept)vnode.get(j)).conceptname))
                {
                vpath.add(((Concept)vnode.get(j)).path);
                }
            }
        return vpath;
    }

public  int[ ] countcategory(Vector vp,Vector vOldInst, Vector vnode)
    {
        int [ ] couInst = new int[vnode.size()];
        Vector OldPath = new Vector();

        for (int i=0;i<vp.size();i++)
            {
            for(int j=0;j<((Vector)vp.get(i)).size();j++)
                {
                for(int k= 0; k<vnode.size();k++)
                    {
                    if ((((Concept)vnode.get(k)).conceptname).equals(((Vector)vp.get(i)).get(j)))
                        {
                        ((Concept)vnode.get(k)).count= ((Concept)vnode.get(k)).count+1;
                        }
                    }
                }
            }

        if (!vOldInst.isEmpty())
            {
            for (int i=0;i<vOldInst.size();i++)
                {
                for(int k= 0; k<vnode.size();k++)
                    {
                    if ((((Concept)vnode.get(k)).conceptname).equals(vOldInst.get(i)))
                        {
                        OldPath.add((Vector)((Concept)vnode.get(k)).path);
                        }
                    }
                }


            for (int i=0;i<OldPath.size();i++)
                {
                for(int j=0;j<((Vector)OldPath.get(i)).size();j++)
                    {
                    for(int k= 0; k<vnode.size();k++)
                        {
                        if
((((Concept)vnode.get(k)).conceptname).equals(((Vector)OldPath.get(i)).get(j)))
                            {
                            ((Concept)vnode.get(k)).count= ((Concept)vnode.get(k)).count-1;
                            }
                }
              }
            }
          }
      for (int i = 0; i < couInst.length; i++)
        {
        couInst[i]=  ((Concept)vnode.get(i)).count;
        System.out.println(couInst[i]);
        }
      return couInst;
  }
  }
```

************Position************

```
package induction;
import java.sql.*;
import java.util.*;
import java.io.*;

public class Position
   {
    String Psname;
    int count;

 public Position (String nn,int vts)
    {
     Psname=nn;
     count=vts;
   }
   }
```

************AllLocations.java************

```
package induction;
import java.sql.*;
import java.util.*;
import java.io.*;

public class AllLocations
   {

    Vector vCountry ;
    Vector vComposition;
    Vector vGeography;

public AllLocations (Vector vcoun, Vector vcomp,Vector vgeog)
   {
   vCountry=vcoun;
   vComposition=vcomp;
   vGeography=vgeog;
```

```
  }

public Vector getVcoun()
   {
   return this.vCountry;
   }

public Vector getVcomp()
  {
  return this.vComposition;
  }

public Vector getVgeog()
  {
  return this.vGeography;
  }

public Vector AddorCountP(Position pst, Vector vpos, Vector voldinst)
  {
      boolean flag =true;
      String psname = pst.Psname;

      if(!(voldinst.isEmpty()))
         {
          for (int i=0;i<vpos.size();i++)
             {
             for (int j=0;j<voldinst.size();j++)
                {
                if (((Position)vpos.get(i)).Psname.equals(voldinst.get(j)))
                   {
                   ((Position)vpos.get(i)).count=((Position)vpos.get(i)).count-1;
                   }
                }
             }
         }

      for (int i=0;i<vpos.size();i++)
         {
         if (((Position)vpos.get(i)).Psname.equals(psname))
             {
             ((Position)vpos.get(i)).count= ((Position)vpos.get(i)).count+1;
             System.out.println("Here is Position L 34: "+((Position)vpos.get(i)).count);
             flag=false;
             }
         }

      if (flag)
         {
          pst.count++;
         vpos.add(pst);
         }
      return(vpos);
  }
```

```
public int[ ] countPosition(Vector vpos)
   {
     int [ ] countP = new int[vpos.size()];
     for (int i = 0; i < countP.length; i++)
        {
        countP[i]=((Position)vpos.get(i)).count;
        }
    return countP;
  }
public Vector getMostP(Vector vpos,int[ ] cInst)
  {
     Vector vinstP= new Vector();
     for (int j=0; j<vpos.size();j++)
        {
        if(((Position)vpos.get(j)).count==cInst[0])
           {
           vinstP.add((Position)vpos.get(j));
           System.out.println("Here is Position:"+((Position)vpos.get(j)).Psname);
           }
        }
     return vinstP;
  }

public Vector getInstofP (Vector vpos1,Vector vpos2,Vector vpos3)
  {
     Vector InstofP = new Vector();
     if (vpos1.size()==1)
        {
         InstofP.add(((Position)vpos1.get(0)).Psname);
         }
     else
        {
         if (vpos2.size()==1)
            {
            InstofP.add(((Position)vpos2.get(0)).Psname);
            }
         else
            {
            if (vpos3.size()==1)
                {
                InstofP.add(((Position)vpos3.get(0)).Psname);
                }
           else
                {
                for(int i=0;i<vpos1.size();i++)
                   {
                   InstofP.add(((Position)vpos1.get(i)).Psname);
                   }
                }
            }
        }
    return InstofP;
  }
  }
```

**********Interest.java**********

```
package induction;

import java.sql.*;
import java.util.*;
import java.io.*;


public class Interest
   {
   int visitNum;
   int repNum;
   Vector interestC;/* interest of category*/
   Vector interestL;/* interest of location*/
   Vector Accinterest;
   Vector AccinterestC;
   Vector AccinterestCp;
   Vector AccinterestG;

 public Interest (int repn,int visn,Vector vinstc, Vector vinstl,Vector vaccinst,Vector vaccinstc,
Vector vaccinstcp,Vector vaccinstg)
   {
    repNum=repn;
    visitNum=visn;
    interestC=vinstc;
    interestL=vinstl;
    Accinterest=vaccinst;
    AccinterestC=vaccinstc;
    AccinterestCp=vaccinstcp;
    AccinterestG=vaccinstg;
   }

public void setinterestC(Vector vinsts1)
   {
   this.interestC = (Vector)vinsts1.clone();
   }

public void setinterestL(Vector vInst)
   {
   this.interestL = (Vector)vInst.clone();
   }

public void setAccinterest(Vector vCa)
   {
   this.Accinterest = (Vector)vCa.clone();
   }
 public void setAccinterestC(Vector vC)
   {
   this.AccinterestC = (Vector)vC.clone();
   }

 public void setAccinterestCp(Vector vCp)
   {
   this.AccinterestCp = (Vector)vCp.clone();
   }

 public void setAccinterestG(Vector vG)
   {
   this.AccinterestG = (Vector)vG.clone();
   }

 public int setvisitNum(int vnum)
   {
   this.visitNum = vnum+1;
   return this.visitNum;
   }

public  int getVisitN()
   {
   return this.visitNum;
   }

public  Vector getInstC()
   {
   return this.interestC;
   }

 public  Vector getInstL()
   {
   return this.interestL;
    }

  public  int  getRept()
   {
   return this.repNum;
   }

 public  Vector getAccinst()
   {
   return this.Accinterest;
   }

public  Vector getAccinstC()
   {
   return this.AccinterestC;
   }

public  Vector getAccinstCp()
   {
```

```
   return this.AccinterestCp;
   }

public  Vector getAccinstG()
   {
   return this.AccinterestG;
   }

public Vector getoutofOldInst(Vector vctg, Vector vacc)
   {
      Vector vOldinst = new Vector();
      for (int i=0;i<vctg.size();i++)
         {
         vacc.add((String)vctg.get(i));
         }

      vacc.add("next");

     if (visitNum >9)
         {
         while(!(vacc.get(0)).equals("next"))
            {
            vOldinst.add(vacc.remove(0));
            }
         vacc.removeElementAt(0);
         }
      return vOldinst;
   }

public  boolean getRepeatTimes(Vector v1, Vector v2)
   {
    boolean flag = false;
    if(v1.equals(v2))
      {
      if (this.repNum<4)
          {
          this.repNum=this.repNum+1;
          flag=false;
          }
       else
         {
         flag=true;
         this.repNum=0;
         }
   else
     {
     flag=false;
     this.repNum=0;
     }
   return flag;
   }
   }
```

**********WConcept.java**********

```
package winduction;

import java.sql.*;
import java.util.*;
import java.io.*;

public class WConcept
   {
   float level;
   String conceptname;
   Vector path = new Vector();
   float weight;
   float uweight;

public WConcept(String nn,int lev,Vector v,float wt, float uwt)
   {
   level=lev;
   conceptname=nn;
   path =v;
   weight=wt;
   uweight=uwt;
   }

public float getUweight()
   {
   return this.uweight;
   }

public Vector getPath()
   {
   return this.path;
   }

public void setUweight(float wght)
   {
   this.uweight = wght;
   }

public static float[] getlevel(Vector vnode)
   {
      float [] lev = new float[vnode.size()];
      for (int i = 0; i < lev.length; i++)
         {
         lev[i]=  ((WConcept)vnode.get(i)).level;
         }
     return lev;
   }
   }
```

***********WCategory.java***********

```
package winduction;

import java.sql.*;
import java.util.*;
import java.io.*;


public class WCategory
   {
   String Wcname;
   int Wcnum;
   Vector wconcepts = new Vector();

public WCategory(String ln, int noden,Vector vnode)
    {
    Wcname=ln;
    Wcnum=noden;
    wconcepts=vnode;
    }

public Vector getWconcepts()
   {
   return this.wconcepts;
   }

 public static Vector initWconcepts(String ln)
   {
       Vector vp0 = new Vector();
       Vector vp1 = new Vector();
       Vector vp2 = new Vector();
       Vector vp3 = new Vector();
       Vector vp4 = new Vector();
       Vector vp5 = new Vector();
       Vector vp6 = new Vector();
       Vector vp7 = new Vector();
       Vector vp8 = new Vector();
       Vector vp9 = new Vector();
       Vector vp10 = new Vector();
       Vector vp11 = new Vector();
       Vector vp12 = new Vector();
       Vector vp13 = new Vector();
       Vector vp14 = new Vector();
       Vector vp15 = new Vector();
       Vector vp16 = new Vector();
       Vector vp17 = new Vector();
       Vector vp18 = new Vector();
       Vector vp19 = new Vector();
       Vector vp20 = new Vector();
       Vector vp21 = new Vector();
       Vector vp22 = new Vector();
Vector vp23 = new Vector();
Vector vp24 = new Vector();
Vector vp25 = new Vector();
Vector vp26 = new Vector();
Vector vp27 = new Vector();
Vector vp28 = new Vector();
Vector vp29 = new Vector();
Vector vp30 = new Vector();
Vector vp31 = new Vector();
Vector vp32 = new Vector();
Vector vp33 = new Vector();
Vector vp34 = new Vector();
Vector vp35 = new Vector();
Vector vp36 = new Vector();
Vector vp37 = new Vector();
Vector vp38 = new Vector();
Vector vp39 = new Vector();
Vector vp40 = new Vector();
Vector vp41 = new Vector();
Vector vp42 = new Vector();
Vector vp43 = new Vector();
Vector vp44 = new Vector();
Vector vp45 = new Vector();
Vector vp46 = new Vector();
Vector vp47 = new Vector();
Vector vp48 = new Vector();
Vector vp49 = new Vector();
Vector vp50 = new Vector();
Vector vp51 = new Vector();
Vector vp52 = new Vector();
Vector vp53 = new Vector();
Vector vp54 = new Vector();
Vector vp55 = new Vector();
Vector vp56 = new Vector();
Vector vp57 = new Vector();
Vector vp58 = new Vector();
Vector vp59 = new Vector();
Vector vp60 = new Vector();
Vector vp61 = new Vector();
Vector vp62 = new Vector();
Vector vp63 = new Vector();
Vector vp64 = new Vector();
Vector vp65 = new Vector();
Vector vp66 = new Vector();
Vector vp67 = new Vector();
Vector vp68 = new Vector();
Vector vp69 = new Vector();
Vector vp70 = new Vector();
Vector vp71 = new Vector();
Vector vp72 = new Vector();
Vector vp73 = new Vector();
Vector vp74 = new Vector();
Vector vp75 = new Vector();
Vector vp76 = new Vector();
Vector vp77 = new Vector();
Vector vp78 = new Vector();
Vector vp79 = new Vector();
Vector vp80 = new Vector();
Vector vp81 = new Vector();
Vector vp82 = new Vector();
Vector vp83 = new Vector();
Vector vp84 = new Vector();
Vector vp85 = new Vector();
Vector vp86 = new Vector();
Vector vp87 = new Vector();
Vector vp88 = new Vector();
Vector vp89 = new Vector();
Vector vp90 = new Vector();
Vector vp91 = new Vector();
Vector vp92 = new Vector();
Vector vp93 = new Vector();
Vector vp94 = new Vector();
Vector vp95 = new Vector();
Vector vp96 = new Vector();
Vector vp97 = new Vector();
Vector vp98 = new Vector();
Vector vp99 = new Vector();
Vector vp100 = new Vector();
Vector vp101 = new Vector();
Vector vp102 = new Vector();
Vector vp103 = new Vector();
Vector vp104 = new Vector();
Vector vp105 = new Vector();
Vector vp106 = new Vector();
Vector vp107 = new Vector();
Vector vp108 = new Vector();
Vector vp109 = new Vector();
Vector vp110 = new Vector();
Vector vp111 = new Vector();
Vector vp112 = new Vector();
Vector vp113 = new Vector();
Vector vp114 = new Vector();
Vector vp115 = new Vector();
Vector vp116 = new Vector();
Vector vp117 = new Vector();
Vector vp118 = new Vector();
Vector vp119 = new Vector();
Vector vp120 = new Vector();
Vector vp121 = new Vector();
Vector vp122 = new Vector();
Vector vp123 = new Vector();
Vector vp124 = new Vector();
Vector vp125 = new Vector();
Vector vp126 = new Vector();
Vector vp127 = new Vector();
Vector vp128 = new Vector();
```

```
 Vector vp129 = new Vector();
 Vector vp130 = new Vector();
 Vector vp131 = new Vector();

Vector vl1 = new Vector();

 System.out.println(ln);

 if ("Natural Sites".equals(ln))
 {
 vp1.add("natural beauty");
 vp2.add("topography");
 vp3.add("geography");
 vp4.add("biodiversity");
 vp5.add("geology");
 vp6.add("ecosystem");
 vp7.add("biology");
 vp8.add("wintering Land");

 vp9.add("mountain");
 vp9.add("topgraphy");

 vp10.add("savannah");
 vp10.add("geography");

 vp11.add("lake");
 vp11.add("geography");

 vp12.add("wetland");
 vp12.add("geography");

 vp13.add("waterfall");
 vp13.add("geography");

 vp14.add("river");
 vp14.add("geography");

  vp15.add("forest");
  vp15.add("geography");

  vp16.add("fossil");
  vp16.add("geology");

 vp17.add("volcano");
  vp17.add("geology");

  vp18.add("george");
  vp18.add("geology");

  vp19.add("geomorphology");
  vp19.add("geology");

  vp20.add("reef");
```

```
vp20.add("geology");

vp21.add("cliff");
vp21.add("geology");

vp22.add("glacier");
vp22.add("geology");

vp23.add("island");
vp23.add("geology");

vp24.add("fauna");
vp24.add("biology");

vp25.add("flora");
vp25.add("biology");

vp26.add("rainforest");
vp26.add("forest");
vp26.add("geography");

vp27.add("tropical forest");
vp27.add("forest");
vp27.add("geography");

vp28.add("temperate forest");
vp28.add("forest");
vp28.add("geography");

vp29.add("special forest");
vp29.add("forest");
vp29.add("geography");

vp30.add("cave");
vp30.add("gemorphology");
vp30.add("geology");

vp31.add("karst");
vp31.add("gemorphology");
vp31.add("geology");

vp32.add("limestone");
vp32.add("gemorphology");
 vp32.add("geology");

vp33.add("birds");
vp33.add("fauna");
vp33.add("biology");

vp34.add("animal");
vp34.add("fauna");
vp34.add("biology");
```

```
vp35.add("mammal");
vp35.add("animal");
vp35.add("fauna");
vp35.add("biology");


WConcept node1 = new WConcept("natural beauty", 1,vp1,0,0);
WConcept node2 = new WConcept("topography", 1,vp2,0,0);
WConcept node3 = new WConcept("geography", 1,vp3,0,0);
WConcept node4 = new WConcept("biodiversity", 1, vp4,0,0);
WConcept node5 = new WConcept("geology", 1, vp5,0,0);
WConcept node6 = new WConcept("ecosystem", 1, vp6,0,0);
WConcept node7 = new WConcept("biology", 1, vp7,0,0);
WConcept node8 = new WConcept("wintering land", 1, vp8,0,0);
WConcept node9 = new WConcept("mountain", 2, vp9,0,0);
WConcept node10 = new WConcept("savannah", 2,vp10,0,0);
WConcept node11 = new WConcept("lake", 2,vp11,0,0);
WConcept node12 = new WConcept("wetland", 2,vp12,0,0);
WConcept node13 = new WConcept("waterfall", 2, vp13,0,0);
WConcept node14 = new WConcept("river", 2, vp14,0,0);
WConcept node15 = new WConcept("forest", 2, vp15,0,0);
WConcept node16 = new WConcept("fossil", 2, vp16,0,0);
WConcept node17 = new WConcept("volcano", 2, vp17,0,0);
WConcept node18 = new WConcept("gorge", 2, vp18,0,0);
WConcept node19 = new WConcept("geomorphology", 2, vp19,0,0);
WConcept node20 = new WConcept("reef", 2, vp20,0,0);
WConcept node21 = new WConcept("cliff", 2,vp21,0,0);
WConcept node22 = new WConcept("glacier", 2,vp22,0,0);
WConcept node23 = new WConcept("island", 2,vp23,0,0);
WConcept node24 = new WConcept("fauna", 2, vp24,0,0);
WConcept node25 = new WConcept("flora", 2, vp25,0,0);
WConcept node26 = new WConcept("rainforest", 3, vp26,0,0);
WConcept node27 = new WConcept("tropical forest", 3, vp27,0,0);
WConcept node28 = new WConcept("temperate forest", 3, vp28,0,0);
WConcept node29 = new WConcept("special forest", 3, vp29,0,0);
WConcept node30 = new WConcept("cave", 3, vp30,0,0);
WConcept node31 = new WConcept("karst", 3, vp31,0,0);
WConcept node32 = new WConcept("limestone", 3, vp32,0,0);
WConcept node33 = new WConcept("birds", 3, vp33,0,0);
WConcept node34 = new WConcept("animal", 3, vp34,0,0);
WConcept node35 = new WConcept("mammal", 4, vp34,0,0);

vl1.add(node1);
vl1.add(node2);
vl1.add(node3);
vl1.add(node4);
vl1.add(node5);
vl1.add(node6);
vl1.add(node7);
vl1.add(node8);
vl1.add(node9);
vl1.add(node10);
vl1.add(node11);
```

```
vl1.add(node12);
vl1.add(node13);
vl1.add(node14);
vl1.add(node15);
vl1.add(node16);
vl1.add(node17);
vl1.add(node18);
vl1.add(node19);
vl1.add(node20);
vl1.add(node21);
vl1.add(node22);
vl1.add(node23);
vl1.add(node24);
vl1.add(node25);
vl1.add(node26);
vl1.add(node27);
vl1.add(node28);
vl1.add(node29);
vl1.add(node30);
vl1.add(node31);
vl1.add(node32);
vl1.add(node33);
vl1.add(node34);
vl1.add(node35);
}
if ("Cultural Sites".equals(ln))
{

 vp1.add("Historic Cities and Areas");
vp2.add("Construction");
vp3.add("Artistic Work");
 vp4.add("Hominid Sites");
 vp5.add("Archaeological Sites");
 vp6.add("Culture Landscape");

 vp7.add("Economy Historic centre");
 vp7.add("Historic Cities and Areas");

  vp8.add("Culture Historic Centre");
 vp8.add("Historic Cities and Areas");

  vp9.add("Spiritual Historic Centre");
 vp9.add("Historic Cities and Areas");

 vp10.add("Military Historic Centre");
 vp10.add("Historic Cities and Areas");

  vp11.add("Politics Historic Centre");
  vp11.add("Historic Cities and Areas");

  vp12.add("Resident Settlement");
  vp12.add("Historic Cities and Areas");
 vp13.add("Economy Construction");
 vp13.add("Construction");

 vp14.add("Cultural Construction");
 vp14.add("Construction");

 vp15.add("Spiritual Construction");
 vp15.add("Construction");

 vp16.add("Military Construction");
 vp16.add("Construction");

 vp17.add("Politics Construction");
 vp17.add("Construction");

 vp18.add("Resident Construction");
 vp18.add("Construction");

 vp19.add("Caving");
 vp19.add("Artistic Work");

 vp20.add("Painting");
 vp20.add("Artistic Work");

 vp21.add("Mosaic");
 vp21.add("Artistic Work");

 vp22.add("Clearly Defined Culture Landscape Designed and Created by Man");
 vp22.add("Culture Landscape");

 vp23.add("Organically Evolved Landscape");
 vp23.add("Culture Landscape");

 vp24.add("Associative Culture Landscape");
 vp24.add("Culture Landscape");

 vp25.add("Agriculture Historic Centre");
 vp25.add("Economy Historic centre");
 vp25.add("Historic Cities and Areas");

vp26.add("Industry Historic Centre");
vp26.add("Economy Historic centre");
vp25.add("Historic Cities and Areas");

vp27.add("Commerce Historic Centre");
vp27.add("Economy Historic centre");
vp27.add("Historic Cities and Areas");

vp28.add("Mining Town");
vp28.add("Industry Historic Centre");
vp28.add("Economy Historic centre");
vp28.add("Historic Cities and Areas");
```

```
vp29.add("Company Town");
vp29.add("Industry Historic Centre");
vp29.add("Economy Historic centre");
vp29.add("Historic Cities and Areas");

vp30.add("Industry Town");
vp30.add("Commerce Historic Centre");
vp30.add("Economy Historic centre");
vp30.add("Historic Cities and Areas");

vp31.add("Wool Industry Complex");
vp31.add("Industry Town");
vp31.add("Industry Historic Centre");
vp31.add("Economy Historic centre");
vp31.add("Historic Cities and Areas");

vp32.add("Copper Mines Industry Complex");
vp32.add("Mining Town");
vp32.add("Industry Historic Centre");
vp32.add("Economy Historic centre");
vp32.add("Historic Cities and Areas");

vp33.add("Sugar Cane Industry Complex");
vp31.add("Industry Town");
vp33.add("Industry Historic Centre");
vp33.add("Economy Historic centre");
vp33.add("Historic Cities and Areas");

vp34.add("Silver Mines Industry Complex");
vp34.add("Mining Town");
vp34.add("Industry Historic Centre");
vp34.add("Economy Historic centre");
vp34.add("Historic Cities and Areas");

vp35.add("Gold Mines Industry Complex");
vp35.add("Mining Town");
vp35.add("Industry Historic Centre");
vp35.add("Economy Historic centre");
vp35.add("Historic Cities and Areas");

vp36.add("Diamond Prospect Industry Complex");
vp36.add("Mining Town");
vp36.add("Industry Historic Centre");
vp36.add("Economy Historic centre");
vp36.add("Historic Cities and Areas");

vp37.add("Trading Centre");
vp37.add("Commerce Historic Centre");
vp37.add("Economy Historic centre");
vp37.add("Historic Cities and Areas");


vp38.add("Harbor");
```

```
vp38.add("Commerce Historic Centre");
vp38.add("Economy Historic centre");
vp38.add("Historic Cities and Areas");

vp39.add("Culture Movement Witness");
vp39.add("Culture Historic Centre");
vp39.add("Historic Cities and Areas");

vp40.add("Culture Blending Witness");
vp40.add("Culture Historic Centre");
vp40.add("Historic Cities and Areas");

vp41.add("Colonial Settlement"); /*two pathes */
vp41.add("Culture Blending Witness");
vp41.add("Culture Historic Centre");
vp41.add("Urban Settlement");
vp41.add("Resident Settlement");
vp41.add("Historic Cities and Areas");

vp42.add("Education Historic Centre");
vp42.add("Culture Historic Centre");
vp42.add("Historic Cities and Areas");

vp43.add("Aesthetic Historic Centre");
vp43.add("Culture Historic Centre");
vp43.add("Historic Cities and Areas");

vp44.add("Architecture Centre");
vp44.add("Culture Historic Centre");
vp44.add("Historic Cities and Areas");

vp45.add("University City");
vp45.add("Education Historic Centre");
vp45.add("Culture Historic Centre");
vp45.add("Historic Cities and Areas");

vp46.add("Music Centre");
vp46.add("Aesthetic Historic Centre");
vp46.add("Culture Historic Centre");
vp46.add("Historic Cities and Areas");


vp47.add("Monument Historic Centre");
vp47.add("Spiritual Historic Centre");
vp47.add("Historic Cities and Areas");

vp48.add("Religious Historic Centre");
vp48.add("Spiritual Historic Centre");
vp48.add("Historic Cities and Areas");

vp49.add("Judaism Historic Centre");
vp49.add("Religious Historic Centre");
vp49.add("Spiritual Historic Centre");
```

```
 vp49.add("Historic Cities and Areas");

 vp50.add("Buddhism Historic Centre");
 vp50.add("Religious Historic Centre");
 vp50.add("Spiritual Historic Centre");
vp50.add("Historic Cities and Areas");

 vp51.add("Christianity Historic Centre");
 vp51.add("Religious Historic Centre");
 vp51.add("Spiritual Historic Centre");
vp51.add("Historic Cities and Areas");

 vp52.add("Islam Historic Centre");
 vp52.add("Religious Historic Centre");
 vp52.add("Spiritual Historic Centre");
vp52.add("Historic Cities and Areas");

 vp53.add("Religious Blending Witness");
 vp53.add("Religious Historic Centre");
 vp53.add("Spiritual Historic Centre");
vp53.add("Historic Cities and Areas");

 vp54.add("Judaism Holy City");
 vp54.add("Judaism Historic Centre");
 vp54.add("Religious Historic Centre");
 vp54.add("Spiritual Historic Centre");
vp54.add("Historic Cities and Areas");

 vp55.add("Christianity Holy City");
 vp55.add("Christianity Historic Centre");
 vp55.add("Religious Historic Centre");
 vp55.add("Spiritual Historic Centre");
vp55.add("Historic Cities and Areas");

 vp56.add("Islam Holy City");
 vp56.add("Islam Historic Centre");
 vp56.add("Religion Historic Centre");
 vp56.add("Spirit Historic Centre");
 vp56.add("Historic Cities and Areas");

 vp57.add("Fortified Town");
 vp57.add("Military Historic Centre");
vp57.add("Historic Cities and Areas");


 vp58.add("Historic Capital");
  vp58.add("Politics Historic Centre");
  vp58.add("Historic Cities and Areas");


 vp59.add("Rural Settlement");
 vp59.add("Resident Settlement");
vp59.add("Historic Cities and Areas");
```

```
vp60.add("Urban Settlement");
vp60 .add("Resident Settlement");
vp60.add("Historic Cities and Areas");


vp61.add("Settlement in Certain Environment");
vp61.add("Urban Settlement");
vp61.add("Resident Settlement");
vp61.add("Historic Cities and Areas");

vp95.add("Settlement Restoration");
vp95.add("Urban Settlement");
vp95.add("Resident Settlement");
vp95.add("Historic Cities and Areas");


vp62.add("Urban Fabric");
vp62.add("Urban Settlement");
vp62.add("Resident Settlement");
vp62.add("Historic Cities and Areas");

vp63.add("Medieval Town");
vp63.add("Urban Settlement");
vp63.add("Resident Settlement");
vp63.add("Historic Cities and Areas");

vp64.add("Islamic City");
vp64.add("Urban Settlement");
vp64.add("Resident Settlement");
vp64.add("Historic Cities and Areas");

vp65.add("Rock Art");
vp65.add("Artistic Work");

vp66.add("park");
vp66.add("Clearly Defined Culture Landscape Designed and Created by Man");
vp66.add("Culture Landscape");

vp67.add("Garden");
vp67.add("Clearly Defined Culture Landscape Designed and Created by Man");
vp67.add("Culture Landscape");

vp68.add("Industry Landscape");
vp68.add("Clearly Defined Culture Landscape Designed and Created by Man");
vp68.add("Culture Landscape");

vp69.add("Agriculture Landscape");
vp69.add("Associative Culture Landscape");
vp69.add("Culture Landscape");

vp70.add("Agriculture Construction");
vp70.add("Economy Construction");
```

```
vp70.add("Construction");

vp71.add("Commerce Construction");
vp71.add("Economy Construction");
vp71.add("Construction");

vp72.add("Industry Construction");
vp72.add("Economy Construction");
vp72.add("Construction");

vp73.add("Transport Construction");
vp73.add("Economy Construction");
 vp73.add("Construction");

vp74.add("Public Facility");
vp74.add("Culture Construction");
 vp74.add("Construction");

vp75.add("Culture Blending Construction");
vp75.add("Culture Construction");
 vp75.add("Construction");

vp76.add("Tower");
vp76.add("Military Construction");
vp76.add("Monument Construction");
vp76.add("Spiritual Construction");
vp76.add("Construction");

vp77.add("Fortress");
vp77.add("Military Construction");
vp77.add("Construction");

vp78.add("Wall");
vp78.add("Military Construction");
 vp78.add("Construction");

vp79.add("Castle");
vp79.add("Military Construction");
vp79.add("Resident Construction");
 vp79.add("Construction");

vp80.add("Palace");
vp80.add("Resident Construction");
 vp80.add("Construction");

vp81.add("Mansion");
vp81.add("Resident Construction");
 vp81.add("Construction");

vp82.add("Religious Construction");
vp82.add("Spiritual Construction");
 vp82.add("Construction");
```

```
vp83.add("Monument Construction");
vp83.add("Spiritual construction");
 vp83.add("Construction");

vp84.add("Buddhism Construction");
vp84.add("Religious Construction");
vp84.add("Spirit Construction");
 vp84.add("Construction");

vp85.add("Christianity Construction");
vp85.add("Religious Construction");
vp85.add("Spiritual Construction");
vp85.add("Construction");

vp86.add("Islam Construction");
vp86.add("Religious Construction");
vp86.add("Spiritual Construction");
vp86.add("Construction");

vp87.add("Tomb");
vp87.add("Monument Construction");
vp87.add("Spiritual Construction");
vp87.add("Construction");

vp88.add("Temple");
vp88.add("Religious Construction");
vp88.add("Spiritual Construction");
vp88.add("Construction");

vp89.add("Monastery");
vp89.add("Religious Construction");
vp89.add("Spiritual Construction");
 vp89.add("Construction");

vp96.add("Convent"); // added late
vp96.add("Religious Construction");
vp96.add("Spiritual Construction");
vp96.add("Construction");

vp90.add("Abbey");
vp90.add("Religious Construction");
vp90.add("Spiritual Construction");
vp90.add("Construction");

vp91.add("Church");
vp91.add("Christianity Construction");
vp91.add("Religious Construction");
vp91.add("Spiritual Construction");
vp91.add("Construction");

vp92.add("Cathedral");
vp92.add("Christianity Construction");
vp92.add("Religious Construction");
```

```
vp92.add("Spiritual Construction");
 vp92.add("Construction");

vp93.add("Mosque");
vp93.add("Islam Construction");
vp93.add("Religious Construction");
vp93.add("Spiritual Construction");
 vp93.add("Construction");

vp94.add("Garden-Tomb");
vp94.add("Tomb");
vp94.add("Monument Construction");
vp94.add("Spiritual Construction");
vp94.add("Construction");


WConcept node1 = new WConcept("Historic Cities and Areas", 1,vp1,0,0);
 WConcept node2 = new WConcept("Construction", 1,vp2,0,0);
 WConcept node3 = new WConcept("Art Work", 1,vp3,0,0);
 WConcept node4 = new WConcept("Hominid Sites", 1,vp4,0,0);
 WConcept node5 = new WConcept("Archaeological", 1,vp5,0,0);
 WConcept node6 = new WConcept("Culture Landscape", 1,vp6,0,0);
  WConcept node7 = new WConcept("Economy Historic Centre", 2,vp7,0,0);
 WConcept node8 = new WConcept("Culture Historic Centre", 2,vp8,0,0);
 WConcept node9 = new WConcept("Spirit Historic Centre", 2,vp9,0,0);
 WConcept node10 = new WConcept("Military Historic Centre", 2, vp10,0,0);
 WConcept node11 = new WConcept("Politics Historic Centre", 2, vp11,0,0);
 WConcept node12 = new WConcept("Resident Settlement", 2, vp12,0,0);
  WConcept node13 = new WConcept("Economy Construction", 2,vp13,0,0);
 WConcept node14 = new WConcept("Culture Construction", 2,vp14,0,0);
 WConcept node15 = new WConcept("Spirit Construction", 2,vp15,0,0);
 WConcept node16 = new WConcept("Military Construction", 2,vp16,0,0);
 WConcept node17 = new WConcept("Politics Construction", 2,vp17,0,0);
 WConcept node18 = new WConcept("Resident Construction", 2,vp18,0,0);
  WConcept node19 = new WConcept("Caving", 2,vp19,0,0);
 WConcept node20 = new WConcept("Painting", 2,vp20,0,0);
 WConcept node21 = new WConcept("Mosaic", 2,vp21,0,0);
  WConcept node22 = new WConcept("Clearly Defined Culture Landscape Designed and
Created by Man", 2,vp22,0,0);
 WConcept node23 = new WConcept("Organically Evolved Landscape", 2,vp23,0,0);
 WConcept node24 = new WConcept("Associative Culture Landscape", 2,vp24,0,0);
 WConcept node25 = new WConcept("Agriculture Historic Centre", 3, vp25,0,0);
 WConcept node26 = new WConcept("Industry Historic Centre", 3, vp26,0,0);
 WConcept node27 = new WConcept("Commerce Historic Centre", 3, vp27,0,0);
 WConcept node28 = new WConcept("Mining Town", 4, vp28,0,0);
 WConcept node29 = new WConcept("Company Town", 4, vp29,0,0);
 WConcept node30 = new WConcept("Industry Town", 4, vp30, 0,0);
 WConcept node31 = new WConcept("Wool Industry Complex", 5, vp31, 0,0);
 WConcept node32 = new WConcept("Copper Mines Industry Complex", 5, vp32, 0,0);
 WConcept node33 = new WConcept("Sugar Cane Industry Complex", 5, vp33, 0,0);
 WConcept node34 = new WConcept("Silver Mines Industry Complex", 5, vp34, 0,0);
 WConcept node35 = new WConcept("Gold Mines Industry Complex", 5, vp35, 0,0);
```

```
        WConcept node36 = new WConcept("Diamond Prospect Industry Complex", 5, vp36,
0,0);
        WConcept node37 = new WConcept("Trading Centre", 4, vp37, 0,0);
        WConcept node38 = new WConcept("Harbor", 4, vp38, 0,0);
        WConcept node39 = new WConcept("Culture Movement Witness", 3, vp39, 0,0);
        WConcept node40 = new WConcept("Culture Blending Witness", 3, vp40, 0,0);
        WConcept node41 = new WConcept("Colonial Settlement", 4, vp41, 0,0);
        WConcept node42 = new WConcept("Education Historic Centre", 3, vp42, 0,0);
        WConcept node43 = new WConcept("Aesthetics Historic Centre", 3, vp43, 0,0);
        WConcept node44 = new WConcept("Architecture Centre", 3, vp44, 0,0);
        WConcept node45 = new WConcept("University City", 4, vp45, 0,0);
        WConcept node46 = new WConcept("Music Centre", 4, vp46, 0,0);
        WConcept node47 = new WConcept("Monument Historic Centre", 3, vp47, 0,0);
        WConcept node48 = new WConcept("Religion Historic Centre", 3, vp48, 0,0);
        WConcept node49 = new WConcept("Judaism Historic Centre", 4, vp49, 0,0);
        WConcept node50 = new WConcept("Buddhism Historic Centre", 4, vp50, 0,0);
        WConcept node51 = new WConcept("Christianity Historic Centre", 4, vp51, 0,0);
        WConcept node52 = new WConcept("Islam Historic Centre", 4, vp52, 0,0);
        WConcept node53 = new WConcept("Religion Blending Witness", 4, vp53, 0,0);
        WConcept node54 = new WConcept("Judaism Holy City", 5, vp54, 0,0);
        WConcept node55 = new WConcept("Christianity Holy City", 5, vp55, 0,0);
        WConcept node56 = new WConcept("Islam Holy City", 5, vp56, 0,0);
        WConcept node57 = new WConcept("Fortified Town", 3, vp57, 0,0);
        WConcept node58 = new WConcept("Historic Capital", 3, vp58, 0,0);
        WConcept node59 = new WConcept("Rural Settlement", 3, vp59, 0,0);
        WConcept node60 = new WConcept("Urban Settlement", 3, vp60, 0,0);
        WConcept node61 = new WConcept("Settlement in Certain Environment", 4, vp61, 0,0);
        WConcept node95 = new WConcept("Settlement Restoration", 4, vp95, 0,0);
        WConcept node62 = new WConcept("Urban Fabric", 4, vp62, 0,0);
        WConcept node63 = new WConcept("Medieval Town", 4, vp63, 0,0);
        WConcept node64 = new WConcept("Islamic City", 4, vp64, 0,0);
         WConcept node65 = new WConcept("Rock Art", 2, vp65,0,0);
         WConcept node66 = new WConcept("Park", 3, vp66,0,0);
         WConcept node67 = new WConcept("Garden", 3,vp67,0,0);
         WConcept node68 = new WConcept("Industry Landscape", 3,vp68,0,0);
         WConcept node69 = new WConcept("Agriculture Landscape", 3, vp69,0,0);
        WConcept node70 = new WConcept("Agriculture Construction", 3,vp70,0,0);
        WConcept node71 = new WConcept("Commerce Construction", 3,vp71,0,0);
        WConcept node72 = new WConcept("Industry Construction", 3, vp72, 0,0);
        WConcept node73 = new WConcept("Transport Construction", 3, vp73, 0,0);
        WConcept node74 = new WConcept("Public Facility", 3,vp74,0,0);
        WConcept node75 = new WConcept("Culture Blending Construction", 3,vp75,0,0);
        WConcept node76 = new WConcept("Tower", 3,vp76,0,0);
        WConcept node77 = new WConcept("Fortress", 3,vp77,0,0);
        WConcept node78 = new WConcept("Wall", 3,vp78,0,0);
        WConcept node79 = new WConcept("Castle", 3,vp79,0,0);
        WConcept node80 = new WConcept("Palace", 3,vp80,0,0);
        WConcept node81 = new WConcept("Mansion", 3,vp81,0,0);

        WConcept node82 = new WConcept("Religion Construction", 3,vp82,0,0);
        WConcept node83 = new WConcept("Monument", 3,vp83,0,0);
        WConcept node84 = new WConcept("Buddhism Construction", 4,vp84,0,0);
        WConcept node85 = new WConcept("Christianity Construction", 4,vp85,0,0);
WConcept node86 = new WConcept("Islam Construction", 4,vp86,0,0);
WConcept node87 = new WConcept("Tomb", 4,vp87,0,0);
WConcept node88 = new WConcept("Temple", 4,vp88,0,0);
WConcept node89 = new WConcept("Monastery", 4,vp89,0,0);
WConcept node90 = new WConcept("Abbey", 4,vp90,0,0);
WConcept node91 = new WConcept("Church", 5,vp91,0,0);
WConcept node92 = new WConcept("Cathedral", 5,vp92,0,0);
WConcept node93 = new WConcept("Mosque", 5,vp93,0,0);
WConcept node94 = new WConcept("Garden-Tomb", 5,vp94,0,0);
WConcept node96 = new WConcept("Convent", 4, vp95, 0,0);

vl1.add(node1);
vl1.add(node2);
vl1.add(node3);
vl1.add(node4);
vl1.add(node5);
vl1.add(node6);
vl1.add(node7);
vl1.add(node8);
vl1.add(node9);
vl1.add(node10);
vl1.add(node11);
vl1.add(node12);
vl1.add(node13);
vl1.add(node14);
vl1.add(node15);
vl1.add(node16);
vl1.add(node17);
vl1.add(node18);
vl1.add(node19);
vl1.add(node20);
vl1.add(node21);
vl1.add(node22);
vl1.add(node23);
vl1.add(node24);
vl1.add(node25);
vl1.add(node26);
vl1.add(node27);
vl1.add(node28);
vl1.add(node29);
vl1.add(node30);
vl1.add(node31);
vl1.add(node32);
vl1.add(node33);
vl1.add(node34);
vl1.add(node35);
vl1.add(node36);
vl1.add(node37);
vl1.add(node38);
vl1.add(node39);
vl1.add(node40);
vl1.add(node41);
vl1.add(node42);
```

```
vl1.add(node43);
vl1.add(node44);
vl1.add(node45);
vl1.add(node46);
vl1.add(node47);
 vl1.add(node48);
vl1.add(node49);
vl1.add(node50);
vl1.add(node51);
vl1.add(node52);
vl1.add(node53);
vl1.add(node54);
vl1.add(node55);
vl1.add(node56);
vl1.add(node57);
vl1.add(node58);
vl1.add(node59);
vl1.add(node60);
vl1.add(node61);
vl1.add(node95);
vl1.add(node62);
vl1.add(node63);
vl1.add(node64);
vl1.add(node65);
vl1.add(node66);
vl1.add(node67);
vl1.add(node68);
vl1.add(node69);
vl1.add(node70);
vl1.add(node71);
vl1.add(node72);
vl1.add(node73);
vl1.add(node74);
vl1.add(node75);
vl1.add(node76);
vl1.add(node77);
vl1.add(node78);
vl1.add(node79);
vl1.add(node80);
vl1.add(node81);
vl1.add(node82);
vl1.add(node83);
vl1.add(node84);
vl1.add(node85);
vl1.add(node86);
vl1.add(node87);
vl1.add(node88);
vl1.add(node89);
vl1.add(node90);
vl1.add(node91);
vl1.add(node92);
vl1.add(node93);
vl1.add(node94);
```

```
      vl1.add(node96);
     }
     return vl1;
     }

public float[ ] sort(float[ ] instArr)
   {
      float pass,temp;
      for(pass=0;pass<instArr.length;pass++)
        {
        for (int i=0;i<instArr.length-pass-1;i++)
           {
           if (instArr[i]<instArr[i+1])
              {
              temp=instArr[i];
              instArr[i]=instArr[i+1];
              instArr[i+1]=temp;
              }
           }
        }
      return instArr;
   }

public  Vector getinterest(Vector vnode,float[ ] cInst)
   {
      Vector vinst= new Vector();
      Vector vminst= new Vector();
      Vector interest = new Vector();
      for (int j=0; j<vnode.size();j++)
        {
         if(((WConcept)vnode.get(j)).weight==cInst[0])
           {
           vinst.add((WConcept)vnode.get(j));
           }
        }

      float [] vlev= WConcept.getlevel(vinst);

      if (vinst.size()==1)
        {
         for (int j=0; j<vnode.size();j++)
           {
           if(((WConcept)vnode.get(j)).weight==cInst[1] && cInst[1]>0)
              {
              vminst.add((WConcept)vnode.get(j));
              }
           }

        for (int i=0; i<vminst.size();i++)
           {

if(((Vector)((WConcept)vminst.get(i)).path).containsAll((Collection)(((WConcept)vinst.get(0)
).path)))
```

```
                {
                interest.add(((WConcept)vminst.get(i)).conceptname);
                }
            }

            if(interest.isEmpty())
                {
                interest.add(((WConcept)vinst.get(0)).conceptname);
                }
          }
       else if (!vinst.isEmpty())
          {
          for (int i = 0; i< vlev.length; i++)
             {
              System.out.println(vlev[i]);
              }
          sort(vlev);
          for (int j=0; j< vinst.size();j++)
             {
              if(((WConcept)vinst.get(j)).level==vlev[0])
                {
                interest.add(((WConcept)vinst.get(j)).conceptname);
                }
             }
          }
       return(interest);
   }

 public  Vector getRelativeInterest(Vector vnode,Vector vi)
   {
       Vector vnInst= new Vector();
       Vector vnb= new Vector();
       for (int i=0; i<vi.size();i++)
          {
          for(int j=0; j<vnode.size();j++)
             {
             if ((((WConcept)vnode.get(j)).conceptname).equals(vi.get(i)))
                {
                vnInst.add((WConcept)vnode.get(j));
                }
             }
          }

       for (int i=0; i<vnInst.size();i++)
          {
          for(int j=0; j<vnode.size();j++)
             {
             if((((WConcept)vnode.get(j)).level==((WConcept)vnInst.get(i)).level)&&(((WConce
             pt)vnode.get(j)).conceptname!=((WConcept)vnInst.get(i)).conceptname)&&(!(compa
             re(((WConcept)vnode.get(j)).path,((WConcept)vnInst.get(i)).path)).isEmpty()))
                {
                vnb.add(((WConcept)vnode.get(j)).conceptname);
                }
            }
         }
      return vnb;

   }

public  Vector compare(Vector v1, Vector v2)
   {
      Vector vr = new Vector();
      if (!v2.isEmpty())
         {
          for(int i=0;i<v2.size();i++)
            {
            if (v1.contains(v2.get(i)))
               {
               vr.add(v2.get(i));
               }
            }
         }
      return vr;
   }

public  Vector getinduce(Vector vp,Vector vnode)
   {
      Vector indResult = new Vector();
      for (int i = 0; i<vnode.size() ; i++)
         {
          for (int j = 0; j <vp.size() ; j++)
            {
             if (((Vector)(((WConcept)vnode.get(i)).path)).contains(vp.get(j)))
               {
               if (!indResult.contains(((Vector)(((WConcept)vnode.get(i)).path)).firstElement()))
                  {
                  indResult.add(((Vector)(((WConcept)vnode.get(i)).path)).firstElement());
                  }
               }
            }
         }
      return (indResult);
   }

public  Vector getpath(Vector vc,Vector vnode)
   {
      Vector vpath = new Vector();
      for (int j=0; j<vnode.size();j++)
         {
        if (vc.contains(((WConcept)vnode.get(j)).conceptname))
            {
            vpath.add(((WConcept)vnode.get(j)).path);
            }
         }
      return vpath;
```

```
  }

public  float[] countcategory(Vector vp,Vector vOldInst, Vector vnode)
  {
     float [] couInst = new float[vnode.size()];
     Vector OldPath = new Vector();
     for (int i=0;i<vp.size();i++)
        {
        for(int j=0;j<((Vector)vp.get(i)).size();j++)
           {
          for(int k= 0; k<vnode.size();k++)
              {
              if ((((WConcept)vnode.get(k)).conceptname).equals(((Vector)vp.get(i)).get(j)))
                 {
                 ((WConcept)vnode.get(k)).weight=((WConcept)vnode.get(k)).weight+((WConc
                 ept)vnode.get(k)).uweight;}
                 }
              }
           }
        }

     if (!vOldInst.isEmpty())
        {
        for (int i=0;i<vOldInst.size();i++)
           {
          for(int k= 0; k<vnode.size();k++)
              {
              if ((((WConcept)vnode.get(k)).conceptname).equals(vOldInst.get(i)))
                 {
                 OldPath.add((Vector)((WConcept)vnode.get(k)).path);
                 }
              }
           }


          for (int i=0;i<OldPath.size();i++)
              {
              for(int j=0;j<((Vector)OldPath.get(i)).size();j++)
                 {
                 for(int k= 0; k<vnode.size();k++)
                    {
                    if
((((WConcept)vnode.get(k)).conceptname).equals(((Vector)OldPath.get(i)).get(j)))
                       {
                       ((WConcept)vnode.get(k)).weight= ((WConcept)vnode.get(k)).weight-
                       ((WConcept)vnode.get(k)).uweight;
                       }
                    }
                 }
              }
           }

     for (int i = 0; i < couInst.length; i++)
```

```
        {
        couInst[i]=  ((WConcept)vnode.get(i)).weight;
        System.out.println(couInst[i]);
        }
      return couInst;
  }
  }
```

**********Wposition.java**********

```
package winduction;
import java.sql.*;
import java.util.*;
import java.io.*;

public class Wposition
   {
   String Psname;
   float count;

public Wposition (String nn,float vts)
   {
   Psname=nn;
   count=vts;
   }
   }
```

***********WAllLocations.java**********

```
package winduction;
import java.sql.*;
import java.util.*;
import java.io.*;

public class WAllLocations
   {
   Vector vCountry ;
   Vector vComposition;
   Vector vGeography;

public WAllLocations (Vector vcoun, Vector vcomp,Vector vgeog)
   {
   vCountry=vcoun;
   vComposition=vcomp;
   vGeography=vgeog;
   }

 public Vector getVcoun()
   {
   return this.vCountry;
   }

 public Vector getVcomp()
```

```
  {
  return this.vComposition;
  }

public Vector getVgeog()
  {
  return this.vGeography;
  }

public Vector AddorCountP(Wposition pst, Vector vpos, Vector voldinst)
  {
     boolean flag =true;
     String psname = pst.Psname;

     if(!(voldinst.isEmpty()))
        {
        for (int i=0;i<vpos.size();i++)
           {
            for (int j=0;j<voldinst.size();j++)
               {
               if (((Wposition)vpos.get(i)).Psname.equals(voldinst.get(j)))
                  {
                  ((Wposition)vpos.get(i)).count=((Wposition)vpos.get(i)).count-1;
                  }
               }
           }
        }

     for (int i=0;i<vpos.size();i++)
        {
         if (((Wposition)vpos.get(i)).Psname.equals(psname))
            {
            ((Wposition)vpos.get(i)).count= ((Wposition)vpos.get(i)).count+1;
            flag=false;
            }
        }

     if (flag)
        {
         pst.count++;
         vpos.add(pst);
         }
     return(vpos);
  }

public float[ ] countPosition(Vector vpos)
  {
     float [ ] countP = new float[vpos.size()];
     for (int i = 0; i < countP.length; i++)
        {
         countP[i]=  ((Wposition)vpos.get(i)).count;
         }
     return countP;
  }

public Vector getMostP(Vector vpos,float[ ] cInst)
  {
     Vector vinstP= new Vector();
     for (int j=0; j<vpos.size();j++)
        {
       if(((Wposition)vpos.get(j)).count==cInst[0])
          {
          vinstP.add((Wposition)vpos.get(j));
          }
       }
     return vinstP;
  }

 public Vector getInstofP (Vector vpos1,Vector vpos2,Vector vpos3)
  {
     Vector InstofP = new Vector();
     if (vpos1.size()==1)
        {
        InstofP.add(((Wposition)vpos1.get(0)).Psname);
        }
     else
        {
        if (vpos2.size()==1)
           {
           InstofP.add(((Wposition)vpos2.get(0)).Psname);
           }
        else
           {
           if (vpos3.size()==1)
              {
              InstofP.add(((Wposition)vpos3.get(0)).Psname);
              }
           else
              {
              for(int i=0;i<vpos1.size();i++)
                 {
                 InstofP.add(((Wposition)vpos1.get(i)).Psname);
                 }
              }
           }
        }
     return InstofP;
  }
  }
```

Appendix B

HTML source code

*********inddex.html*********

```
<html>
<head>
<title>Our World Heritage Explorer</title>
<link rel="stylesheet" type="text/css" href="/engine/css/style.css">
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<body bgcolor="#ccddee" link="#000000" vlink="#000000" alink="#aa4400" leftmargin="0"
topmargin="0" marginwidth="0" marginheight="0">
<center>
   <table width="800" cellspacing="0" cellpadding="0" border="0">
   <h2>Inducing Queries from examples in World Heritage </h2>
   <p><i>Jing Zhou and Yukun Zheng</i></p>
   <tr>
  <td>
  <br><br>                <!-- Indhold start -->

      <table width="800" border="0" cellspacing="0" cellpadding="0">
         <tr>
         <td width="130" valign="top">
            <table width="130" border="0" cellpadding="8" cellspacing="8">
            <tr>
            <td>&nbsp;</td>
            </tr>
            <tr>
            <td><img src="/projsp/picture/100.jpg" width="80" height="60" hspace="10"
border="0"></td>
            </tr>
            <tr>
            <td><img src="/projsp/picture/101.jpg" width="80" height="60" hspace="10"
border="0"></td>
            </tr>
            <tr>
            <td><img src="/projsp/picture/104.jpg" width="80" height="60" hspace="10"
border="0"></td>
            </tr>
            <tr>
            <td><img src="/projsp/picture/102.jpg" width="80" height="60" hspace="10"
border="0"></td>
            </tr>
            <tr>
            <td><img src="/projsp/picture/103.jpg" width="80" height="60" hspace="10"
border="0"></td>
            </tr>
            <tr>
            <td>&nbsp;</td>
            </tr>
          </tr>
          </table>
          <br>
          </td>

          <td>
          <table width="660" border="0" cellpadding="8" cellspacing="8">
            <tr valign="top">
            <td>
            <!-- Indhold start -->
            <font color="blue"><b>PROTOTYPE</b></font><br>
            We design and implement a query induction system that can computationally induce
            the user's interests on the basis of his/her browsing of the World Heritage sites.
            </p>
            <p> This query induction system can induce the queries from examples in World
            Heritage, and suggest the user the sites based on these queries. This system also has
            basic information on all 754 World Heritage sites and gives an introduction to the
            outstanding and universal aspects of each site.
            </p>
             In order to illustrate the process of query induction, we also provide the following
information:
               <ul>
                 <li>Category: the cateogies of the visited site </li>
                 <li>Geographical Location: the country, composition of continient and
continient of the    selected site</li>
                 <li>Sorted count: the counts of the concepts in ascending order </li>
                 <li>Results of Concept Generalisation of Category  </li>
                 <li>Results of Concept Specialisation of Category </li>
                 <li>Repeat Times: the repeat times of a concept as the result of Concept
Generalisation </li>
                 <li>Visiting Times: the number of the sites visited by the user </li>
                 <li>Results of Concept Specialisation of Geographical Location</li>
                 <li>Query Induction Results: the sites suggested to the uesr, based on the user's
visited sites       </li>
                </ul>
                <br><br>

                 Relative Link:<br>
                 <ul>
                 <li><a href="http://whc.unesco.org/">World Heritage offical Web
Site</a><br></li>
                 <li><a href="http://www.world-heritage-explorer.org">World Heritage
Explorer</a><br></li>

                 </ul>

                <a href="http://130.225.76.156:8080/projsp/Mainclass.html"><b>Enter</b></a>

                <!-- Indhold slut -->
                </td>
            </tr>
          </table></td>
          </tr>
        </table>
        <br>
      </td>
      </tr>
            <!-- Indhold slut -->
<br><br>

</td>
</tr>
</table>
</center>
</body>
</html>
```

***********Mainclass.html***********

```
<html>
<head>
<title>World Heritage Explorer</title>
<link rel="stylesheet" type="text/css" href="/engine/css/style.css">
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<body bgcolor="#ccddee" link="#000000" vlink="#000000" alink="#aa4400" leftmargin="0"
topmargin="0" marginwidth="0" marginheight="0">
<center>
  <table width="800" cellspacing="0" cellpadding="0" border="0">
  <tr >
  <td >
  <br><br>              <!-- Indhold start -->
     <table width="800" border="0" cellspacing="0" cellpadding="0">
     <tr>
     <td width="130" valign="top">
        <table width="130" border="0" cellpadding="8" cellspacing="8">
           <tr>
           <td>&nbsp;</td>
           </tr>
           <tr>
           <td><img src="/projsp/picture/100.jpg" width="80" height="60" hspace="10"
border="0"></td>
           </tr>
           <tr>
           <td><img src="/projsp/picture/101.jpg" width="80" height="60" hspace="10"
border="0"></td>
           </tr>
           <tr>
           <td><img src="/projsp/picture/104.jpg" width="80" height="60" hspace="10"
border="0"></td>
           </tr>
           <tr>
```

```
            <td><img src="/projsp/picture/102.jpg" width="80" height="60" hspace="10"
border="0"></td>
            </tr>
            <tr>
            <td><img src="/projsp/picture/103.jpg" width="80" height="60" hspace="10"
border="0"></td>
            </tr>
            <tr>
            <td>&nbsp;</td>
            </tr>

            </tr>
         </table>
        <br>
        </td>

     <td>
         <table width="660" border="0" cellpadding="8" cellspacing="8">
         <tr><th align=left>Show all Sites </th></tr>
         <tr>
         <td colspan="2">
             <li><a href="jsp/allsites.jsp?CategoryName=All%20Sites&Nodenum=130">All
Sites by Country</a>
         </td>
         </tr>
          <tr><th align=left>Search by Category (Version1)</th> </tr>
          <tr>
          <td>
            <li><a href='jsp/CNallsites.jsp?CategoryName=Natural%20Sites&Nodenum=35
                   &subcategoryid=1'>Natural sites</a><ul>
            <li><a href='jsp/CNcategory.jsp?CategoryName=Geography&Nodenum=11
                   &Subcid=9'> Geography</a>
            <li><a href='jsp/CNcategory.jsp?CategoryName=Natural+Beauty&Nodenum=0
                   &Subcid=7'> Natural Beauty</a>
            <li><a href='jsp/CNcategory.jsp?CategoryName=Biography&Nodenum=0
                   &Subcid=10'> Biodiversity</a>
            <li><a href='jsp/CNcategory.jsp?CategoryName=Geology&Nodenum=11
                   &Subcid=11'> Geology</a>
            <li><a href='jsp/CNcategory.jsp?CategoryName=Ecosystem&Nodenum=0
                   &Subcid=12'> Ecosystem</a>
            <li><a href='jsp/CNcategory.jsp?CategoryName=Biology&Nodenum=8
                   &Subcid=13'> Biology</a>
            <li><a href='jsp/CNcategory.jsp?CategoryName=Topography&Nodenum=1
                   &Subcid=8'> Topography</a>
            <li><a href='jsp/CNcategory.jsp?CategoryName=Wintering+Land&Nodenum=0
                   &Subcid=14'> "Wintering" Land</a>
            </ul>
            <br>
            <li><a href='jsp/CNallsites.jsp?CategoryName=Cultural+Sites&Nodenum=95
                   &subcategoryid=6'>Cultural sites</a><ul>
            <li><a
href='jsp/CNcategory.jsp?CategoryName=Historic+Cities+and+Areas&Nodenum=47
                   &Subcid=1'> Historic City and Area</a>
            <li><a href='jsp/CNcategory.jsp?CategoryName=Construction&Nodenum=20
                    &Subcid=2'> Construction</a>
            <li><a href='jsp/CNcategory.jsp?CategoryName=Artistic+Work&Nodenum=20
                    &Subcid=3'> Artistic Work</a>
            <li><a
href='jsp/CNcategory.jsp?CategoryName=Culture+Landscape&Nodenum=20
                    &Subcid=4'> Cultural Landscape</a>
           <li><a href='jsp/CNcategory.jsp?CategoryName=Homonid+Sites&Nodenum=20
                    &Subcid=5'> Hominid Sites</a>
           <li><a
href='jsp/CNcategory.jsp?CategoryName=Archaeological+Sites&Nodenum=20
                    &Subcid=6'> Archaeological Sites</a>
           </ul>
         </td>
         </tr>

         <tr><th align=left>Search by Category (Version2)</th> </tr>
         <tr>
         <td>
            <li><a href='jsp/Nsv2.html'>Natural sites</a>
            <li><a href='jsp/Csv2.html'>Cultural sites</a>
         </td>
          </tr>
       </table>
     </td>
     </tr>
  </table>
  <!-- Indhold slut -->

<br><br>
</td>
</tr>
</table>
</center>
</body>
</html>
```

***********Nsv2.html***********

```
<html>
<head><title>Concept Generalisation by Weight in Category</title>
</head>
<body bgcolor="#ccddee" link="#000000" vlink="#000000" alink="#aa4400" leftmargin="0"
topmargin="0" marginwidth="0" marginheight="0" >
<div align="center">
<h2>Concept Generalisation by Weight in Category</h2>
</div>
<form method="POST" action="WNsites.jsp">
<p align="left">
Concept Generalisation by Weight allows a user assign the unit weights to the concepts that
they feel interested in. So it is more possible for the concepts with higher unit weight to be the
result of Concept Generalisation, because they are more important to the user. The value of
```

```
weight is a float number from 0 to 1. For example, if you are most interested in the Geography
category , you can define its weight as 1. Then if you feel a littele interested in Biology
category, you can define its weight as 0.8. If you are not interested in the other categories, you
only defined their weight as 0. We give the important categories in Natural sites as below,
please assign the weights for them.
</P>
<br>

<pre>
<div align="center">
<b> Please input a float number from 0 to 1</b>
<br>
Weight of Geography:       <input type="text" name="gpy" size="3">   Weight of Ecosystem:
<input type="text" name="esm" size="3">
Weight of Natural Beauty:  <input type="text" name="nby" size="3">   Weight of Biology:
<input type="text" name="boy" size="3">
Weight of Biodiversity:    <input type="text" name="bdy" size="3">   Weight of Topography:
<input type="text" name="tgy" size="3">
Weight of Geology:         <input type="text" name="ggy" size="3">   Weight of Witering
land:  <input type="text" name="wl" size="3">
<input type="submit" value="Submit">   <input type="reset" value="Modify">
</div>
</pre>
</form>
</body>
</html>
```

*********Csv2.html*********

```
<html>
<head><title>Concept Generalisation by Weight in Category</title>
</head>
<body bgcolor="#ccddee" link="#000000" vlink="#000000" alink="#aa4400" leftmargin="0"
topmargin="0" marginwidth="0" marginheight="0" >
<div align="center">
<h2>Concept Generalisation by Weight in Category</h2>
</div>
<form method="POST" action="WCsites.jsp">
<p align="left">
Concept Generalisation by Weight allows a user assign the unit weights to the concepts that
they feel interested in. So it is more possible for the concepts with higher unit weight to be the
result of Concept Generalisation, because they are more important to the user. The value of
weight is a float number from 0 to 1. For example, if you are interested in the Construction
category at most, you can define its weight as 1. In order, you are interested in Art Work
category, and then you can define its weight as 0.8. If you are not interested other categories,
you only defined their weight as 0. We give the important categories in Cultural sites as
below, please assign the weights for them.
</P>
<br>

<pre>
<div align="center">
```

```
<b>Note:please input a number from 0 to 1</b>
<br>
Weight of Historic Cities and Areas:  <input type="text" name="hca" size="3">   Weight of
Construction:        <input type="text" name="woc" size="3">
Weight of Art Work:                  <input type="text" name="art" size="3">   Weight of
Cultural Landscape:   <input type="text" name="cls" size="3">
Weight of Hominid Sites:             <input type="text" name="hms" size="3">   Weight of
Archaeological Sites: <input type="text" name="arc" size="3">

<input type="submit" value="Submit">   <input type="reset" value="Modify">
</div>
</pre>
</form>
</body>
</html>
```

Appendix C

JSP source code

************allsites.jsp**********

```
<html>
<head>
<title>All Sites of World Heritage </title>
<link rel="stylesheet" type="text/css" href="/engine/css/style.css">
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<%@ page language= "java" session = "true"
import="java.sql.*,java.util.*,java.io.*,induction.*" %>

<body bgcolor="#ccddee" link="#000000" vlink="#000000" alink="#000000" leftmargin="0"
topmargin="0" marginwidth="0" marginheight="0">
<%
String Ln = request.getParameter("CategoryName"); %>


<center>
<table width="660" border="0" cellpadding="8" cellspacing="0">

<tr><h1> <%= Ln%> by Country</h>
<hr width=80% color="#000000" noshade size="1">
<tr><th align="left" width="160" >Country</th><th align="left" width="500" >Site
</th></tr>

<%
Vector vNode = new Vector();
Vector vposC = new Vector(); /* save the positions of visited sites*/
Vector vposCp = new Vector();
Vector vposG = new Vector();

AllLocations Aloc= new AllLocations(vposC,vposCp,vposG);
session.setAttribute("Aloc", Aloc);

Vector vinstNC = new Vector();
Vector vinstL = new Vector();
Vector vaccInst=new Vector();
Vector vaccInstC=new Vector();
Vector vaccInstCp=new Vector();
Vector vaccInstG=new Vector();

Interest AINST = new Interest(0,0,vinstNC,vinstL,vaccInst,vaccInstC,vaccInstCp,vaccInstG);
session.setAttribute("AINST", AINST);

vNode=Category.initConcepts(Ln);
```

```
int nodeNum=Integer.parseInt(request.getParameter("Nodenum"));
if (nodeNum!=0)
  {
  Category ALat = new Category(Ln,nodeNum,vNode);
  session.setAttribute("ALat", ALat);
  }

Class.forName("com.mysql.jdbc.Driver").newInstance();
java.sql.Connection conn1;
conn1 =
DriverManager.getConnection("jdbc:mysql://130.225.76.177/exwh?user=kun&password=som
e_pass");

Statement stmt1 = conn1.createStatement ();
ResultSet rs = stmt1.executeQuery("select distinct country from country, LOCcountry where
country.countryid=LOCcountry.countryid order by country");

String psquery="select distinct site_name from site, LOCcountry, country where
country.country=? and country.countryid=LOCcountry.countryid and LOCcountry.siteid
=site.siteid";

PreparedStatement ps=conn1.prepareStatement(psquery);
String cn = "";
if (rs!=null)
  {
  while(rs.next())
     {
     int css=0;
     cn=rs.getString("country");
     ps.setString(1,cn);
     ResultSet myResultSet = ps.executeQuery();
%>
<tr><td>
<%
     while(myResultSet.next())
        {
        css++;
%>
<tr><td><font color="blue">
<%
        if (css<=1)
           {
           out.println(cn);
           }
%>
</font></td><td></td></tr>
<%
           String sn=myResultSet.getString("site_name");
%>
<tr><td></td><td><dd><a href='Displaysites.jsp?site_name=<%=sn.replace(' ','+')%>'><%=
sn%></a></td></tr>
</tr>
<%
```

```
        }
      }
    }
stmt1.close();
conn1.close();
%>
</table>                  <!-- Indhold slut -->
</td>
</tr>
</table>
</center>
</body>
</html>
```

************CNallsites.jsp************

```
<html>
<head>
<title>
ALl natural sites
</title>
</head>
<%@ page language= "java" import="java.sql.*,java.util.*,java.io.*,induction.*" %>
<body bgcolor="#ccddee" link="#000000" vlink="#000000" alink="#000000" leftmargin="0"
topmargin="0" marginwidth="0" marginheight="0">
<center>

<%
String Ln = request.getParameter("CategoryName"); %>


<table width="660" border="0" cellpadding="8" cellspacing="0">
<h1><%= Ln%></h>
<hr width=80% color="#000000" noshade size="1">
<tr><th align="left" width="160" >Country</th><th align="left" width="500" >Site
</th></tr>

<%
Vector vNode = new Vector();

Vector vposC = new Vector(); /* save the positions of visited sites*/
Vector vposCp = new Vector();
Vector vposG = new Vector();

AllLocations Aloc= new AllLocations(vposC,vposCp,vposG);
session.setAttribute("Aloc", Aloc);

Vector vinstNC = new Vector();
Vector vinstL = new Vector();
Vector vaccInst=new Vector();
Vector vaccInstC=new Vector();
Vector vaccInstCp=new Vector();
Vector vaccInstG=new Vector();

Interest AINST = new Interest(0,0,vinstNC,vinstL,vaccInst,vaccInstC,vaccInstCp,vaccInstG);
session.setAttribute("AINST", AINST);

vNode=Category.initConcepts(Ln);
int nodeNum=Integer.parseInt(request.getParameter("Nodenum"));

if (nodeNum!=0)
   {
   Category ALat = new Category(Ln,nodeNum,vNode);
   session.setAttribute("ALat", ALat);
   }

Class.forName("com.mysql.jdbc.Driver").newInstance();
java.sql.Connection conn1;
conn1 =
DriverManager.getConnection("jdbc:mysql://130.225.76.177/exwh?user=kun&password=som
e_pass");

String psq="select distinct country from country, LOCcountry, category where
category.subcategoryid=? and category.siteid = LOCcountry.siteid and
LOCcountry.countryid=country.countryid order by country";
int subcid=Integer.parseInt(request.getParameter("subcategoryid"));

PreparedStatement stmt1 = conn1.prepareStatement (psq);
stmt1.setInt(1,subcid);
ResultSet rs = stmt1.executeQuery();

String psquery="select distinct site_name from site,category, LOCcountry, country where
site.siteid=category.siteid and category.subcategoryid=? and country.country=? and
country.countryid = LOCcountry.countryid and LOCcountry.siteid =site.siteid";

PreparedStatement ps=conn1.prepareStatement(psquery);
ps.setInt(1,subcid);

String cn = "";
if (rs!=null)
   {
   while(rs.next())
      {
      int css =0;
      cn=rs.getString("country");
      ps.setString(2,cn);
      ResultSet myResultSet = ps.executeQuery();

%>
<tr><td>
<%
      while(myResultSet.next())
         {
         css++;
%>
<tr><td><font color="blue">
<%
        if (css<=1)
            {
            out.println(cn);
             }
%>
</font></td><td></td></tr>
<%
        String sn=myResultSet.getString("site_name");
%>

<tr><td></td><td><dd><a href='Displaysites.jsp?site_name=<%=sn.replace(' ','+')%>'><%=
sn%></a></td></tr>
</tr>
<%
        }
     }
  }
stmt1.close();
conn1.close();
%>
</table>                  <!-- Indhold slut -->

</td>
</tr>
</table>
</center>
</body>
</html>
```

************WNsites.jsp************

```
<html>
<head>
<title>
All natural sites
</title>
</head>
<%@ page language= "java" import="java.sql.*,java.util.*,java.io.*,winduction.*" %>

<body bgcolor="#ccddee" link="#000000" vlink="#000000" alink="#000000" leftmargin="0"
topmargin="0" marginwidth="0" marginheight="0">

<center>
<%
String Ln = "Natural Sites";
int nodeNum=35;
%>

<table width="660" border="0" cellpadding="8" cellspacing="0">
<h1><%= Ln%></h>
```

```
<hr width=80% color="#000000" noshade size="1">
<tr><th align="left" width="160" >Country</th><th align="left" width="500" >Site
</th></tr>

<%
Vector vNode = new Vector();

Vector vposC = new Vector(); /* save the positions of visited sites*/
Vector vposCp = new Vector();
Vector vposG = new Vector();

WAllLocations WNloc= new WAllLocations(vposC,vposCp,vposG);
session.setAttribute("WNloc", WNloc);

Vector vinstNC = new Vector();
Vector vinstL = new Vector();
Vector vaccInst=new Vector();
Vector vaccInstC=new Vector();
Vector vaccInstCp=new Vector();
Vector vaccInstG=new Vector();

Winterest AINST = new
Winterest(0,0,vinstNC,vinstL,vaccInst,vaccInstC,vaccInstCp,vaccInstG);
session.setAttribute("AINST", AINST);

vNode=WCategory.initWconcepts(Ln);

for (int i=0;i<vNode.size();i++)
  {
  if (((Vector)(((WConcept)(vNode.get(i))).getPath())).contains("geography"))
      {
      ((WConcept)(vNode.get(i))).setUweight(Float.parseFloat(request.getParameter("gpy")));
      }

  if (((Vector)(((WConcept)(vNode.get(i))).getPath())).contains("natural beauty"))
      {
      ((WConcept)(vNode.get(i))).setUweight(Float.parseFloat(request.getParameter("nby")));
      }
  if (((Vector)(((WConcept)(vNode.get(i))).getPath())).contains("biodiversity"))
      {
      ((WConcept)(vNode.get(i))).setUweight(Float.parseFloat(request.getParameter("bdy")));
      }
  if (((Vector)(((WConcept)(vNode.get(i))).getPath())).contains("geology"))
      {
       ((WConcept)(vNode.get(i))).setUweight(Float.parseFloat(request.getParameter("ggy")));
      }
  if (((Vector)(((WConcept)(vNode.get(i))).getPath())).contains("ecosystem"))
     {
      ((WConcept)(vNode.get(i))).setUweight(Float.parseFloat(request.getParameter("esm")));
     }
  if (((Vector)(((WConcept)(vNode.get(i))).getPath())).contains("biology"))
     {
     ((WConcept)(vNode.get(i))).setUweight(Float.parseFloat(request.getParameter("boy")));
```

```
    }
  if (((Vector)(((WConcept)(vNode.get(i))).getPath())).contains("topography"))
    {
    ((WConcept)(vNode.get(i))).setUweight(Float.parseFloat(request.getParameter("tgy")));
    }
  if (((Vector)(((WConcept)(vNode.get(i))).getPath())).contains("wintering land"))
    {
    ((WConcept)(vNode.get(i))).setUweight(Float.parseFloat(request.getParameter("wl")));
    }
  }

WCategory NWLat = new WCategory(Ln,nodeNum,vNode);
session.setAttribute("NWLat", NWLat);

Class.forName("com.mysql.jdbc.Driver").newInstance();
java.sql.Connection conn1;
conn1 =
DriverManager.getConnection("jdbc:mysql://130.225.76.177/exwh?user=kun&password=som
e_pass");

Statement stmt1 = conn1.createStatement ();
ResultSet rs = stmt1.executeQuery("select distinct country from country, LOCcountry,
category where category.subcategoryid=1 and category.siteid = LOCcountry.siteid and
LOCcountry.countryid=country.countryid order by country");

String psquery="select distinct site_name from site,category, LOCcountry, country where
site.siteid=category.siteid and category.subcategoryid=1 and country.country=? and
country.countryid = LOCcountry.countryid and LOCcountry.siteid =site.siteid";

PreparedStatement ps=conn1.prepareStatement(psquery);

String cn = "";
if (rs!=null)
  {
  while(rs.next())
     {
       int css =0;
       cn=rs.getString("country");
       ps.setString(1,cn);
       ResultSet myResultSet = ps.executeQuery();

%>
<tr><td>
<%

       while(myResultSet.next()){
       css++;
%>
<tr><td><font color="blue">
<%
       if (css<=1)
         {
         out.println(cn);
```

```
            }
%>
</font></td><td></td></tr>
<%
        String sn=myResultSet.getString("site_name");
%>

<tr><td></td><td><dd><a href='DisplayWNsites.jsp?site_name=<%=sn.replace('
','+')%>'><%= sn%></a></td></tr>
</tr>
<%
        }
      }
  }

stmt1.close();
conn1.close();
%>
</table>                <!-- Indhold slut -->

</td>
</tr>
</table>
</center>
</body>
</html>
```

***********<b>WCsites.jsp</b>***********

```
<html>
<head>
<title>
All cultural sites
</title>
</head>
<%@ page language= "java" import="java.sql.*,java.util.*,java.io.*,winduction.*" %>

<body bgcolor="#ccddee" link="#000000" vlink="#000000" alink="#000000" leftmargin="0"
topmargin="0" marginwidth="0" marginheight="0">


<center>
<%
String Ln = "Cultural Sites";
int nodeNum=95;
%>

<table width="660" border="0" cellpadding="8" cellspacing="0">
<h1><%= Ln%></h>
<hr width=80% color="#000000" noshade size="1">
<tr><th align="left" width="160" >Country</th><th align="left" width="500" >Site
</th></tr>
```

```
<%
Vector vNode = new Vector();

Vector vposC = new Vector(); /* save the positions of visited sites*/
Vector vposCp = new Vector();
Vector vposG = new Vector();

WAllLocations WNloc= new WAllLocations(vposC,vposCp,vposG);
session.setAttribute("WNloc", WNloc);

Vector vinstNC = new Vector();
Vector vinstL = new Vector();
Vector vaccInst=new Vector();
Vector vaccInstC=new Vector();
Vector vaccInstCp=new Vector();
Vector vaccInstG=new Vector();

Winterest AINST = new
Winterest(0,0,vinstNC,vinstL,vaccInst,vaccInstC,vaccInstCp,vaccInstG);
session.setAttribute("AINST", AINST);

vNode=WCategory.initWconcepts(Ln);

for (int i=0;i<vNode.size();i++)
  {
  if  (((Vector)(((WConcept)(vNode.get(i))).getPath())).contains("Historic Cities and Areas"))
      {
     ((WConcept)(vNode.get(i))).setUweight(Float.parseFloat(request.getParameter("hca")));
      }
  if (((Vector)(((WConcept)(vNode.get(i))).getPath())).contains("Construction"))
      {
     ((WConcept)(vNode.get(i))).setUweight(Float.parseFloat(request.getParameter("woc")));
      }
  if (((Vector)(((WConcept)(vNode.get(i))).getPath())).contains("Hominid Sites"))
      {
      ((WConcept)(vNode.get(i))).setUweight(Float.parseFloat(request.getParameter("hms")));
      }
  if (((Vector)(((WConcept)(vNode.get(i))).getPath())).contains("Art Work"))
      {
      ((WConcept)(vNode.get(i))).setUweight(Float.parseFloat(request.getParameter("art")));
      }
  if (((Vector)(((WConcept)(vNode.get(i))).getPath())).contains("Archaeological Sites"))
      {
      ((WConcept)(vNode.get(i))).setUweight(Float.parseFloat(request.getParameter("arc")));
      }
  if (((Vector)(((WConcept)(vNode.get(i))).getPath())).contains("Culture Landscape"))
      {
      ((WConcept)(vNode.get(i))).setUweight(Float.parseFloat(request.getParameter("cls")));
      }
  }

WCategory NWLat = new WCategory(Ln,nodeNum,vNode);
```

```
session.setAttribute("NWLat", NWLat);

Class.forName("com.mysql.jdbc.Driver").newInstance();
java.sql.Connection conn1;
conn1 =
DriverManager.getConnection("jdbc:mysql://130.225.76.177/exwh?user=kun&password=som
e_pass");

Statement stmt1 = conn1.createStatement ();
ResultSet rs = stmt1.executeQuery("select distinct country from country, LOCcountry,
category where category.subcategoryid=6 and category.siteid = LOCcountry.siteid and
LOCcountry.countryid=country.countryid order by country");

String psquery="select distinct site_name from site,category, LOCcountry, country where
site.siteid=category.siteid and category.subcategoryid=6 and country.country=? and
country.countryid = LOCcountry.countryid and LOCcountry.siteid =site.siteid";

PreparedStatement ps=conn1.prepareStatement(psquery);

String cn = "";
if (rs!=null)
   {
  while(rs.next())
      {
         int css =0;
         cn=rs.getString("country");
         ps.setString(1,cn);
         ResultSet myResultSet = ps.executeQuery();
%>
<tr><td>
<%
         while(myResultSet.next())
           {
           css++;
%>
<tr><td><font color="blue">
<%
           if (css<=1)
             {
             out.println(cn);
             }
%>
</font></td><td></td></tr>
<%
          String sn=myResultSet.getString("site_name");
%>

<tr><td></td><td><dd><a href='DisplayWNsites.jsp?site_name=<%=sn.replace('
','+')%>'><%= sn%></a></td></tr>
</tr>
<%
        }
      }
  }
stmt1.close();
conn1.close();
%>
</table>                 <!-- Indhold slut -->

</td>
</tr>
</table>
</center>
</body>
</html>
```

***********Displaysites.jsp***********

```
<html>
<head>
<title>World Heritage Sites Display</title>
<link rel="stylesheet" type="text/css" href="/engine/css/style.css">
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<%@ page language= "java" import="java.sql.*,java.util.*,java.io.*,induction.*" %>

<body bgcolor="#ccddee" link="#000000" vlink="#000000" alink="#aa4400" leftmargin="0"
topmargin="0" marginwidth="0" marginheight="0">
 <%

Category asLat =(Category)session.getAttribute("ALat");
AllLocations Aloc =(AllLocations)session.getAttribute("Aloc");
Interest AINST =(Interest)session.getAttribute("AINST");

Connection conn = null;
Vector vl = new Vector();

Vector vcc   = new Vector();/* vector of categories about one site*/
Vector vPc= new Vector();/* vector of country abot one site*/
Vector vPcp= new Vector();/*vector of composition about one site*/
Vector vPg= new Vector(); /*Vector of region about one site*/

Vector vInstC = new Vector(); /* save the interest of positions from the method getMostP*/
Vector vInstCp = new Vector();
Vector vInstG = new Vector();

int [ ] countC = new int[236];
int [ ] countCp = new int[36];
int [ ] countG = new int[6];

Vector vInst = new Vector();/* the interst of Location*/
Vector vCp = new Vector(); /* all the compositions */
Vector vReg = new Vector();/* all the Regions*/

Vector vpth1 = new Vector();
Vector vpth2 = new Vector();
Vector vinduce = new Vector();
Vector vinsts = new Vector();
Vector vres = new Vector();
Vector vlv = new Vector();

Vector vNC = new Vector(); /*natural property or cultural*/

vl=asLat.getConcepts();

int [] curInst = new int[vl.size()];

String Inst="";
String userName;
String password;
String url;

try
   {
       userName= "kun";
       password="some_pass";
       url= "jdbc:mysql://130.225.76.177/exwh";
       conn= DriverManager.getConnection (url,userName,password);
   }
catch(Exception e){}

String Sn= request.getParameter("site_name");
%>

<center>
  <table width="800" border="0" cellspacing="0" cellpadding="0">
        <tr>
        <td width="800" valign="top"  >
            <table width="800" border="0" cellpadding="8" cellspacing="0">
               <tr>
               <td colspan= "2" valign="top" class="overskrift"><font size=4><b><%=
Sn%></font></td>
               </tr>

               <tr>
               <td height="1" colspan="3"><hr color="#000000" noshade size="1"></td>
               </tr>
<%
String psqueryD1="select distinct country from country, site, LOCcountry where site_name=?
and country.countryid = LOCcountry.countryid and LOCcountry.siteid =site.siteid";

String psqueryD2="select distinct location_name from site, location where site_name=? and
location.siteid =site.siteid";

PreparedStatement psD1=conn.prepareStatement(psqueryD1);
PreparedStatement psD2=conn.prepareStatement(psqueryD2);

psD1.setString(1,Sn);
```

```
psD2.setString(1,Sn);
ResultSet myResultSet1 = psD1.executeQuery();
ResultSet myResultSet2 = psD2.executeQuery();

if (myResultSet1 !=null)
   {
   while (myResultSet1.next())
       {
       String snm = myResultSet1.getString("country");
%>
              <tr>
              <td  class='underOverskrift'><%= snm%></td>
<%
       }
   }
%>
<%
if (myResultSet2 !=null)
   {
   while (myResultSet2.next())
       {
       String ln = myResultSet2.getString("location_name");
%>
              <td colspan="2" align='right' ><%= ln%></td>
<%
       }
   }
%>
              </tr>
              <tr>
              <td height="1" colspan="3"><hr color="#000000" noshade size="1"></td>
              </tr>
           </table>
       </td>

        <tr>
        <table width="800" border="0" cellspacing="2" cellpadding="0">
        <tr>
        <td width="400" valign="top">
            <table width="400" border="0" cellpadding="8" cellspacing="0">
                <tr>
                <td>
<%
 String psquerycn = "select distinct cncategory from cncategory, site where site.site_name = ?
and site.siteid = cncategory.siteid";

 String psqueryPc = "select distinct country from country,LOCcountry,site where
site.site_name = ? and site.siteid = LOCcountry.siteid and LOCcountry.countryid =
country.countryid";

String psqueryPcp = "select distinct composition_name from
composition,country,LOCcountry,site where site.site_name = ? and site.siteid =
LOCcountry.siteid and LOCcountry.countryid = country.countryid and
composition.compositionid=country.compositionid";

String psqueryPg= "select distinct continent_name from
continent,composition,country,LOCcountry,site where site.site_name = ? and site.siteid =
LOCcountry.siteid and LOCcountry.countryid = country.countryid and
composition.compositionid=country.compositionid and
composition.continentid=continent.continentid";

try
  {
      vcc=Connect.getAttribute(Sn,psquerycn,conn,"cncategory");
      vPc=Connect.getAttribute(Sn,psqueryPc,conn,"country");
      vPcp=Connect.getAttribute(Sn,psqueryPcp,conn,"composition_name");
      vPg=Connect.getAttribute(Sn,psqueryPg,conn,"continent_name");

      out.println("Category:");
      out.println(vcc);
%>
<br><br>
<%
       out.println("Geographical Location:");
%>

<dl>
<dd>
<%
      out.println("Country:"+vPc);
%>
<dd>
<%
      out.println("Compositin of Continent:"+vPcp);
%>
<dd>
<%
      out.println("Continent:"+vPg);
%>
</dl>
<%
      vpth1=asLat.getpath(vcc,vl);

curInst=asLat.countcategory(vpth1,AINST.getoutofOldInst(vcc,AINST.getAccinstC()),vl);
      asLat.sort(curInst);
%>
<br><br>
<%
      out.println("Sorted count:");
      for (int i = 0; i < curInst.length; i++)
         {
          out.println(curInst[i]);
         }

      Vector vnbs = new Vector();
      vinsts=asLat.getinterest(vl,curInst);

      if(AINST.getRepeatTimes(vinsts,AINST.getInstC()))
         {
         vnbs=asLat.getRelativeInterest(vl,vinsts);
         vinduce = asLat.getinduce(vnbs,vl);
         }
      else
         {
%>
<br><br>
<%
         out.println("Results of Concept Generalisation of Category: "+vinsts);
         vinduce = asLat.getinduce(vinsts,vl);
         }
%>
<br><br>
<%
         out.println("Results of Concept Specialisation of Category: "+ vinduce);
%>
<br><br>
<%
         out.println("Repeat Times:"+AINST.getRept());

         for (int i = 0; i <vPc.size() ; i++)
            {

            Aloc.AddorCountP(new Position((String)vPc.get(i),0),Aloc.getVcoun(),
                                   AINST.getoutofOldInst(vPc,AINST.getAccinstC()));
            }

         for (int i = 0; i <vPcp.size() ; i++)
            {
            Aloc.AddorCountP(new Position((String)vPcp.get(i),0),Aloc.getVcomp(),
                                  AINST.getoutofOldInst(vPcp,AINST.getAccinstCp()));
            }

        for (int i = 0; i <vPg.size() ; i++)
           {
           Aloc.AddorCountP(new Position((String)vPg.get(i),0),Aloc.getVgeog(),
                                 AINST.getoutofOldInst(vPg,AINST.getAccinstG()));
           }

         countC=asLat.sort(Aloc.countPosition(Aloc.getVcoun()));
         countCp=asLat.sort(Aloc.countPosition(Aloc.getVcomp()));
         countG=asLat.sort(Aloc.countPosition(Aloc.getVgeog()));

         vInstC = Aloc.getMostP(Aloc.getVcoun(),countC);
         vInstCp = Aloc.getMostP(Aloc.getVcomp(),countCp);
         vInstG = Aloc.getMostP(Aloc.getVgeog(),countG);
         vInst = Aloc.getInstofP(vInstC, vInstCp, vInstG);

         out.println("Visiting Times:"+AINST.setvisitNum(AINST.getVisitN()));
        AINST.setinterestC(vinsts);
        AINST.setinterestL(vInst);
        AINST.setAccinterest(AINST.getAccinst());
        AINST.setAccinterestC(AINST.getAccinstC());
        AINST.setAccinterestCp(AINST.getAccinstCp());
        AINST.setAccinterestG(AINST.getAccinstG());

        String psqueryCp = "select distinct  composition_name from composition";
        String psqueryG = "select distinct continent_name from continent";

        vCp=Connect.getCporReg(psqueryCp,conn,"composition_name");
        vReg=Connect.getCporReg(psqueryG,conn,"continent_name");

        String psquery2 = Connect.getAQuery(vInst,vReg,vCp);
%>
<br><br>
<%

        out.println("Results of Concept Specialisation of Geographical Location: ");

        for (int i = 0; i <vInst.size() ; i++)
          {
          out.println(vInst.get(i));
          vres.add(Connect.getIndOfresult(vinduce,conn, psquery2,(String)vInst.get(i),Sn));
          }

        for (int i = 0; i <vres.size() ; i++)
          {
          for (int j = 0; j <((Vector)(vres.get(i))).size(); j++) {
%>
</td>

<tr>
<td>
<ul>
  <a href='Displaysites.jsp?site_name=<%=((String)(((Vector)vres.get(i)).get(j))).replace('
','+')%>'><%=(String)(((Vector)vres.get(i)).get(j))%></a>
</ul>
</td>
</tr>
<%
          }
        }
     }
  catch(Exception e){};
%>
</table>
</td>

<td width="400" valign="top">
<table width="400" border="0" cellpadding="8" cellspacing="0">
<tr valign="top">
```

```
<%
String psqueryD3="select distinct site_briefdesc from site where site_name=?";
PreparedStatement psD3=conn.prepareStatement(psqueryD3);

psD3.setString(1,Sn);
ResultSet myResultSet3 = psD3.executeQuery();

if (myResultSet3 !=null)
   {
   while (myResultSet3.next())
       {
       String desc = myResultSet3.getString("site_briefdesc");
%>

 <td colspan="3"><b>Description:</b><br><%= desc%></td>
 </tr>
<%
       }
   }
conn.close();
%>
                 <tr>
                <td colspan="3"><b>Links with Partner Institutions:</b><br>
                    There's no links available                      </td>
                </tr>
           </table>
          </td>
       </table>
                 <!-- Indhold slut -->

</td>
</tr>
</table>
</center>
</body>
</html>
```

*********DisplayWnsites.jsp**********

```
<html>
<head>
<title>World Heritage Sites Display</title>
<link rel="stylesheet" type="text/css" href="/engine/css/style.css">
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<%@ page language= "java" import="java.sql.*,java.util.*,java.io.*,winduction.*" %>

<body bgcolor="#ccddee" link="#000000" vlink="#000000" alink="#aa4400" leftmargin="0"
topmargin="0" marginwidth="0" marginheight="0">
<%
WCategory NWLat =(WCategory)session.getAttribute("NWLat");
```

```
WAllLocations WNloc =(WAllLocations)session.getAttribute("WNloc");
Winterest AINST =(Winterest)session.getAttribute("AINST");

Connection conn = null;
Vector vl = new Vector();

Vector vcc   = new Vector();/* vector of categories about one site*/
Vector vPc= new Vector();/* vector of country abot one site*/
Vector vPcp= new Vector();/*vector of composition about one site*/
Vector vPg= new Vector(); /*Vector of region about one site*/

Vector vInstC = new Vector(); /* save the interest of positions from the method getMostP*/
Vector vInstCp = new Vector();
Vector vInstG = new Vector();

float [] countC = new float[236];
float [] countCp = new float[36];
float [] countG = new float[6];

Vector vInst = new Vector();/* the interst of Location*/
Vector vCp = new Vector(); /* all the compositions */
Vector vReg = new Vector();/* all the Regions*/
Vector vinduce = new Vector();
Vector vinsts = new Vector();
Vector vres = new Vector();
Vector vlv = new Vector();

vl=NWLat.getWconcepts();

float [] curInst = new float[vl.size()];

try
  {
   userName= "kun";
   password="some_pass";
   url= "jdbc:mysql://130.225.76.177/exwh";
   conn= DriverManager.getConnection (url,userName,password);
  }

catch(Exception e){}
String Sn= request.getParameter("site_name");
%>

<center>
<table width="800" border="0" cellspacing="0" cellpadding="0">
        <tr>
        <td width="800" valign="top"  >
            <table width="800" border="0" cellpadding="8" cellspacing="0">
                <tr>
                <td colspan= "2" valign="top" class="overskrift"><font size=4><b><%=
Sn%></font></td>
                </tr>
                 <tr>
                <td height="1" colspan="3"><hr color="#000000" noshade size="1"></td>
                </tr>
<%
String psqueryD1="select distinct country from country, site, LOCcountry where site_name=?
and country.countryid = LOCcountry.countryid and LOCcountry.siteid =site.siteid";

String psqueryD2="select distinct location_name from site, location where site_name=? and
location.siteid =site.siteid";

PreparedStatement psD1=conn.prepareStatement(psqueryD1);
PreparedStatement psD2=conn.prepareStatement(psqueryD2);

psD1.setString(1,Sn);
psD2.setString(1,Sn);

ResultSet myResultSet1 = psD1.executeQuery();
ResultSet myResultSet2 = psD2.executeQuery();

if (myResultSet1 !=null)
   {
   while (myResultSet1.next())
      {
      String snm = myResultSet1.getString("country");
%>
               <tr>
               <td  class='underOverskrift'><%= snm%></td>
<%
      }
   }
%>
<%
if (myResultSet2 !=null)
   {
   while (myResultSet2.next())
      {
      String ln = myResultSet2.getString("location_name");
%>
               <td colspan="2" align='right' ><%= ln%></td>
<%
      }
   }
%>
               </tr>
               <tr>
               <td height="1" colspan="3"><hr color="#000000" noshade size="1"></td>
               </tr>
             </table>
             </td>

        <tr>
        <table width="800" border="0" cellspacing="2" cellpadding="0">
        <tr>
        <td width="400" valign="top">
```

```
          <table width="400" border="0" cellpadding="8" cellspacing="0">
          <tr>
          <td>
<%
String psquerycn = "select distinct cncategory from cncategory, site where site.site_name = ?
and site.siteid = cncategory.siteid";

String psqueryPc = "select distinct country from country,LOCcountry,site where
site.site_name = ? and site.siteid = LOCcountry.siteid and LOCcountry.countryid =
country.countryid";

String psqueryPcp = "select distinct composition_name from
composition,country,LOCcountry,site where site.site_name = ? and site.siteid =
LOCcountry.siteid and LOCcountry.countryid = country.countryid and
composition.compositionid=country.compositionid";

String psqueryPg= "select distinct continent_name from
continent,composition,country,LOCcountry,site where site.site_name = ? and site.siteid =
LOCcountry.siteid and LOCcountry.countryid = country.countryid and
composition.compositionid=country.compositionid and
composition.continentid=continent.continentid";

try
  {
  vcc=WConnect.getAttribute(Sn,psquerycn,conn,"cncategory");
  vPc=WConnect.getAttribute(Sn,psqueryPc,conn,"country");
  vPcp=WConnect.getAttribute(Sn,psqueryPcp,conn,"composition_name");
  vPg=WConnect.getAttribute(Sn,psqueryPg,conn,"continent_name");

  out.println("Category:");
  out.println(vcc);
%>
<br><br>
<%
  out.println("Geographical Location:");
%>
<dl>
<dd>
<%
  out.println("Country:"+vPc);
%>
<dd>
<%
  out.println("Compositin of Continent:"+vPcp);
%>
<dd>
<%
  out.println("Continent:"+vPg);
%>
</dl>
<%
  vpth1=NWLat.getpath(vcc,vl);
```

```
curInst=NWLat.countcategory(vpth1,AINST.getoutofOldInst(vcc,AINST.getAccinstC()),vl);
   NWLat.sort(curInst);
%>
<br><br>
<%
   out.println("Sorted count:");

   for (int i = 0; i < curInst.length; i++)
      {
      out.println(curInst[i]);
      }

   Vector vnbs = new Vector();
   vinsts=NWLat.getinterest(vl,curInst);

   if(AINST.getRepeatTimes(vinsts,AINST.getInstC()))
      {
      vnbs=NWLat.getRelativeInterest(vl,vinsts);
      vinduce = NWLat.getinduce(vnbs,vl);
      }
   else
      {
%>
<br><br>
<%
     out.println("Results of Concept Generalisation of Category: "+vinsts);
     vinduce = NWLat.getinduce(vinsts,vl);
     }
%>
<br><br>
<%
     out.println("Results of Concept Specialisation of Category: "+ vinduce);
%>
<br><br>
<%
     out.println("Repeat Times:"+AINST.getRept());

     for (int i = 0; i <vPc.size() ; i++)
        {
        WNloc.AddorCountP(new Wposition((String)vPc.get(i),0),WNloc.getVcoun(),
                                   AINST.getoutofOldInst(vPc,AINST.getAccinstC()));
        }

     for (int i = 0; i <vPcp.size() ; i++)
        {
        WNloc.AddorCountP(new Wposition((String)vPcp.get(i),0),WNloc.getVcomp(),
                                     AINST.getoutofOldInst(vPcp,AINST.getAccinstCp()));
        }

        for (int i = 0; i <vPg.size() ; i++)
        {
        WNloc.AddorCountP(new Wposition((String)vPg.get(i),0),WNloc.getVgeog(),
                                        AINST.getoutofOldInst(vPg,AINST.getAccinstG()));
         }

         countC=NWLat.sort(WNloc.countPosition(WNloc.getVcoun()));
         countCp=NWLat.sort(WNloc.countPosition(WNloc.getVcomp()));
         countG=NWLat.sort(WNloc.countPosition(WNloc.getVgeog()));

         vInstC = WNloc.getMostP(WNloc.getVcoun(),countC);
         vInstCp = WNloc.getMostP(WNloc.getVcomp(),countCp);
         vInstG = WNloc.getMostP(WNloc.getVgeog(),countG);
         vInst = WNloc.getInstofP(vInstC, vInstCp, vInstG);

         out.println("Visiting Times:"+AINST.setvisitNum(AINST.getVisitN()));

         AINST.setinterestC(vinsts);
         AINST.setinterestL(vInst);
         AINST.setAccinterest(AINST.getAccinst());
         AINST.setAccinterestC(AINST.getAccinstC());
         AINST.setAccinterestCp(AINST.getAccinstCp());
         AINST.setAccinterestG(AINST.getAccinstG());

         String psqueryCp = "select distinct  composition_name from composition";
         String psqueryG = "select distinct continent_name from geography";

         vCp=WConnect.getCporReg(psqueryCp,conn,"composition_name");
         vReg=WConnect.getCporReg(psqueryG,conn,"continent_name");

         String psquery2 = WConnect.getAQuery(vInst,vReg,vCp);
%>
<br><br>
<%

         out.println("Results of Concept Specialisation of Geographical Location: ");
         for (int i = 0; i <vInst.size() ; i++)
           {
           out.println(vInst.get(i));
           vres.add(WConnect.getIndOfresult(vinduce,conn, psquery2,(String)vInst.get(i),Sn));
           }

         for (int i = 0; i <vres.size() ; i++)
           {
            for (int j = 0; j <((Vector)(vres.get(i))).size(); j++)
              {
%>
</td>
<tr>
<td>
<ul>
<a href='DisplaySites.jsp?site_name=<%=((String)(((Vector)vres.get(i)).get(j))).replace('
','+')%>'><%=(String)(((Vector)vres.get(i)).get(j))%></a>
</ul>
</td>
</tr>
```

```
<%
                }
            }
        }
    catch(Exception e){};
%>

</table>
</td>
<td width="400" valign="top">
<table width="400" border="0" cellpadding="8" cellspacing="0">
<tr valign="top">

<%
String psqueryD3="select distinct site_briefdesc from site where site_name=?";
PreparedStatement psD3=conn.prepareStatement(psqueryD3);

psD3.setString(1,Sn);
ResultSet myResultSet3 = psD3.executeQuery();

if (myResultSet3 !=null)
   {
   while (myResultSet3.next())
      {
      String desc = myResultSet3.getString("site_briefdesc");
%>
 <td colspan="3"><b>Description:</b><br><%= desc%></td>
 </tr>
<%
      }
   }
conn.close();
%>
                 <tr>
                <td colspan="3"><b>Links with Partner Institutions:</b><br>
                     There's no links available                      </td>
                 </tr>

                 </table>
                </td>
            </table>
            <!-- Indhold slut -->
</td>
</tr>
</table>
</center>
</body>
</html>
```