ON-TO-METHODOLOGY

Ontology Development Methodology

A Dissertation submitted in partial fulfillment of the requirement for the award of degree of

MASTER OF TECHNOLOGY

IN

SOFTWARE ENGINEERING

By

MAGENDRA SINGH

College Roll No. - 08/SE/2010

Under the esteemed guidance of

Dr. DAYA GUPTA

Department of Computer Engineering

Delhi Technological University

2011-2012

CERTIFICATE

DELHI TECHNOLOGICAL UNIVERSITY

BAWANA ROAD, DELHI – 110042

Date:_____

This is to certify that dissertation entitled "**On-to-Methodology: Ontology Development Methodology**" has been completed by Magendra Singh in partial fulfillment of the requirement of major project of **Master of Technology** in **Software Engineering**.

This is a record of his work carried out by him under my supervision and support during the academic session 2011 -2012.

Dr. DAYA GUPTA

Prof., HOD & PROJECT GUIDE

(Dept. of Computer Engineering)

DELHI TECHNOLOGICAL UNIVERSITY

BAWANA ROAD, DELHI – 110042

ACKNOWLEDGEMENT

It is distinct pleasure to express my deep sense of gratitude and indebtedness to my learned supervisor **Dr. Daya Gupta, HOD, Department of Computer Engineering, Delhi Technological University**, for her invaluable guidance, encouragement and patient reviews. Her continuous inspiration only has made me complete this dissertation. She kept on boosting me time to time for putting an extra ounce of effort to realize this work.

I would also like to take this opportunity to present my sincere regards to my teacher Dr. Ruchika Malhotra, for encouragement and perpetual motivation and other faculty members of Computer Engineering Department for providing me unconditional and anytime access to the resources and guidance. I would like to thank my friends for their unconditional support and motivation during this work.

Finally, I would thank my parents who bless me & always have been there for me.

(MAGENDRA SINGH)
Master of Technology
(Software Engineering)
Dept. of Computer Engineering
DELHI TECHNOLOGICAL UNIVERSITY
BAWANA ROAD, DELHI – 110042

ABSTRACT

For about a decade, ontologies have been known in computer science as explicit specifications of shared conceptualizations. Researchers have written much about the potential benefits of using them, and most of us regard ontologies as central building blocks of the Semantic Web and other semantic systems. There is much work already existent on their definitions, construction and development and their applications. All these literature define the set of activities that concern the ontology development process, the ontology lifecycle, the principles, methods and methodologies for building ontologies and the tool suites and languages that support them.

The construction of ontology can allow users or agents of software/service to arrive at consistent views about organization structure of information with same semantics. However, since domains differ in principles, theories and techniques underlying them, there is no existing methodology that could work as the standard method for ontology construction at the present time.

Unfortunately, still not much quality ontologies have been developed. This implies that the Semantic Web community has yet to build practically useful ontologies for a lot of relevant domains in order to make the Semantic Web a reality. Indeed, several social and technical issues exist that cause problems in development of ontologies.

In this work we provide an overview of what ontology is, describing the current trends, issues and problems in constructing them. We also propose an ontology development methodology *On-to-Methodology* that could be used as a standard model for ontology development tasks across various domains. We have automated the development process by implementing a tool for Ontology design process. We illustrate our methodology by developing Ontology of Bikes. We then compare our methodology with other existing ontology development methodologies.

Table of Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

The American Heritage Dictionary defines "ontology" as "the branch of metaphysics that deals with the nature of being." Ontology, as a formal specification of a shared conceptualization [3], defines the thesaurus of the concept, relations between concepts and properties even relations between properties. Moreover, it describes axioms and individuals and relations between them, and provides sharing knowledge [4]. More specifically, it is an explicit specification of a conceptualization [5, 10].

The origin of ontologies in computer science can be referred back to 1991, in the context of the DARPA Knowledge Sharing Effort (Neches, Fikes, Finin, Gruber, Senator, & Swartout, 1991). The aim of this project was to devise new ways of constructing knowledge-based systems, so that the knowledge bases upon which they are based did not have to be constructed from scratch, but by assembling reusable components. This reuse applies both to static knowledge, which is modeled by means of ontologies, and dynamic problem-solving knowledge, which is modeled by means of problem solving methods [1].

Initial application of ontologies in computer science was that it acted as a means of providing the semantics for the semantic web i.e. web 2.0. By use of ontologies, the information retrieval could then be done based on the context (/meaning) rather than just simple string matching mechanisms.

In this work we provide an overview of what ontology is, describing the current trends, issues and problems in constructing them. We also propose an ontology development methodology *On-to-Methodology* that could be used as a standard model for ontology development tasks across various domains. We have automated the development process by implementing a tool for Ontology design process. We illustrate our methodology by

developing Ontology of Bikes. We then compare our methodology with other existing ontology development methodologies.

1.1 Motivation

The Ontologies created by different creators are different and inconsistent. There is a huge difference between different domains and currently there is no defined methodology for constructing Ontologies [12, 14, 19]. And inspite much being written in literature about Ontologies, the number and quality of actual, "non-toy" ontologies available on the Web today is remarkably low [19,20].

The development of ontologies generally consists of four main tasks:

- *Scope Definition*
  This activity involves establishment of scope of the ontology in terms of purpose that the ontology would serve. This is accompanied by the advantages that the ontology would provide after it has been developed.

- *Knowledge Acquisition*
  The target of this activity is to acquire the knowledge required for successful development of the ontology. This activity specifies the knowledge sources from where the knowledge would be compiled.

- *Design*
  The concepts identified during *Knowledge Acquisition* activity integrated to form an Ontology. The design specifies the architecture of the ontology and acts as an important step for successful formalization of the ontology.

- *Formalization*
  This activity deals with formalization of the ontology and represents it in a form so that it is understandable by the machines. The ontology is formalized using ontology languages and tools.

The development of ontologies is still a matter of craft skill rather than an understood engineering process. For the successful utilization of the potential of ontologies, it is required to formally define this practice.

1.2 Related Work

Fernández et. al. proposed "Methontology" and defined the ontology development process and the ontology life cycle [21]. Farooq et. al. proposed "Design of Ontology in Semantic Web Engineering Process and illustrated their method with the help on a case study [24]. Gaoyun et. al. identified the reasons for the inconsistencies in ontology construction and put forward a method for Consistent Ocean Ontology Construction [19]. Zhang et. al. proposed a methodology namely MADRE and illustrated their work by focusing on healthy housing [25]. Jia et. al. put forward "Automatic ontology construction approaches and its application on military intelligence" [27]. Chen et. al proposed Concept feature based ontology construction and maintenance which was backed up by a detailed framework [15]. A structured ontology construction by using data clustering and pattern tree mining was proposed by Yao-Tang et. al. [14].

These proposals seem to suffer from following drawbacks [19]

1. *Inconsistency of domain cognition*
   This kind inconsistency could be classified to two cases. One is about scope of domain that is people have no consistent definition to the scope of domain. The other is about contents of domain that is which contents belong to the domain people have their own opinions.

2. *Inconsistency of viewpoint*
   In many cases constructions which are restricted to the same scope and are built by domain experts with the same background knowledge still have essential dissimilarities. The reason is viewpoint. As we know one matter may

get contrary conclusions from different perspectives, so does ontology construction. Different viewpoints could also lead to inconsistent ontologies.

3. *Ontology construction is affected by subjective factors*

   From the definitions discussed in Section 1.1 above, it is clearly that ontology is category of subjective field, therefore ontology construction would be influenced by subjective factors heavily. Even in the same field with same viewpoint ontologies may be constructed various from one another due to creators' interests, tastes, preferences, ideas, abilities etc.

4. *Lexical inconsistency*

   Conceptualization is independent from vocabulary, but ontology is language-related. That is ontology should be expressed by a special formal language. Then there is another kind inconsistency that creators from different nationalities use their own language to build ontologies. Even the language is unified, the situation is still not improved. That is because all the languages existed cannot avoid the following two problems: "polysemy" and "thesaurus".

Martin Hepp [20] classifies the ontology-related tasks into two main groups: building or contributing to the development of ontologies and committing to a particular ontology. Committing to a given ontology, explicitly or implicitly, means agreeing that it properly represents the domain's conceptual elements.

So to solve the aforesaid problems with ontology engineering, we propose a methodology that is generic and comprehensive and narrows down the gap between constructing and committing to the ontologies.

1.3 Problem of the thesis

In this work we focus on proposing a method that is comprehensive, generic and makes it easy to commit and construct ontologies of various domains specifically transportation like bikes, cars, planes etc. We aim to produce documents similar to the documents in Software Engineering like Software Requirement Specification document known as IEEE std. 830. These documents will help the ontology to evolve. Also we wish to provide computer based support for these activities. The proposed methodology will have following features:

- *Better domain cognition*

  As listed in Section 1.3, there exists inconsistency in domain cognition. Our method handles this problem by adapting group oriented activities thus enabling experts and users to define the domain in better terms.
- *Unification of viewpoints*

  Inconsistency in viewpoints arises as different people have different perspectives and due to this contrary conclusions may be obtained. Our methodology resolves this problem by instrumenting the development process with evaluation activities after each phase of the development process. Also, the unification is achieved due to the adaptation of group oriented development activities.

- *Exhaustive documentation*

  On-To-Methodology provides an extensive and exhaustive documentation support thus making the development process more formal and helps in achieving traceability across phases. Documentation also simplifies the maintenance of the ontology by providing the backward traceability of various components of the ontology.

- *Extensive Verification & Validation*

  These two tasks of verification and validation are interleaved between various phases of the development process. This prevents introduction of bugs into the system right from the beginning of the development process and prevents faults till the maintenance phase.

So problem of the thesis is:

> **"Develop a comprehensive ontology methodology that is generic and construct a tool which can automate the construction process."**

1.4 Scope of the thesis

Our ontology construction methodology will have the following steps as shown in figure :

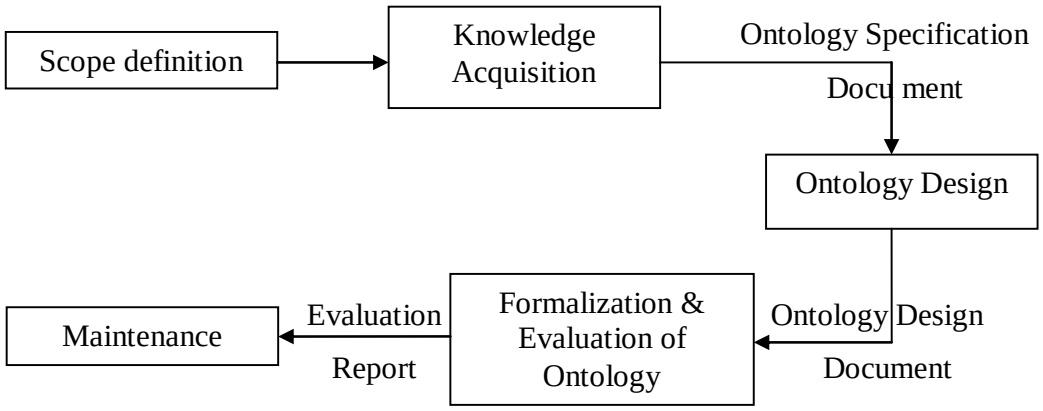


Figure 1: Steps involved in On-To-Methodology

Knowledge elicitation and representation plays an important role construction of Ontology. It is done through a formal group oriented technique based on FAST (Facilitated Application Specification Technique). We have designed set of forms to carry out these activities to formalize knowledge elicitation and then conceptual model of

the knowledge is built which is the base of ontology construction that will narrow down the gap between ontology building and commitment. Output of this activity is the Ontology Specification Document (OSD) which is similar to the Software Specification document in Software Engineering. Hence our knowledge is elicitated & represented in a formal manner.

For automating the ontology design process we present an algorithmic approach that will generate the ontology structure from the conceptual model of the knowledge. In this way we will automate the design process. This tool will be available online so that ontology engineers can access it from anywhere and anytime.

We shall apply On-To-Methodology to domain of Automobile, specifically Bikes. The intended ontology of Bikes will be representation of bikes in the domain of ontology. The purpose of this ontology is to provide information on bikes based on the criteria specified by the users.

- The ontology would hold information to answer queries of customers based on single (/combination of) parameter(s) which are Make, Engine capacity, Power, Price, Fuel tank capacity, Mileage, Brake type, Weight, Wheel type, Ignition and Number of gears.
- Bike manufacturing organizations can use this ontology to identify the bike configurations that are suitable for a particular market and can also use it to analyze current sales and make future predictions. This will guide them to plan their production & inventory.
- This ontology can prove to be beneficial for bike retailers as they can use it to plan their inventory and analyze their sales.

The advantage that this ontology would provide is its capability to answer the queries of the customers across a large information base of different bikes, based on multiple search criteria with complex inter-relations.

Broadly, the scope of this work can be summarized as:

- Develop a detailed process for knowledge elicitation to build the conceptual model with comprehensive documentation support. The output is the Ontology Specification Document (OSD).
- Develop an algorithm that will automate the ontology design activity.
- Develop an ontology of Bikes using the proposed methodology.
- Evaluate our methodology with other existing methodology including Methontology, Methodology by Farooq et. al., Methodology by Gaoyun et. al., MADRE, approach by Jia et. al, Concept feature based Ontology Construction and Maintenance and Structured Ontology Construction by using data clustering & Maintenance.
- Develop a tool that will support our methodology.

1.5 Thesis Statement and Outline

This aim of this dissertation is to propose a well defined methodology for construction of ontology.

The rest of the thesis is organized as follows:

In chapter 2, we review the concept of methodology for creation of ontologies and also describe the current state of the art in ontology engineering.

Chapter 3 provides the proposed methodology for ontology construction.

In chapter 4, we illustrate our methodology by constructing an Ontology of Bikes.

In chapter 5, we evaluate the Ontology of Bikes for various competency questions and compare our approach with other existent methodologies.

Chapter 6 concludes the thesis and proposes future work.

Chapter 7 gives the list of references that I have gone through during my research.

Chapter 2

ONTOLOGY DEVELOPMENT METHODOLOGIES

Here we analyze the state of the art of various existent methodologies that can be used for ontology construction process.

2.1 METHONTOLOGY

METHONTOLOGY [21] is a methodology that is based on software engineering and knowledge engineering. It is structured methodology used to build ontologies from scratch. It also gives a life cycle to build ontologies based on evolving prototypes. METHONTOLOGY is divided in to management activity, development activity. These activities are integrated to work together. The lifecycle model of METHONTOLOGY is waterfall but uses prototyping strategy. METHONTOLOGY consists of the following tasks:

2.1.1 Specification

The goal of the specification phase is to produce either an informal, semi-formal or formal ontology specification document written in natural language, using a set of intermediate representations or using competency questions, respectively.

2.1.2 Knowledge Acquisition

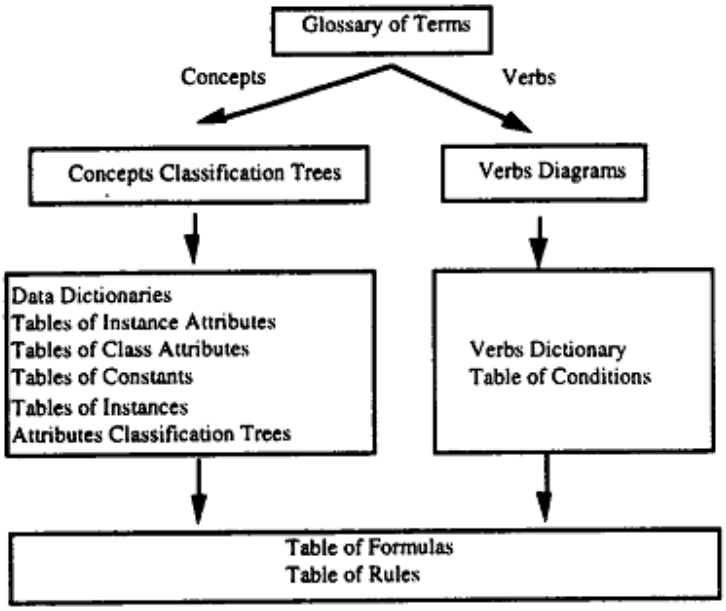
It is coincident with other activities. Most of the acquisition is done simultaneously with the requirements specification phase, and decreases as the ontology development process moves forward. Experts, books, handbooks, figures, tables and even other ontologies are sources of knowledge from which the knowledge can be elucidated using in conjunction techniques such as: brainstorming, interviews, formal and informal analysis of texts and knowledge acquisition tools.

2.1.3 Conceptualization

In this activity, you will structure the domain knowledge in a conceptual model that describes the problem and its solution in terms of the domain vocabulary identified in the ontology specification activity. The first thing to do is to build a complete Glossary of Terms (GT). Terms include concepts, instances, verbs and properties. So, the GT identifies and gathers all the useful and potentially usable domain knowledge and its meanings. Note that you do not start from scratch when you develop your GT.

For each set of related concepts and related verbs, a concepts classification tree and a verbs diagram is built. After they have been built, you can split your ontology development process into different, but related, teams. Figure 2 graphically summarizes the intermediate representations used in the conceptualization phase.

Fig. 2: Set of Intermediate Representations in the conceptualization phase.



2.1.4 Integration

With the goal of speeding up the construction of your ontology, you might consider reuse of definitions already built into other ontologies instead of starting from scratch.

2.1.5 Implementation

Ontologies implementation requires the use of an environment that supports the meta-ontology and ontologies selected at the integration phase. The result of this phase is the ontology codified in a formal language such as: CLASSIC, BACK, LOOM, Ontolingua, Prolog, C++ or in your favorite language.

2.1.6 Evaluation

Evaluation means to carry out a technical judgment of the ontologies, their software environment and documentation with respect to a frame of reference (here it the requirements specification document) during each phase and between phases of their life cycle. Evaluation subsumes the terms Verification and Validation.

The output proposed by METHONTOLOGY for this activity is many evaluation document in which the ontologist will describe how the ontology has beenevaluated, the techniques used, the kind of errors found in each activity, and the sources of knowledge used in the evaluation.

2.1.7 Documentation

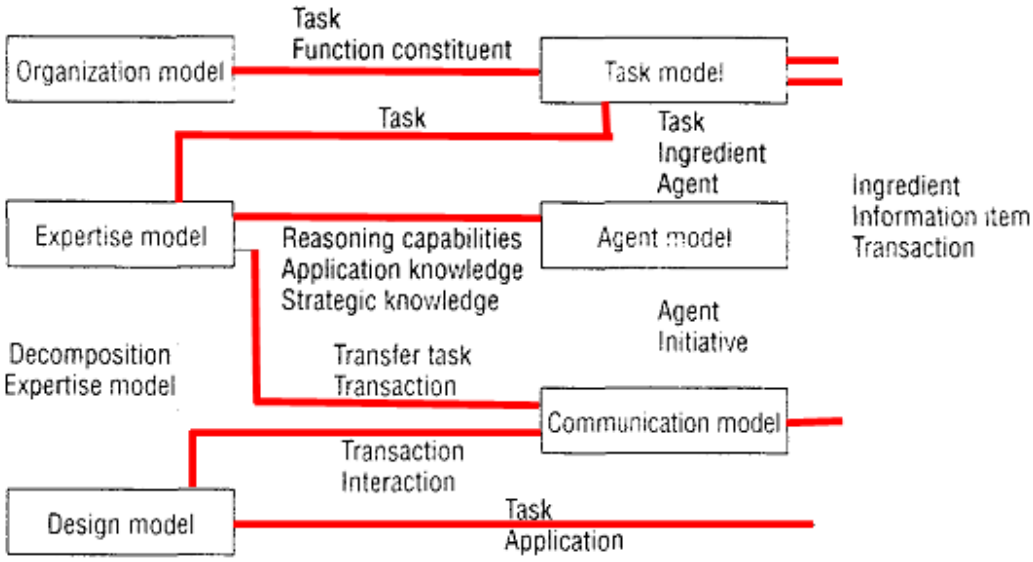
There are no standard guidelines on how to document ontologies. In many cases, the only documentation available is in the code of the ontology, the natural language text attached to formal definitions, and papers published.

METHONTOLOGY pretends to break this circle including the documentation as an activity to be done during the whole ontology development process. In fact, after the specification phase, you get a requirements specification document; after the knowledge acquisition phase, a knowledge acquisition document; after the conceptualization, a conceptual model document that includes a set of intermediate representations that describe the application domain; after the formalization, a formalization document; after the integration, an integration document; after the implementation, the implementation document; and during the evaluation, an evaluation document.

2.2 CommonKADS

CommonKADS [22] is widely used ontology development methodology. CommonKADS is focused on the work product-based rather than process-oriented. It supports every aspect of knowledge engineering including project management, group analysis, knowledge acquisition, conceptual modeling, user interface, system integration and architecture [23]. Fig. 3 summarizes the suite of models involved in a ComrnonKADS project. A central model in the CommonKADS methodology is the expertise model, which models the problem solving behavior of an agent in terms of the knowledge that is applied to perform a certain task.

Fig. 3: The CommonKADS suite of models



The CommonKADS model set provides four models that are specifically geared to modeling the organizational environment of a KBS: the organization, task, agent and communication models.

The *organization model* supports the analysis of the major features of an organization to discover problems and opportunities for KBS development, as well as possible effects a KBS could have when fielded. A template that defines object and relation types is associated with each model in the model set.

The *task model* describes, at a general level, the tasks that are performed or will be performed in the organization where the expert system will be installed. The tasks it covers are those that help realize an organizational function. The task model is represented as a hierarchy of tasks. In addition, aspects like inputs and outputs of tasks, task features, and task requirements can be modeled. The task model also specifies the distribution of tasks over agents.

An *agent* is an executor of a task. It can be human, computer software, or any other "entity" capable of executing a task. In the *agent model*, the capabilities of each agent are described. The model can also be used to represent constraints on an agent, such as norms, preferences, and permissions that apply to the agent.

Because several agents are usually involved in a task, it is important to model the communication between agents. This is the purpose of the CommonKADS *communication model*. The transactions here are modeled at a level that is still independent of a computational realization.

2.3 Methodology by Farooq et. al.

In their paper [24], they proposed a technique for ontology design during SW application engineering process. By incorporating their technique, existing web applications design methods may easily be upgraded for semantic web (SW) applications. They made some deliberate efforts in design phase of ontology development which they found missing in other methods.

2.3.1 Adaptation at Specification Level

For ontology, requirements should be specified accordingly, in specification phase. A preliminary web-ontology model should be prepared at specification level. The activities involved, are as below:

a) Domain Vocabulary Declaration
b) Identifying resources and assigning them to proper groups
c) Identifying Axioms
d) Identifying relationships and assigning them proper names
e) Identifying data-characteristics and assigning them, proper names
f) Applying constraints
g) Verification

2.3.2 Adaptation at Design Level

The processing of design phase mainly uses the report generated by specification phase, and transforms it into some algorithmic or pseudo form so that it can be coded easily in any computer language in order to make it executable. Since ontology (schema and document) is based on Resource Description Framework model, therefore they designed a model, so-called RDF model, from preliminary ontology model generated in previous phase. This model consists of triples. A triple contains three components: (i) subject (ii) predicate and (iii) object. Each name in RDF model is a URI reference or a literal.

2.4 Methodology by Gaoyun et. al.

In their paper [19], they firstly analyzed four reasons for occurrences of inconsistencies in ontologies. Then based on the reasons they propose a methodology for ontology construction to minimize the distinctions as follows:

2.4.1 Clarifying domain

In order to clarify the domain, following three problems should be solved: (1) the scope the domain covers, (2) the contents the domain describe and (3) the viewpoint of the domain uses. All the answers of the problems depend on the users of ontology. Consequently the following questions should be considered as we are users: Who would be the users of ontology? What can we do by this ontology? What problems can we solve by this ontology?

2.4.2 Building domain model

After clarifying domain, the domain model construction is separated into two phases: building physical model of domain and building concept model of domain to minimize inconsistencies brought by subjective factors.

2.4.2.1 Building physical model of domain

As above analysis ontology construction highly depends on subjective factors, which is a major reason why ontologies built have such inconsistencies from one another. To construct consistent ontologies, the influences affected by subjective factors should be reduced and the process of concepts extracting should be suspended. To model the domain we should begin with objective description of real world, describe objects and relations between them and then build physical model of domain. This less abstract physical model is easier to reach agreement among creators, and then abstract it unceasingly to higher concept model.

2.4.2.2 Building concept model of domain

What's more based on a widely recognized physical model, in concept model if there are inconsistencies which are caused by the degree of abstraction, we just need to unify the inconsistencies by identifying a proper granularity for concept. Attribute of concept can aid this identifying process which would be introduced in next subsection.

2.4.3 Identifying attributes

By attribute two concepts are contacted with each other, and thanks to attribute the simple concept tree which has inherited relations only could be converted to concept web which has abundant relations and could describe complex relations of the world more exactly.

2.4.4 Layered vocabulary

Another problem that may lead to inconsistencies is caused by the case that several terms aim at one meaning. To unify the terms used in the ontology, requiring creators to use the controlled vocabulary is a good choice. But it is not the same good to compel users to use words in the controlled vocabulary either. For this reason the vocabulary is divided into three grades: controlled vocabulary, domain special vocabulary and common vocabulary. The terms in controlled vocabulary which is used in ontology construction originate from various metadata of domain to guarantee consistency of terms using. The terms in domain special vocabulary and common vocabulary come from daily words used by experts and ordinary people respectively to ensure richness of words that can be used.

2.4.5 Merging ontology

Reusing existed ontologies could reduce the work of construction and improve efficiency. Even developing ontology by one community independently, ontology division is also a good choice. These two cases are both involved in ontology merging finally. This process would bring inconsistencies if the chosen ontologies are inappropriate.

Therefore, when choosing ontologies the following two points should be ensured. Firstly ensure the viewpoints of ontologies should be the same. Secondly, part concepts of the one chosen ontology should be equivalent to or subsumption of some concepts of the other one that is parts of ontologies should be compatible without conflictions. Moreover for ontology reuse it is crucial to transform terms with controlled vocabulary.

2.4.6 Ontology formalization

The fundamental purpose of ontology is to aid machine comprehending the meaning of information which is coded by ontology language. OWL has been accepted as the ontology description language by the authors.

2.5 MADRE

The authors, Zhang et. al., realized that because of the differences between domains, there was no existing technical route as the standard method for ontology construction at the present time. The paper [25] offers an improved ontology construction method MADRE (Method of Analysis, Design, Representation and Evaluation) based on IDEF5 and seven steps methodologies. This method uses graphic language to explicit represent domain knowledge for research domain. Moreover, it verifies correctness of ontology construction on relation and hierarchy in the phase of ontology evaluation.

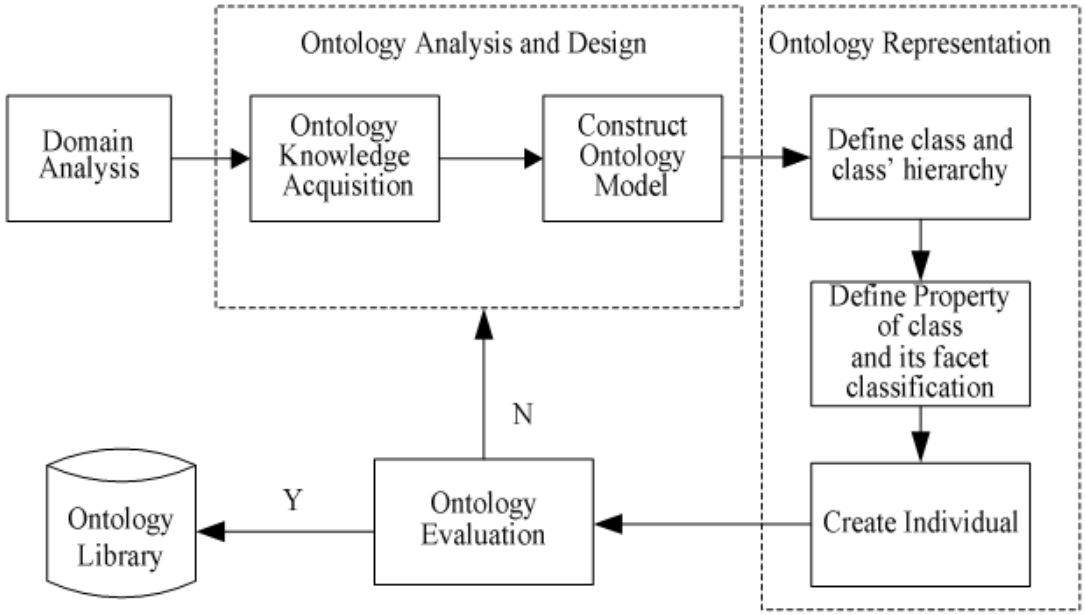
Following the Seven Steps method, the construction is divided into four steps, which are domain analysis, ontology analysis and design, ontology representation and ontology evaluation. Meanwhile, in the phase of ontology analysis and design, MADRE adopts expression of the graphic language of IDEF5. The phase of ontology evaluation in MADRE is able to make an accurate judgment of created ontology. The specific flowchart to construct ontology with MADRE is as follows:

- Use Phase of Domain Analysis: determine the professional field extension and reusability of domain ontology.
- Phase of Ontology Analysis and Design: acquires semantic information about core concept, relations, actions and so forth, define the hierarchy of concepts and relations between different activities, formally translates the professional

knowledge and raw data of the current domain to commonly used information based on ontology, and therefore establishes the ontology model.

- Phase of Ontology Representation: according to the ontology model, detailed specific the class and properties of the domain ontology, and creates individual. Meanwhile, represents the domain ontology with structural language-OWL.
- Phase of Ontology Evaluation: establishes rule set, evaluates the accurateness and coincidence of ontology by the First-Order Logic inference. If it is consistent with the domain knowledge, the construction of ontology succeeds; otherwise, go back to Phase of Ontology Analysis and Design.

Fig.4: The process of MADRE ontology construction

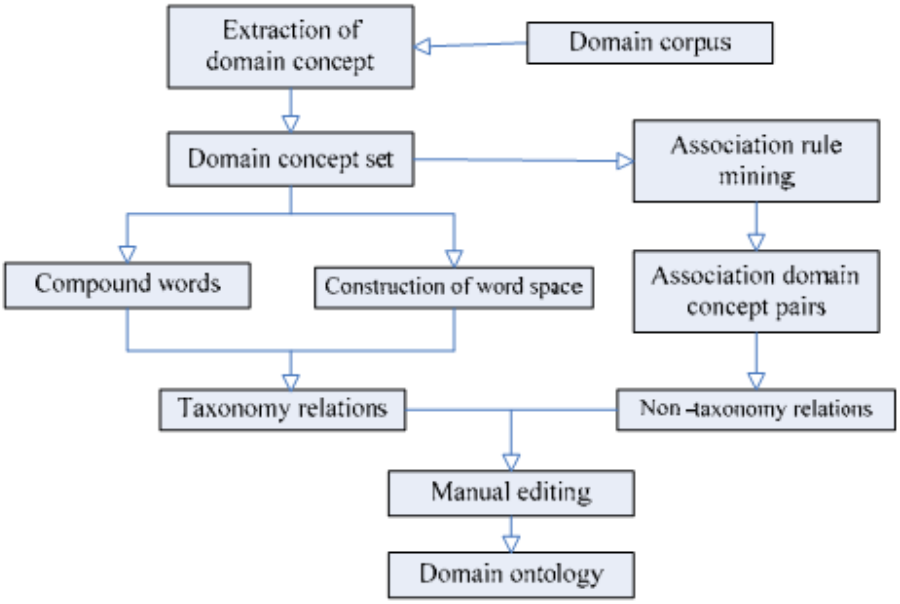


2.6 Semi-automatic Domain Ontology Construction methodology by Dan et. al.

In their paper [26], an approach to semi-automatically constructing domain ontology based on Chinese word partition and data mining is proposed.

The semi-automatic domain ontology system designed in this paper mainly consists of three parts: the extraction module of domain concepts and the extraction module of taxonomy and non-taxonomy relations among domain concepts. Statistical analysis method, generalized suffix tree and clustering method and association rule mining method are respectively adopted in the above three parts. The system framework is illustrated in Figure 5.

Fig.5: The system framework



2.7 Automatic Ontology Construction Approach by Jia et. al

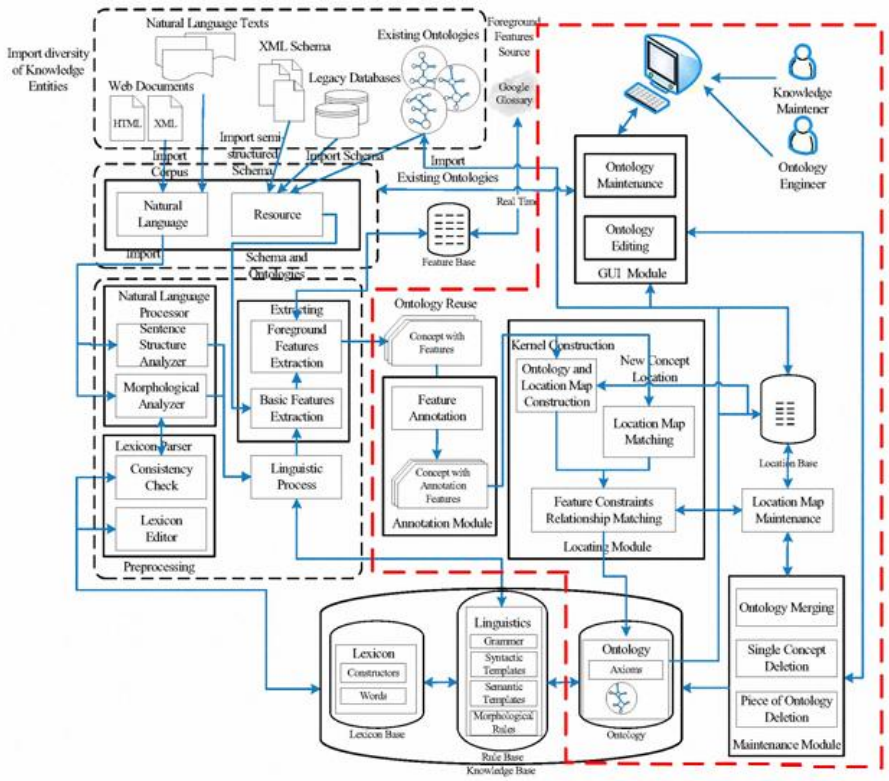
In this paper, thesaurus and oracle database are selected as the existing resources in the military intelligence. The thesaurus is relatively complete and accurate knowledge resource, and it contains 61895 terms and 3 kinds of relationships between concepts. The classes of the ontology mainly are obtained from thesaurus, while the database provides the instances for the ontology. Protégé is used as the development tool and OWL as ontology language.

The authors combine the characteristic thesaurus. First, they select the standardized thematic terminology as the classes of ontology and give up the non-thematic terminology; second, semantic relations should be adjusted in order to obtain accurate semantic relations, which is the core job needed to be done in the thesaurus. In order to resolve this problem, all of the concepts in the thesaurus are classified by the semantic categories, and then the relations will be converted based on the concepts of each category. At the same time, considering the characteristic of relational database, the authors propose a series of rules to achieve the target that append the instances into the ontology automatically. Finally, following some certain steps as they show4, the ontology is constructed completely.

2.8 Concept Feature-based Ontology Construction and Maintenance

The study [15] develops a concept feature-based mechanism for constructing and maintaining ontology. In addition to assisting enterprise knowledge engineers in constructing and classifying knowledge more precisely to increase knowledge maintenance efficiency, the proposed mechanism assists knowledge users in accurately searching for required knowledge based on use of concept features. They designed a concept feature-based ontology construction and maintenance framework, developed techniques related to the concept feature-based ontology construction and maintenance, and implemented a concept feature-based ontology construction and maintenance mechanism. The proposed framework includes six modules of import, preprocessing, annotation, locating, maintenance, and graphic user interface (GUI), as shown in figure 6.

Fig.6: Concept feature-based ontology construction and maintenance framework.



2.9 A structured ontology construction by using data clustering and pattern tree mining

The paper [14] proposes an automatic system of ontology construction to link the relations between concepts. The system contains two major functions: document clustering and ontology construction. The former mainly groups related documents by clustering method. The latter discovers inter-concept relations from each group of related documents and integrates all the generated conceptual relations in light of sequence patterns to construct ontology. The system is hence suitable for domains with a larger scope. It can construct an ontology which is able to describe inter-concept relations in more detail as well as more finely and reasonably structured.

2.9.1 Ontology construction

Traditionally, ontology construction usually uses the following methods for ontology construction: *relational analysis*, *clustering*, and *formal concept analysis (FCA)*.

Relational analysis discovers and clusters the relations of keywords, such as synonyms, roots, hypernyms, and hyponyms. It then constructs the ontology by manmade or other methods. *Clustering* usually groups keywords of documents into clusters and constructs ontology by selecting representative concepts from each cluster. Formal concept analysis uses the binary relation matrix between documents and vocabulary to generate the supremum concept set. The inter-conceptual hierarchical relation and a complete partial sort is formed by the sets of all the concepts. After constructing the inter-concept level relation by means of *FCA* construction, the concept figure of ontology is thus constructed.

2.9.2 System Architecture

The architecture of the structured ontology construction system contains two modules: *document clustering* and *ontology construction*. Basically, document clustering is responsible for document vector computation and document clustering. Ontology construction establishes the concept tree of the clustered documents and integrates them into complete document ontology.

Figure 7 gives the document vector algorithm. Figure 8 gives the concept tree construction algorithm and Figure 9 gives the sequence pattern mining algorithm used in ontology construction. Finally figure 10 represents the ontology construction algorithm.

Fig.7: Document vector algorithm

1. Establish the matrix of latent semantic singular value decomposition: $R \circ S' \circ M^T = A$
2. Set up the dimension of document importance and establish the dimension-reduction matrix (S) of new documents and keywords in the latent semantic space.
3. Calculate the feature vector of documents: $F=R_{\circ}\ S$.

Fig.8: Concept tree construction algorithm

1. Establish the two-dimensional matrix of documents and keywords for each group of documents. The documents to which Keyword A corresponded were more than those to which Keyword B corresponded, and no other keywords corresponding to documents existed between A and B, so a link was established between A and B, the result of which constructed a concept lattice.

2. Delete the empty concept nodes which were not root nodes in the concept lattice in order to form concept trees.

Fig.9: Sequence pattern mining algorithm

1. Take all the root-to-leave path patterns of the concept trees and sequence the concepts in all the patterns to generate candidate concept sets.
2. In the candidate concept sets, all the single concepts construct a single concept set. In single concept sets, find out the set which is greater than or equal to the minimum support, namely the highly frequent single-concept set. Repeat the steps to discover the highly frequent two-concept set, the highly frequent three-concept set, the highly frequent four-concept set, etc. until no highly frequent concept set can be discovered, so that the referentially valuable sequence pattern structure will be found.
3. Delete sub-sequences from the mined longest sequence pattern. For example, the longest 4-concept set is <1,2,3,4>. If the 3-concept set includes <1,2,3> and <1,2,4>, delete them, so the rest is the sequence pattern set.

Fig.10: Ontology construction algorithm

1. Select the longest pattern structure as the ontology skeleton and add the mined patterns to the structure tree according to the pattern length and occurrence frequency.
2. Through WordNet, combine with the Jaccard similarity (Sim) to obtain the distance (Dis) between root nodes. That is, the more similar they are, the shorter the distance is. The closest root nodes are adopted to generate common hypernyms; the common hypernyms of the keywords of the root nodes in the ontology skeleton are established, and the new root nodes of hypernyms are created.

$$Sim(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

$$Dis(A,B) = 1 - Sim(A,B)$$

   where A and B are two different root nodes, and S is between 0 and 1. The greater the value is, the more similar A and B are.
3. Repeat Step 2 until all the sub-trees in all the ontology skeletons find their common hypernyms.
4. Link the hypernyms with the structure trees to form the ontology structures.

Chapter 3

ON-TO-METHODOLOGY:
PROPOSED METHODOLOGY FOR ONTOLOGY CONSTRUCTION

In this chapter we shall describe our process of methodology. This process is generic and extension of proposed methodology by Fernández et. al. [21]. We create various documwnts at the end of each activity in the proposed methodology. These documents are analogous to various documents in software engineering. We call this methodology On-To-Methodology.

3.1 Process Description of Ontology Methodology

The different phases in On-To-Methodology are described in Figure 11. These steps are generic and consist of four main activities namely Knowledge Acquisition, Design Ontology, Formalization and Evaluation. The output of Knowledge Acquisition is Ontology Specification document. This document is analogous to Software Requirements Specification (SRS) document IEEE standard 830. The output of Design Ontology is Ontology Design Document (ODD) and consists of logical design of the ontology being developed. The output of Evaluation is Evaluation Report. It consists of the evaluation of ontology for knowledge representation efficiency.

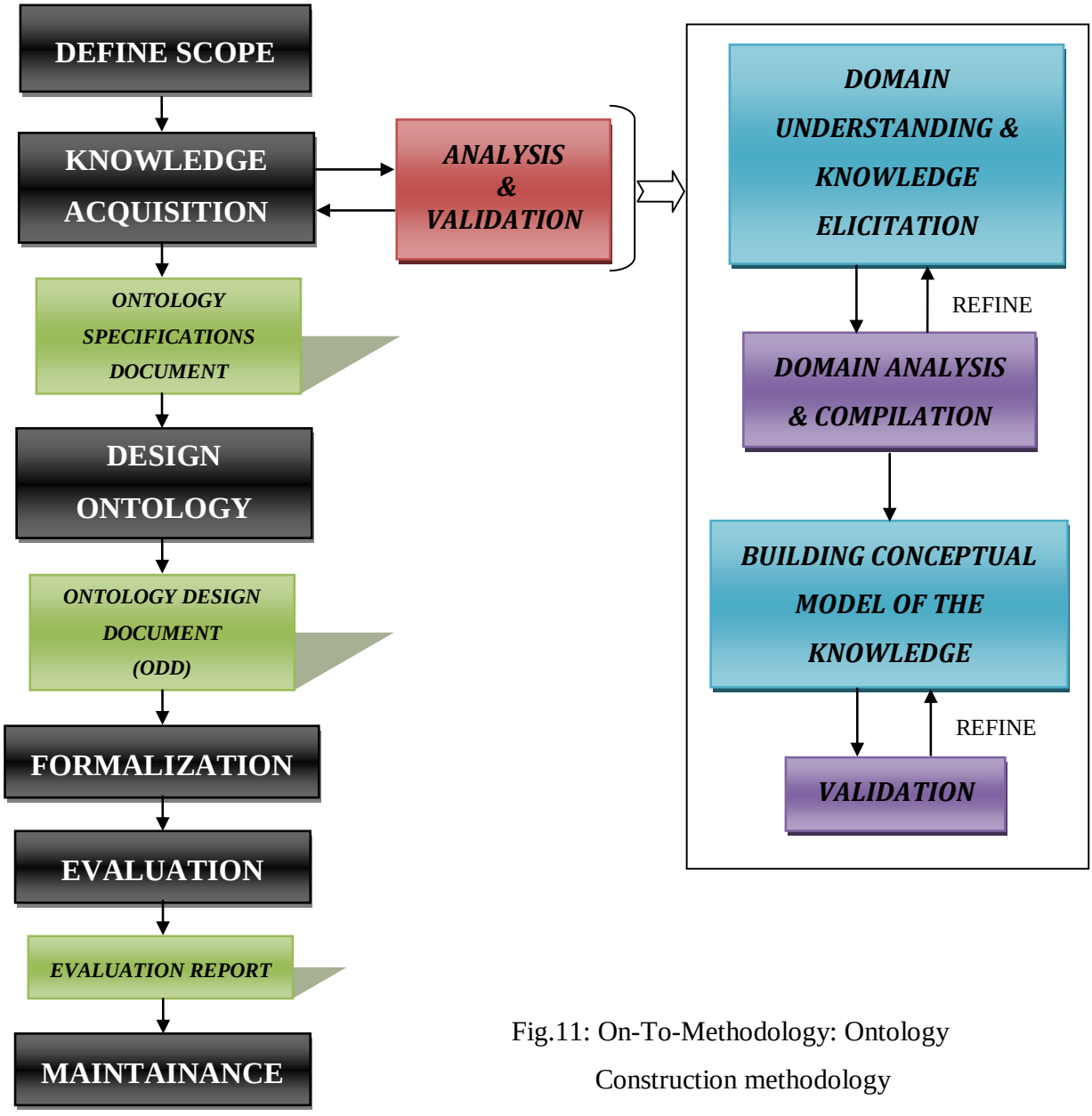


Fig.11: On-To-Methodology: Ontology Construction methodology

3.2 Define Ontology Scope

The target of this phase is to establish the scope of the ontology in terms of purpose that the ontology would serve with associated advantages. It should provide an objective description of the intended users, various scenarios in which it would be used and the end users.

3.3 Knowledge Acquisition

This activity would require the identification of *knowledge source* from where the knowledge is to be acquired. It can include the following:

- Books
- Experts
- Figures
- Tables
- Files
- Webpages
- Other ontologies etc.

Various techniques can be used for defining this such as brainstorming, interviews, group discussions, formal and informal reviews etc. The output of this phase is a *Ontology Specifications Document*.

We propose the use of *FAST* (Facilitated Application Specification Technique) here. It is a team oriented Software Engineering approach for gathering of requirements. This approach encourages the creation of a joint team of domain experts and users who work together to understand the expectations and propose a set of requirements.

We have adapted FAST to elicitate conceptual knowledge required to construct ontology in a formal way. We have designed various forms which are used to gather this knowledge. We identify the various stakeholders of the ontology and generate different forms to acquire the required knowledge. Figure 12 illustrates the various forms that we have generated.

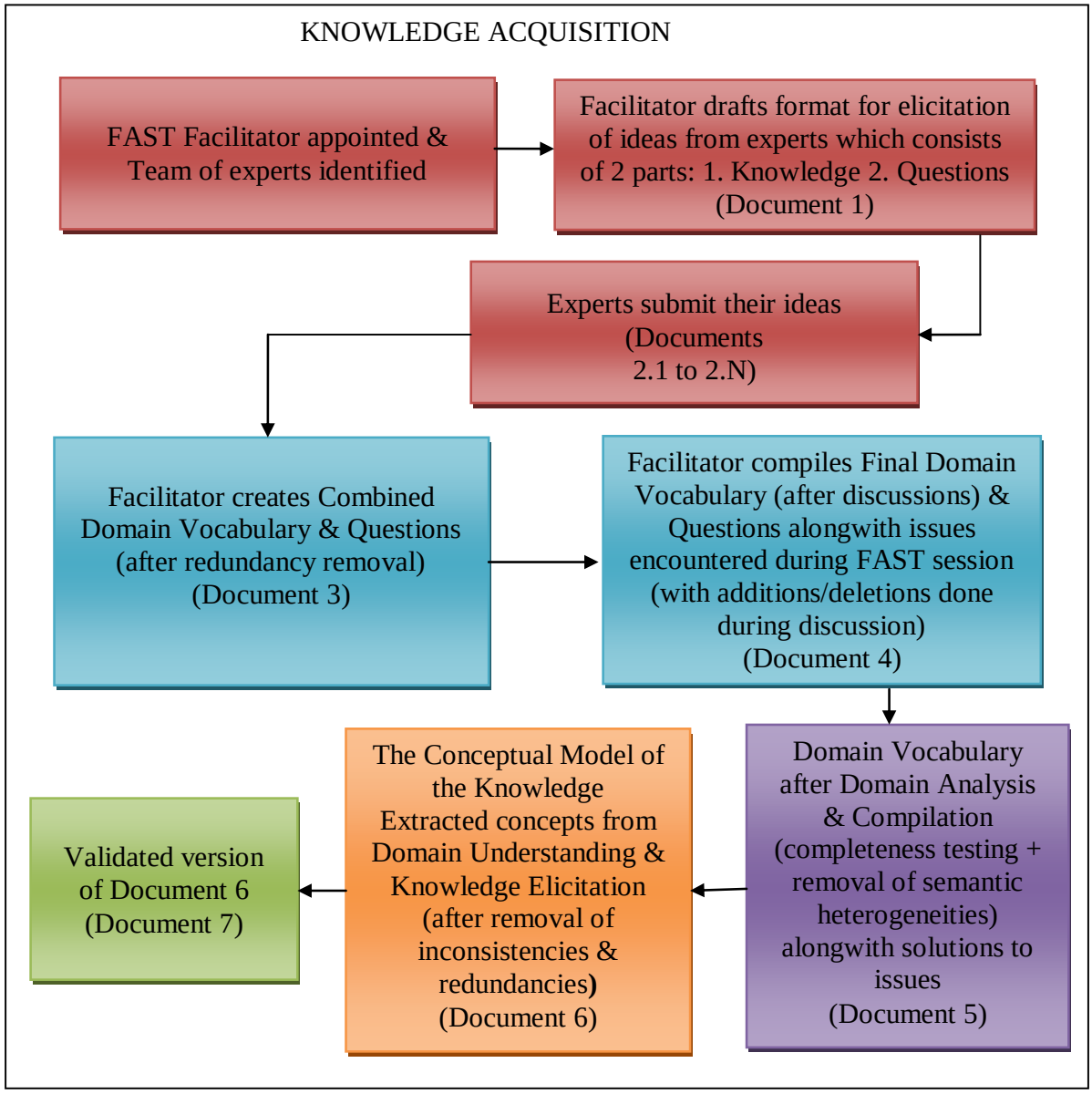


Figure 12: Various forms generated during different activities

As presented in figure, in the first phase a facilitator & team of experts are identified. The facilitator controls the various activities of the FAST session and prepares an informal agenda to encourage the free flow of ideas. The facilitator drafts a form (Document 1) to elicit ideas from the experts which consists of two parts which are required to be filled by the experts. The two parts are Knowledge and Questions. This form is filled by the various experts and submitted

back to the facilitator (Documents 2.1 to 2.N where N is the number of experts). The facilitator then compiles the ideas from these forms into a separate document (Document 3). The consolidated ideas are then discusses within the team and a final document i.e. document 4 is created which consists of the domain vocabulary and questions alongwith the issues encountered during the FAST session. The domain vocabulary listed in document 4 is analysed for the given domain and thus document 5 is created after compilatation. The concepts are then extracted from the Domain Understanding & Knowledge Elicitation as a result document 6 is obtained which is then validated and the final document consisting of the Conceptual model of the knowledge is obtained in the form of document 7.

We consider this activity consisting of four main tasks as described below. All these activities are carried out in a formal manner with the support of forms generated.

3.3.1 Domain Understanding & Knowledge Elicitation

Based on prelude

In this phase, the basic elements of the ontology are identified and enumerated. The elements of ontology are Class, Relationships, Constraints, Forms, Instances, Constants, and Instance attributes. These are explained as follows [28]:

- *Class* or concepts for a domain such as location, city, travel, destination etc.
- *Relationships* are the properties between two classes such as ‘isa’.
- *Constraints* are conditions that must be satisfied during the design.
- *Instance* is values for particular categories in ontology.

Here we don’t categorize the domain keywords according to the above categories but the focus is on comprehensive list of constituents without worrying about redundancies or overlaps.

Competency questions are also elicitated in this phase. These questions are queries related to specific situations from real life problems that the ontology would help provide a solution. The competency questions are therefore used to evaluate the ontology thus developed.

3.3.2 Domain Analysis & Compilation

An expert for particular domain can easily examine the ontology in better way and changes suggested by expert will remove some of the inconsistencies at the earlier stage to make the design phase simpler. The expert review makes knowledge acquisition evolutionary [28].

So after drafting the basic model as described above as document 4, it is reviewed by the experts for following rules:

I. *Consistency rule*: various defects are analyzed for example usage of different names for a same concept.

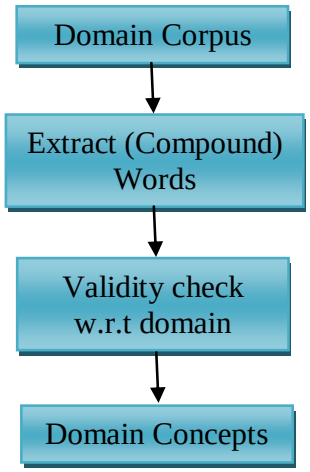
II. C*ompleteness rule*: the defects such as omission of domain resources and the omission of relationship are diagnosed. The issues encountered during FAST sessions are also resolved here.

If any consistency or completeness check fails, it is refined by the expert and the output is called as document 5.

3.3.3 Building Conceptual Model of the Knowledge

A conceptual model is developed on the basis of the domain model which we call as the document 5. Here the knowledge is represented in form of trees and table of rules/ formulas. The inconsistencies and the redundancies are resolved. The domain vocabulary identified in the *Domain Understanding & Knowledge Elicitation* is classified as classes, sub-classes, properties, constraints and individuals. The mechanism as shown in figure 13, can be employed.

Fig.:13 Framework for extracting the domain concepts



The definition of class hierarchy can be done in different ways [29] as following:

- Top-down development process begins with the definition of the concept of the field of the most common, followed by characterization of the concept.

- Bottom-up development process begins with the definition of the most specific classes, namely, class hierarchy the leaves, and then grouped into these classes more general concept.

- Combination of the two- defining more clearly the concept of the first, followed by their proper places generalization and characterization, it is the easier process.

The final result is document 6 we call as the conceptual model of the ontology.

3.3.4 Validation

The conceptual model of the knowledge, developed in the last step is reviewed here. Expert review is again used here. An expert for particular domain will examine the ontology for the following rules and the changes suggested will be incorporated [24]:

I. *Correctness rule*: The use of incorrect relationship, aggregation & specialization of classes and cardinality of relationship are tested.
II. *Rule for Semantic heterogeneities*: Semantic heterogeneities are also determined and dissolved in this step.
III. *Constraints*: Constraints on domain and range values of each object property and datatype property are also verified in the testing activity.

The result of this activity is document 7 which is the validated version of document 6.

3.3.5 The format for the Ontology Specification document is described as follows:

ONTOLOGY SPECIFICATION DOCUMENT FORMAT

Domain: ……
Date: ……..
Author: …….....

1. Introduction

Analogous to IEEE std. 830-1998 for Software Requirement Specification (SRS), this document aims at defining the overall requirements for Ontology being developed. The final ontology will be having only features/functionalities mentioned in this document and assumptions for any additional functionality/feature should not be made by any of the parties involved in developing/testing/implementing/using this product. In case it is required to have some additional features, a formal change request will need to be raised and subsequently a new release of this document and/or product will be produced.

1.1. Purpose

The purpose of this document is to record the requirements of the ontology. This document is also the starting point for design phase of ontology development methodology and is also used for testing the ontology when developed.

1.2. Scope

Here the scope of the ontology is established in terms of purpose that the ontology would serve with associated advantages. An objective description of the intended users is also provided here alongwith the various scenarios in which it would be used and the end users.

1.3. Definitions, acronyms & abbreviations

Here the various terms are defined and all the acronyms & abbreviations using during the documentation are listed.

1.4. References

A complete list of all the referenced documents is provided here.

1.5. Sources of knowledge

The various sources of knowledge from where the elements of the ontology are derived are listed under this sub-section.

1.6. Overview

The contents & organization of the rest of the Ontology specification document is described here.

2. **Overall Description**

The general factors that affect the ontology are described under this section.

2.1. Ontology Functions

The ontology will store the following elements:

- Class hierarchy
- Properties
- Inverse properties
- Instances

Based on the above information, the ontology will help solve some problem at hand. The major functions of such a system are listed here

2.2. User Characteristics

The characteristics of the intended users are listed here with the following details:

- Educational Qualification:
  The required educational qualification that the user should have to use the system is provided here.

- Experience Requirements:
  Any experience requirements of the user are provided here.

- Technical Expertise:
  The technical expertise that the user should possess is provided here.

2.3. Constraints

An overview of the constraints applicable to the ontology system is provided here.

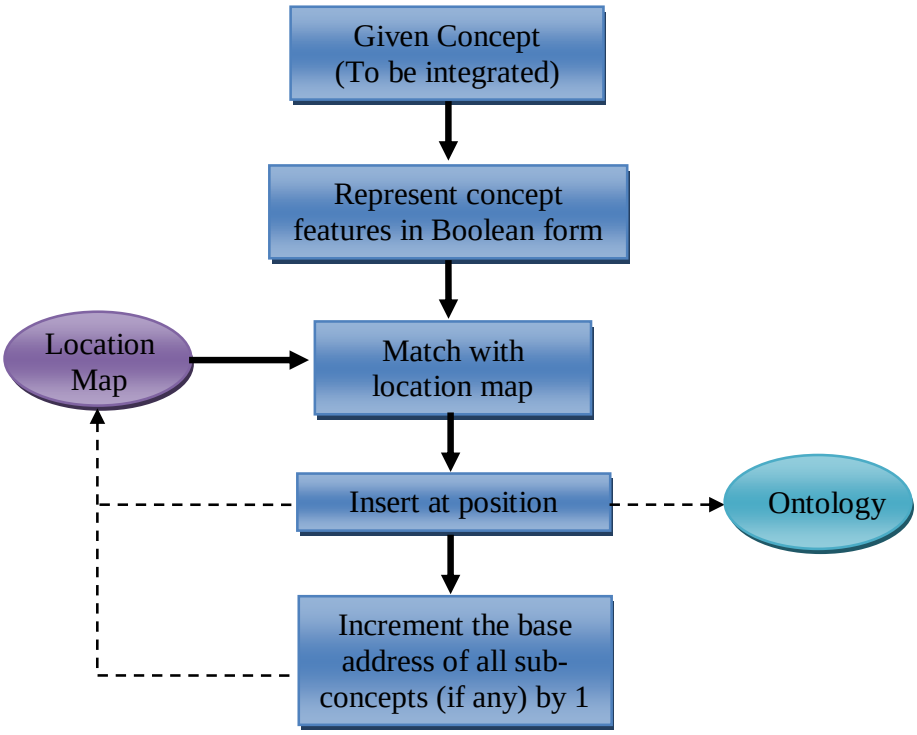
3.4 Design Ontology

This phase consists of two main tasks. One is the design task where with the help of the Ontology Specification Document and the Conceptual model of the knowledge, the concepts are mapped to location in location map. In this phase the concepts in the conceptual model are structured so that knowledge can be accessed. Second task is to produce a design document called Ontology Design Document (ODD) which can be helpful in evolution of the ontology.

3.4.1 Design

This is the core task of building ontology where different concepts such as classes, instances etc. in the conceptual model are processed to form the structure of ontology. We have adapted an algorithm for this and apply the technique [15] as shown in figure 14 below:

Fig.14: Adding new concept



The steps as shown in figure 15 can be explained as follows:

i. A concept is selected from the conceptual; model.
ii. It is then converted to Boolean form by representing its features in Boolean form.
iii. The concept as represented in above step is then matched with other concepts already present in location map.
iv. On determination of the correct position of the concept in the ontology structure (/location map), the concept is inserted in that position.
v. If there exist any sub-concepts concepts then increment their base address by 1.

We have automated the above task by implementing the following algorithm:

i. Given an Ontology structure with a concept *Thing* which is super concept of all concepts.
ii. A concept with from Conceptual model of the knowledge is selected. Let it be say X.
iii. All features for that concept are identified and labeled with a feature number as follows:
{X} → f^+ <feature no. from 1 to n>
iv. The concept is then represented in the Boolean form as follows:
C (X): { $f^+ 1 \bullet f^+ 2 \bullet \ldots \bullet f^+ n$ }
v. The concept represented in Boolean form is compared to other concepts in the location map one by one. Say the concept currently chosen from the Ontology structure be Y with address in location map as (a, b). Following cases can happen:
    a. If {Y}: { $f^+ 1 \bullet f^+ 2 \bullet \ldots \bullet f^+ (n-k)$ } where k<n, and we have
    {X}: { $f^+ 1 \bullet f^+ 2 \bullet \ldots \bullet f^+ (n-k) \bullet \ldots \bullet f^+ n$ }
    In this case there is match between X and Y for $f^+ 1$ to $f^+ (n-k)$. This means that X is a child of Y with $f^+ (n-k+1)$ to $f^+ n$ as additional features. Thus X is inserted in the Location map with address (a+1, xx).
    b. If {Y}: { $f^+ 1 \bullet f^+ 2 \bullet \ldots \bullet f^+ m$ }, and we have
    {X}: { $f^+ 1 \bullet f^+ 2 \bullet \ldots \bullet f^+ n$ }
    If there exists a feature in Y that is not present in X, then X is not a sub-concept of Y thus it is either a brother concept of this concept or a sub-concept of some other concept.

vi. Steps ii to v are repeated for all concepts in the conceptual model and the end result is the complete ontology structure.

The following are the snapshots of our tool:

Figure 15: Home screen

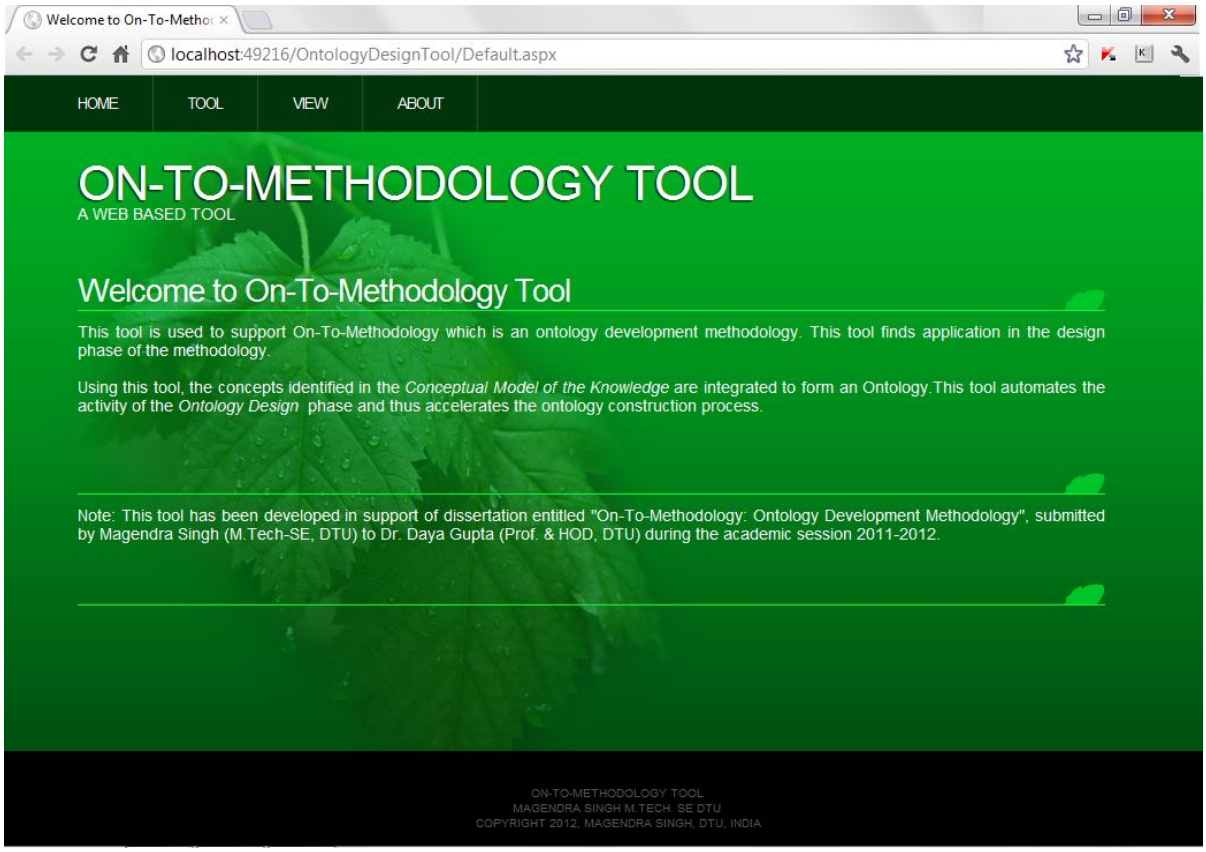


Fig.16 (a): Interface for designing

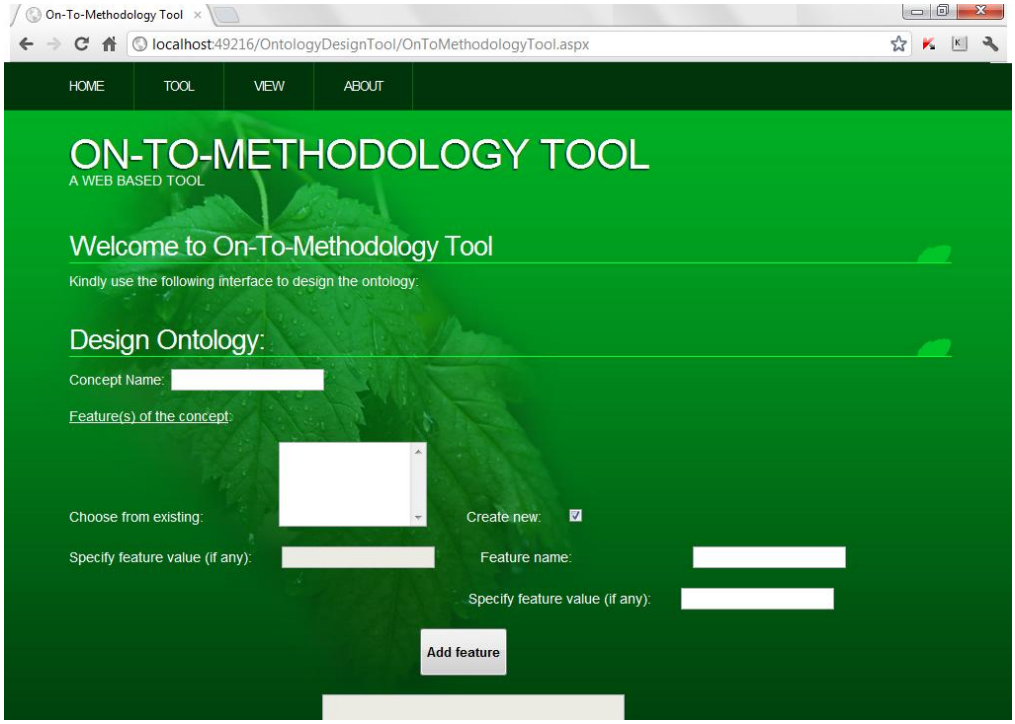
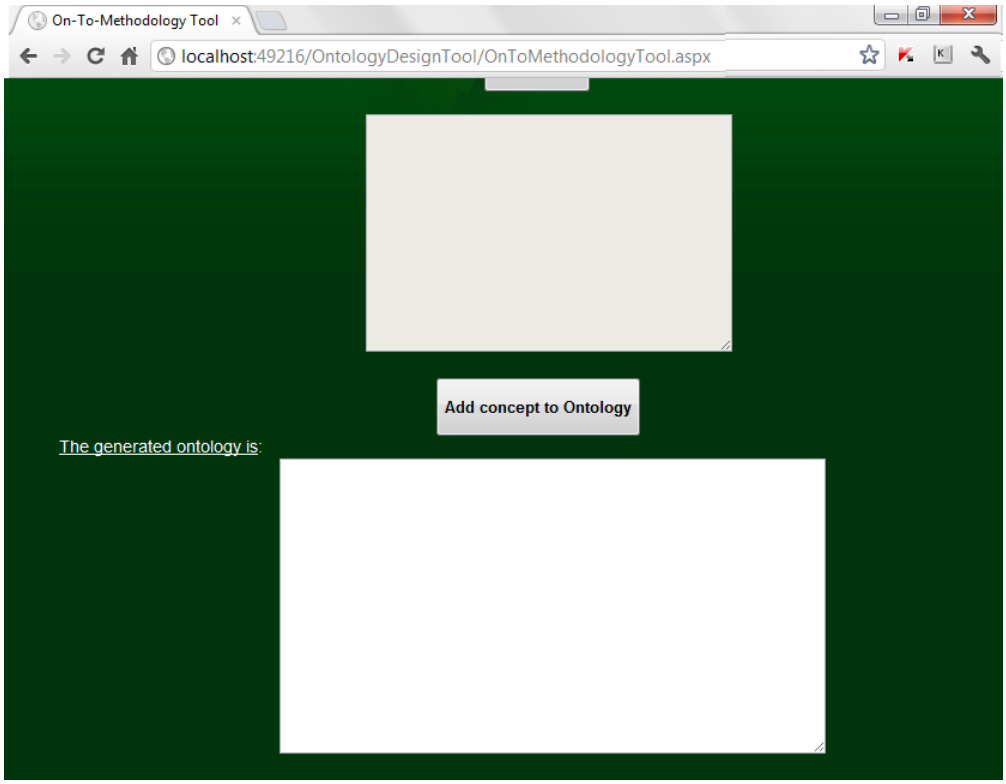


Fig.16 (b): Interface for designing



3.4.2 Ontology Design Document

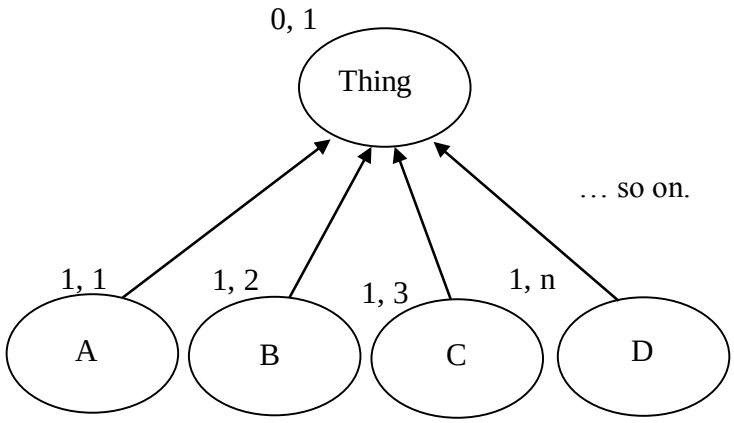
The output of this phase is the Ontology Design Document (ODD). This document is a formal record of the ontology structure and consists of the Location map alongwith the graphical representation of the ontology structure. The format for ODD is as follows:

ONTOLOGY DESIGN DOCUMENT

The location map is as below:

CONCEPT ADDRESS	CONCEPT FEATURE
Thing (0, 1)	X
A (a1, a2)	f^{+} 1 • f^{+} 2 • …
B (b1, b2)	f^{+} x • f^{+} y • …
…	…

The graphical representation of ontology structure is as follows:



3.5 Formalization

The main aim behind ontology development is to make the information to be understood by the machines. Thus, the ontology is finally formalized using the selected ontology languages and tools.

Semantic Web is not a single technology and it comprises of a number of components including ontology languages (RDF/RDFS, OWL etc.), editing tools (Protégé, Ontolingua etc.) and standards (WSMO, OWL-S etc.). As quoted in [28], different ontology languages provide different facilities. The most recent development in standard ontology languages is OWL from the W3C. OWL [8] makes it possible to describe concepts but it also provides new facilities. It has a richer set of operators - e.g. and, or and negation. It is based on a different logical model which makes it possible for concepts to be defined as well as described. Complex concepts can therefore be built up in definitions out of simpler concepts. OWL ontologies may be categorized into three sub languages: OWL-Lite, OWL-DL and OWL-Full. A defining feature of each sub-language is its expressiveness. OWL-Lite is the least expressive sub-language. OWL-Full is the most expressive sub-language. The expressiveness of OWL-DL falls between that of OWL-Lite and OWL-Full. Islam et. al. [30] have compared several editing tools as shown in Table 1 below.

Table 1: A comparison of ontology editing tools

Tools	Free	Open Source	Dot Net and Web Based	Import Languages	Export Languages	Inferencing Reasoning Tools
Protégé	√	√	×	RDF, OWL	RDF, OWL, FLogic, CLIPS	√
OntoEdit (Free)	√	×	×	DAML + OIL, RDFS	DAML + OIL, RDFS	×
Differential Ontology Editor (DOE)	√	×	×	RDFS, OWL	DAML+OIL, OIL, RDFS, OWL	×
IsaViz	√	√	×	RDF/XML, N-Triples	RDF/XML, N-Triples	√
Ontolingua	√	×	×	DAML+OIL, KIF,CLIPS	DAML+OIL, KIF, CLIPS	√
Altova Semantic Works™	×	×	×	N-triples, OWL, RDF and RDFS	N-triples, OWL, RDF and RDFS	×
WebODE	√	×	×	RDF(S), DAML+OIL, UML, OWL	CLIPS, DAML+OIL, UML, OWL	√
TopBraid Composer	×	×	×	RDBMS, OWL, RDF	OWL,RDF, XML	√
Morla	√	√	×	RDF, OWL	RDF,OWL	×
Hozo	√	√	×	RDF, OWL (subset)	OWL, RDF	×
TODE	√	×	√	RDF, OWL-Lite, N-3, RDBMS, N-Triple,	RDF, OWL-Lite, N-Triple, N-3, RDBMS	×

3.6 Evaluation

After the ontology has been drafted, it should be checked for quality and knowledge representation efficiency. If any constraints are violated then they are corrected. The ontology engineers also verify the developed ontology against the requirement document.

The term 'Evaluation' encompasses both- verification & validation. Based on [25], we identify following two ways of evaluation:

- *Relation Evaluation*: Relation evaluation infers the new individual based on constructed rule set, and judges whether the relation is coincident with the professionals' knowledge. This is done using the *Reasoner* that comes inbuilt in the tool, like Fact++ in Protégé.

- *Evaluation of hierarchy*: Evaluation of hierarchy develops limits definition of created class and individual properties, makes use of ontology tool (like OntoGraf, OWLViz) to infer and create Ontology Graph automatically, and judge whether Subordination between the class and individual in the graph are coincident.

Also, in above steps, the answers to competency questions are verified by the experts. The output of this phase is the *Evaluation Report*. It consists of following:

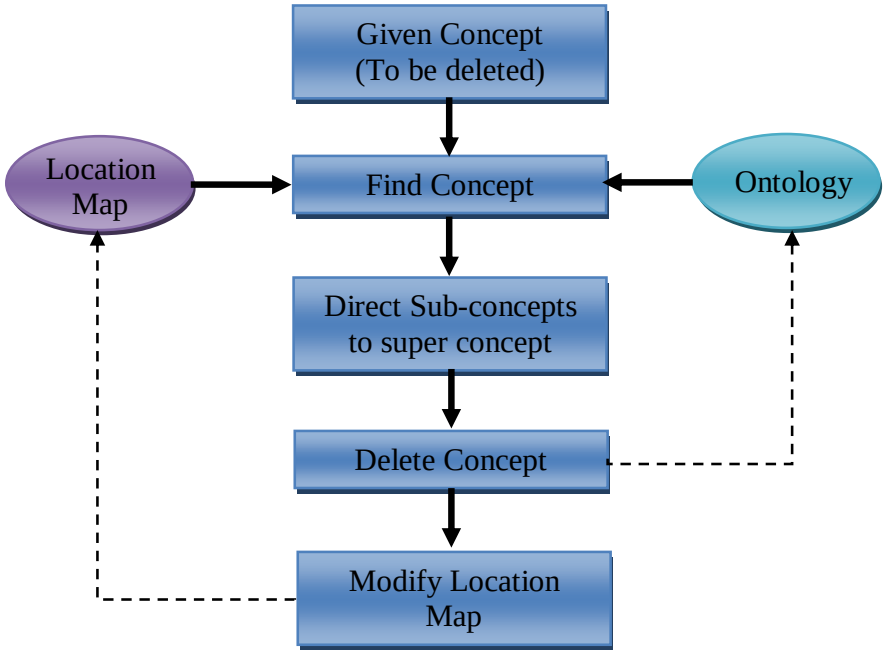
I. Evaluation of competency questions by the experts to check if the results are as per the expectation.
II. Evaluation of hierarchy using OntoGraf & OWLViz.

3.7 Maintenance

Ontologies need to adapt to the changing specifications to become more mature. This can be accomplished by adding, updating and removing concepts/ parts of the ontology. For this, following techniques can be employed:

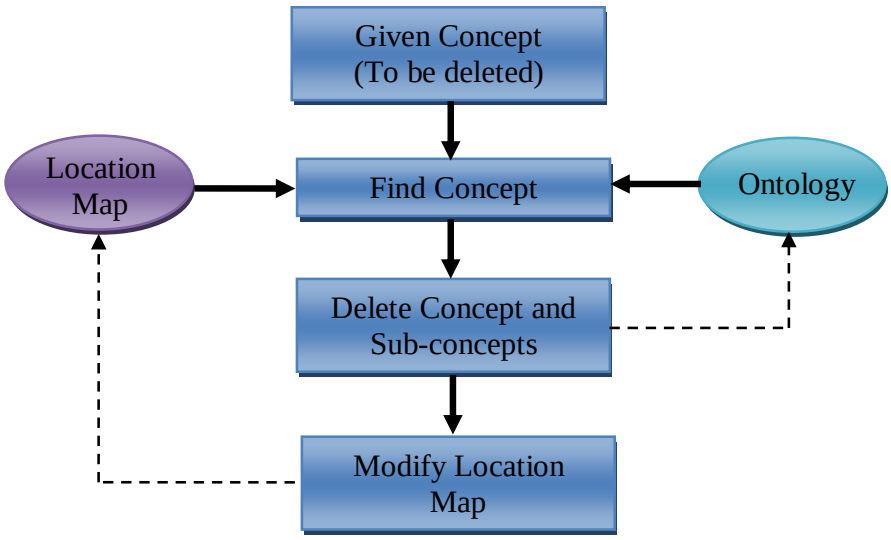
- Addition of new concept: Technique as used in ontology design can be used.
- Deletion of a single concept: Technique as shown in figure 17 can be used.

Fig.17: Deletion process of single concept



- Deletion Operation for a Portion of the Ontology: Technique as shown in figure 18 can be used.

Fig.18: Deletion process of a portion of ontology



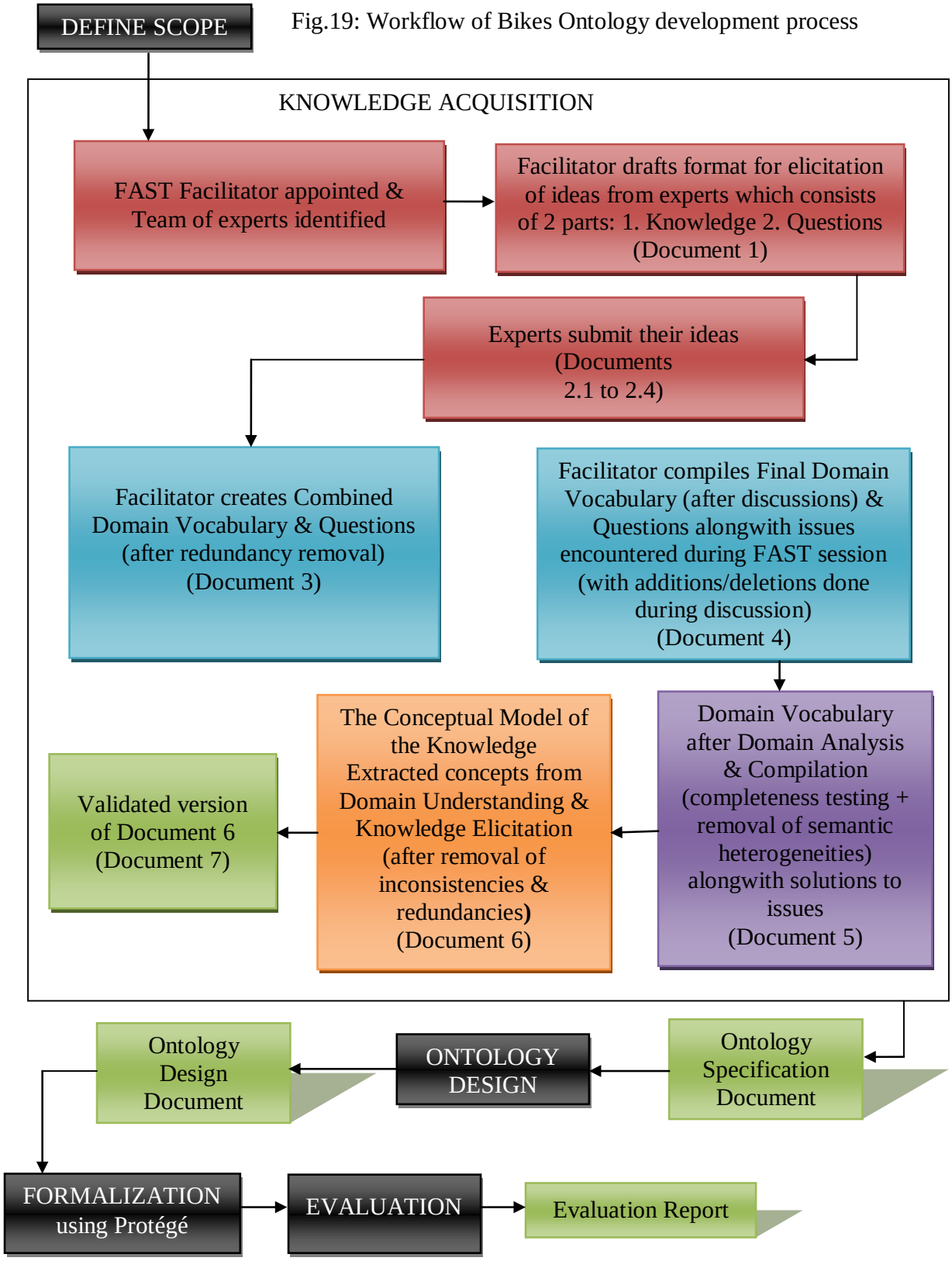
Chapter 4

DISCUSSION

Here we illustrate our methodology by working out an Ontology of Bikes. The intended ontology of Bikes will be representation of bikes in the domain of ontology. Millions of people all over the world choose bikes over automobiles for the thrill, speed, and high performance capabilities. Bikes have become a major means of transportation over the years. There exists a large group of bike users who either use it for transportation or for sports. India in one of the largest markets around the world and all has become a key market of all manufactures.

The purpose of this ontology is to provide information on bikes based on the criteria specified by the users like Make, Engine capacity, Power, Price, Fuel tank capacity, Mileage etc.
Following is the workflow of the Bikes Ontology development process (figure 19):

Fig.19: Workflow of Bikes Ontology development process



4.1 SCOPE

Date: ……

The intended ontology of Bikes will be representation of bikes in the domain of ontology. The following are the related details:

- Target Users:
  The end users of the bikes ontology are targeted to be customers who want some information regarding bikes. Other users include Bike manufacturers and retailers.

- Purpose:
  The purpose of this ontology is to provide information on bikes based on the criteria specified by the users:
  - The ontology would hold information to answer queries of customers based on single (/combination of) parameter(s) which are Make, Engine capacity, Power, Price, Fuel tank capacity, Mileage, Brake type, Weight, Wheel type, Ignition and Number of gears.
  - Bike manufacturing organizations can use this ontology to identify the bike configurations that are suitable for a particular market and can also use it to analyze current sales and make future predictions. This will guide them to plan their production & inventory.
  - This ontology can prove to be beneficial for bike retailers as they can use it to plan their inventory and analyze their sales.

- Pros:
  The advantage that this ontology would provide is its capability to answer the queries of the customers across a large information base of different bikes, based on multiple search criteria with complex inter-relations.

4.2 **DOCUMENT 1:** Date: ……

FORMAT FOR ELICITATION OF IDEAS

(Created by: Mr. Magendra Singh)

Name: ……

Educational Qualification: …….

Domain corpus consisting of various keywords:

……

……

……

……

……

……

……..

Date: …. /…./……..

Signature: ……..

( )

4.3 DOCUMENT: 6

The following document represents the conceptual model of the knowledge. The Ontology Specification document is presented next in section 4.4. The various other documents namely Document 2.1 to 2.4, 3, 4, 5, 7 (see figure -18) are listed in *Appendix-A* at the end of the thesis.

Date: ……

THE CONCEPTUAL MODEL OF THE KNOWLEDGE:

EXTRACTED CONCEPTS FROM DOMAIN UNDERSTANDING & KNOWLEDGE ELICITATION

(AFTER REMOVAL OF INCONSISTENCIES & REDUNDANCIES)

Step 1: Resolve inconsistencies and the redundancies

Engine Capacity

Power

Make

Mileage

Brakes (Disk/Drum)

Fuel Tank Capacity

Price

Ignition (KickStart/SelfStart)

Seat type (Split/Single/Step/Normal)

Weight

Wheel type (Alloy/Wirespoke)

Weight

Gears

~~(0-60)~~

~~Looks~~

~~Ground clearance~~

~~Tire size~~

~~Tire Width~~

~~Visor~~

~~Warranty (upto kms?)~~

~~Maintenance~~

~~Free Service~~

~~Suspension~~

Hero Honda

Hero

Bajaj

~~Kawasaki~~

Royal Enfield

Yamaha

TVS

~~Harley Davidson~~

Suzuki

Karizma (Normal)→ 223cc, 17bhp @7000 rpm, Rs. 74000, 15ltr., 40km/ltr, Combo, 150kg, Alloy, self, 5

Karizma (ZMR) → 223cc, 17.6bhp @ 7000rpm, Rs.120000 ,16 ltr., 40km/ltr, Disk, 159kg, Alloy, Self, 5

Splendor (Plus) → 97.2cc, 7.5bhp @ 8000rpm, Rs. 45000, 10.5ltr., 75km/ltr, Drum, 109kg, WireSpoke, Kick, 4

Splendor (NXG) → 97.2cc, 7.7bhp @ 7500rpm, Rs.48000 , 10.3ltr., 65km/ltr, Drum, 107kg, Alloy, Kick, 4

Splendor (Super) → 125cc, 9bhp @ 7000rpm, Rs.47000 ,12 ltr., 70km/ltr, Drum, 121kg, Alloy, Self, 4

Splendor (Pro)→97.2 cc, 7.6bhp @ 7500rpm, Rs.55000 , 11ltr., 80km/ltr, Drum, 109kg, Alloy, Self, 4

Paasion Pro → 97.2cc, 7.6bhp @ 7500rpm, Rs.53000 , 12.8ltr., 75km/ltr, Combo, 119kg, Alloy, Self, 4

CD-Dawn → 97.2cc, 7.7bhp @ 7500rpm, Rs.37000 , 10.5ltr., 75km/ltr, Drum, 107kg, Wire Spoke, Kick, 4

CD-Deluxe → 97.2cc, 7.7bhp @ 7500rpm, Rs.43000 , 10.5ltr., 75km/ltr, Drum, 107kg, Alloy, Kick, 4

Glamour (Normal) → 125cc, 9bhp @ 7000rpm, Rs.55000 ,13.6 ltr., 60km/ltr, Combo, 125kg, Alloy, Self, 4

Glamour (PGMFi) → 125cc, 9bhp @ 7000rpm, Rs.58500 ,12 ltr., 70km/ltr, Combo, 125kg, Alloy, Self, 4

Achiever → 150cc, 13.4bhp @ 8000rpm, Rs.60000 ,12.5 ltr., 55km/ltr, Combo, 134kg, Alloy, Self, 5

CBZ Xtreme → 150cc, 14.4bhp @ 8500rpm, Rs. 65000, 12.3ltr., 50km/ltr, Combo, 141kg, Alloy, Self, 5

Hunk→ 150cc, 14.2bhp @ 8500rpm, Rs. 63000 , 12.2ltr., 50km/ltr, Disk, 145kg, Alloy, Self, 5

Impulse→ 150cc, 13bhp @ 75000rpm, Rs.79000 , 11ltr., 55km/ltr, Combo, 119kg, Alloy, Self, 5

CT100 → 100cc, 8.2bhp @ 7500rpm, Rs. 32000, 10.5ltr., 80km/ltr, Drum, 109kg, Wire Spoke, Kick, 4

Pulsar 135 LS → 135cc, 13.3bhp @ 9000rpm, Rs. 57000, 8ltr., 68km/ltr, Combo, 122kg, Alloy, Self, 5

Pulsar 150 DTS-i → 150cc, 14.09bhp @ 8500rpm, Rs. 63000, 15ltr., 48km/ltr, Combo, 130kg, Alloy, Self, 5

Pulsar 180 DTS-i → 180cc, 16.5bhp @ 8000rpm, Rs. 67000, 15ltr., 55km/ltr, Combo, 140kg, Alloy, Self, 5

Pulsar 200 DTS-i → 200cc, 18bhp @ 8000rpm, Rs. 70000, 15ltr., 40km/ltr, Disk, 145kg, Alloy, Self, 5

Pulsar 220 DTS-i → 220cc, 20bhp @ 8500rpm, Rs. 90000, 15ltr., 35km/ltr, Disk, 150kg, Alloy, Self, 5

Avenger 220 DTS-i →220 cc, 16.5bhp @ 8000rpm, Rs. 75000, 14ltr., 40km/ltr, Combo, 152kg,

Alloy, Self, 5

Discover 135 → 135cc, 13.1bhp @ 8500rpm, Rs. 55000, 10ltr., 60km/ltr, Combo, 125kg, Alloy, Self, 4

Discover 125 → 125cc, 11bhp @ 8000rpm, Rs. 52000, 8ltr., 85km/ltr, Drum, 125kg, Alloy, Self, 4

Discover 100 → 100cc, 7.5bhp @ 7500rpm, Rs. 44500, 10.3ltr., 91km/ltr, Drum, 115kg, Alloy, Self, 4

Platina 100 → 99.27cc, 8.2bhp @ 7500rpm, Rs. 35500, 13ltr., 108km/ltr, Drum, 113kg, Alloy, Kick, 4

XCD → 125cc, 7.01bhp @ 7000rpm, Rs. 46000, 13ltr., 109km/ltr, Drum, 115kg, Alloy, Self, 4

Duke200 → 200cc, 25bhp @ 10000rpm, Rs. 130000, 10.5ltr., 35km/ltr, Disk, 136kg, Alloy, Self, 6

~~Ninja 250R~~

~~Ninja 650R~~

Bullet Electra Twinspark →350 cc, 19.8bhp @ 5250rpm, Rs. 111000, 13.5ltr., 40km/ltr, Combo, 183kg, Wire Spoke, Self, 5

Bullet Electra EFI → 500cc, 27.2bhp @ 5250rpm, Rs. 125000, 14.5ltr., 40km/ltr, Combo, 185kg, Wire Spoke, Self, 5

Bullet Electra Deluxe →500 cc, 27.2bhp @ 5250rpm, Rs. 140000, 14.5ltr., 45km/ltr, Combo, 187kg, Wire Spoke, Self, 5

Bullet 350 Twinspark → 350cc, 19.8bhp @ 5250rpm, Rs. 100000, 13.5ltr., 45km/ltr, Combo, 180kg, Wire Spoke, Self, 5

Royal Enfield Classic 500 → 500cc, 27.2bhp @ 5250rpm, Rs. 155000, 13.5ltr., 35km/ltr, Combo, 187 kg, Wire Spoke, Self, 5

Royal Enfield Classic 350 → 350cc, 19.8bhp @ 5250rpm, Rs. 117000, 13.5ltr., 45km/ltr, Combo, 182kg, Wire Spoke, Self, 5

Thunderbird Twinspark → 350cc, 19.8bhp @ 5250rpm, Rs. 116300, 15.5ltr., 45km/ltr, Combo, 182kg, Wire Spoke, Self, 5

R15 → 150cc, 16.8bhp @ 8500rpm, Rs. 119000, 12ltr., 45km/ltr, Disk, 136kg, Alloy, Self, 6

FZ → 153cc, 14bhp @ 7500rpm, Rs. 74900, 12ltr., 50km/ltr, Combo, 135kg, Alloy, Self, 5

Victor → 110cc, 8.1bhp @ 7250rpm, Rs. 50000, 11ltr., 85km/ltr, Drum, 113kg, Wire Spoke, Kick, 4

CBR → 250cc, 26.4bhp @ 8500rpm, Rs. 160000, 13ltr., 30km/ltr, Disk, 165kg, Alloy, Self, 6

Shine → 125cc, 10.3bhp @ 7500rpm, Rs. 53500, 11ltr., 60km/ltr, Drum, 122kg, Alloy, Self, 4

Unicorn → 150cc, 62000bhp @ 8000rpm, Rs. 62000, 13ltr., 60km/ltr, Combo, 165kg, Alloy, Self, 5

~~Superlow (560000, 883)~~

~~Iron883 (883, 660000)~~

~~Roadster (883, 765000)~~

~~Forty-Eight (1202,8,65000)~~

~~Nightster (1202,110000)~~

~~XR1200X(1200,121000)~~

~~StreetBob(1010000,1585)~~

~~SuperglideCustom (1165000,1585)~~

~~Mountain Bikes~~

~~Dipper~~

~~Model~~

~~Affordable~~

~~Light Weight~~

~~Looks~~

Step 2: Identify Classes, sub-classes & individuals

Root class → *Thing*

Sub-class of *Thing* → *Bikes*

Subclasses of *Bikes* →

Make
EngineCapacity
Power
Price
FuelTankCapacity
Mileage
NamedBikes (It is supposed to be merely a container class.)
Brakes
Weight
WheelType
Ignition
Gears

Subclasses of *Make* →*Hero Honda*, *Hero*, *Bajaj*, *Royal Enfield*, *Yamaha*, *TVS* and *Honda*

Subclasses of *Brakes* → *Combo*, *DiskBrakes*, *DrumBrakes*

Subclasses of *WheelType* → *Alloy*, *Wirespoke*

Subclasses of *Ignition* → *KickStart*, *SelfStart*

Subclasses of *Gears* → *4*, *5*, *6*

Subclasses of *NamedBikes* →

HeroHondaBikes
HeroBikes
BajajBikes
RoyalEnfieldBikes
YamahaBikes
TVSBikes
HondaBikes

Subclasses of *HeroHondaBikes* →

Karizma (Normal) Model:
◊ Karizma (Normal)
Karizma (ZMR) Model:
◊ Karizma (ZMR)
Splendor (Plus) Model:
◊ Splendor (Plus)
Splendor (NXG) Model:
◊ Splendor (NXG)
Splendor (Super) Model:
◊ Splendor (Super)
Splendor (Pro) Model:
◊ Splendor (Pro)
Passion Pro Model:
◊ Passion Pro

CD-Dawn Model:
◊ CD-Dawn
CD-Deluxe Model:
◊ CD-Deluxe
Glamour (Normal) Model:
◊ Glamour (Normal)
Glamour (PGMFi) Model:
◊ Glamour (PGMFi)
Achiever Model:
◊ Achiever
CBZ Xtreme Model:
◊ CBZ Xtreme
Hunk Model:
◊ Hunk

Subclasses of *HeroBikes* → *Impulse Model*: ◊ Impulse

Subclasses of *BajajBikes* →

CT100 Model: ◊ CT100
Pulsar 135 LS Model:
◊ Pulsar 135 LS
Pulsar 150 DTS-i Model:
◊ Pulsar 150 DTS-i
Pulsar 180 DTS-i Model:
◊ Pulsar 180 DTS-i
Pulsar 220 DTS-i Model:
◊ Pulsar 220 DTS-i
Avenger 220 DTS-i Model:
◊ Avenger 220 DTS-i
Discover 135 Model:
◊ Discover 135
Discover 125 Model:
◊ Discover 125
Discover 100 Model:
◊ Discover 100
Platina 100 Model:
◊ Platina 100
Duke200 Model:
◊ Duke200

Subclasses of *RoyalEnfieldBikes* →

Bullet Electra Twinspark Model:
◊ Bullet Electra Twinspark
Bullet 350 Twinspark Model:
◊ Bullet 350 Twinspark
Bullet Electra EFI Model:
◊ Bullet Electra EFI
Bullet Electra Deluxe Model:
◊ Bullet Electra Deluxe
Royal Enfield Classic 500 Model:
◊ Royal Enfield Classic 500
Royal Enfield Classic 350 Model:
◊ Royal Enfield Classic 350
Thunderbird Twinspark Model:
◊ Thunderbird Twinspark

Subclasses of *YamahaBikes* →

R15 Model: ◊ R15

FZ Model: ◊ FZ

Subclasses of *TVSBikes* →

Victor Model: ◊ Victor

Subclasses of *HondaBikes* →

CBR Model: ◊ CBR

Shine Model: ◊ Shine

Unicorn Model: ◊ Unicorn

Step 3.1: Identify Properties

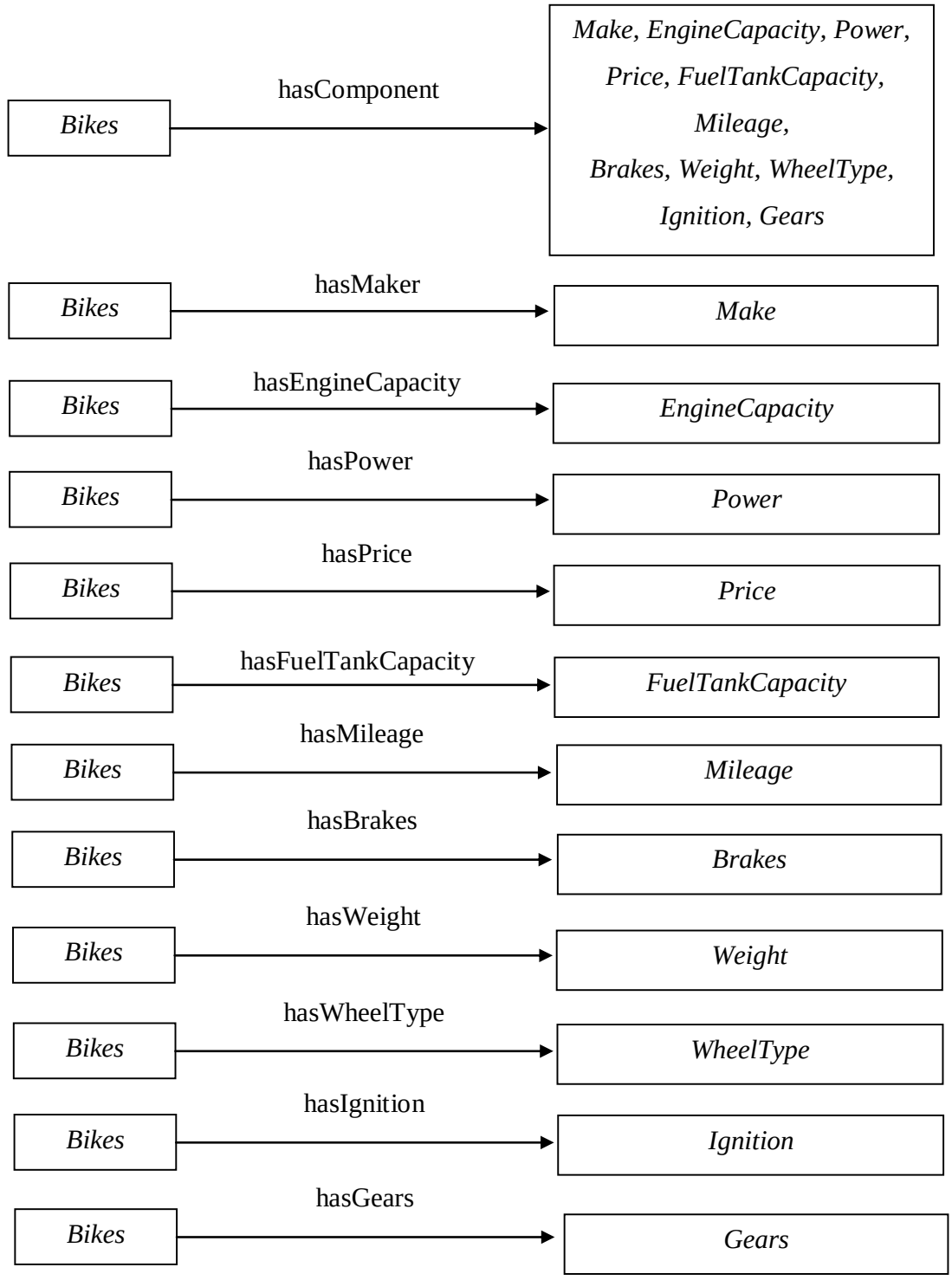
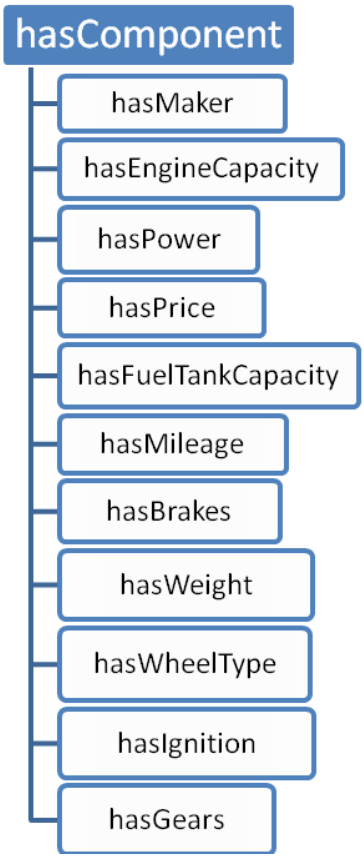


Fig. 19: Properties

Thus property hierarchy is:

Fig. 20: Property hierarchy



Step 3.2: Specify the Domains and Ranges of properties:

Table 2: Domain & Ranges of properties

Name	Domain	Range
hasMaker	Bikes	Make
hasEngineCapacity	Bikes	Float
hasPower	Bikes	String
hasPrice	Bikes	Integer
hasFuelTankCapacity	Bikes	Float
hasMileage	Bikes	Float
hasBrakes	Bikes	Brakes
hasWeight	Bikes	Float
hasWheelType	Bikes	WheelType
hasIgnition	Bikes	Ignition
hasBrakes	Bikes	Brakes

Step 3.3: Identify Inverse Properties

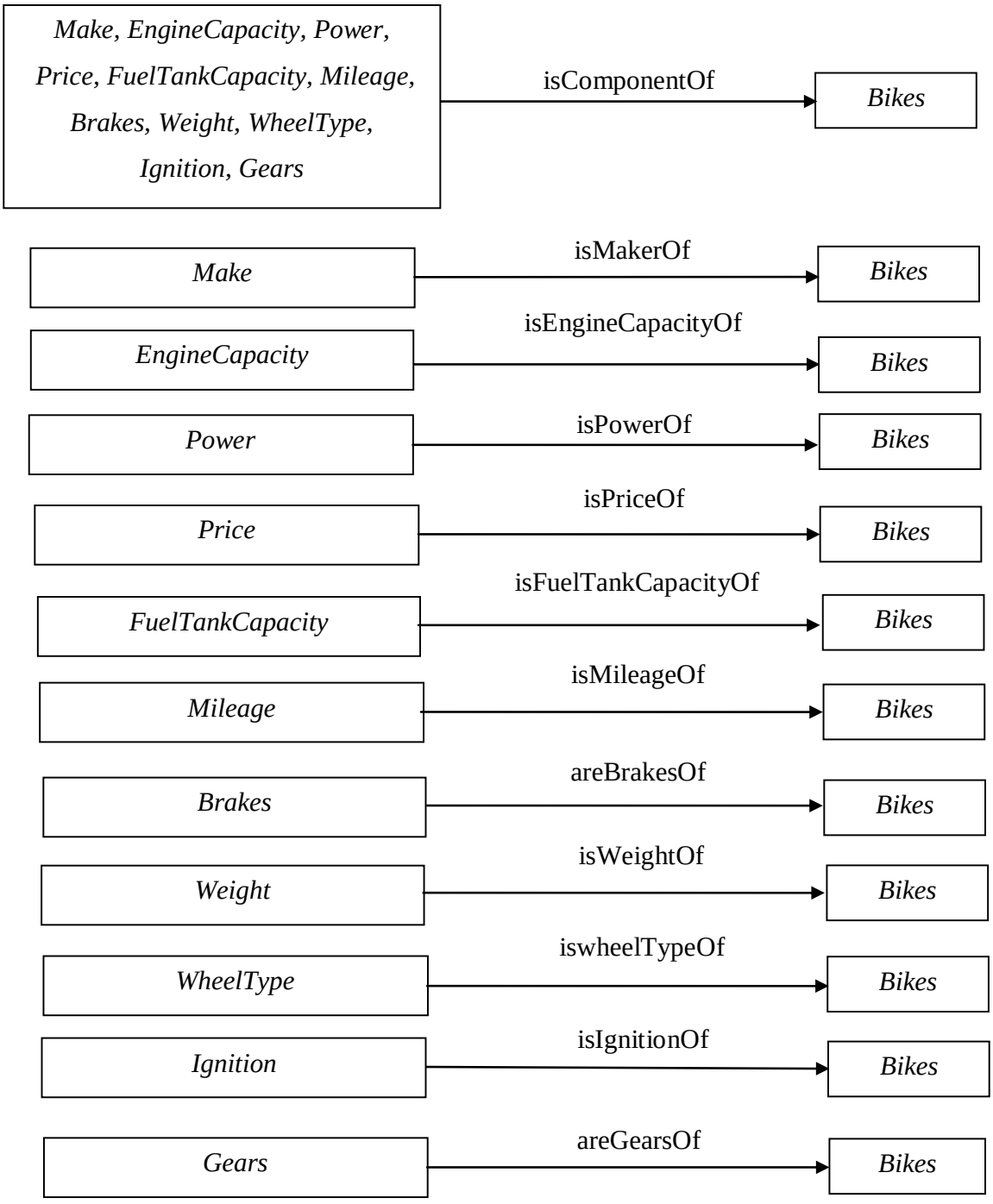
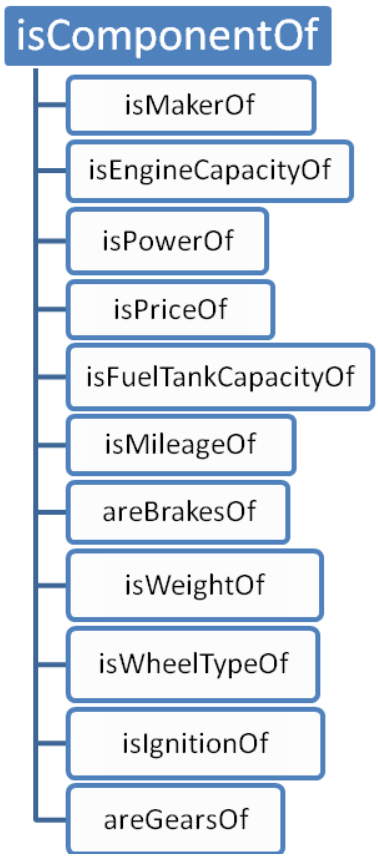


Fig. 21: Inverse Properties

Thus the inverse-property hierarchy is:

Fig. 22: Inverse-property hierarchy



Step 3.4: Specify the Domains and Ranges of inverse properties:

Table 3: Domain & Ranges of Inverse Properties

Name	Domain	Range
isMakerOf	Make	Bikes
isEngineCapacityOf	Float	Bikes
isPowerOf	String	Bikes
isPriceOf	Integer	Bikes
isFuelTankCapacityOf	Float	Bikes
isMileageOf	Float	Bikes
areBrakesOf	Brakes	Bikes
isWeightOf	Float	Bikes
isWheelTypeOf	WheelType	Bikes
isIgnitionOf	Ignition	Bikes
areBrakesOf	Brakes	Bikes

4.4 ONTOLOGY SPECIFICATIONS DOCUMENT

Domain: Automobile/Vehicles/Bikes

Date: 10 May 2012

Author: Mr. Magendra Singh

1. **Introduction**

This document aims at defining the overall requirements for Ontology of Bikes. The final ontology will be having only features/functionalities mentioned in this document and assumptions for any additional functionality/feature should not be made by any of the parties involved in developing/testing/implementing/using this product. In case it is required to have some additional features, a formal change request will need to be raised and subsequently a new release of this document and/or product will be produced.

1.1. Purpose

The purpose of this document is to record the requirements of an ontology of bikes. This document is also the starting point for design phase of ontology development methodology and is also used for testing the ontology when developed.

1.2. Scope

The intended ontology of Bikes will be representation of bikes in the domain of ontology. The purpose of this ontology is to provide information on bikes based on the criteria specified by the users.

- The ontology would hold information to answer queries of customers based on single (/combination of) parameter(s) which are Make, Engine capacity, Power, Price, Fuel tank capacity, Mileage, Brake type, Weight, Wheel type, Ignition and Number of gears.
- Bike manufacturing organizations can use this ontology to identify the bike configurations that are suitable for a particular market and can also use it to analyze current sales and make future predictions. This will guide them to plan their production & inventory.
- This ontology can prove to be beneficial for bike retailers as they can use it to plan their inventory and analyze their sales.

The advantage that this ontology would provide is its capability to answer the queries of the customers across a large information base of different bikes, based on multiple search criteria with complex inter-relations.

1.3. Definitions, acronyms & abbreviations

- Kms: Kilometers
- Ltr. : Liter
- Rs. : Rupees (Indian currency)
- UI: User Interface
- BE: Bachelor of Engineering
- B.Tech: Bachelor of Technology
- OWL: Web Ontology Language
- Customer: A person who either desires to purchase a bike or needs some information related to bikes.
- Bike retailer: An organization who sells bikes to customers on retail in market.
- Make: The name of the company that produces that bike.
- Engine capacity: The capacity of a particular bike's engine.
- Power: The maximum power produced by the engine of the bike.
- Price: The price of 1 unit of the particular bike.
- Fuel tank capacity: The amount of fuel (in liters) that can be held in the fuel tank of the bike.
- Mileage: The distance covered (in kilometers) by the bike in consumption of one liter of fuel.
- Type of brake: The braking mechanism (Drum/ Disk/ Combo) that is employed in a particular model of bike.

1.4. References

Document 1: Format for elicitation of ideas

Document 2: Domain corpus as identified by Expert 1: Mr. Magendra Singh

Document 3: Domain corpus as identified by Expert 2: Mr. Vipin Sharma

Document 4: Domain corpus as identified by Expert 3: Mr. Sandeep Saini

Document 5: Domain corpus as identified by Expert 4: Mr. Kushal Verma

Document 6: Combined domain vocabulary & questions (after redundancy removal).

Document 7: Final domain vocabulary & questions alongwith issues encountered during FAST session (with additions/deletions done during discussion).

Document 8: Domain Vocabulary after review (completeness testing + Removal of semantic heterogeneities) alongwith solutions to issues.

Document 9: Extracted concepts from Domain Understanding & Knowledge Elicitation (after removal of inconsistencies & redundancies).

Document 10: The Conceptual Model of the Knowledge (Reviewed version of Document 9).

1.5. Sources of knowledge

Books:

- Autocar India
- Overdrive

Experts:

- Mr. Magendra Singh
- Mr. Vipin Sharma
- Mr. Sandeep Saini
- Mr. Kushal Verma

Websites:

- zigwheels.com/bikes
- heromotorcorp.com/two-wheeler-motorcycles

- bajajauto.com
- autos.maxabout.com
- http://www.royalenfield.com/motorcycles/motor-cycles-landing.aspx
- http://www.harley-davidson.in/harley-davidson-india-our-motorcycles.html
- http://www.yamaha-motor-india.com/product/index.html
- http://www.priceindia.in/bike/yamaha-bike-price-list/
- http://www.tvsmotor.in/index.asp#
- http://www.tvsapache.in/apache-rtr-160-specifications.html
- http://www.bmwmotorcycles.com/us/en/index.html
- http://www.infibeam.com
- http://www.bikedekho.com

1.6. Overview

Section 2 of this document describes the overview of the system in terms of general characteristics of the ontology, information about the possible users of the ontology, possible constraints on the ontology, functions of ontology and user characteristics.

2. **Overall Description**

There are various vehicles that are being used as a mode of transportation in today's world. One specific type of vehicle is a *bike*, also known as *motorcycle*. There exist many bikes enthusiasts who love bikes. Moreover, it is also one of the most popular modes of transportation which is apparent from the fact that there are millions of bikes being sold each year in India.

The ontology of bikes will be a representation of bikes in domain of ontology. The customers who wish to buy a bike can use this ontology to seek information about the optimal bike for them based on their preferences for different criterion such as price, engine capacity, type of brakes & make of the bike etc.

2.1. Ontology Functions

The ontology will store the following elements:

- Class hierarchy
- Properties
- Inverse properties
- Instances

Based on the above information, the customer can classify bikes according to the following parameters separately or in combination with each other- Make, Engine capacity, Power, Price, Fuel tank capacity, Mileage, Brake type, Weight, Wheel type, Ignition and Number of gears.

2.2. User Characteristics

Users of the system are customers. Assuming that they have very less or no knowledge of using such systems, another layer consisting of UI should be added on top of this ontology.

- Educational Qualification:
  An engineer with BE/B.Tech or equivalent at minimum.
- Experience Requirements:
  The user should have knowledge of basic characteristics of bikes.
- Technical Expertise:
  The user should have knowledge of OWL & Protégé.

2.3. Constraints

The customers will have option to only explore and search from information base consisting of information related to bikes available in India.

// INSERT Ontology Design Document HERE // (10) PAGES

Magendra Singh
M.Tech- Software Engineering
Delhi Technological University

4.6 FORMALIZATION

The ontology was formalized using the Protégé tool from Stanford University [31].

Fig.29: Class hierarchy for Ontology of Bikes

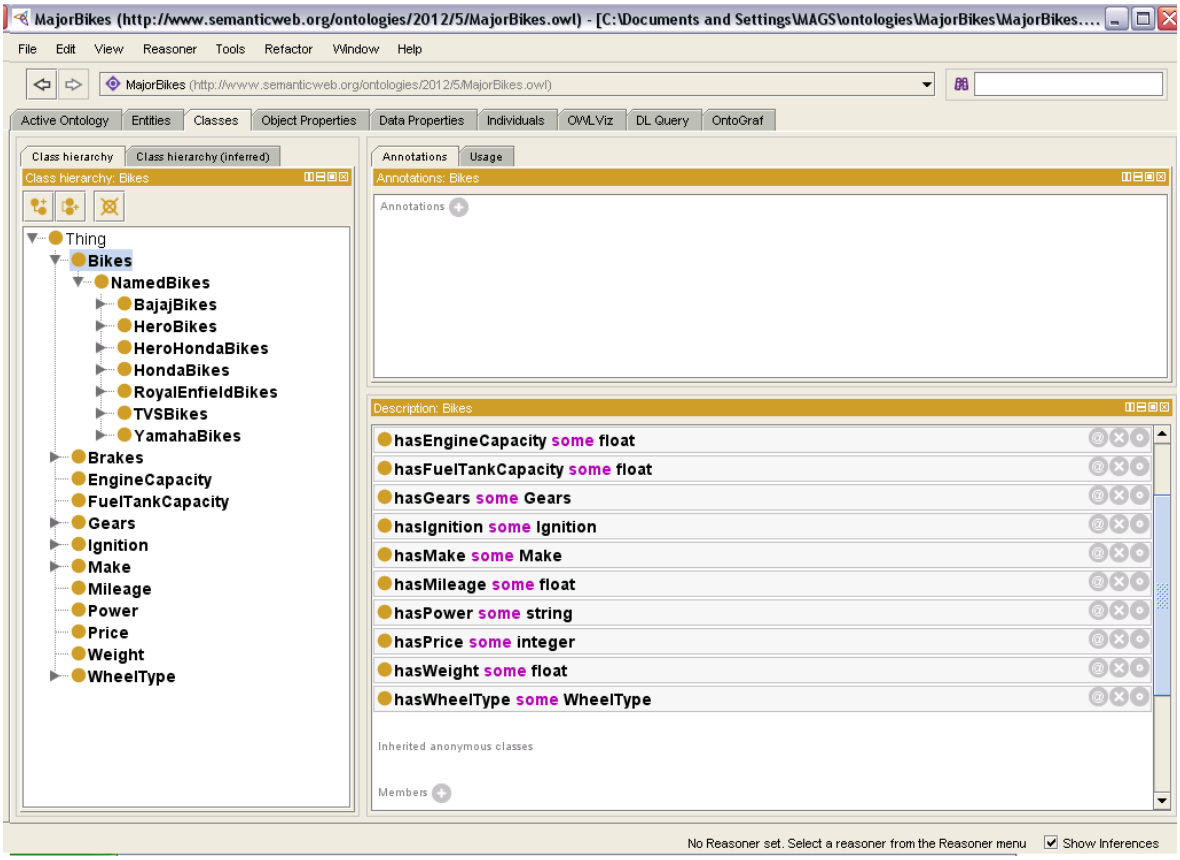


Fig. 30: Object Properties for Ontology of Bikes

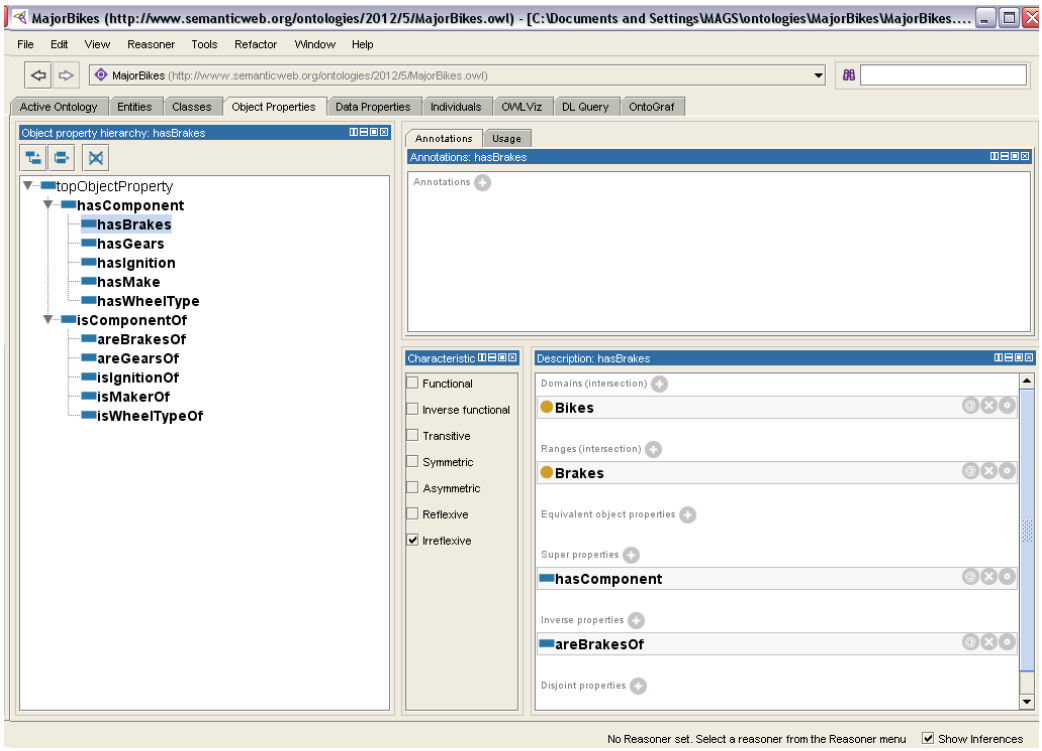


Fig. 31: Data Properties for Ontology of Bikes

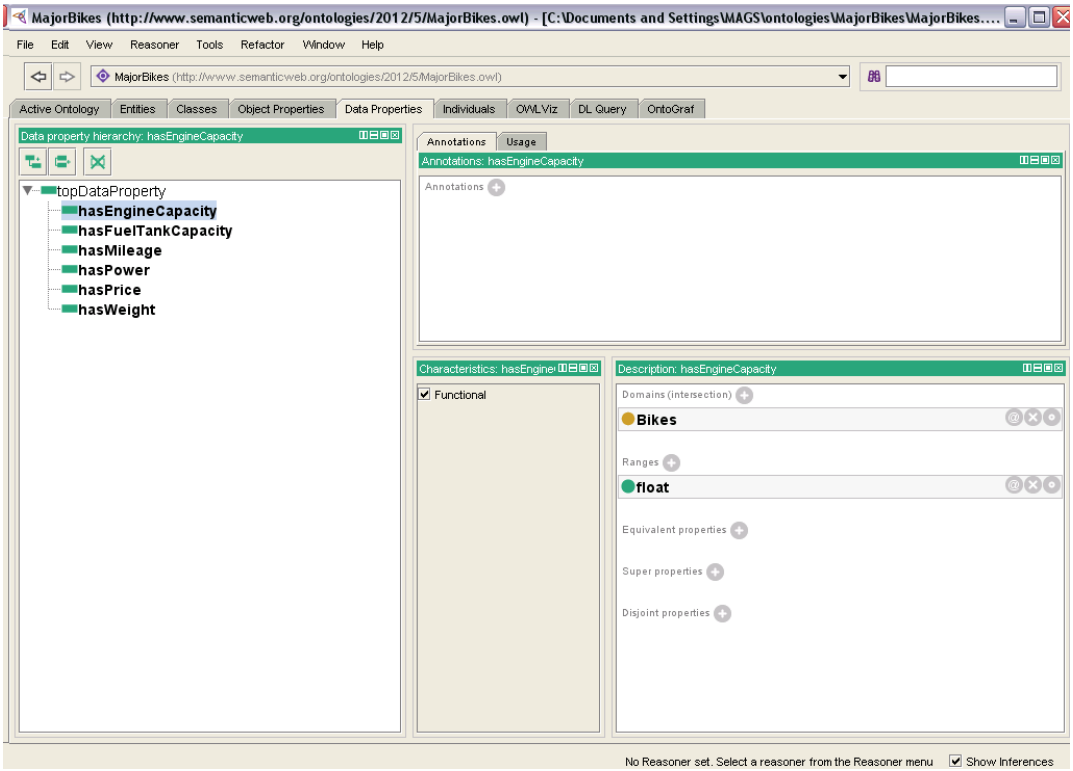


Fig. 32(a): Individuals for various Bikes- Hero Honda Karizma(ZMR)

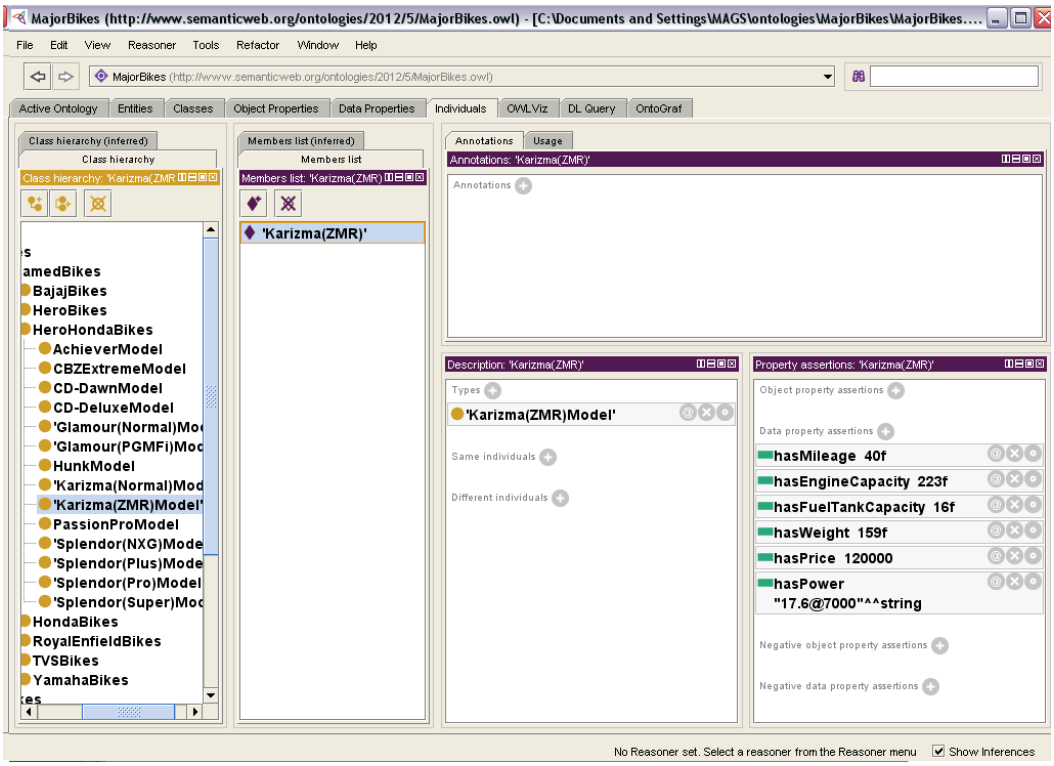
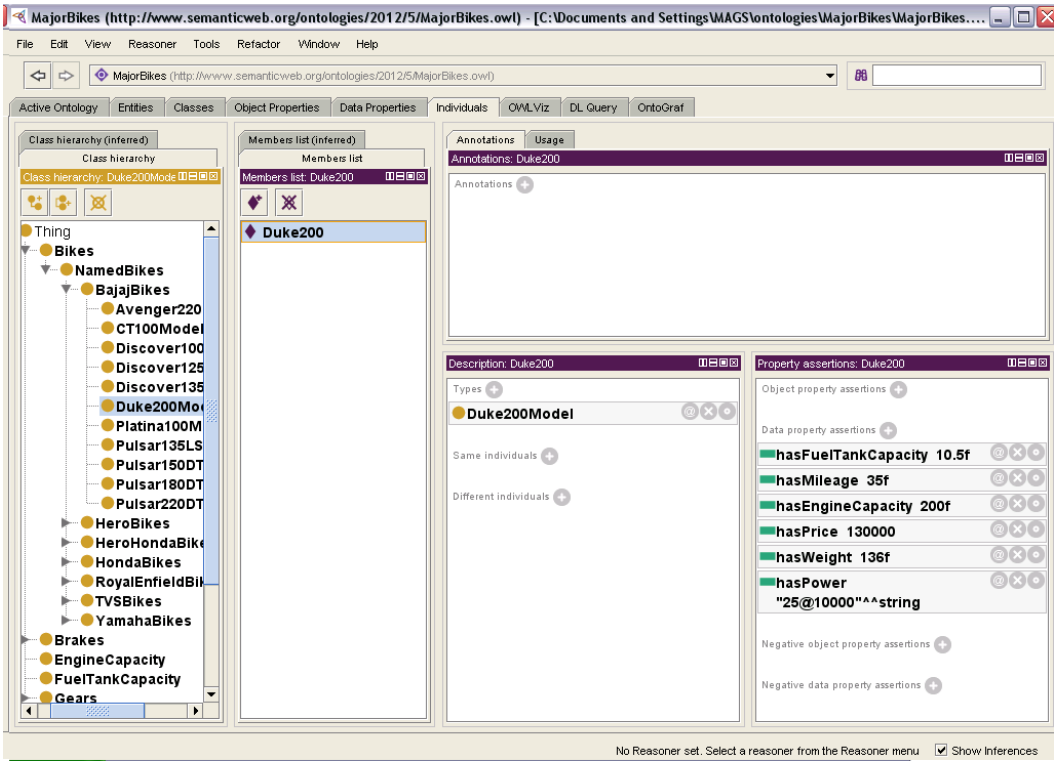


Fig. 32(b): Individuals for various Bikes- Bajaj Duke200



4.5 ONTOLOGY DESIGN DOCUMENT (ODD)

We employed our tool in this activity to design the ontology automatically. Following snapshots show the working of our tool in the designing process:

Figure 23(a): Employment of our tool for design of Ontology of Bikes

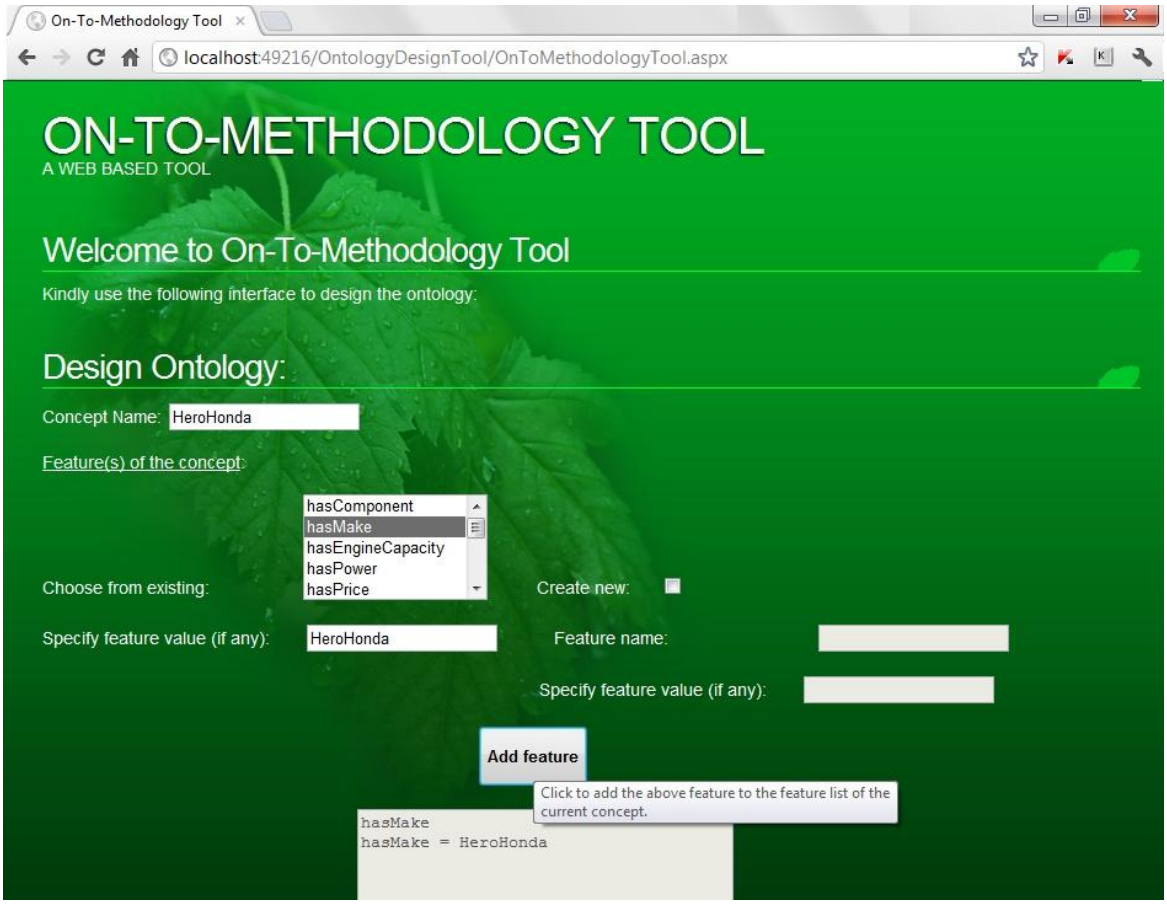
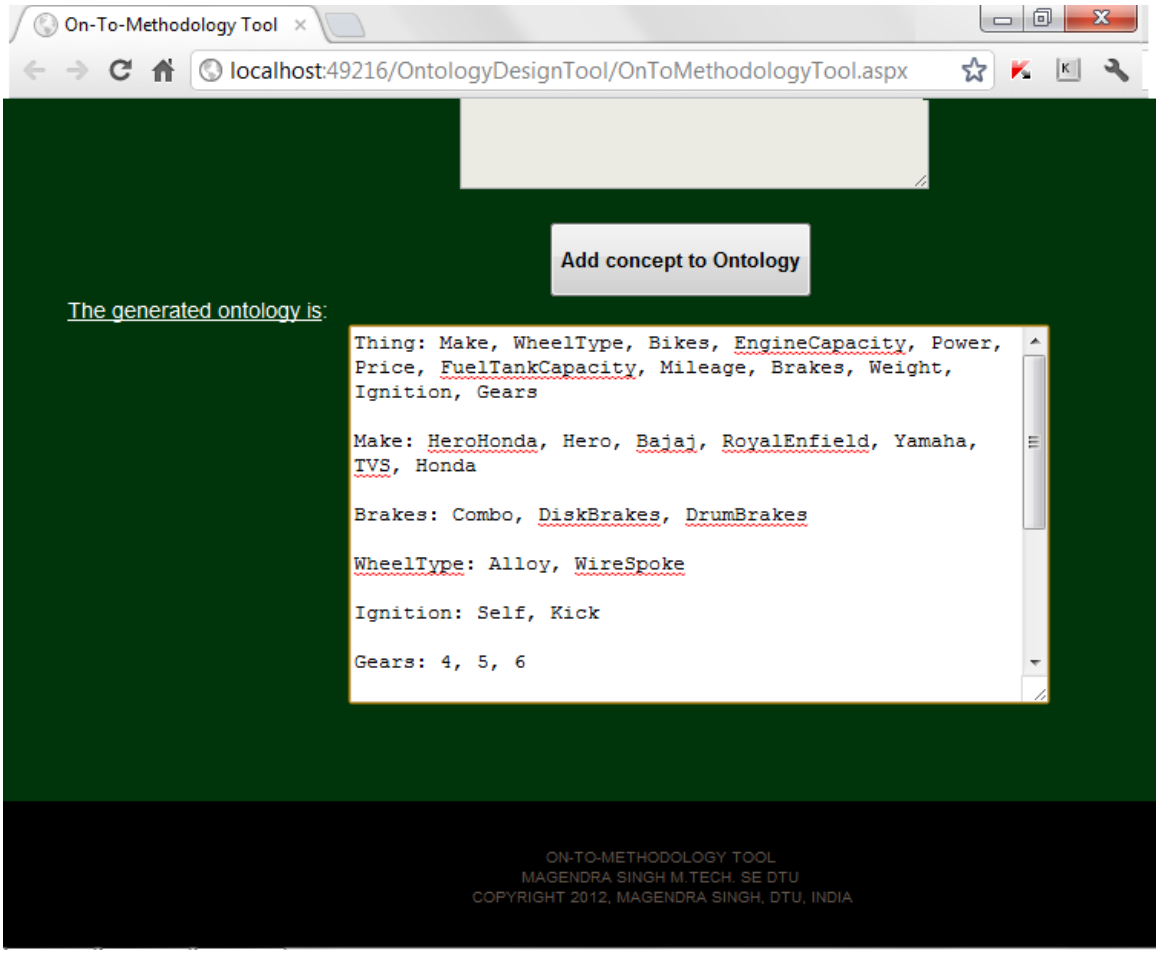


Figure 23(b): Employment of our tool for design of Ontology of Bikes



The following is Ontology Design Document for the Ontology of Bikes. *Appendix-B* illustrates the step-by-step (manual) procedure to derive the Ontology Design Document.

Table 4 : Final Location map

CONCEPT ADDRESS	CONCEPT FEATURE
Thing (0, 1)	X
Bikes (1, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
Make (1, 2)	f^+2
EngineCapacity (1, 3)	f^+3
Power (1, 4)	f^+4
Price (1, 5)	f^+5
FuelTankCapacity (1, 6)	f^+6
Mileage (1, 7)	f^+7

Brakes (1, 8)	$f^+ 8$
Weight (1, 9)	$f^+ 9$
WheelType (1, 10)	$f^+ 10$
Ignition (1,11)	$f^+ 11$
Gears (1 ,12)	$f^+ 12$
NamedBikes (2, 1)	$f^+ 1 \bullet f^+ 2 \bullet f^+ 3 \bullet f^+ 4 \bullet f^+ 5 \bullet f^+ 6 \bullet f^+ 7 \bullet f^+ 8 \bullet f^+ 9 \bullet f^+ 10 \bullet f^+ 11 \bullet f^+ 12$
HeroHonda (2, 2)	$f^+ 2 \bullet f^+ 13$
Hero (2, 3)	$f^+ 2 \bullet f^+ 14$
Bajaj (2, 4)	$f^+ 2 \bullet f^+ 15$
RoyalEnfield (2, 5)	$f^+ 2 \bullet f^+ 16$
Yamaha (2, 6)	$f^+ 2 \bullet f^+ 17$
TVS (2, 7)	$f^+ 2 \bullet f^+ 18$
Honda (2, 8)	$f^+ 2 \bullet f^+ 19$
HeroHondaBikes (3, 1)	$f^+ 2 \bullet f^+ 3 \bullet f^+ 4 \bullet f^+ 5 \bullet f^+ 6 \bullet f^+ 7 \bullet f^+ 8 \bullet f^+ 9 \bullet f^+ 10 \bullet f^+ 11 \bullet f^+ 12 \bullet f^+ 13$
HeroBikes (3, 2)	$f^+ 2 \bullet f^+ 3 \bullet f^+ 4 \bullet f^+ 5 \bullet f^+ 6 \bullet f^+ 7 \bullet f^+ 8 \bullet f^+ 9 \bullet f^+ 10 \bullet f^+ 11 \bullet f^+ 12 \bullet f^+ 14$
BajajBikes (3, 3)	$f^+ 2 \bullet f^+ 3 \bullet f^+ 4 \bullet f^+ 5 \bullet f^+ 6 \bullet f^+ 7 \bullet f^+ 8 \bullet f^+ 9 \bullet f^+ 10 \bullet f^+ 11 \bullet f^+ 12 \bullet f^+ 15$
RoyalEnfieldBikes (3, 4)	$f^+ 2 \bullet f^+ 3 \bullet f^+ 4 \bullet f^+ 5 \bullet f^+ 6 \bullet f^+ 7 \bullet f^+ 8 \bullet f^+ 9 \bullet f^+ 10 \bullet f^+ 11 \bullet f^+ 12 \bullet f^+ 16$
YamahaBikes (3, 5)	$f^+ 2 \bullet f^+ 3 \bullet f^+ 4 \bullet f^+ 5 \bullet f^+ 6 \bullet f^+ 7 \bullet f^+ 8 \bullet f^+ 9 \bullet f^+ 10 \bullet f^+ 11 \bullet f^+ 12 \bullet f^+ 17$
TVSBikes (3, 6)	$f^+ 2 \bullet f^+ 3 \bullet f^+ 4 \bullet f^+ 5 \bullet f^+ 6 \bullet f^+ 7 \bullet f^+ 8 \bullet f^+ 9 \bullet f^+ 10 \bullet f^+ 11 \bullet f^+ 12 \bullet f^+ 18$
HondaBikes (3, 7)	$f^+ 2 \bullet f^+ 3 \bullet f^+ 4 \bullet f^+ 5 \bullet f^+ 6 \bullet f^+ 7 \bullet f^+ 8 \bullet f^+ 9 \bullet f^+ 10 \bullet f^+ 11 \bullet f^+ 12 \bullet f^+ 19$
Combo (2, 9)	$f^+ 8 \bullet f^+ 20$
DiskBrakes (2, 10)	$f^+ 8 \bullet f^+ 21$
DrumBrakes (2, 11)	$f^+ 8 \bullet f^+ 22$
Alloy (2, 12)	$f^+ 10 \bullet f^+ 23$
WireSpoke (2, 13)	$f^+ 10 \bullet f^+ 24$
Self (2, 14)	$f^+ 11 \bullet f^+ 25$
Kick (2, 15)	$f^+ 11 \bullet f^+ 26$
6 (2, 16)	$f^+ 12 \bullet f^+ 27$
5 (2, 17)	$f^+ 12 \bullet f^+ 28$
4 (2, 18)	$f^+ 12 \bullet f^+ 29$
Karizma(Normal)Model (4, 1)	$f^+ 2 \bullet f^+ 3 \bullet f^+ 4 \bullet f^+ 5 \bullet f^+ 6 \bullet f^+ 7 \bullet f^+ 8 \bullet f^+ 9 \bullet f^+ 10 \bullet f^+ 11 \bullet f^+ 12 \bullet f^+ 13 \bullet f^+ 20 \bullet f^+ 23 \bullet f^+ 25 \bullet f^+ 28$
Karizma(ZMR)Model (4, 2)	$f^+ 2 \bullet f^+ 3 \bullet f^+ 4 \bullet f^+ 5 \bullet f^+ 6 \bullet f^+ 7 \bullet f^+ 8 \bullet f^+ 9 \bullet f^+ 10 \bullet f^+ 11 \bullet f^+ 12 \bullet f^+ 13 \bullet f^+ 21 \bullet f^+ 23 \bullet f^+ 25 \bullet f^+ 28$
Splendor(Plus)Model (4, 3)	$f^+ 2 \bullet f^+ 3 \bullet f^+ 4 \bullet f^+ 5 \bullet f^+ 6 \bullet f^+ 7 \bullet f^+ 8 \bullet f^+ 9 \bullet f^+ 10 \bullet f^+ 11 \bullet f^+ 12 \bullet f^+ 13 \bullet f^+ 22 \bullet f^+ 24 \bullet f^+ 26 \bullet f^+ 29$
Splendor(NXG)Model (4, 4)	$f^+ 2 \bullet f^+ 3 \bullet f^+ 4 \bullet f^+ 5 \bullet f^+ 6 \bullet f^+ 7 \bullet f^+ 8 \bullet f^+ 9 \bullet f^+ 10 \bullet f^+ 11 \bullet f^+ 12 \bullet f^+ 13 \bullet f^+ 23 \bullet f^+ 23 \bullet f^+ 26 \bullet f^+ 29$
Splendor(Super)Model (4, 5)	$f^+ 2 \bullet f^+ 3 \bullet f^+ 4 \bullet f^+ 5 \bullet f^+ 6 \bullet f^+ 7 \bullet f^+ 8 \bullet f^+ 9 \bullet f^+ 10 \bullet f^+ 11 \bullet f^+ 12 \bullet f^+ 13 \bullet f^+ 22 \bullet f^+ 23 \bullet f^+ 25 \bullet f^+ 29$
Splendor(Pro)Model (4, 6)	$f^+ 2 \bullet f^+ 3 \bullet f^+ 4 \bullet f^+ 5 \bullet f^+ 6 \bullet f^+ 7 \bullet f^+ 8 \bullet f^+ 9 \bullet f^+ 10 \bullet f^+ 11 \bullet f^+ 12 \bullet f^+ 13 \bullet f^+ 22 \bullet f^+ 23 \bullet f^+ 25 \bullet f^+ 29$
PassionProModel (4, 7)	$f^+ 2 \bullet f^+ 3 \bullet f^+ 4 \bullet f^+ 5 \bullet f^+ 6 \bullet f^+ 7 \bullet f^+ 8 \bullet f^+ 9 \bullet f^+ 10 \bullet f^+ 11 \bullet f^+ 12 \bullet f^+ 13 \bullet f^+ 20 \bullet f^+ 23 \bullet f^+ 25 \bullet f^+ 29$
CD-DawnModel (4, 8)	$f^+ 2 \bullet f^+ 3 \bullet f^+ 4 \bullet f^+ 5 \bullet f^+ 6 \bullet f^+ 7 \bullet f^+ 8 \bullet f^+ 9 \bullet f^+ 10 \bullet f^+ 11 \bullet f^+ 12 \bullet f^+ 13 \bullet f^+ 22 \bullet f^+ 24 \bullet f^+ 26 \bullet f^+ 29$
CD-DeluxeModel (4, 9)	$f^+ 2 \bullet f^+ 3 \bullet f^+ 4 \bullet f^+ 5 \bullet f^+ 6 \bullet f^+ 7 \bullet f^+ 8 \bullet f^+ 9 \bullet f^+ 10 \bullet f^+ 11 \bullet f^+ 12 \bullet f^+ 13 \bullet f^+ 22 \bullet f^+ 23 \bullet f^+ 26 \bullet f^+ 29$

Magendra Singh
M.Tech- Software Engineering
Delhi Technological University

Glamour(Normal)Model (4, 10)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+29$
Glamour(PGMFi)Model (4, 11)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+29$
AchieverModel (4, 12)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+28$
CBZXtremeModel (4, 13)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+28$
HunkModel (4, 14)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+21 \bullet f^+23 \bullet f^+25 \bullet f^+28$
ImpulseModel (4, 15)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+14 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+28$
CT100Model (4, 16)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15 \bullet f^+22 \bullet f^+24 \bullet f^+26 \bullet f^+29$
Pulsar135LSModel (4, 17)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+28$
Pulsar150DTS-iModel (4, 18)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+28$
Pulsar180DTS-iModel (4, 19)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+28$
Pulsar220DTS-iModel (4, 20)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15 \bullet f^+21 \bullet f^+23 \bullet f^+25 \bullet f^+28$
Avenger220DTS-iModel (4, 21)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+28$
Discover135Model (4, 22)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+29$
Discover125Model (4, 23)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15 \bullet f^+22 \bullet f^+23 \bullet f^+25 \bullet f^+28$
Discover100Model (4, 24)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15 \bullet f^+22 \bullet f^+23 \bullet f^+25 \bullet f^+29$
Platina100Model (4, 25)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15 \bullet f^+22 \bullet f^+23 \bullet f^+26 \bullet f^+29$
Duke200Model (4, 26)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15 \bullet f^+21 \bullet f^+23 \bullet f^+25 \bullet f^+27$
BulletElectraTwinspark Model (4, 27)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+16 \bullet f^+20 \bullet f^+24 \bullet f^+25 \bullet f^+28$
Bullet 350 Twinspark Model (4, 28)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+16 \bullet f^+20 \bullet f^+24 \bullet f^+25 \bullet f^+28$
Bullet Electra EFI Model (4, 29)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+16 \bullet f^+20 \bullet f^+24 \bullet f^+25 \bullet f^+28$
Bullet Electra Deluxe Model (4, 30)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+16 \bullet f^+20 \bullet f^+24 \bullet f^+25 \bullet f^+28$
Royal Enfield Classic 500 Model (4, 31)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+16 \bullet f^+20 \bullet f^+24 \bullet f^+25 \bullet f^+28$
Royal Enfield Classic 350	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+$

Magendra Singh
M.Tech- Software Engineering
Delhi Technological University

Model (4, 32)	16 • f^+20 • f^+24 • f^+25 • f^+28
Thunderbird Twinspark Model (4, 33)	f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12 • f^+16 • f^+20 • f^+24 • f^+25 • f^+28
R15Model (4, 34)	f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12 • f^+17 • f^+21 • f^+23 • f^+25 • f^+27
FZModel (4, 35)	f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12 • f^+17 • f^+20 • f^+23 • f^+25 • f^+28
VictorModel (4, 36)	f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12 • f^+18 • f^+22 • f^+24 • f^+26 • f^+29
CBRModel (4, 37)	f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12 • f^+19 • f^+21 • f^+23 • f^+25 • f^+27
ShineModel (4, 38)	f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12 • f^+19 • f^+22 • f^+23 • f^+25 • f^+29
UnicornModel (4, 39)	f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12 • f^+19 • f^+20 • f^+23 • f^+25 • f^+28

Fig. 24: Bike Ontology Structure

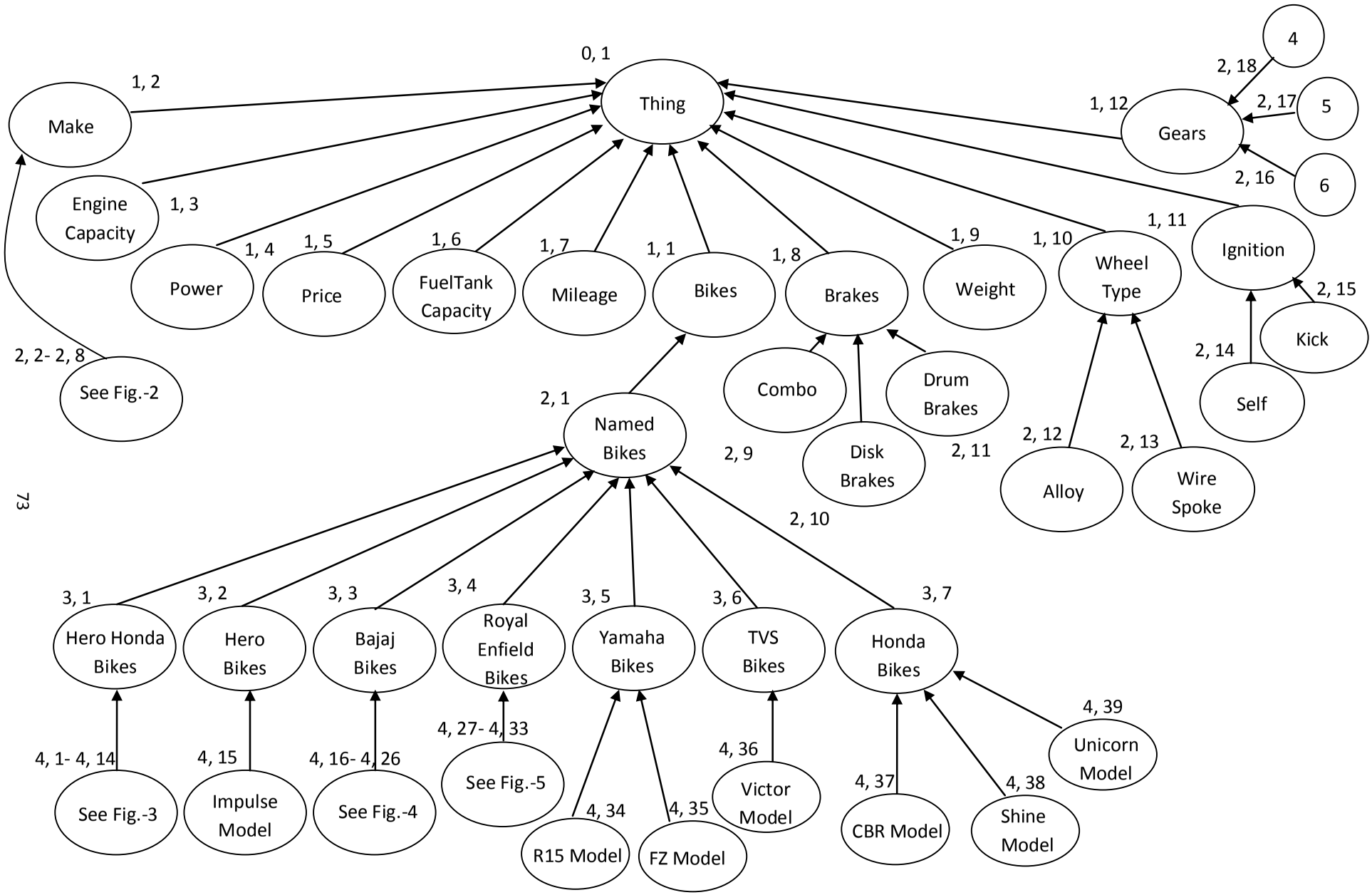


Fig. 25: Sub Ontology Structure of *HeroHondaBikes*

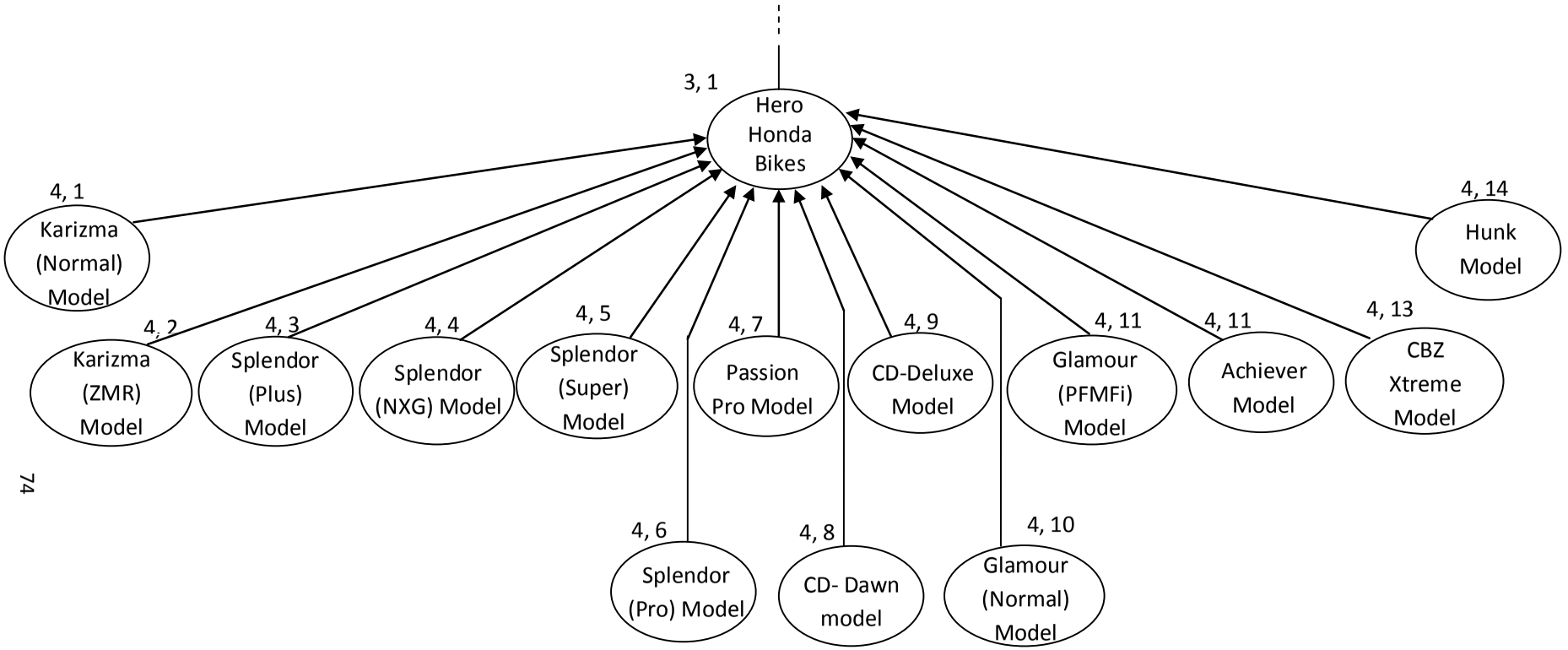


Fig. 26: Sub Ontology Structure of *BajajBikes*

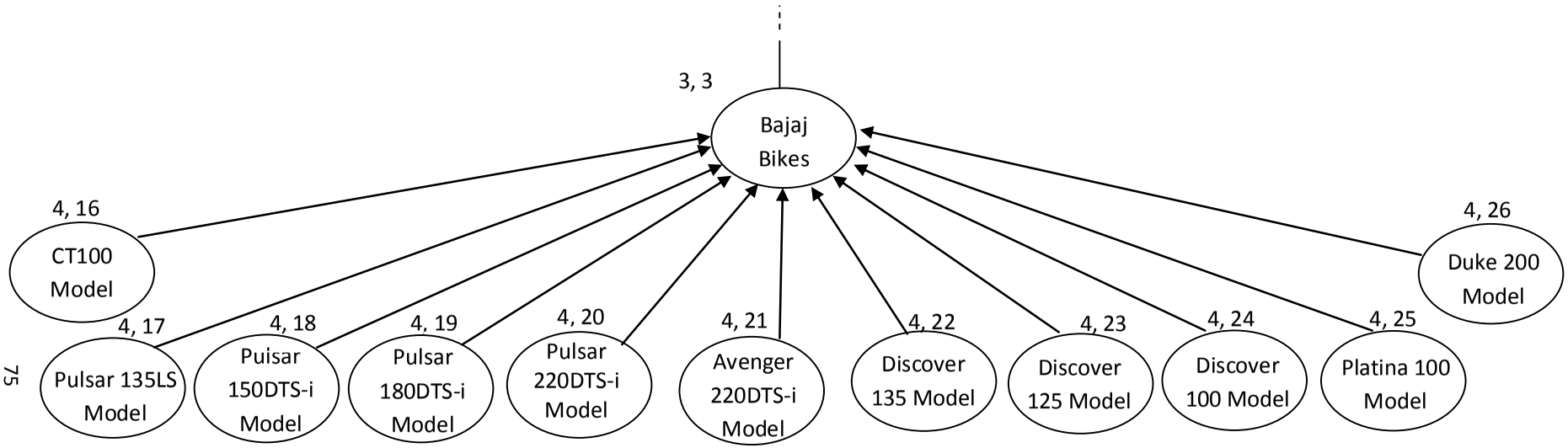


Fig. 27: Sub Ontology Structure of *Make*

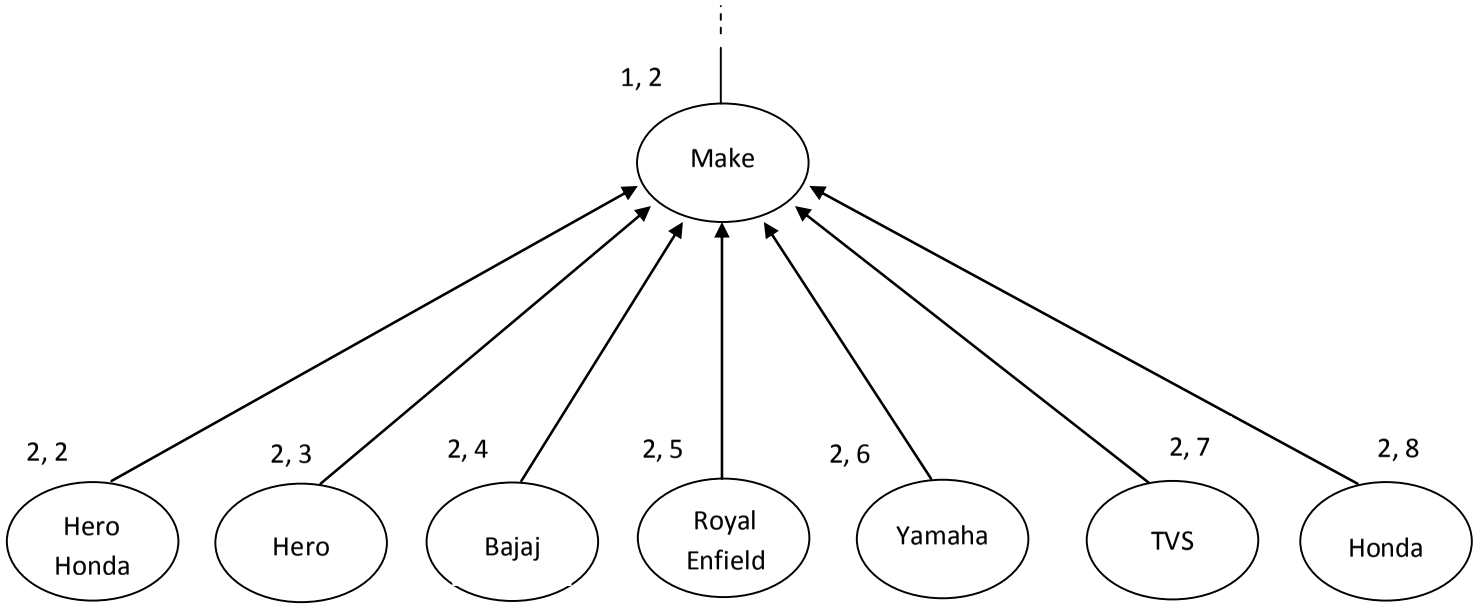
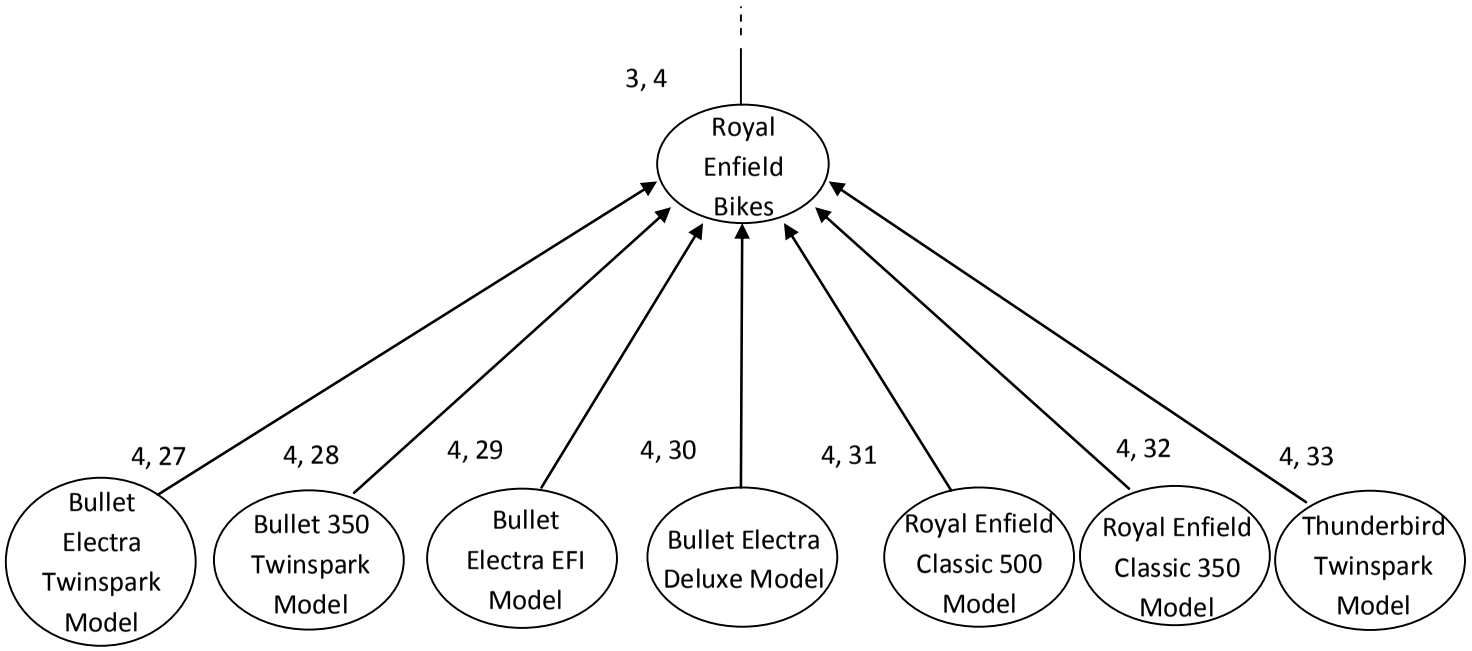


Fig. 28: Sub Ontology Structure of *RoyalEnfieldBikes*



Chapter 5

Evaluation & Comparison

5.1 Evaluation

The developed ontology was evaluated by experts by querying it with competency questions and verifying the results.

The fig.- 33(a) shows the following query:

" Average >=40 & price should be between Rs. 50000 & Rs. 60000, and engine capacity should be >=125"

The fig.- 33(b), shows the results after execution of the *Resoner*, the results were verified by the experts and were found to be consistent.

Fig.33(a): Evaluation of competency question

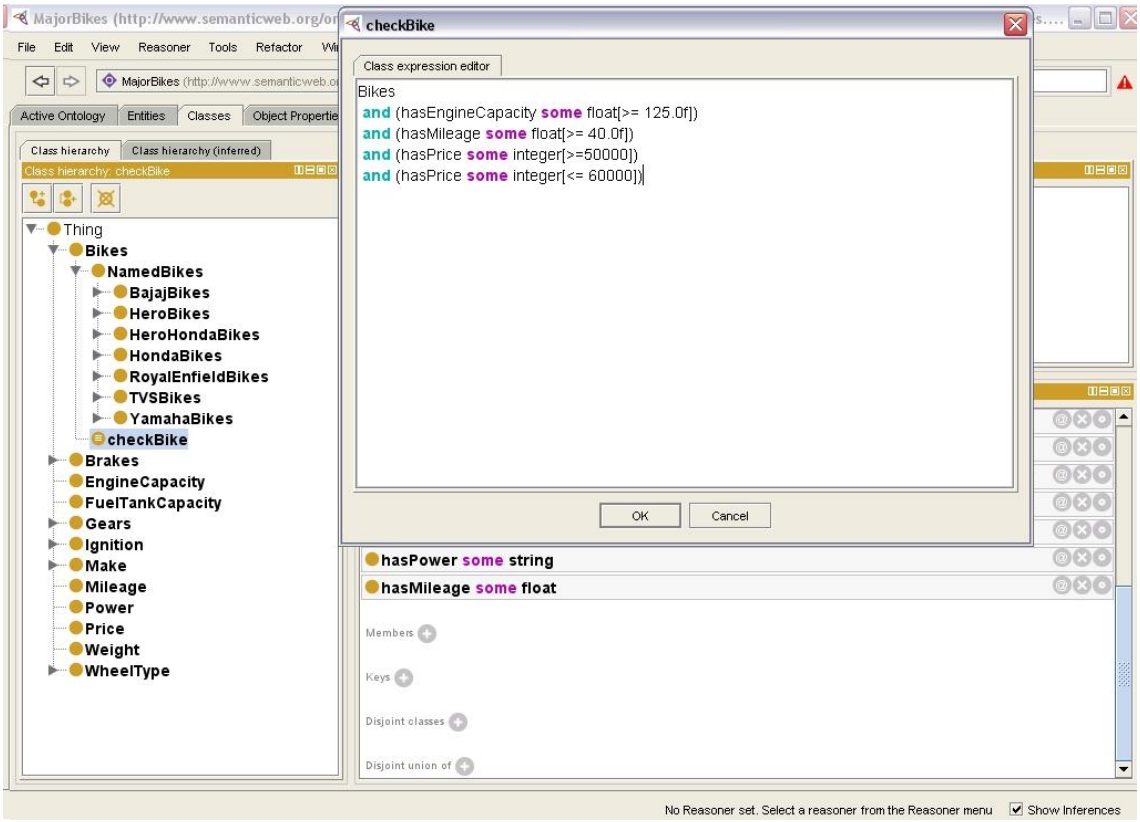
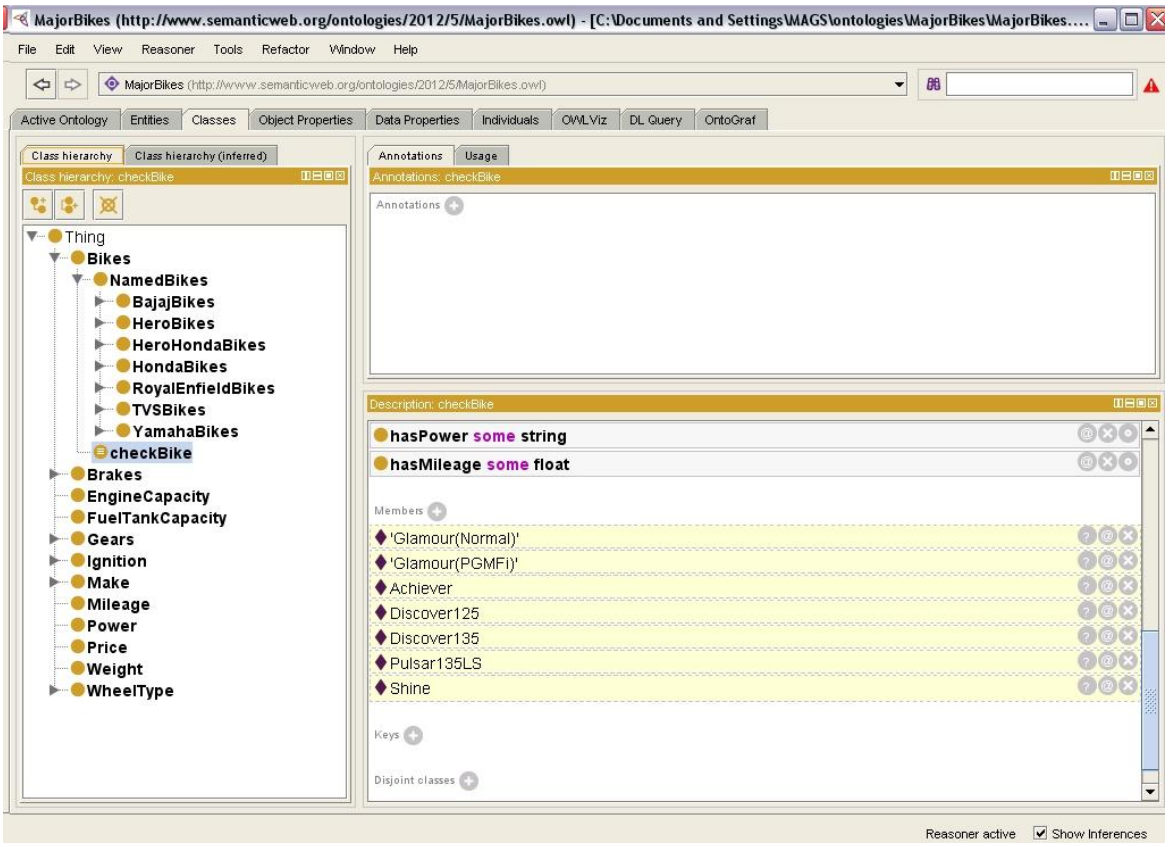


Fig. 33(b): Evaluation of competency question



The hierarchy was also evaluated by experts for created class and individual properties by using ontology tool (like OntoGraf & OWLViz). These tools infer and create Ontology Graph automatically and thus help to judge whether subordination between the class and individual in the graph are coincident. The fig.:34 shows the output of OntoGraph and fig.: 35(a), 35(b) shows the output of OWLViz.

Fig. 34: The output of OntoGraf

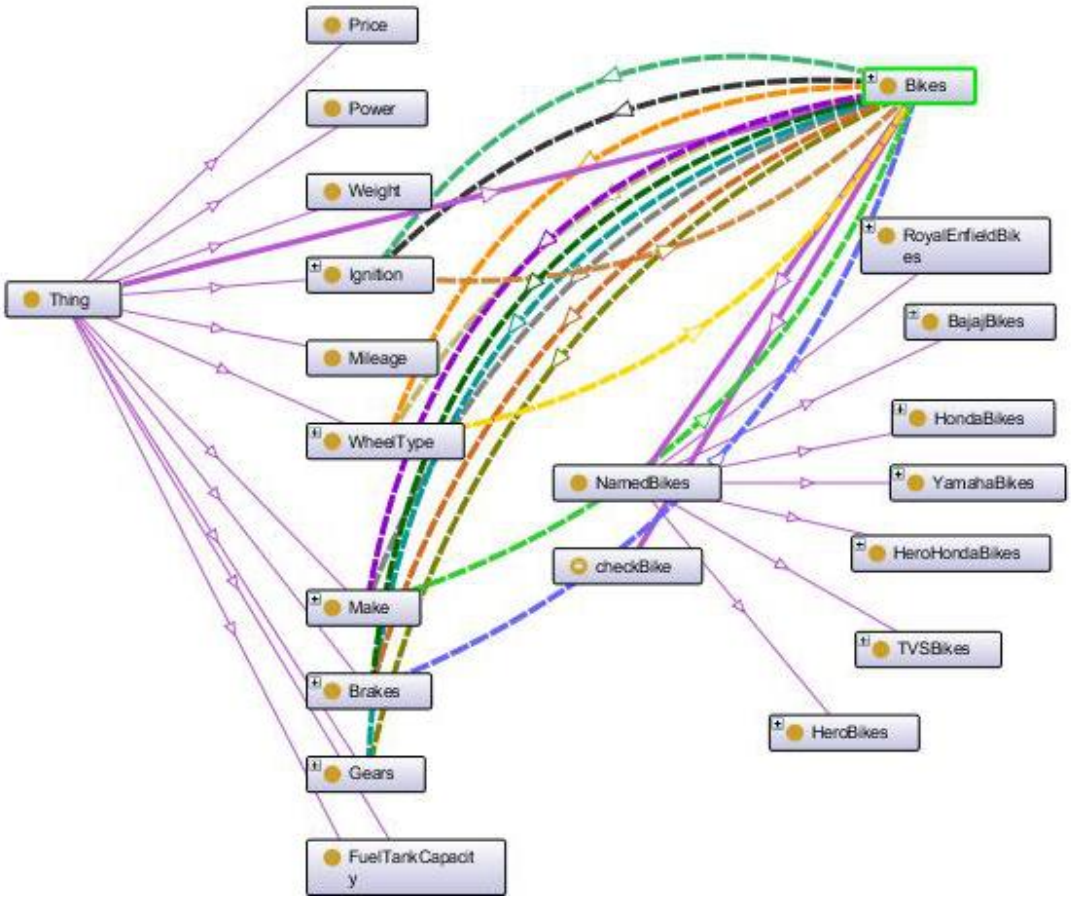


Fig. 35(a): The output of OWLViz

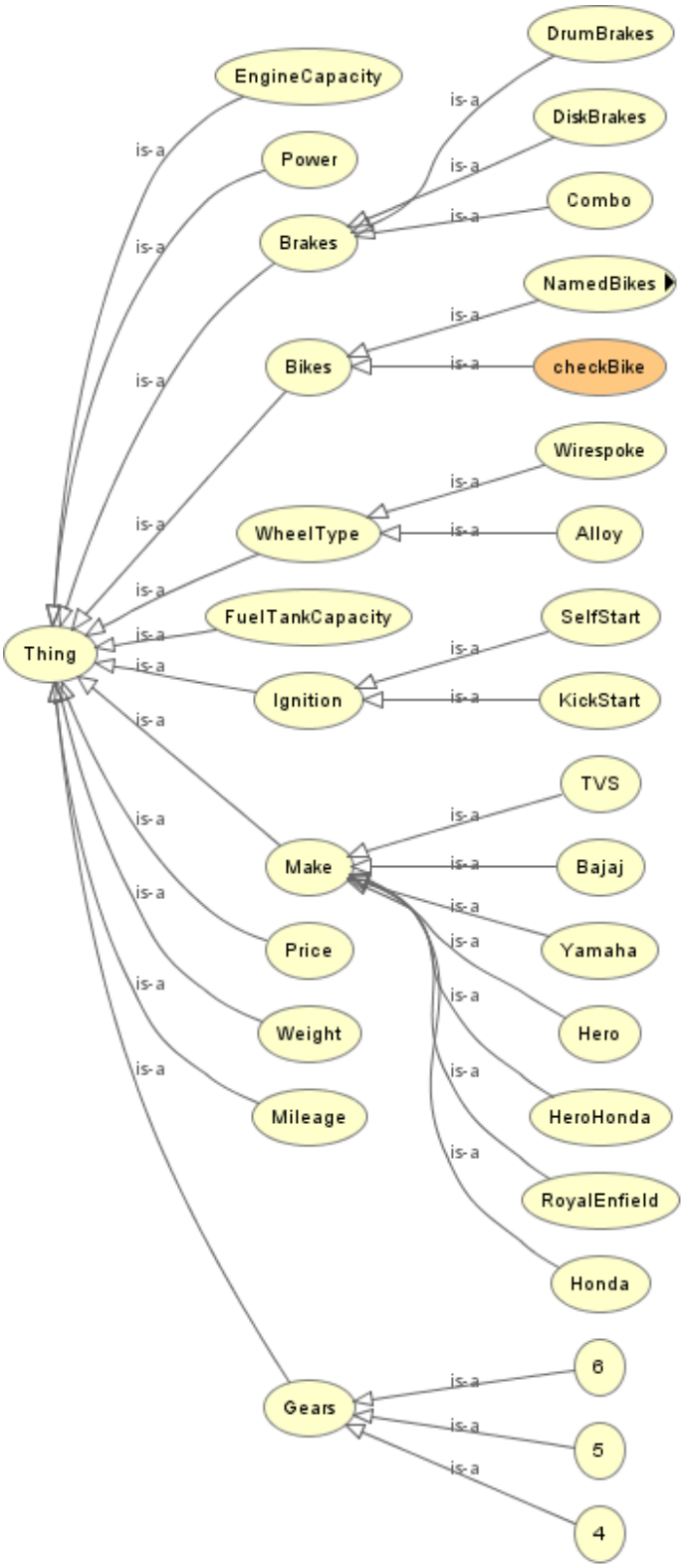
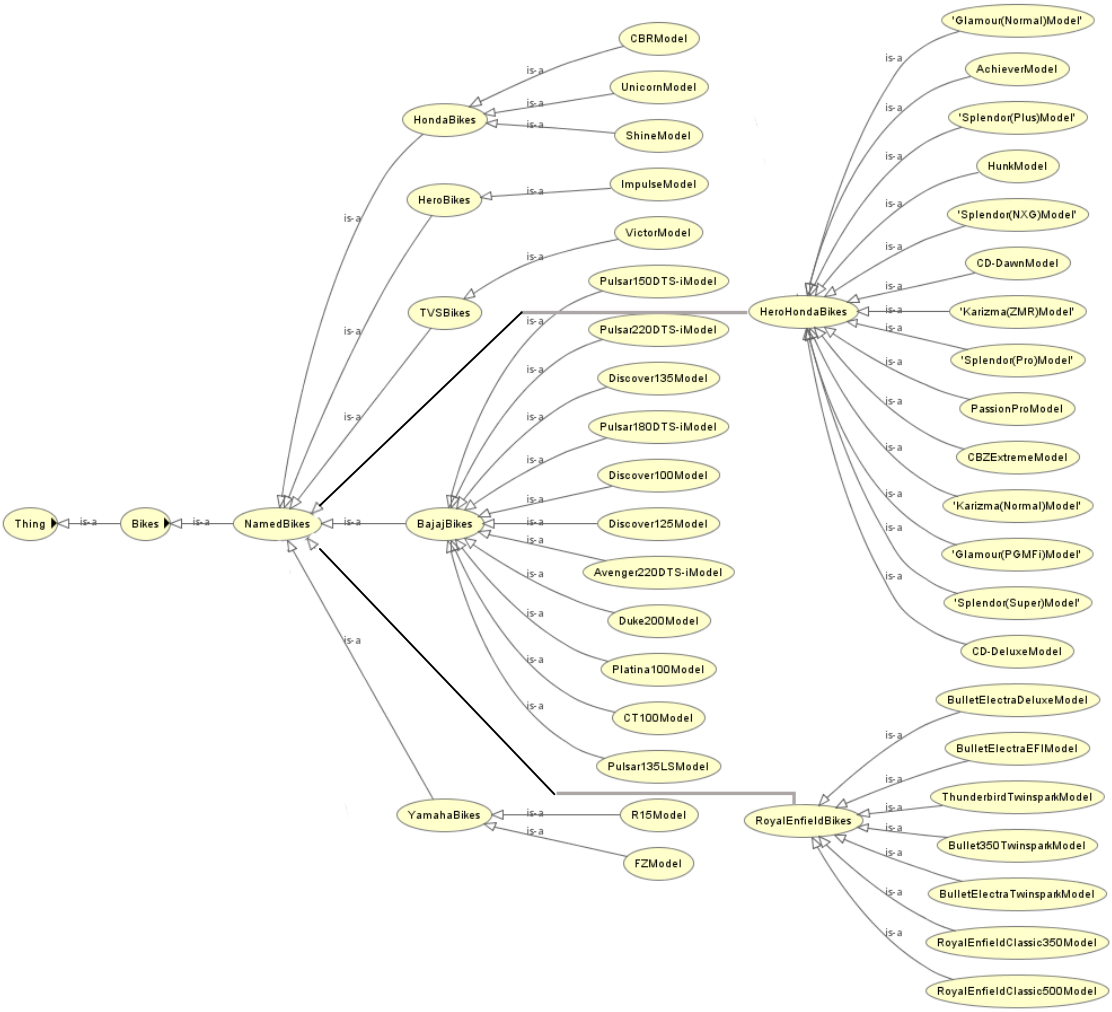


Fig. 35(b): The output of OWLViz



5.2 **Comparison of Ontology Engineering Methods:**

Our methodology namely *On-To-Methodology* overcomes the various problems that are left unaddressed by existent methodologies. The following are the advantages that *On-To-Methodology* offers:

- *Better domain cognition*

  As listed in Section 1.3, there exists inconsistency in domain cognition. Our method handles this problem by adapting group oriented activities thus enabling experts and users to define the domain in better terms.

- *Unification of viewpoints*

  Inconsistency in viewpoints arises as different people have different perspectives and due to this contrary conclusions may be obtained. Our methodology resolves this problem by instrumenting the development process with evaluation activities after each phase of the development process. Also, the unification is achieved due to the adaptation of group oriented development activities.

- *Exhaustive documentation*

  On-To-Methodology provides an extensive and exhaustive documentation support thus making the development process more formal and helps in achieving traceability across phases. Documentation also simplifies the maintenance of the ontology by providing the backward traceability of various components of the ontology.

- *Extensive Verification & Validation*

  These two tasks of verification and validation are interleaved between various phases of the development process. This prevents introduction of bugs into the system right from the beginning of the development process and prevents faults till the maintenance phase.

Table- 5 presents the mapping of main concepts of our On-To-Methodology to notions used by previously presented Ontology engineering methods.

Chapter 6

CONCLUSION & FUTURE WORK

The work carried out in this report provides with strong base in Ontology Engineering. Here we have presented the theories and principles for construction of ontology according to various currently existent methodologies. We also noted the various issues related to the field of ontology development.

Since there exists no one methodology that can be applied in all scenarios of ontology development, our main emphasis here is on building a methodology that can be used as a standard method for ontology construction under various domains. For this we have proposed a method namely *On-To-Methodology*. Then we illustrated our methodology by working out an Ontology of bikes. We also subjectively compared our methodology with various other existent methodologies.

The future work entails development of a tool that will automatically generate the Conceptual model of the knowledge. Another future direction is to develop software that would work on the ontology of Bikes that we created and has simpler graphical user interface so that it could be operated by non-technical users.

Chapter 7

REFERENCES

[1] Corcho O., Fernández-López M., Gómez-Pérez A., "Ontological Engineering: What are ontologies and How can we build them?"

[2] William Swartout, "Ontologies" USC Information Sciences Institute Austin Tate, Artificial Intelligence Applications Institute, University of Edinburgh

[3] Guarino, N., Formal ontology in information systems. 1998, IOS Pr.

[4] Borst, P., H. Akkermans,and J. Top, "Engineering ontologies", International Journal of Human Computer Studies, 1997. 46: p. 365-406.

[5] Gruber T R., "A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition", 1993,5(2): 199-220

[6] Richard Cyganiak, Holger Stenzhorn, Renaud Delbru, Stefan Decker, and Giovanni Tummarello, Semantic Sitemaps: Efficient and Flexible Access to Datasets on the Semantic Web. In Proc. of ESWC,2008.

[7] Kitamura, Y., Takafuji, S., Mizoguchi, R., Towards a Reference Ontology for Functional knowledge InteroperabiIity, In Proc. of ASME IDETC/CIE 2007, DETC2007-35473, to appear.

[8] N. F. Noy, and D. L. McGuinness, "Ontology Development 101: A guide to creating your first ontology", Technical Report SMI-2001-0880, Stanford Medical Informatics, 2001

[9] Fensel D, "Ontologies: Silver Bullet for Knowledge Management and Electronic Commence", 2001, SpringerVerlag

[10] Gruber T R, 1993, "Towards principles for the design of ontologiesused for knowledge sharing", International Workshop on Formal Ontology in Conceptual analysis and Knowledge Representation, eds. G N & P R, Kluver Academic Publishers, Deventer, The Netherlands, Padova, Italy

[11] Dillion T S, Chang E, Wongthonghtam P, “Ontology-based Software Engineering-Software Engineering 2.0”, 2008 19th Australian Conference on Software Engineering IEEE, 13-23

[12] Zhang Qing,Tian Yuexin, "Requirements capturement and analysis",In China Science and Technology Information,Feb,2009:1-2.

[13] Neches R, Fikes R E, Gurber T R, etal. Enabling Technology for Knowledge Sharing [J]. AI Magazine, 1999, 12（3）:36-56

[14] Yao-Tang Yu, Chien-Chang Hsu, “A structured ontology construction by using data clustering and pattern tree mining”, Proceedings of the 2011 International Conference on Machine Learning and Cybernetics, Guilin, 10-13 July, 2011, 45-50

[15] Chen Y J, Chen Y M, Huang C Y, Chao C Y, “Concept Feature-based Ontology Construction and Maintenance”, 2010 IEEE, 28-32

[16] Yang-ying Y, Zong-yong L, Zhi-Xue Wang, “Domain Knowledge Consistency Checking for Ontology-based Requirement Engineering”, 2008 International Conference on Computer Science and Software Engineering, 302-305

[17] Shunxin L, Leijun S, “Requirements Engineering Based on Domain Ontology”, 2010 International Conference of Information Science and Management Engineering, 120-122

[18] Zhao Y, Dong J, Peng T, “Ontology Classification for Semantic-Web-Based Software Engineering”, IEEE Transactions On Services Computing, vol. 2, no. 4, October-December 2009, 303-317

[19] Gaoyun J, Jianliang W, Shaohua Y, “A Method for Consistent Ocean Ontology Construction”, IEEE 201O 2nd International Conference on Industrial and Information Systems, 37-42

[20] Hepp M, “Possible Ontologies- How Reality Constrains the Development of Relevant Ontologies”, 2007

[21] Fernández M, Gómez-Pérez A, Juristo N, "METHONTOLOGY: From Ontological Art Towards Ontological Engineering", AAAI Technical Report SS-97-06, available at *http://www.aaai.org/Papers/Symposia/Spring/1997/SS-97-06/SS97-06-005.pdf*

[22] Schreiber G, Wielinga B, Hoog R, "CommonKADS: A Comprehensive Methodology for KBS Development", December 1994 IEEE, 28-36

[23] Kim J A, Choi S Y, "Identifying the process capability of the ontology development processes", 2010, 9th IEEE/ACIS International Conference on Computer and Information Science, 708-713

[24] Farooq A, Shah A, Asif K H, "Design of Ontology in Semantic Web Engineering Process"

[25] Zhang Y, tan L, Liu J, Yu CC, "A Domain Ontology Construction Method Towards Healthy Housing", 2010 IEEE

[26] Dan Z, Li Z, Hao J, "Research on Semi-automatic Domain Ontology Construction", 2010 Seventh Web Information Systems and Applications Conference, 115-118

[27] Jia M, Zheng D, Liu L, Yang B, Sun W, Yang J, "Automatic Ontology Construction Approaches and its Application on Military Intelligence", 2009 Asia-Pacific Conference on Information Processing, 348-351

[28] Gupta D, Jat K, "Knowledge representation with Ontology and Relational database to RDF converter"

[29] Wei QUI, "Development and Application of Knowledge Engineering Based on Ontology", 2010 Third International Conference on Knowledge Discovery and Data Mining, 518-521

[30] Islam N, Siddiqui MS, Shaikh ZA, "TODE : A Dot Net Based Tool for Ontology Development and Editing", 2010 2nd International Conference on Computer Engineering and Technology.

[31] http://protege.stanford.edu/ : Protégé tool from Stanford University

Table 5: Mapping of concepts between On-To-Methodology & other methodologies

Variable Name	On-To-Methodology	METHONTOLOGY	Methodology by Farooq et. al.	Methodology by Gaoyun et. al.	MADRE	Automatic ontology construction approach by Jia et. al.	Concept feature based Ontology Construction and Maintenance	Structured Ontology Construction by using data clustering & Maintenance
TYPE	Manual	Manual	Manual	Manual	Manual	Automatic	Automatic	Semi Automatic
Documentation	✔	✓	X	X	X	X	X	✓
Validation/ Verification	✔/✔	✓/✓	✓/✓	X/✓	X/✓	✓/X	✓/X	--
	Define Scope	✓ ≡Specification	✓ ≡Adaptation at Specification level	✓ ≡ Clarifying domain	✓ ≡ Domain Analysis	--	--	--
	Knowledge Acquisition: Domain Understanding & Knowledge Elicitation	✓ ≡Knowledge Acquisition	✓ ≡Domain vocabulary declaration	✓	✓ ≡Ontology Knowledge Acquisition	✓ ≡Extraction from Free Text + Dictionary	✓	✓ ≡Relational analysis + Clustering (Ontology construction)
	Knowledge Acquisition: Building Conceptual Model of the Knowledge	✓ Conceptualize	✓ ≡ Identifying and grouping resources + Identifying axioms, relationships, data-characteristics and naming them	✓	✓ ≡Phase of Ontology analysis & design (Construct Ontology Model)	✓ ≡Extraction from Knowledge Base	✓	✓ ≡Concept analysis (Ontology Construction)
	Analysis & Validation	X	✓ ≡Verification	--	X	✓ ≡Extract Relational patterns	X	--
	Design Ontology	✓ ≡Integration	✓ ≡Adaptation at design level	✓ ≡Identifying attributes	✓ ≡Phase of Ontology analysis & design		--	✓ ≡ System architecture)

	Formalization	✓ ≡ Implementation		✓	✓ ≡Phase of Ontology representation	✓ ≡Sub-ontology Extraction	--	--
	Evaluation	✓	X	X	✓	X	✓	--
	Maintenance	X	X	X	X	X	✓	X

APPENDIX A

A.1 **DOCUMENT: 2.1** Date: ……

Name: ……

Educational Qualification: ……

Organization:

……

……

Domain corpus consisting of various keywords:

Engine	Splendor
Brakes	Pulsar (135/150/180/200)
Tires	Passion
Fuel Tank capacity	Victor
Price	R15
Seat type (Split/Single/Step/Normal)	FZ
Bullet350	Harley Davidson
Bullet500	CBR
Thunderbird	Ninja
Karizma (Normal/ZMR)	
Royal Enfield	Harley Davidson
Hero Honda	Yamaha
Bajaj	Kawasaki
TVS	Make

Zigwheels.com/Bikes:

Swish125

Suzuki

Mountain Bikes

Heromotocorp.com/two-wheeler-motorcycles:

Impulse	Glamour (Normal/PGMFi)
CD-Dawn	Achiever
CD-Deluxe	CBZ Xtreme
Pleasure	Hunk
Splendor (Plus/NXG/Super/Pro)	Hero
Passion Pro	

Bajajauto.com:

Avenger 220 DTS-i	Discover 125
Pulsar 135 LS	Discover 100
Pulsar 150 DTS-I	Platina 100
Pulsar 180 DTS-I	Ninja 250R
Pulsar 220 DTS-I	Ninja 650R
Discover 150	Duke

Autos.maxabout.com/bikes/ktm/duke/990-super-duke:

Weight	0-60
Wheel type	Fuel Tank capacity
Gears	

Date: …./…./……..

Signature: ……..

( )

A.2 **DOCUMENT: 2.2** Date: ……

Name: ……

Educational Qualification: ……

Organization:

……

……..

Domain corpus consisting of various keywords:

……	……
……	……
……	……
……	……
……	……
……	……
……	……
……	……
……	……
……	……

http://www.royalenfield.com/motorcycles/motor-cycles-landing.aspx:

Royal Enfield Classic	Bullet 350 Twinspark
Royal Enfield Classic 500	Bullet Electra EFI
Royal Enfield Classic 350	Bullet Electra Deluxe
Thunderbird Twinspark	Bullet 500
Bullet Electra Twinspark	

http://www.harley-davidson.in/harley-davidson-india-our-motorcycles.html :

Superlow (560000, 883)	Nightster (1202,110000)
Iron883 (883, 660000)	XR1200X(1200,121000)
Roadster (883, 765000)	StreetBob(1010000,1585)
Forty-Eight (1202,8,65000)	SuperglideCustom (1165000,1585)

Comments (if any):

……

……

……

……

……

……

……..

Date: …./…./……..

Signature: ……..

( )

A.3 **DOCUMENT: 2.3** Date: ……

Name: ……

Educational Qualification: ……

Organization:

……

……

Domain corpus consisting of various keywords:

……	……
……	……
……	……
……	……
……	……
……	……
……	……
……	……
……	……
……	……

http://www.yamaha-motor-india.com/product/index.html

+http://www.priceindia.in/bike/yamaha-bike-price-list/:

VMAX (1679, 2000000, Disk)

YZF R1 (998, 1050000,disk)

FZ1 (998,870000,disk)

YZF R15 (150, 105000,disk)

Fazer (153, 72000, combo)

SZR (153,55000)

SZX (153,52000)

YBR(125, 47000,drum)

YBR110(110, 42000, drum, 4)

http://www.tvsmotor.in/index.asp#

+ **http://www.tvsapache.in/apache-rtr-160-specifications.html:**

Apache RTR 160 (161, , combo, 5)	Flame SR 125 (124.8, , Drum, 4)
Apache RTR 180 (177, , Disk, 5)	Star City (110, , Drum, 4)
TVS Flame DS 125(124.8, , , 4)	Scooty Streak (87.8, , Drum, 0)

Comments (if any):

……

……

……

……

……

……

……..

Date: …./…./……..

Signature: ……..

( )

A.4 **DOCUMENT: 2.4** Date: ……

Name: ……

Educational Qualification: ……

Organization:

……

……..

Domain corpus consisting of various keywords:

……	……
……	……
……	……
……	……
……	……
……	……
……	……
……	……
……	……
……	……

http://www.bmwmotorcycles.com/us/en/index.html:

Category-Enduro:

G 650 GS Sertão (8650 USD, 652cc, disk, 5, Wire spoke wheels)

F 800GS (12355USD, 798, Disc, Wike spoke wheel)

R1200 GS (18350 USD,1170cc, Disc, Cross-spoke wheels)

Category-Roadster:

F 800R (10840USD, 798, Disc, Cast Aluminum Wheels)

R1200 R (13880, 1170., Disc, Cast Aluminum wheels)

Category-Sport:

S 1000RR (15050, 999, Disc, Cast Aluminum wheels)

K 1300 S (15555,1293, Disc, Cast Aluminum Wheels)

Comments (if any):

……

……

……

……

……

……

……..

Date: …./…./……..

Signature: ……..

( )

A.5 **DOCUMENT: 3** Date: ……

COMBINED DOMAIN VOCABULARY & QUESTIONS

(AFTER REDUNDANCY REMOVAL)

1. Combining domain keywords & competency questions as identified by FAST participants:

Engine--cubic centimeter (cc)--Engine capacity→ **Engine Capacity**

Power→**Power**

Make--Make--make→**Make**

Mileage--Mileage--Mileage--Average→**Mileage**

Brakes-Brakes→**Brakes**

Tires→**Tires**

Fuel Tank capacity-- Fuel Tank capacity--Tank Capacity→**Fuel Tank Capacity**

Price--Price→**Price**

Ignition (Kick/Self)→ **Ignition (Kick/Self)**

Seat type (Split/Single/Step/Normal)→ **Seat type (Split/Single/Step/Normal)**

Weight→ **Weight**

Wheel type--Wheel type (Alloy/Wirespoke)→ **Wheel type (Alloy/Wirespoke)**

Gears→**Weight**

0-60→**(0-60)**

Looks→**Looks**

Ground clearance→ **Ground clearance**

Tire size→ **Tire size**

Tire Width→ **Tire Width**

Visor--Head Lamps→ **Visor**

Warranty (upto kms?)→ **Warranty (upto kms?)**

Maintenance→ **Maintenance**

Free Service→ **Free Service**

Suspension→ **Suspension**

Hero Honda--Hero Honda--Hero Honda→ **Hero Honda**

Hero--Hero→ **Hero**

Bajaj--Bajaj→ **Bajaj**

Kawasaki→ **Kawasaki**

Royal Enfield--Royal Enfield→ **Royal Enfield**

Yamaha--Yamaha→ **Yamaha**

TVS→ **TVS**

Harley Davidson→ **Harley Davidson**

Suzuki→ **Suzuki**

Karizma (Normal/ZMR)--ZMR→ **Karizma (Normal/ZMR)**

Splendor-- Splendor (Plus/NXG/Super/Pro)--Splendor→ **Splendor (Plus/NXG/Super/Pro)**

Passion-- Passion Pro--Passion Pro→ **Passion/Paasion Pro**

CD-Dawn→ **CD-Dawn**

CD-Deluxe→ **CD-Deluxe**

Pleasure→ **Pleasure**

Glamour (Normal/PGMFi)→ **Glamour (Normal/PGMFi)**

Achiever→ **Achiever**

CBZ Xtreme→ **CBZ Xtreme**

Hunk→ **Hunk**

Impulse → **Impulse**

CT100→ **CT100**

Pulsar (135/150/180/200)-- Pulsar 135 LS/Pulsar 150 DTS-i/Pulsar 180 DTS-i/Pulsar 220 DTS-i--Pulsar→ **Pulsar 135 LS/Pulsar 150 DTS-i/Pulsar 180 DTS-i/Pulsar 220 DTS-i**

Avenger 220 DTS-i→ **Avenger 220 DTS-i**

Discover 150/Discover 125/Discover 100→ **Discover 150/Discover 125/Discover 100**

Platina 100→ **Platina 100**

Duke→ **Duke**

Ninja--Ninja 250R/Ninja 650R→ **Ninja 250R/Ninja 650R**

Swish125→ **Swish125**

Bullet350/Bullet500--Bullet-- Bullet Electra Twinspark/Bullet 350 Twinspark /Bullet Electra EFI/Bullet Electra Deluxe /Bullet 500→ **Bullet Electra Twinspark/Bullet 350 Twinspark /Bullet Electra EFI/Bullet Electra Deluxe /Bullet 500**

Royal Enfield Classic/Royal Enfield Classic 500/Royal Enfield Classic 350→ **Royal Enfield Classic/Royal Enfield Classic 500/Royal Enfield Classic 350**

Thunderbird-- Thunderbird Twinspark → **Thunderbird Twinspark**

R15→ **R15**

FZ→**FZ**

CBR→ **CBR**

Victor→ **Victor**

Superlow (560000, 883)→ **Superlow (560000, 883)**

Iron883 (883, 660000)→ **Iron883 (883, 660000)**

Roadster (883, 765000)→ **Roadster (883, 765000)**

Forty-Eight (1202,8,65000)→ **Forty-Eight (1202,8,65000)**

Nightster (1202,110000)→ **Nightster (1202,110000)**

XR1200X(1200,121000)→ **XR1200X(1200,121000)**

StreetBob(1010000,1585)→ **StreetBob(1010000,1585)**

SuperglideCustom (1165000,1585)→ **SuperglideCustom (1165000,1585)**

Mountain Bikes→ **Mountain Bikes**

Dipper→**Dipper**

Model→ **Model**

Affordable→ **Affordable**

Light Weight→ **Light Weight**

Looks→ **Looks**

Competency questions:

- Average of bike >50?
- Price <=1.25 lakhs?
- Engine capacity >=150?
- Price <=1.25 lakhs and engine capacity >=150?
- Average of some particular model?
- CC of some particular model?
- CC >=150 and mileage >=60 and make should be either Hero Honda or Yamaha, should have alloy wheels with disk brakes. Price range [Rs. 50000, Rs. 100000]
- More than 200cc, mileage more than atleast 30km/ltr, Hero Honda or Royal Enfield, both disk brakes, top speed more than 130 kmph, cost under 1.2lakh, wide rear tyre, riding comfort.
- Average >=40 & price should be between Rs. 50000 & Rs. 60000, and engine capacity should be >=125
- Price of some particular model like Apache 160?

2. Combined domain vocabulary & competency questions:

Engine Capacity

Power

Make

Mileage

Brakes

Tires

Fuel Tank Capacity

Price

Ignition (Kick/Self)

Seat type (Split/Single/Step/Normal)

Weight

Wheel type (Alloy/Wire)

Weight

(0-60)

Looks

Ground clearance

Tire size

Tire Width

Visor

Warranty (upto kms?)

Maintenance

Free Service

Suspension

Hero Honda

Hero

Bajaj

Kawasaki

Royal Enfield

Yamaha

TVS

Harley Davidson

Suzuki

Karizma (Normal/ZMR)

Splendor (Plus/NXG/Super/Pro)

Passion/Paasion Pro

CD-Dawn

CD-Deluxe

Pleasure

Glamour (Normal/PGMFi)

Achiever

CBZ Xtreme

Hunk

Impulse

CT100

Pulsar 135 LS/Pulsar 150 DTS-i/Pulsar 180 DTS-i/Pulsar 220 DTS-i

Avenger 220 DTS-i

Discover 150/Discover 125/Discover 100

Platina 100

Duke

Ninja 250R/Ninja 650R

Swish125

Bullet Electra Twinspark/Bullet 350 Twinspark /Bullet Electra EFI/Bullet Electra Deluxe /Bullet 500

Royal Enfield Classic/Royal Enfield Classic 500/Royal Enfield Classic 350

Thunderbird Twinspark

R15

FZ

CBR

Victor

Superlow (560000, 883)

Iron883 (883, 660000)

Roadster (883, 765000)

Forty-Eight (1202,8,65000)

Nightster (1202,110000)

XR1200X(1200,121000)

StreetBob(1010000,1585)

SuperglideCustom (1165000,1585)

Mountain Bikes

Dipper

Model

Affordable

Light Weight

Looks

Competency questions:

- Average of bike >50?
- Price <=1.25 lakhs?
- Engine capacity >=150?
- Price <=1.25 lakhs and engine capacity >=150?
- Average of some particular model?
- CC of some particular model?
- CC >=150 and mileage >=60 and make should be either Hero Honda or Yamaha, should have alloy wheels with disk brakes. Price range [Rs. 50000, Rs. 100000]
- More than 200cc, mileage more than atleast 30km/ltr, Hero Honda or Royal Enfield, both disk brakes, top speed more than 130 kmph, cost under 1.2lakh, wide rear tyre, riding comfort.
- Average >=40 & price should be between Rs. 50000 & Rs. 60000, and engine capacity should be >=125
- Price of some particular model like Apache 160?

A.6 **DOCUMENT: 4** Date: ……

FINAL DOMAIN VOCABULARY & QUESTIONS ALONGWITH ISSUES ENCOUNTERED DURING FAST SESSION

(WITH ADDITIONS/DELETIONS DONE DURING DISCUSSION)

Domain Vocabulary after discussion:

Engine Capacity

Power

Make

Mileage

Brakes (Disk/Drum)

~~Tires~~

Fuel Tank Capacity

Price

Ignition (Kick/Self)

Seat type (Split/Single/Step/Normal)

Weight

Wheel type (Alloy/Wirespoke)

Weight

Gears

(0-60)

Looks

Ground clearance

Tire size

Tire Width

Visor

Warranty (upto kms?)

Maintenance

Free Service

Suspension

Hero Honda

Hero

Bajaj

Kawasaki

Royal Enfield

Yamaha

TVS

Harley Davidson

Suzuki

Karizma (Normal/ZMR)

Splendor (Plus/NXG/Super/Pro)

~~Passion~~/Paasion Pro

CD-Dawn

CD-Deluxe

~~Pleasure~~

Glamour (Normal/PGMFi)

Achiever

CBZ Xtreme

Hunk

Impulse

CT100

Pulsar 135 LS/Pulsar 150 DTS-i/Pulsar 180 DTS-i/Pulsar 200 DTS-i/Pulsar 220 DTS-i

Avenger 220 DTS-i

Discover 135/Discover 125/Discover 100

Platina 100

XCD

Duke→**Duke200**

Ninja 250R/Ninja 650R

~~Swish125~~

Bullet Electra Twinspark/Bullet 350 Twinspark /Bullet Electra EFI/Bullet Electra Deluxe

Royal Enfield Classic 500/Royal Enfield Classic 350

Thunderbird Twinspark

R15

FZ

CBR

Victor

Superlow (560000, 883)

Iron883 (883, 660000)

Roadster (883, 765000)

Forty-Eight (1202,8,65000)

Nightster (1202,110000)

XR1200X(1200,121000)

StreetBob(1010000,1585)

SuperglideCustom (1165000,1585)

Mountain Bikes

Dipper

Model

Affordable

Light Weight

Looks

Competency questions:

- Average of bike >50?
- Price <=1.25 lakhs?
- Engine capacity >=150?
- Price <=1.25 lakhs and engine capacity >=150?
- Average of some particular model?
- CC of some particular model?
- CC >=150 and mileage >=60 and make should be either Hero Honda or Yamaha, should have alloy wheels with disk brakes. Price range [Rs. 50000, Rs. 100000]
- More than 200cc, mileage more than atleast 30km/ltr, Hero Honda or Royal Enfield, both disk brakes, top speed more than 130 kmph, cost under 1.2lakh, wide rear tyre, riding comfort.
- Average >=40 & price should be between Rs. 50000 & Rs. 60000, and engine capacity should be >=125
- Price of some particular model like Apache 160?

Issues encountered:

1. The values of properties such as Engine capacity, brakes, Wheel type etc. are not listed above for different types of bikes.
2. Whether to include bikes that are being imported then being sold in India?

A.7 **DOCUMENT: 5** Date: ……

DOMAIN VOCABULARY AFTER REVIEW

(Completeness Testing + Removal of Semantic heterogeneities)

ALONGWITH SOLUTIONS TO ISSUES

Final Domain Vocabulary:

Engine Capacity

Power

Make

Mileage

Brakes (Disk/Drum)

Fuel Tank Capacity

Price

Ignition (Kick/Self)

Seat type (Split/Single/Step/Normal)

Weight

Wheel type (Alloy/Wirespoke)

Weight

Gears

(0-60)

Looks

Ground clearance

Tire size

Tire Width

Visor

Warranty (upto kms?)

Maintenance

Free Service

Suspension

Hero Honda

Hero

Bajaj

Kawasaki

Royal Enfield

Yamaha

TVS

Harley Davidson

Suzuki

Karizma (Normal)→ 223cc, 17bhp @7000 rpm, Rs. 74000, 15ltr., 40km/ltr, Combo, 150kg, Alloy, self, 5

Karizma (ZMR) → 223cc, 17.6bhp @ 7000rpm, Rs.120000 ,16 ltr., 40km/ltr, Disk, 159kg, Alloy, Self, 5

Splendor (Plus) → 97.2cc, 7.5bhp @ 8000rpm, Rs. 45000, 10.5ltr., 75km/ltr, Drum, 109kg, WireSpoke, Kick, 4

Splendor (NXG) → 97.2cc, 7.7bhp @ 7500rpm, Rs.48000 , 10.3ltr., 65km/ltr, Drum, 107kg, Alloy, Kick, 4

Splendor (Super) → 125cc, 9bhp @ 7000rpm, Rs.47000 ,12 ltr., 70km/ltr, Drum, 121kg, Alloy, Self, 4

Splendor (Pro)→97.2 cc, 7.6bhp @ 7500rpm, Rs.55000 , 11ltr., 80km/ltr, Drum, 109kg, Alloy, Self, 4

Paasion Pro → 97.2cc, 7.6bhp @ 7500rpm, Rs.53000 , 12.8ltr., 75km/ltr, Combo, 119kg, Alloy, Self, 4

CD-Dawn → 97.2cc, 7.7bhp @ 7500rpm, Rs.37000 , 10.5ltr., 75km/ltr, Drum, 107kg, Wire Spoke, Kick, 4

CD-Deluxe → 97.2cc, 7.7bhp @ 7500rpm, Rs.43000 , 10.5ltr., 75km/ltr, Drum, 107kg, Alloy, Kick, 4

Glamour (Normal) → 125cc, 9bhp @ 7000rpm, Rs.55000 ,13.6 ltr., 60km/ltr, Combo, 125kg, Alloy, Self, 4

Glamour (PGMFi) → 125cc, 9bhp @ 7000rpm, Rs.58500 ,12 ltr., 70km/ltr, Combo, 125kg, Alloy, Self, 4

Achiever → 150cc, 13.4bhp @ 8000rpm, Rs.60000 ,12.5 ltr., 55km/ltr, Combo, 134kg, Alloy, Self, 5

CBZ Xtreme → 150cc, 14.4bhp @ 8500rpm, Rs. 65000, 12.3ltr., 50km/ltr, Combo, 141kg, Alloy, Self, 5

Hunk→ 150cc, 14.2bhp @ 8500rpm, Rs. 63000 , 12.2ltr., 50km/ltr, Disk, 145kg, Alloy, Self, 5

Impulse→ 150cc, 13bhp @ 75000rpm, Rs.79000 , 11ltr., 55km/ltr, Combo, 119kg, Alloy, Self, 5

CT100 → 100cc, 8.2bhp @ 7500rpm, Rs. 32000, 10.5ltr., 80km/ltr, Drum, 109kg, Wire Spoke, Kick, 4

Pulsar 135 LS → 135cc, 13.1bhp @ 8500rpm, Rs.55000, 10ltr., 60km/ltr, Combo, 125kg, Alloy, Self, 4

Pulsar 150 DTS-i → 150cc, 14.09bhp @ 8500rpm, Rs. 63000, 15ltr., 48km/ltr, Combo, 130kg, Alloy, Self, 5

Pulsar 180 DTS-i → 180cc, 16.5bhp @ 8000rpm, Rs. 67000, 15ltr., 55km/ltr, Combo, 140kg, Alloy, Self, 5

Pulsar 200 DTS-i → 200cc, 18bhp @ 8000rpm, Rs. 70000, 15ltr., 40km/ltr, Disk, 145kg, Alloy, Self, 5

Pulsar 220 DTS-i → 220cc, 20bhp @ 8500rpm, Rs. 90000, 15ltr., 35km/ltr, Disk, 150kg, Alloy, Self, 5

Avenger 220 DTS-i →220 cc, 16.5bhp @ 8000rpm, Rs. 75000, 14ltr., 40km/ltr, Combo, 152kg, Alloy, Self, 5

Discover 135 → 135cc, 13.1bhp @ 8500rpm, Rs. 55000, 10ltr., 60km/ltr, Combo, 125kg, Alloy, Self, 4

Discover 125 → 125cc, 11bhp @ 8000rpm, Rs. 52000, 8ltr., 85km/ltr, Drum, 125kg, Alloy, Self, 4

Discover 100 → 100cc, 7.5bhp @ 7500rpm, Rs. 44500, 10.3ltr., 91km/ltr, Drum, 115kg, Alloy, Self, 4

Platina 100 → 99.27cc, 8.2bhp @ 7500rpm, Rs. 35500, 13ltr., 108km/ltr, Drum, 113kg, Alloy, Kick, 4

XCD → 125cc, 7.01bhp @ 7000rpm, Rs. 46000, 13ltr., 109km/ltr, Drum, 115kg, Alloy, Self, 4

Duke200 → 200cc, 25bhp @ 10000rpm, Rs. 130000, 10.5ltr., 35km/ltr, Disk, 136kg, Alloy, Self, 6

Ninja 250R

Ninja 650R

Bullet Electra Twinspark →350 cc, 19.8bhp @ 5250rpm, Rs. 111000, 13.5ltr., 40km/ltr, Combo, 183kg, Wire Spoke, Self, 5

Bullet Electra EFI → 500cc, 27.2bhp @ 5250rpm, Rs. 125000, 14.5ltr., 40km/ltr, Combo, 185kg, Wire Spoke, Self, 5

Bullet Electra Deluxe →500 cc, 27.2bhp @ 5250rpm, Rs. 140000, 14.5ltr., 45km/ltr, Combo, 187kg, Wire Spoke, Self, 5

Bullet 350 Twinspark → 350cc, 19.8bhp @ 5250rpm, Rs. 100000, 13.5ltr., 45km/ltr, Combo, 180kg, Wire Spoke, Self, 5

Royal Enfield Classic 500 → 500cc, 27.2bhp @ 5250rpm, Rs. 155000, 13.5ltr., 35km/ltr, Combo, 187 kg, Wire Spoke, Self, 5

Royal Enfield Classic 350 → 350cc, 19.8bhp @ 5250rpm, Rs. 117000, 13.5ltr., 45km/ltr, Combo, 182kg, Wire Spoke, Self, 5

Thunderbird Twinspark → 350cc, 19.8bhp @ 5250rpm, Rs. 116300, 15.5ltr., 45km/ltr, Combo, 182kg, Wire Spoke, Self, 5

R15 → 150cc, 16.8bhp @ 8500rpm, Rs. 119000, 12ltr., 45km/ltr, Disk, 136kg, Alloy, Self, 6

FZ → 153cc, 14bhp @ 7500rpm, Rs. 74900, 12ltr., 50km/ltr, Combo, 135kg, Alloy, Self, 5

Victor → 110cc, 8.1bhp @ 7250rpm, Rs. 50000, 11ltr., 85km/ltr, Drum, 113kg, Wire Spoke, Kick, 4

CBR → 250cc, 26.4bhp @ 8500rpm, Rs. 160000, 13ltr., 30km/ltr, Disk, 165kg, Alloy, Self, 6

Shine → 125cc, 10.3bhp @ 7500rpm, Rs. 53500, 11ltr., 60km/ltr, Drum, 122kg, Alloy, Self, 4

Unicorn → 150cc, 62000bhp @ 8000rpm, Rs. 62000, 13ltr., 60km/ltr, Combo, 165kg, Alloy, Self, 5

Superlow (560000, 883)

Iron883 (883, 660000)

Roadster (883, 765000)

Forty-Eight (1202,8,65000)

Nightster (1202,110000)

XR1200X(1200,121000)

StreetBob(1010000,1585)

SuperglideCustom (1165000,1585)

Mountain Bikes

Dipper

Model

Affordable

Light Weight

Looks

Competency questions:

- Mileage of bike >50?
- Price <=1.25 lakhs?
- Engine capacity >=150?
- Price <=1.25 lakhs and engine capacity >=150?
- Mileage of some particular model?
- Engine capacity of some particular model?
- Engine capacity >=150 and mileage >=60 and make should be either Hero Honda or Yamaha, should have alloy wheels with disk brakes. Price range [Rs. 50000, Rs. 100000]
- More than 200cc, mileage more than atleast 30km/ltr, Hero Honda or Royal Enfield, both disk brakes, cost under 1.2lakh.
- Average >=40 & price should be between Rs. 50000 & Rs. 60000, and engine capacity should be >=125.
- Price of some particular model like Duke 200?

Issues encountered:

1. The values of properties such as Engine capacity, brakes, Wheel type etc. are not listed above for different types of bikes.

   ***Ans.*:** The values for following parameters were included in domain vocabulary:

   - Make
   - Engine capacity
   - Power
   - Price
   - Fuel tank capacity
   - Mileage
   - Brake type
   - Weight
   - Wheel type
   - Ignition
   - Number of gears

2. Whether to include bikes that are being imported then being sold in India?

   ***Ans.*:** No

A.8 **DOCUMENT: 7** Date: ……

VALIDATED VERSION OF DOCUMENT 6

1. Domain Vocabulary:

Engine Capacity

Power

Make

Mileage

Brakes (Disk/Drum)

Fuel Tank Capacity

Price

Ignition (KickStart/SelfStart)

Seat type (Split/Single/Step/Normal)

Weight

Wheel type (Alloy/Wirespoke)

Weight

Gears

Hero Honda

Hero

Bajaj

Royal Enfield

Yamaha

TVS

Suzuki

Karizma (Normal)→ 223cc, 17bhp @7000 rpm, Rs. 74000, 15ltr., 40km/ltr, Combo, 150kg, Alloy, self, 5

Karizma (ZMR) → 223cc, 17.6bhp @ 7000rpm, Rs.120000 ,16 ltr., 40km/ltr, Disk, 159kg, Alloy, Self, 5

Splendor (Plus) → 97.2cc, 7.5bhp @ 8000rpm, Rs. 45000, 10.5ltr., 75km/ltr, Drum, 109kg,

WireSpoke, Kick, 4

Splendor (NXG) → 97.2cc, 7.7bhp @ 7500rpm, Rs.48000 , 10.3ltr., 65km/ltr, Drum, 107kg, Alloy, Kick, 4

Splendor (Super) → 125cc, 9bhp @ 7000rpm, Rs.47000 ,12 ltr., 70km/ltr, Drum, 121kg, Alloy, Self, 4

Splendor (Pro)→97.2 cc, 7.6bhp @ 7500rpm, Rs.55000 , 11ltr., 80km/ltr, Drum, 109kg, Alloy, Self, 4

Paasion Pro → 97.2cc, 7.6bhp @ 7500rpm, Rs.53000 , 12.8ltr., 75km/ltr, Combo, 119kg, Alloy, Self, 4

CD-Dawn → 97.2cc, 7.7bhp @ 7500rpm, Rs.37000 , 10.5ltr., 75km/ltr, Drum, 107kg, Wire Spoke, Kick, 4

CD-Deluxe → 97.2cc, 7.7bhp @ 7500rpm, Rs.43000 , 10.5ltr., 75km/ltr, Drum, 107kg, Alloy, Kick, 4

Glamour (Normal) → 125cc, 9bhp @ 7000rpm, Rs.55000 ,13.6 ltr., 60km/ltr, Combo, 125kg, Alloy, Self, 4

Glamour (PGMFi) → 125cc, 9bhp @ 7000rpm, Rs.58500 ,12 ltr., 70km/ltr, Combo, 125kg, Alloy, Self, 4

Achiever → 150cc, 13.4bhp @ 8000rpm, Rs.60000 ,12.5 ltr., 55km/ltr, Combo, 134kg, Alloy, Self, 5

CBZ Xtreme → 150cc, 14.4bhp @ 8500rpm, Rs. 65000, 12.3ltr., 50km/ltr, Combo, 141kg, Alloy, Self, 5

Hunk→ 150cc, 14.2bhp @ 8500rpm, Rs. 63000 , 12.2ltr., 50km/ltr, Disk, 145kg, Alloy, Self, 5

Impulse→ 150cc, 13bhp @ 75000rpm, Rs.79000 , 11ltr., 55km/ltr, Combo, 119kg, Alloy, Self, 5

CT100 → 100cc, 8.2bhp @ 7500rpm, Rs. 32000, 10.5ltr., 80km/ltr, Drum, 109kg, Wire Spoke, Kick, 4

Pulsar 135 LS → 135cc, 13.3bhp @ 9000rpm, Rs. 57000, 8ltr., 68km/ltr, Combo, 122kg, Alloy, Self, 5

Pulsar 150 DTS-i → 150cc, 14.09bhp @ 8500rpm, Rs. 63000, 15ltr., 48km/ltr, Combo, 130kg, Alloy, Self, 5

Pulsar 180 DTS-i → 180cc, 16.5bhp @ 8000rpm, Rs. 67000, 15ltr., 55km/ltr, Combo, 140kg, Alloy, Self, 5

Pulsar 200 DTS-i → 200cc, 18bhp @ 8000rpm, Rs. 70000, 15ltr., 40km/ltr, Disk, 145kg, Alloy, Self, 5

Pulsar 220 DTS-i → 220cc, 20bhp @ 8500rpm, Rs. 90000, 15ltr., 35km/ltr, Disk, 150kg, Alloy, Self, 5

Avenger 220 DTS-i →220 cc, 16.5bhp @ 8000rpm, Rs. 75000, 14ltr., 40km/ltr, Combo, 152kg, Alloy, Self, 5

Discover 135 → 135cc, 13.1bhp @ 8500rpm, Rs. 55000, 10ltr., 60km/ltr, Combo, 125kg, Alloy, Self, 4

Discover 125 → 125cc, 11bhp @ 8000rpm, Rs. 52000, 8ltr., 85km/ltr, Drum, 125kg, Alloy, Self, 4

Discover 100 → 100cc, 7.5bhp @ 7500rpm, Rs. 44500, 10.3ltr., 91km/ltr, Drum, 115kg, Alloy, Self, 4

Platina 100 → 99.27cc, 8.2bhp @ 7500rpm, Rs. 35500, 13ltr., 108km/ltr, Drum, 113kg, Alloy, Kick, 4

XCD → 125cc, 7.01bhp @ 7000rpm, Rs. 46000, 13ltr., 109km/ltr, Drum, 115kg, Alloy, Self, 4

Duke200 → 200cc, 25bhp @ 10000rpm, Rs. 130000, 10.5ltr., 35km/ltr, Disk, 136kg, Alloy, Self, 6

Bullet Electra Twinspark →350 cc, 19.8bhp @ 5250rpm, Rs. 111000, 13.5ltr., 40km/ltr, Combo, 183kg, Wire Spoke, Self, 5

Bullet Electra EFI → 500cc, 27.2bhp @ 5250rpm, Rs. 125000, 14.5ltr., 40km/ltr, Combo, 185kg, Wire Spoke, Self, 5

Bullet Electra Deluxe →500 cc, 27.2bhp @ 5250rpm, Rs. 140000, 14.5ltr., 45km/ltr, Combo, 187kg, Wire Spoke, Self, 5

Bullet 350 Twinspark → 350cc, 19.8bhp @ 5250rpm, Rs. 100000, 13.5ltr., 45km/ltr, Combo, 180kg, Wire Spoke, Self, 5

Royal Enfield Classic 500 → 500cc, 27.2bhp @ 5250rpm, Rs. 155000, 13.5ltr., 35km/ltr, Combo, 187 kg, Wire Spoke, Self, 5

Royal Enfield Classic 350 → 350cc, 19.8bhp @ 5250rpm, Rs. 117000, 13.5ltr., 45km/ltr, Combo, 182kg, Wire Spoke, Self, 5

Thunderbird Twinspark → 350cc, 19.8bhp @ 5250rpm, Rs. 116300, 15.5ltr., 45km/ltr, Combo, 182kg, Wire Spoke, Self, 5

R15 → 150cc, 16.8bhp @ 8500rpm, Rs. 119000, 12ltr., 45km/ltr, Disk, 136kg, Alloy, Self, 6

FZ → 153cc, 14bhp @ 7500rpm, Rs. 74900, 12ltr., 50km/ltr, Combo, 135kg, Alloy, Self, 5

Victor → 110cc, 8.1bhp @ 7250rpm, Rs. 50000, 11ltr., 85km/ltr, Drum, 113kg, Wire Spoke, Kick, 4

CBR → 250cc, 26.4bhp @ 8500rpm, Rs. 160000, 13ltr., 30km/ltr, Disk, 165kg, Alloy, Self, 6

Shine → 125cc, 10.3bhp @ 7500rpm, Rs. 53500, 11ltr., 60km/ltr, Drum, 122kg, Alloy, Self, 4

Unicorn → 150cc, 62000bhp @ 8000rpm, Rs. 62000, 13ltr., 60km/ltr, Combo, 165kg, Alloy, Self, 5

Step 2: Identified Classes, sub-classes & individuals:

Root class → *Thing*

Sub-class of *Thing* → *Bikes*

Subclasses of *Bikes* →

Make	*Brakes*
EngineCapacity	*Weight*
Power	*WheelType*
Price	*Ignition*
FuelTankCapacity	*Gears*
Mileage	

NamedBikes (It is supposed to be merely a container class.)

Subclasses of *Make* →*Hero Honda*, *Hero*, *Bajaj*, *Royal Enfield*, *Yamaha*, *TVS* and *Honda*

Subclasses of *Brakes* → *Combo*, *DiskBrakes*, *DrumBrakes*

Subclasses of *WheelType* → *Alloy*, *Wirespoke*

Subclasses of *Ignition* → *KickStart*, *SelfStart*

Subclasses of *Gears* → *4*, *5*, *6*

Subclasses of *NamedBikes* →

HeroHondaBikes	*YamahaBikes*
HeroBikes	*TVSBikes*
BajajBikes	*HondaBikes*
RoyalEnfieldBikes	

Subclasses of *HeroHondaBikes* →

Karizma (Normal) Model:
◊ Karizma (Normal)

Karizma (ZMR) Model:
◊ Karizma (ZMR)

Splendor (Plus) Model:
◊ Splendor (Plus)

Splendor (NXG) Model:
◊ Splendor (NXG)

Splendor (Super) Model:
◊ Splendor (Super)

Splendor (Pro) Model:
◊ Splendor (Pro)

Passion Pro Model:
◊ Passion Pro

CD-Dawn Model:
◊ CD-Dawn

CD-Deluxe Model:
◊ CD-Deluxe

Glamour (Normal) Model:
◊ Glamour (Normal)

Glamour (PGMFi) Model:
◊ Glamour (PGMFi)

Achiever Model:
◊ Achiever

CBZ Xtreme Model:
◊ CBZ Xtreme

Hunk Model:
◊ Hunk

Subclasses of *HeroBikes* → *Impulse Model*: ◊ Impulse

Subclasses of *BajajBikes* →

CT100 Model: ◊ CT100

Pulsar 135 LS Model:
◊ Pulsar 135 LS

Pulsar 150 DTS-i Model:
◊ Pulsar 150 DTS-i

Pulsar 180 DTS-i Model:
◊ Pulsar 180 DTS-i

Pulsar 220 DTS-i Model:
◊ Pulsar 220 DTS-i

Avenger 220 DTS-i Model:
◊ Avenger 220 DTS-i

Discover 135 Model:
◊ Discover 135

Discover 125 Model:
◊ Discover 125

Discover 100 Model:
◊ Discover 100

Platina 100 Model:
◊ Platina 100

Duke200 Model:
◊ Duke200

Subclasses of *RoyalEnfieldBikes* →

Bullet Electra Twinspark Model:

◊ Bullet Electra Twinspark

Bullet 350 Twinspark Model:

◊ Bullet 350 Twinspark

Bullet Electra EFI Model:

◊ Bullet Electra EFI

Bullet Electra Deluxe Model:

◊ Bullet Electra Deluxe

Royal Enfield Classic 500 Model:

◊ Royal Enfield Classic 500

Royal Enfield Classic 350 Model:

◊ Royal Enfield Classic 350

Thunderbird Twinspark Model:

◊ Thunderbird Twinspark

Subclasses of *YamahaBikes* →

R15 Model: ◊ R15

FZ Model: ◊ FZ

Subclasses of *TVSBikes* →

Victor Model: ◊ Victor

Subclasses of *HondaBikes* →

CBR Model: ◊ CBR

Shine Model: ◊ Shine

Unicorn Model: ◊ Unicorn

Step 3.1: Identified Properties:

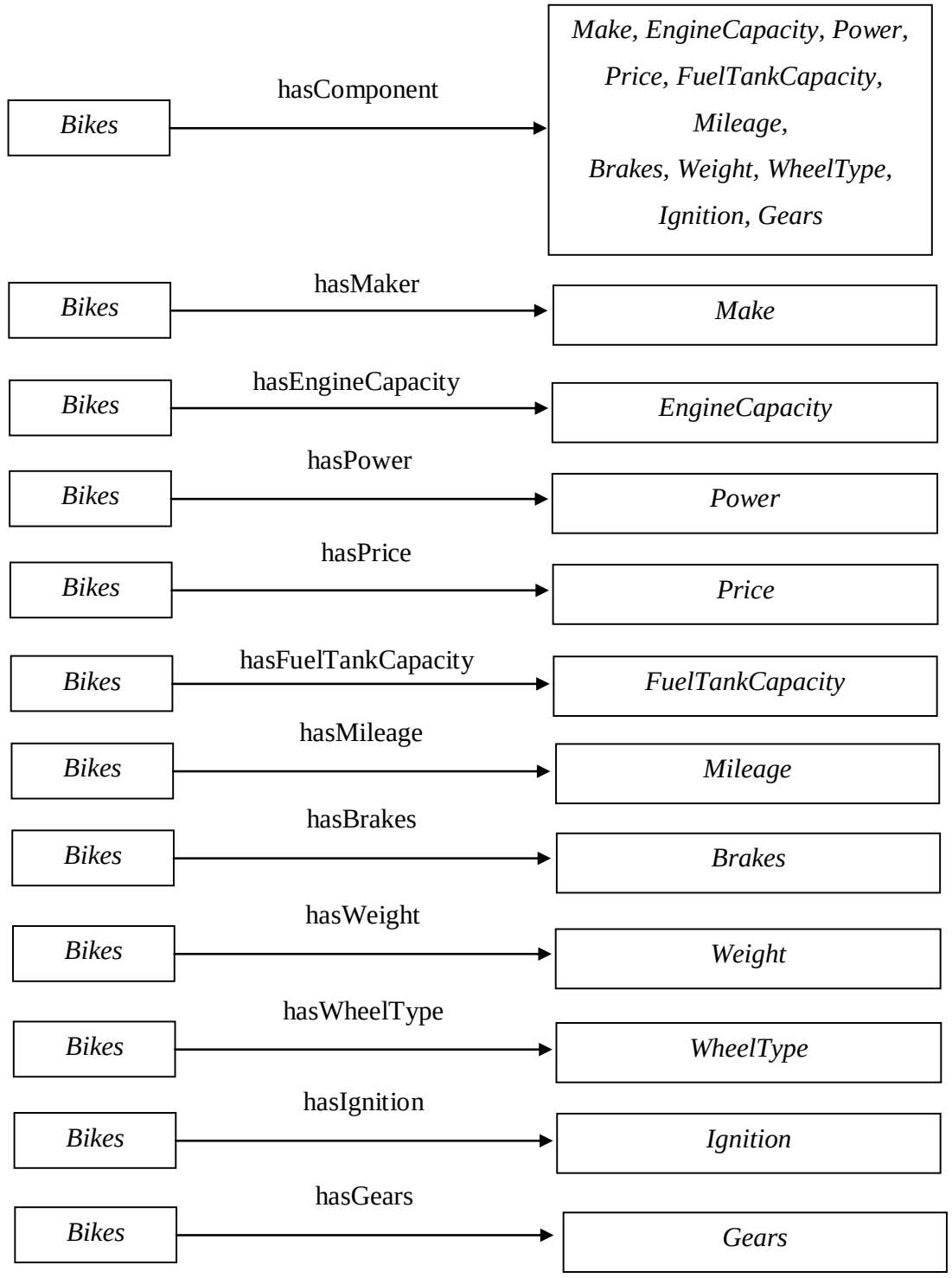
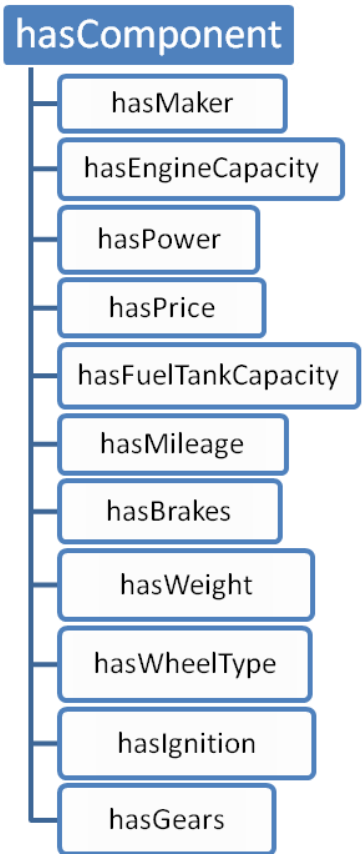


Fig. 36: Identified-properties

Thus property hierarchy is:

Fig. 37: Identified-property hierarchy



Step 3.2: Domains and Ranges of properties:

Table 6: Domain & Ranges of identified properties

Name	Domain	Range
hasMaker	Bikes	Make
hasEngineCapacity	Bikes	Float
hasPower	Bikes	String
hasPrice	Bikes	Integer
hasFuelTankCapacity	Bikes	Float
hasMileage	Bikes	Float
hasBrakes	Bikes	Brakes
hasWeight	Bikes	Float
hasWheelType	Bikes	WheelType
hasIgnition	Bikes	Ignition
hasBrakes	Bikes	Brakes

Step 3.3: Identified Inverse Properties:

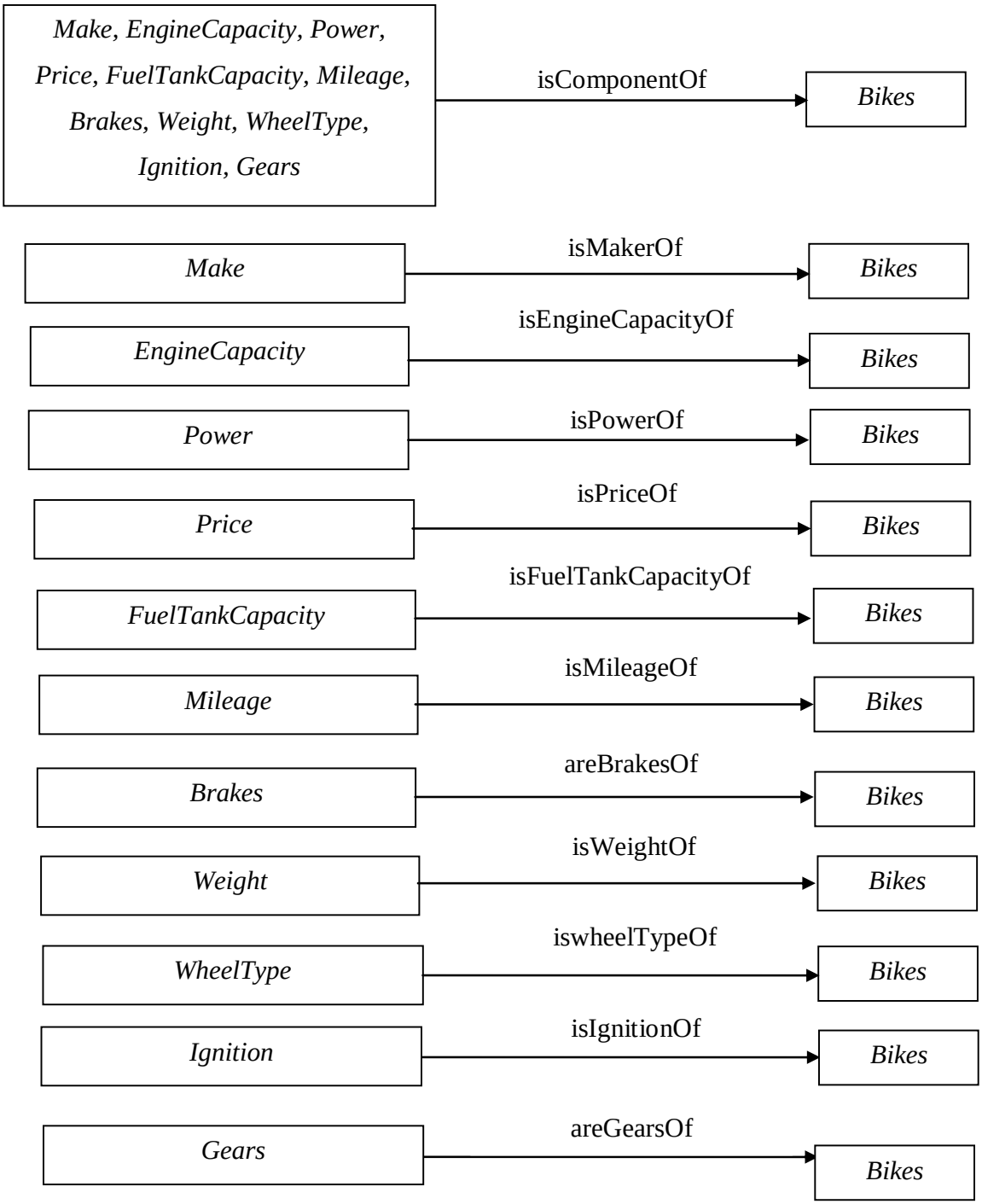
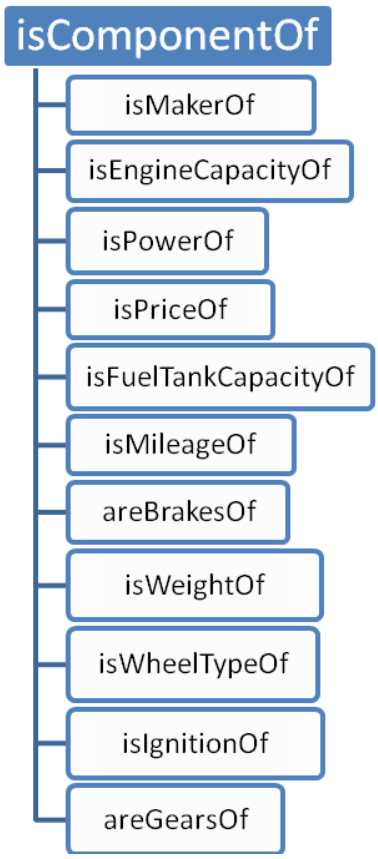


Fig. 38: Identified inverse-properties

Thus the inverse-property hierarchy is:

Fig. 39: Inverse-property hierarchy



Step 3.4: Domains and Ranges of inverse properties:

Table 7: Domain & Ranges of inverse-property hierarchy

Name	Domain	Range
isMakerOf	Make	Bikes
isEngineCapacityOf	Float	Bikes
isPowerOf	String	Bikes
isPriceOf	Integer	Bikes
isFuelTankCapacityOf	Float	Bikes
isMileageOf	Float	Bikes
areBrakesOf	Brakes	Bikes
isWeightOf	Float	Bikes
isWheelTypeOf	WheelType	Bikes
isIgnitionOf	Ignition	Bikes
areBrakesOf	Brakes	Bikes

APPENDIX B

B.1 ONTOLOGY DESIGN DOCUMENT

Date: ……

1. Initial ontology structure:

Fig. 40: Initial Ontology structure

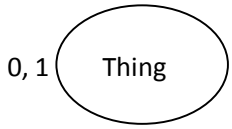


Table8: Initial Location map

CONCEPT ADDRESS	CONCEPT FEATURE
Thing (0,1)	X

2. Concept encountered: *Bikes* with concept features:

{ $hasComponent^+$ } → f^+1

{ $hasMake^+$ } → f^+2

{ $hasEngineCapacity^+$ } → f^+3

{ $hasPower^+$ } → f^+4

{ $hasPrice^+$ } → f^+5

{ $hasFuelTankCapacity^+$ } → f^+6

{ $hasMileage^+$ } → f^+7

{ $hasBrakes^+$ } → f^+8

{ $hasWeight^+$ } → f^+9

{ $hasWheelType^+$ } → f^+10

{ $hasIgnition^+$ } → f^+11

{ $hasGears^+$ } → f^+12

(a.) This can be represented as a Boolean equation as:
C (Bikes): { $f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$ }

(b.) Matching the above Boolean equation and Concept *Thing* in the location map, it is seen that *Bikes* is a sub-concept of *Thing* as *Thing* encompasses everything.

(c.) The location of new concept *Bikes* is (1, 1). The new concept is added to the location (1,1) in location map as below:

Fig. 41: Modified Ontology structure

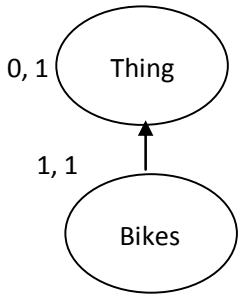


Table9: Modified Location map

CONCEPT ADDRESS	CONCEPT FEATURE
Thing (0,1)	X
Bikes (1,1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$

3. Concept encountered: *Make* with concept features:

{ $hasMake^+$ } → f^+2

(a.) This can be represented as a Boolean equation as:
C (Make): { f^+2}

(b.) Matching the above Boolean equation and concepts *Thing* & *Bikes* in the location map, it is seen that *Make* is a sub-concept of *Thing* and is either a brother concept or sub-concept of *Bike*. Matching C (Make): { f^+2} with C (Bikes): { $f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$ }, it is found that *Make* is a brother-concept of *Bikes*.

(c.) The location of new concept *Make* is (1, 2). The new concept is added to the location (1, 2) in location map as below:

Fig. 42: Modified Ontology structure

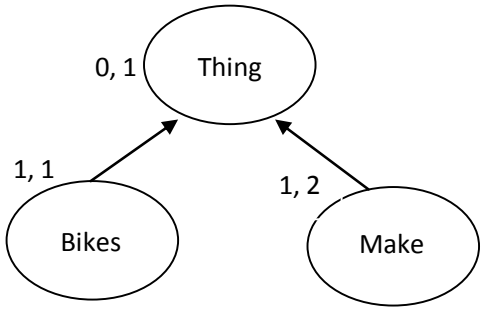


Table10: Modified Location map

CONCEPT ADDRESS	CONCEPT FEATURE
Thing (0,1)	X
Bikes (1,1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
Make (1,2)	f^+2

4. Concept encountered: *EngineCapacity* with concept features:

   { hasEngineCapacity$^+$ } → f^+3

   (a.) This can be represented as a Boolean equation as:
   C (EngineCapacity): { f^+3}

   (b.) Matching the above Boolean equation and concepts *Thing*, *Bikes* and *Make* in the location map, it is seen that *EngineCapacity* is a sub-concept of *Thing* and is either a brother concept or sub-concept of *Bike* or *Make*. Matching C (EngineCapacity): { f^+3} with C (Bikes): { $f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$ } and C (Make): { f^+2}, it is found that *EngineCapacity* is a not a sub-concept of either *Bikes* or *Make*.
   Thus, it is found that *EngineCapacity* is a brother concept of *Bikes* and *Make*.

(c.) The location of new concept *Make* is (2, 2). The new concept is added to the location (2, 2) in location map as below:

Fig. 43: Modified Ontology structure

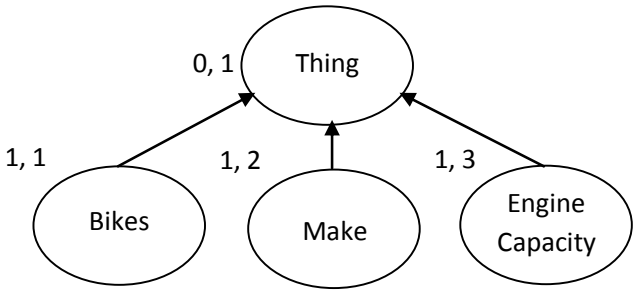


Table11: Modified Location map

CONCEPT ADDRESS	CONCEPT FEATURE
Thing (0,1)	X
Bikes (1,1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
Make (1,2)	f^+2
EngineCapacity (1,3)	f^+3

5. Following the above methodology, we develop the ontology structure for the following subsequent concepts:

Table12: List of subsequent concepts

CONCEPT NAME	CONCEPT FEATURE	BOOLEAN EQUATION
Power	$\{hasPower^+\} \rightarrow f^+4$	C (Power): $\{f^+4\}$
Price	$\{hasPrice^+\} \rightarrow f^+5$	C (Price): $\{f^+5\}$
FuelTankCapacity	$\{hasFuelTankCapacity^+\} \rightarrow f^+6$	C (FuelTankCapacity): $\{f^+6\}$
Mileage	$\{hasMileage^+\} \rightarrow f^+7$	C (Mileage): $\{f^+7\}$
Brakes	$\{hasBrakes^+\} \rightarrow f^+8$	C (Brakes): $\{f^+8\}$
Weight	$\{hasWeight^+\} \rightarrow f^+9$	C (Weight): $\{f^+9\}$
WheelType	$\{hasWheelType^+\} \rightarrow f^+10$	C (WheelType): $\{f^+10\}$
Ignition	$\{hasIgnition^+\} \rightarrow f^+11$	C (Ignition): $\{f^+11\}$
Gears	$\{hasGears^+\} \rightarrow f^+12$	C (Gears): $\{f^+12\}$

Fig. 44: Modified Ontology structure

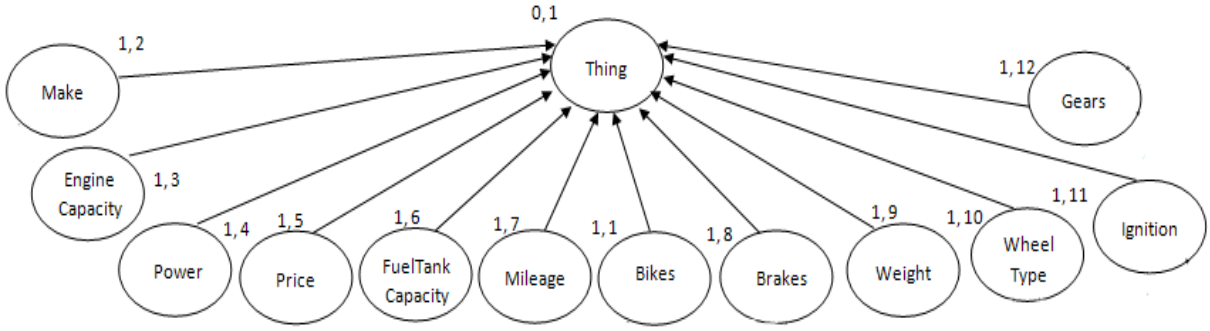


Table13: Modified Location map

CONCEPT ADDRESS	CONCEPT FEATURE
Thing (0, 1)	X
Bikes (1, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
Make (1, 2)	f^+2
EngineCapacity (1, 3)	f^+3
Power (1, 4)	f^+4
Price (1, 5)	f^+5
FuelTankCapacity (1, 6)	f^+6
Mileage (1, 7)	f^+7
Brakes (1, 8)	f^+8
Weight (1, 9)	f^+9
WheelType (1, 10)	f^+10
Ignition (1,11)	f^+11
Gears (1 ,12)	f^+12

6. Since we created *NamedBikes* to be merely a container for all named bikes, so we add it as a sub-concept of *Bikes* with same concept feature as of *Bikes* as follows:

Fig. 45: Modified Ontology structure

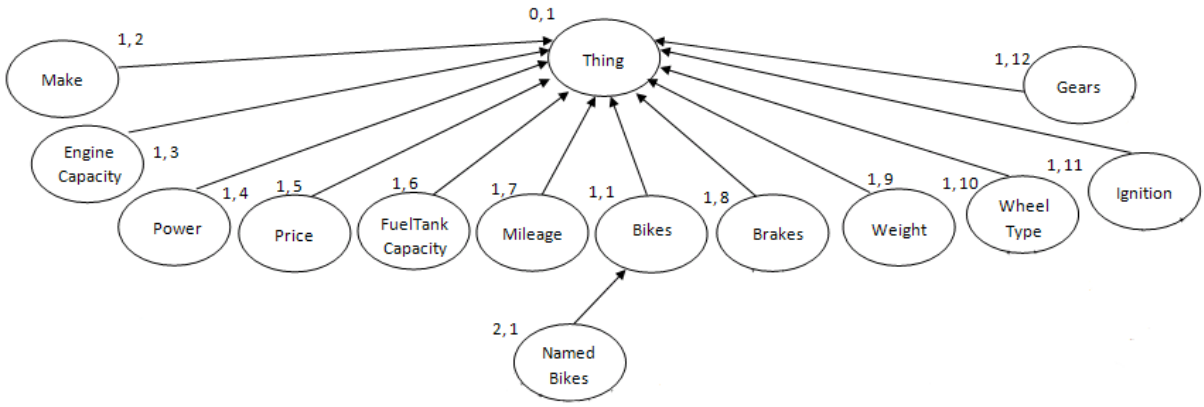


Table14: Modified Location map

CONCEPT ADDRESS	CONCEPT FEATURE
Thing (0, 1)	X
Bikes (1, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
Make (1, 2)	f^+2
EngineCapacity (1, 3)	f^+3
Power (1, 4)	f^+4
Price (1, 5)	f^+5
FuelTankCapacity (1, 6)	f^+6
Mileage (1, 7)	f^+7
Brakes (1, 8)	f^+8
Weight (1, 9)	f^+9
WheelType (1, 10)	f^+10
Ignition (1,11)	f^+11
Gears (1 ,12)	f^+12
NamedBikes (2, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$

7. Concept encountered: *HeroHonda* with concept features:

   $\{ \text{hasMake}^+ \} \rightarrow f^+2$

   $\{ \text{hasMake}^+ = \text{HeroHonda} \} \rightarrow f^+13$

   (a.) This can be represented as a Boolean equation as:
   C (HeroHonda): $\{ f^+2 \bullet f^+13 \}$

   (b.) Matching the above Boolean equation with concepts *Thing, Bikes, Make, EngineCapacity, Power, Price, FuelTankCapacity, Mileage, Brakes, Weight, WheelType, Ignition, Gears* and *NamedBikes* in the location map, it is seen that *HeroHonda* is a sub-concept of *Thing* and is either a brother concept or sub-concept of *Bikes, Make, EngineCapacity, Power, Price, FuelTankCapacity, Mileage, Brakes, Weight, WheelType, Ignition or Gears*. Matching C (HeroHonda): $\{ f^+2 \bullet f^+13 \}$ with them, it is found that *HeroHonda* is a sub-concept of *Make.*

(c.) The location of new concept *HeroHonda* is (2, 2). The new concept is added to the location (3, 1) in location map as below:

Fig. 46: Modified Ontology structure

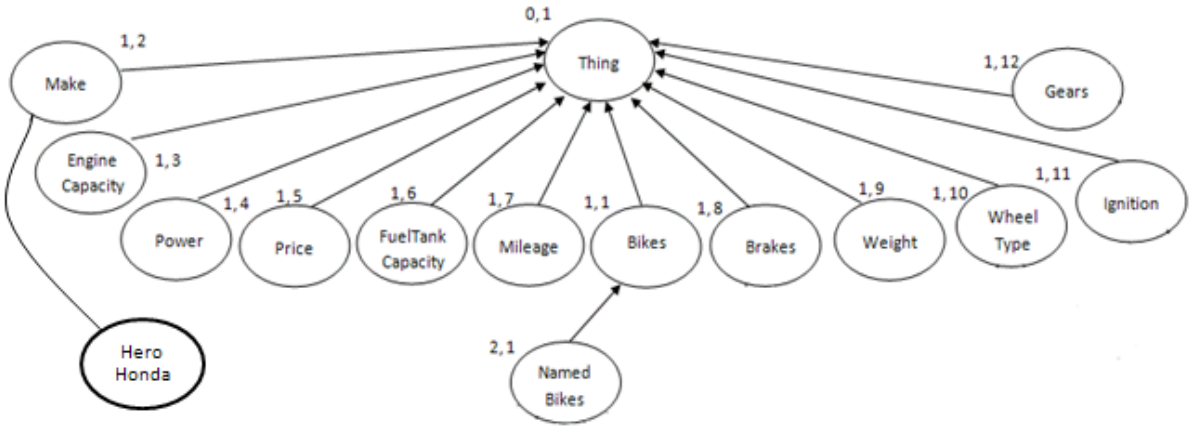


Table15: Modified Location map

CONCEPT ADDRESS	CONCEPT FEATURE
Thing (0, 1)	X
Bikes (1, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
Make (1, 2)	f^+2
EngineCapacity (1, 3)	f^+3
Power (1, 4)	f^+4
Price (1, 5)	f^+5
FuelTankCapacity (1, 6)	f^+6
Mileage (1, 7)	f^+7
Brakes (1, 8)	f^+8
Weight (1, 9)	f^+9
WheelType (1, 10)	f^+10
Ignition (1,11)	f^+11
Gears (1 ,12)	f^+12
NamedBikes (2, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
HeroHonda (2, 2)	$f^+2 \bullet f^+13$

8. Following the above methodology, we develop the ontology structure for the following subsequent concepts:

Table16: List of subsequent concepts

CONCEPT NAME	CONCEPT FEATURE	BOOLEAN EQUATION
Hero	{ hasMake$^+$ } → f$^+$2 { hasMake$^+$ = Hero } → f$^+$14	C (HeroHonda): { f$^+$2 • f$^+$14}
Bajaj	{ hasMake$^+$ } → f$^+$2 { hasMake$^+$ = Bajaj } → f$^+$15	C (Bajaj): { f$^+$2 • f$^+$15}
RoyalEnfield	{ hasMake$^+$ } → f$^+$2 { hasMake$^+$ = RoyalEnfield } → f$^+$16	C (RoyalEnfield): { f$^+$2 • f$^+$16}
Yamaha	{ hasMake$^+$ } → f$^+$2 { hasMake$^+$ = Yamaha} → f$^+$17	C (Yamaha): { f$^+$2 • f$^+$17}
TVS	{ hasMake$^+$ } → f$^+$2 { hasMake$^+$ = TVS } → f$^+$18	C (TVS): { f$^+$2 • f$^+$18}
Honda	{ hasMake$^+$ } → f$^+$2 { hasMake$^+$ = Honda } → f$^+$19	C (Honda): { f$^+$2 • f$^+$19}

Fig. 47: Modified Ontology structure

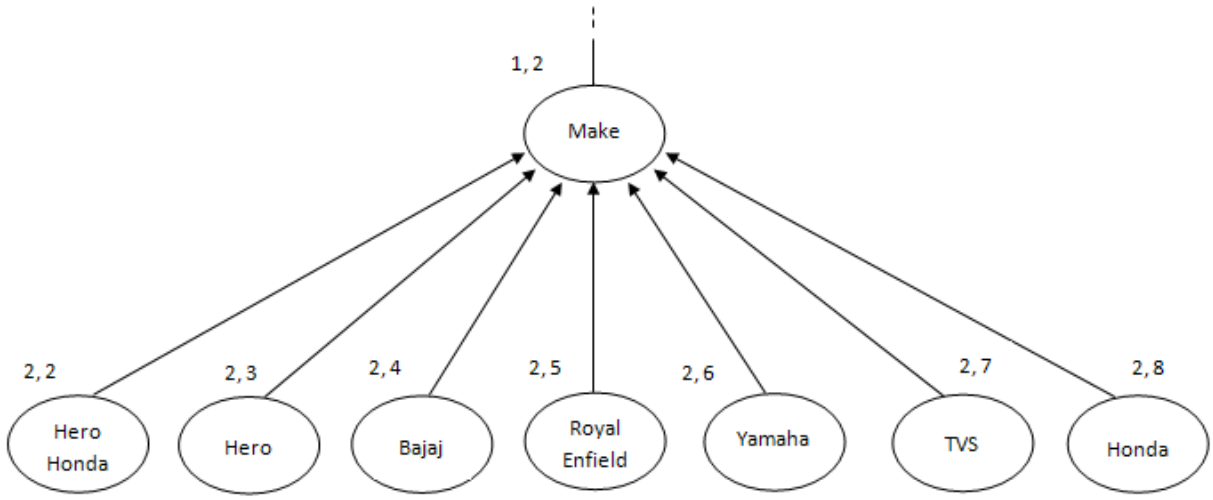


Table17: Modified Location map

CONCEPT ADDRESS	CONCEPT FEATURE
Thing (0, 1)	X
Bikes (1, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
Make (1, 2)	f^+2
EngineCapacity (1, 3)	f^+3
Power (1, 4)	f^+4
Price (1, 5)	f^+5
FuelTankCapacity (1, 6)	f^+6
Mileage (1, 7)	f^+7
Brakes (1, 8)	f^+8
Weight (1, 9)	f^+9
WheelType (1, 10)	f^+10
Ignition (1,11)	f^+11
Gears (1 ,12)	f^+12
NamedBikes (2, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
HeroHonda (2, 2)	$f^+2 \bullet f^+13$
Hero (2, 3)	$f^+2 \bullet f^+14$
Bajaj (2, 4)	$f^+2 \bullet f^+15$
RoyalEnfield (2, 5)	$f^+2 \bullet f^+16$
Yamaha (2, 6)	$f^+2 \bullet f^+17$
TVS (2, 7)	$f^+2 \bullet f^+18$
Honda (2, 8)	$f^+2 \bullet f^+19$

9. Concept encountered: *HeroHondaBikes* with concept features:

{ $hasMake^+$ } → f^+2

{ $hasEngineCapacity^+$ } → f^+3

{ $hasPower^+$ } → f^+4

{ $hasPrice^+$ } → f^+5

{ $hasFuelTankCapacity^+$ } → f^+6

{ $hasMileage^+$ } → f^+7

{ $hasBrakes^+$ } → f^+8

{ $hasWeight^+$ } → f^+9

{ $hasWheelType^+$ } → f^+10

{ $hasIgnition^+$ } → f^+11

{ $hasGears^+$ } → f^+12

{ hasMake^{+} = HeroHonda } → f^{+} 13

(a.) This can be represented as a Boolean equation as:
C (HeroHondaBikes): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 13 }

(b.) Matching the above Boolean equation with concepts *Thing*, *Bikes*, *Make*, *EngineCapacity*, *Power*, *Price*, *FuelTankCapacity*, *Mileage*, *Brakes*, *Weight*, *WheelType*, *Ignition*, *Gears* and *NamedBikes* in the location map, it is seen that *HeroHondaBikes* is a sub-concept of *Thing* and is either a brother concept or sub-concept of *Bikes*, *Make*, *EngineCapacity*, *Power*, *Price*, *FuelTankCapacity*, *Mileage*, *Brakes*, *Weight*, *WheelType*, *Ignition*, *Gears* or *NamedBikes*. Matching C (HeroHondaBikes): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 13} with them, it is found that *HeroHondaBikes* cannot be a sub-concept of *Make*, *EngineCapacity*, *Power*, *Price*, *FuelTankCapacity*, *Mileage*, *Brakes*, *Weight*, *WheelType*, *Ignition* and *Gears*. It is then found that *HeroHondaBikes* is a sub-concept of *Bikes* and thus a sub-concept of *NamedBikes*.

(c.) The location of new concept *HeroHondaBikes* is (3, 1). The new concept is added to the location (3, 1) in location map as below:

Fig. 48: Modified Ontology structure

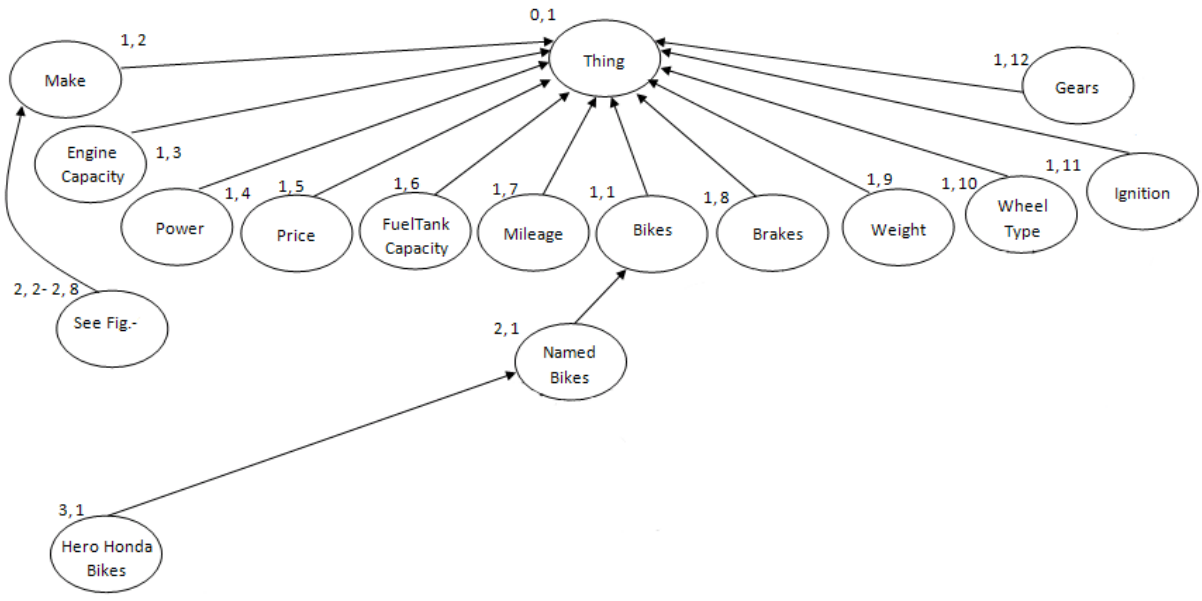


Table18: Modified Location map

CONCEPT ADDRESS	CONCEPT FEATURE
Thing (0, 1)	X
Bikes (1, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
Make (1, 2)	f^+2
EngineCapacity (1, 3)	f^+3
Power (1, 4)	f^+4
Price (1, 5)	f^+5
FuelTankCapacity (1, 6)	f^+6
Mileage (1, 7)	f^+7
Brakes (1, 8)	f^+8
Weight (1, 9)	f^+9
WheelType (1, 10)	f^+10
Ignition (1,11)	f^+11
Gears (1 ,12)	f^+12
NamedBikes (2, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
HeroHonda (2, 2)	$f^+2 \bullet f^+13$
Hero (2, 3)	$f^+2 \bullet f^+14$
Bajaj (2, 4)	$f^+2 \bullet f^+15$
RoyalEnfield (2, 5)	$f^+2 \bullet f^+16$
Yamaha (2, 6)	$f^+2 \bullet f^+17$
TVS (2, 7)	$f^+2 \bullet f^+18$
Honda (2, 8)	$f^+2 \bullet f^+19$
HeroHondaBikes (3, 1)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13$

10. Following the above methodology, we develop the ontology structure for the following subsequent concepts:

Table19: List of subsequent concepts

CONCEPT NAME	CONCEPT FEATURE	BOOLEAN EQUATION
HeroBikes	$\{ hasMake^+ \} \rightarrow f^+2$ $\{ hasEngineCapacity^+ \} \rightarrow f^+3$ $\{ hasPower^+ \} \rightarrow f^+4$ $\{ hasPrice^+ \} \rightarrow f^+5$ $\{ hasFuelTankCapacity^+ \} \rightarrow f^+6$ $\{ hasMileage^+ \} \rightarrow f^+7$ $\{ hasBrakes^+ \} \rightarrow f^+8$ $\{ hasWeight^+ \} \rightarrow f^+9$ $\{ hasWheelType^+ \} \rightarrow f^+10$ $\{ hasIgnition^+ \} \rightarrow f^+11$ $\{ hasGears^+ \} \rightarrow f^+12$ $\{ hasMake^+ = Hero \} \rightarrow f^+14$	C (HeroBikes): $\{ f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+14 \}$

BajajBikes	{ hasMake$^+$ } → f$^+$ 2 { hasEngineCapacity$^+$ } → f$^+$ 3 { hasPower$^+$ } → f$^+$ 4 { hasPrice$^+$ } → f$^+$ 5 { hasFuelTankCapacity$^+$ } → f$^+$ 6 { hasMileage$^+$ } → f$^+$ 7 { hasBrakes$^+$ } → f$^+$ 8 { hasWeight$^+$ } → f$^+$ 9 { hasWheelType$^+$ } → f$^+$ 10 { hasIgnition$^+$ } → f$^+$ 11 { hasGears$^+$ } → f$^+$ 12 { hasMake$^+$ = Bajaj } → f$^+$ 15	C (BajajBikes): { f$^+$ 2 • f$^+$ 3 • f$^+$ 4 • f$^+$ 5 • f$^+$ 6 • f$^+$ 7 • f$^+$ 8 • f$^+$ 9 • f$^+$ 10 • f$^+$ 11 • f$^+$ 12 • f$^+$ 15 }
RoyalEnfieldBikes	{ hasMake$^+$ } → f$^+$ 2 { hasEngineCapacity$^+$ } → f$^+$ 3 { hasPower$^+$ } → f$^+$ 4 { hasPrice$^+$ } → f$^+$ 5 { hasFuelTankCapacity$^+$ } → f$^+$ 6 { hasMileage$^+$ } → f$^+$ 7 { hasBrakes$^+$ } → f$^+$ 8 { hasWeight$^+$ } → f$^+$ 9 { hasWheelType$^+$ } → f$^+$ 10 { hasIgnition$^+$ } → f$^+$ 11 { hasGears$^+$ } → f$^+$ 12 { hasMake$^+$ = RoyalEnfield} → f$^+$ 16	C (RoyalEnfieldBikes): { f$^+$ 2 • f$^+$ 3 • f$^+$ 4 • f$^+$ 5 • f$^+$ 6 • f$^+$ 7 • f$^+$ 8 • f$^+$ 9 • f$^+$ 10 • f$^+$ 11 • f$^+$ 12 • f$^+$ 16 }
YamahaBikes	{ hasMake$^+$ } → f$^+$ 2 { hasEngineCapacity$^+$ } → f$^+$ 3 { hasPower$^+$ } → f$^+$ 4 { hasPrice$^+$ } → f$^+$ 5 { hasFuelTankCapacity$^+$ } → f$^+$ 6 { hasMileage$^+$ } → f$^+$ 7 { hasBrakes$^+$ } → f$^+$ 8 { hasWeight$^+$ } → f$^+$ 9 { hasWheelType$^+$ } → f$^+$ 10 { hasIgnition$^+$ } → f$^+$ 11 { hasGears$^+$ } → f$^+$ 12 { hasMake$^+$ = Yamaha } → f$^+$ 17	C (YamahaBikes): { f$^+$ 2 • f$^+$ 3 • f$^+$ 4 • f$^+$ 5 • f$^+$ 6 • f$^+$ 7 • f$^+$ 8 • f$^+$ 9 • f$^+$ 10 • f$^+$ 11 • f$^+$ 12 • f$^+$ 17 }
TVSBikes	{ hasMake$^+$ } → f$^+$ 2 { hasEngineCapacity$^+$ } → f$^+$ 3 { hasPower$^+$ } → f$^+$ 4 { hasPrice$^+$ } → f$^+$ 5 { hasFuelTankCapacity$^+$ } → f$^+$ 6 { hasMileage$^+$ } → f$^+$ 7 { hasBrakes$^+$ } → f$^+$ 8 { hasWeight$^+$ } → f$^+$ 9 { hasWheelType$^+$ } → f$^+$ 10 { hasIgnition$^+$ } → f$^+$ 11	C (TVSBikes): { f$^+$ 2 • f$^+$ 3 • f$^+$ 4 • f$^+$ 5 • f$^+$ 6 • f$^+$ 7 • f$^+$ 8 • f$^+$ 9 • f$^+$ 10 • f$^+$ 11 • f$^+$ 12 • f$^+$ 18 }

	{ hasGears^{+} } → f^{+} 12 { hasMake^{+} = TVS } → f^{+} 18	
HondaBikes	{ hasMake^{+} } → f^{+} 2 { hasEngineCapacity^{+} } → f^{+} 3 { hasPower^{+} } → f^{+} 4 { hasPrice^{+} } → f^{+} 5 { hasFuelTankCapacity^{+} } → f^{+} 6 { hasMileage^{+} } → f^{+} 7 { hasBrakes^{+} } → f^{+} 8 { hasWeight^{+} } → f^{+} 9 { hasWheelType^{+} } → f^{+} 10 { hasIgnition^{+} } → f^{+} 11 { hasGears^{+} } → f^{+} 12 { hasMake^{+} = Honda } → f^{+} 19	C (HondaBikes): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 19 }

Fig. 49: Modified Ontology structure

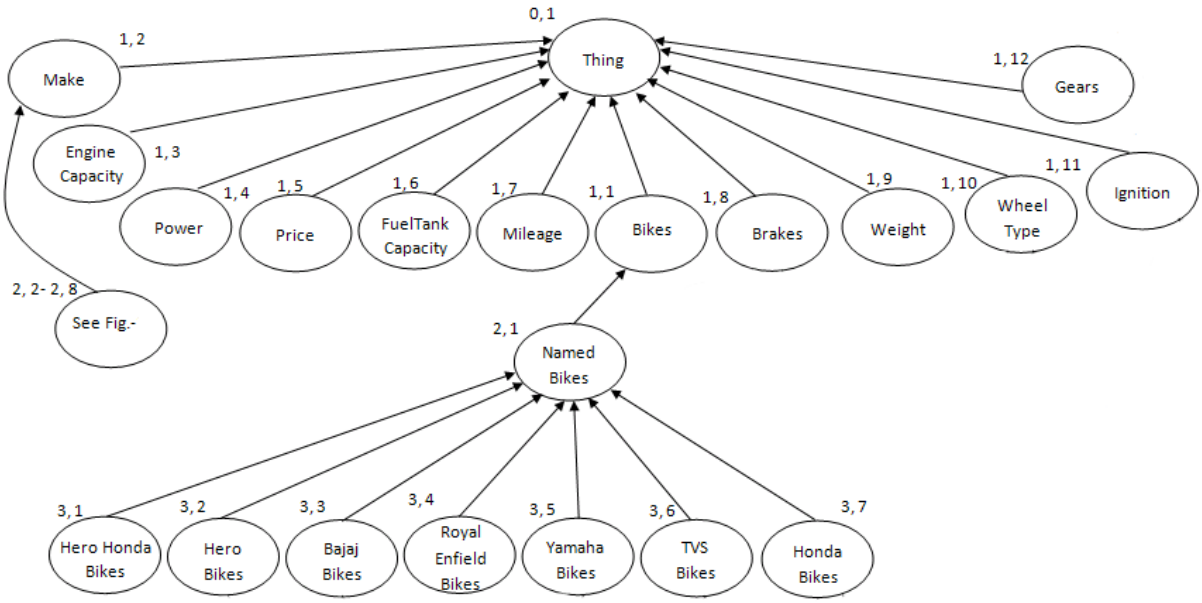


Table20: Modified Location map

CONCEPT ADDRESS	CONCEPT FEATURE
Thing (0, 1)	X
Bikes (1, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
Make (1, 2)	f^+2
EngineCapacity (1, 3)	f^+3
Power (1, 4)	f^+4
Price (1, 5)	f^+5
FuelTankCapacity (1, 6)	f^+6
Mileage (1, 7)	f^+7
Brakes (1, 8)	f^+8
Weight (1, 9)	f^+9
WheelType (1, 10)	f^+10
Ignition (1,11)	f^+11
Gears (1 ,12)	f^+12
NamedBikes (2, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
HeroHonda (2, 2)	$f^+2 \bullet f^+13$
Hero (2, 3)	$f^+2 \bullet f^+14$
Bajaj (2, 4)	$f^+2 \bullet f^+15$
RoyalEnfield (2, 5)	$f^+2 \bullet f^+16$
Yamaha (2, 6)	$f^+2 \bullet f^+17$
TVS (2, 7)	$f^+2 \bullet f^+18$
Honda (2, 8)	$f^+2 \bullet f^+19$
HeroHondaBikes (3, 1)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13$
HeroBikes (3, 2)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+14$
BajajBikes (3, 3)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15$
RoyalEnfieldBikes (3, 4)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+16$
YamahaBikes (3, 5)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+17$
TVSBikes (3, 6)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+18$
HondaBikes (3, 7)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+19$

11. Concept encountered: *Combo* with concept features:

$\{hasBrakes^+\} \rightarrow f^+8$

$\{hasBrakes^+ = Combo\} \rightarrow f^+20$

(a.) This can be represented as a Boolean equation as:
C (Combo): $\{ f^+8 \bullet f^+20 \}$

(b.) Matching the above Boolean equation with concepts *Thing, Bikes, Make, EngineCapacity, Power, Price, FuelTankCapacity, Mileage, Brakes, Weight, WheelType,*

Ignition, *Gears* and *NamedBikes* in the location map. It is then found that C (Combo): { $f^+8 \bullet f^+20$ } matches C (Brakes): { f^+8}, thus *Combo* is a sub-concept of *Brakes.*

(c.) The location of new concept Combo is (2, 9). The new concept is added to the location (2, 9) in location map as below:

Fig. 50: Modified Ontology structure

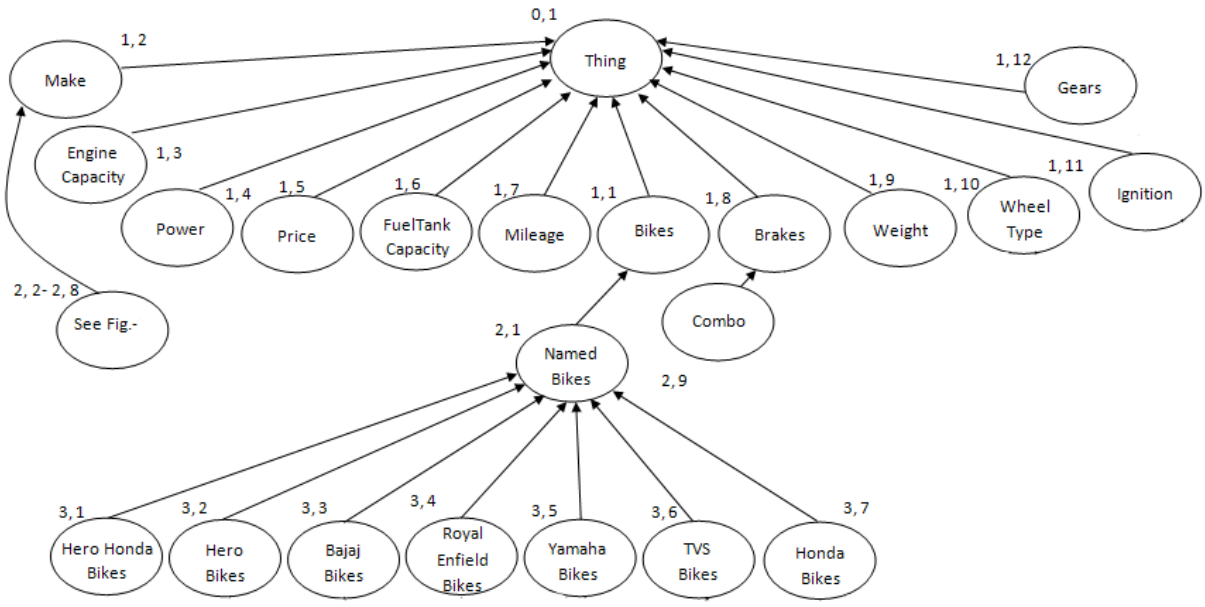


Table21: Modified Location map

CONCEPT ADDRESS	CONCEPT FEATURE
Thing (0, 1)	X
Bikes (1, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
Make (1, 2)	f^+2
EngineCapacity (1, 3)	f^+3
Power (1, 4)	f^+4
Price (1, 5)	f^+5
FuelTankCapacity (1, 6)	f^+6
Mileage (1, 7)	f^+7
Brakes (1, 8)	f^+8
Weight (1, 9)	f^+9
WheelType (1, 10)	f^+10
Ignition (1,11)	f^+11
Gears (1 ,12)	f^+12
NamedBikes (2, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
HeroHonda (2, 2)	$f^+2 \bullet f^+13$
Hero (2, 3)	$f^+2 \bullet f^+14$

Bajaj (2, 4)	f^+2 • f^+15
RoyalEnfield (2, 5)	f^+2 • f^+16
Yamaha (2, 6)	f^+2 • f^+17
TVS (2, 7)	f^+2 • f^+18
Honda (2, 8)	f^+2 • f^+19
HeroHondaBikes (3, 1)	f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12 • f^+13
HeroBikes (3, 2)	f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12 • f^+14
BajajBikes (3, 3)	f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12 • f^+15
RoyalEnfieldBikes (3, 4)	f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12 • f^+16
YamahaBikes (3, 5)	f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12 • f^+17
TVSBikes (3, 6)	f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12 • f^+18
HondaBikes (3, 7)	f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12 • f^+19
Combo (2, 9)	f^+8 • f^+20

12. Following the above methodology, we develop the ontology structure for the following subsequent concepts:

Table22: List of subsequent concepts

CONCEPT NAME	CONCEPT FEATURE	BOOLEAN EQUATION
DiskBrakes	$\{hasBrakes^+\} \rightarrow f^+8$ $\{hasBrakes^+ = DiskBrakes\} \rightarrow f^+21$	C (DiskBrakes): $\{f^+8 \bullet f^+21\}$
DrumBrakes	$\{hasBrakes^+\} \rightarrow f^+8$ $\{hasBrakes^+ = DrumBrakes\} \rightarrow f^+22$	C (DrumBrakes): $\{f^+8 \bullet f^+22\}$
Alloy	$\{hasWheelType^+\} \rightarrow f^+10$ $\{hasWheelType^+ = Alloy\} \rightarrow f^+23$	C (Alloy): $\{f^+10 \bullet f^+23\}$
WireSpoke	$\{hasWheelType^+\} \rightarrow f^+10$ $\{hasWheelType^+ = WireSpoke\} \rightarrow f^+24$	C (WireSpoke): $\{f^+10 \bullet f^+24\}$
Self	$\{hasIgnition^+\} \rightarrow f^+11$ $\{hasIgnition^+ = Self\} \rightarrow f^+25$	C (Self): $\{f^+11 \bullet f^+25\}$
Kick	$\{hasIgnition^+\} \rightarrow f^+11$ $\{hasIgnition^+ = Kick\} \rightarrow f^+26$	C (Kick): $\{f^+11 \bullet f^+26\}$
6	$\{hasGears^+\} \rightarrow f^+12$ $\{hasGears^+ = 6\} \rightarrow f^+27$	C (6): $\{f^+12 \bullet f^+27\}$
5	$\{hasGears^+\} \rightarrow f^+12$ $\{hasGears^+ = 5\} \rightarrow f^+28$	C (5): $\{f^+12 \bullet f^+28\}$
4	$\{hasGears^+\} \rightarrow f^+12$ $\{hasGears^+ = 4\} \rightarrow f^+29$	C (4): $\{f^+12 \bullet f^+29\}$

Fig. 51: Modified Ontology structure

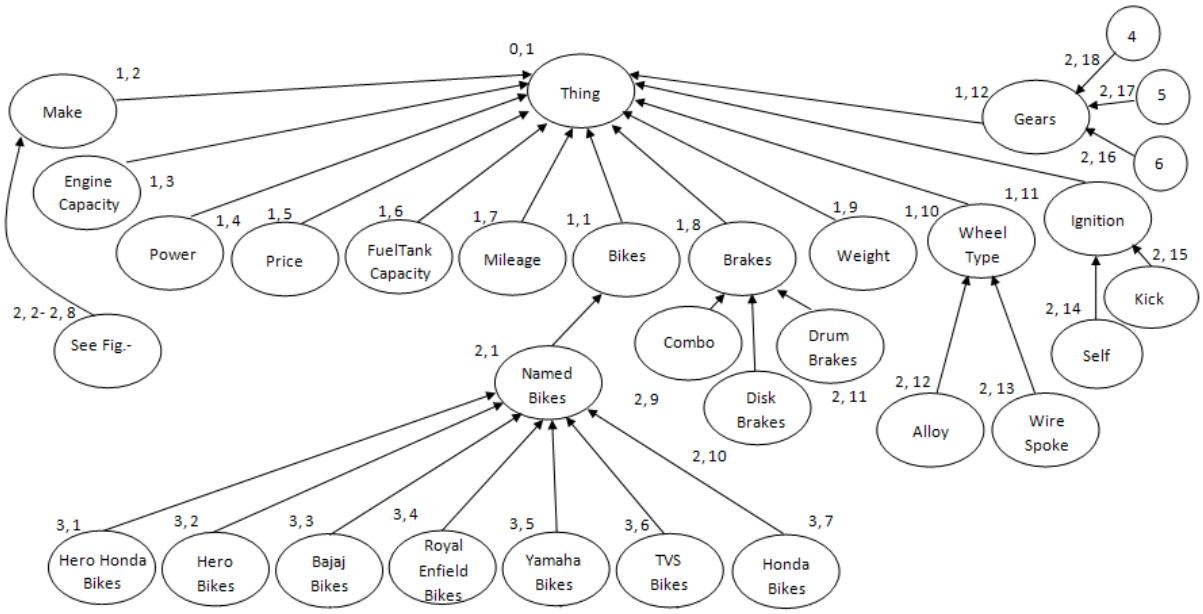


Table23: Modified Location map

CONCEPT ADDRESS	CONCEPT FEATURE
Thing (0, 1)	X
Bikes (1, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
Make (1, 2)	f^+2
EngineCapacity (1, 3)	f^+3
Power (1, 4)	f^+4
Price (1, 5)	f^+5
FuelTankCapacity (1, 6)	f^+6
Mileage (1, 7)	f^+7
Brakes (1, 8)	f^+8
Weight (1, 9)	f^+9
WheelType (1, 10)	f^+10
Ignition (1,11)	f^+11
Gears (1 ,12)	f^+12
NamedBikes (2, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
HeroHonda (2, 2)	$f^+2 \bullet f^+13$
Hero (2, 3)	$f^+2 \bullet f^+14$
Bajaj (2, 4)	$f^+2 \bullet f^+15$
RoyalEnfield (2, 5)	$f^+2 \bullet f^+16$
Yamaha (2, 6)	$f^+2 \bullet f^+17$
TVS (2, 7)	$f^+2 \bullet f^+18$
Honda (2, 8)	$f^+2 \bullet f^+19$
HeroHondaBikes (3, 1)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13$
HeroBikes (3, 2)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+14$
BajajBikes (3, 3)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15$

RoyalEnfieldBikes (3, 4)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+16$
YamahaBikes (3, 5)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+17$
TVSBikes (3, 6)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+18$
HondaBikes (3, 7)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+19$
Combo (2, 9)	$f^+8 \bullet f^+20$
DiskBrakes (2, 10)	$f^+8 \bullet f^+21$
DrumBrakes (2, 11)	$f^+8 \bullet f^+22$
Alloy (2, 12)	$f^+10 \bullet f^+23$
WireSpoke (2, 13)	$f^+10 \bullet f^+24$
Self (2, 14)	$f^+11 \bullet f^+25$
Kick (2, 15)	$f^+11 \bullet f^+26$
6 (2, 16)	$f^+12 \bullet f^+27$
5 (2, 17)	$f^+12 \bullet f^+28$
4 (2, 18)	$f^+12 \bullet f^+29$

13. Concept encountered: *Karizma(Normal)Model* with concept features:

$\{ hasMake^+ \} \rightarrow f^+2$

$\{ hasEngineCapacity^+ \} \rightarrow f^+3$

$\{ hasPower^+ \} \rightarrow f^+4$

$\{ hasPrice^+ \} \rightarrow f^+5$

$\{ hasFuelTankCapacity^+ \} \rightarrow f^+6$

$\{ hasMileage^+ \} \rightarrow f^+7$

$\{ hasBrakes^+ \} \rightarrow f^+8$

$\{ hasWeight^+ \} \rightarrow f^+9$

$\{ hasWheelType^+ \} \rightarrow f^+10$

$\{ hasIgnition^+ \} \rightarrow f^+11$

$\{ hasGears^+ \} \rightarrow f^+12$

$\{ hasMake^+ = HeroHonda \} \rightarrow f^+13$

$\{ hasBrakes^+ = Combo \} \rightarrow f^+20$

$\{ hasWheelType^+ = Alloy \} \rightarrow f^+23$

$\{ hasIgnition^+ = Self \} \rightarrow f^+25$

{ hasGears^{+} = 5} → f^{+} 28

(a.) This can be represented as a Boolean equation as:
C (Karizma(Normal)Model): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 13 • f^{+} 20 • f^{+} 23 • f^{+} 25 • f^{+} 28}

(b.) Matching the above Boolean equation with concepts in the location map, it is found that C (Karizma(Normal)Model) matches C (HeroHondaBikes), thus Karizma(Normal)Model is a sub-concept of *HeroHondaBikes.*

(c.) The location of new concept Karizma(Normal)Model is (4, 1). The new concept is added to the location (4, 1) in location map as below:

Fig. 52: Modified Ontology structure

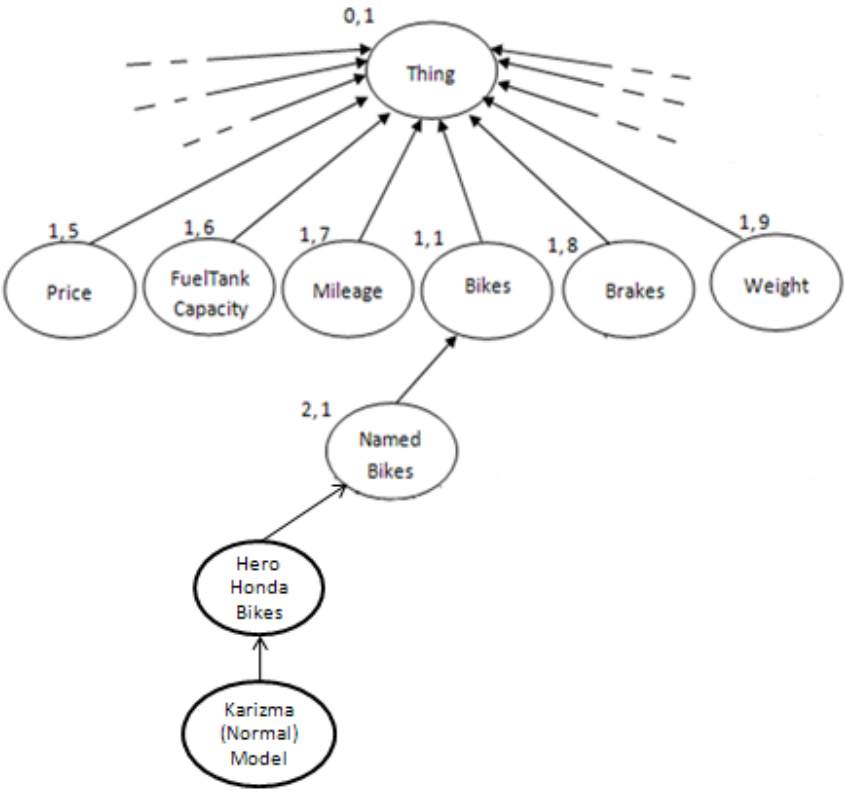


Table24: Modified Location map

CONCEPT ADDRESS	CONCEPT FEATURE
Thing (0, 1)	X
Bikes (1, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
Make (1, 2)	f^+2
EngineCapacity (1, 3)	f^+3
Power (1, 4)	f^+4
Price (1, 5)	f^+5
FuelTankCapacity (1, 6)	f^+6
Mileage (1, 7)	f^+7
Brakes (1, 8)	f^+8
Weight (1, 9)	f^+9
WheelType (1, 10)	f^+10
Ignition (1,11)	f^+11
Gears (1 ,12)	f^+12
NamedBikes (2, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
HeroHonda (2, 2)	$f^+2 \bullet f^+13$
Hero (2, 3)	$f^+2 \bullet f^+14$
Bajaj (2, 4)	$f^+2 \bullet f^+15$
RoyalEnfield (2, 5)	$f^+2 \bullet f^+16$
Yamaha (2, 6)	$f^+2 \bullet f^+17$
TVS (2, 7)	$f^+2 \bullet f^+18$
Honda (2, 8)	$f^+2 \bullet f^+19$
HeroHondaBikes (3, 1)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13$
HeroBikes (3, 2)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+14$
BajajBikes (3, 3)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15$
RoyalEnfieldBikes (3, 4)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+16$
YamahaBikes (3, 5)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+17$
TVSBikes (3, 6)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+18$
HondaBikes (3, 7)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+19$
Combo (2, 9)	$f^+8 \bullet f^+20$
DiskBrakes (2, 10)	$f^+8 \bullet f^+21$
DrumBrakes (2, 11)	$f^+8 \bullet f^+22$
Alloy (2, 12)	$f^+10 \bullet f^+23$
WireSpoke (2, 13)	$f^+10 \bullet f^+24$
Self (2, 14)	$f^+11 \bullet f^+25$
Kick (2, 15)	$f^+11 \bullet f^+26$
6 (2, 16)	$f^+12 \bullet f^+27$
5 (2, 17)	$f^+12 \bullet f^+28$
4 (2, 18)	$f^+12 \bullet f^+29$
Karizma(Normal)Model (4, 1)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+28$

14. Concept encountered: *Karizma(ZMR)Model* with concept features:

{ $hasMake^+$ } → $f^+ 2$

{ $hasEngineCapacity^+$ } → $f^+ 3$

{ $hasPower^+$ } → $f^+ 4$

{ $hasPrice^+$ } → $f^+ 5$

{ $hasFuelTankCapacity^+$ } → $f^+ 6$

{ $hasMileage^+$ } → $f^+ 7$

{ $hasBrakes^+$ } → $f^+ 8$

{ $hasWeight^+$ } → $f^+ 9$

{ $hasWheelType^+$ } → $f^+ 10$

{ $hasIgnition^+$ } → $f^+ 11$

{ $hasGears^+$ } → $f^+ 12$

{ $hasMake^+$ = HeroHonda } → $f^+ 13$

{ $hasBrakes^+$ = DiskBrakes } → $f^+ 21$

{ $hasWheelType^+$ = Alloy} → $f^+ 23$

{ $hasIgnition^+$ = Self} → $f^+ 25$

{ $hasGears^+$ = 5} → $f^+ 28$

(a.) This can be represented as a Boolean equation as:
C (Karizma(ZMR)Model): { $f^+ 2 \bullet f^+ 3 \bullet f^+ 4 \bullet f^+ 5 \bullet f^+ 6 \bullet f^+ 7 \bullet f^+ 8 \bullet f^+ 9 \bullet f^+ 10 \bullet f^+ 11 \bullet f^+ 12 \bullet f^+ 13 \bullet f^+ 21 \bullet f^+ 23 \bullet f^+ 25 \bullet f^+ 28$}

(b.) Matching the above Boolean equation with concepts in the location map, it is found that C (Karizma(ZMR)Model) matches C (HeroHondaBikes), thus Karizma(ZMR)Model is a sub-concept of *HeroHondaBikes* and a brother concept of *Karizma(Normal)Model*.

(c.) The location of new concept Karizma(ZMR)Model is (4, 2). The new concept is added to the location (4, 2) in location map as below:

Fig. 53: Modified Ontology structure

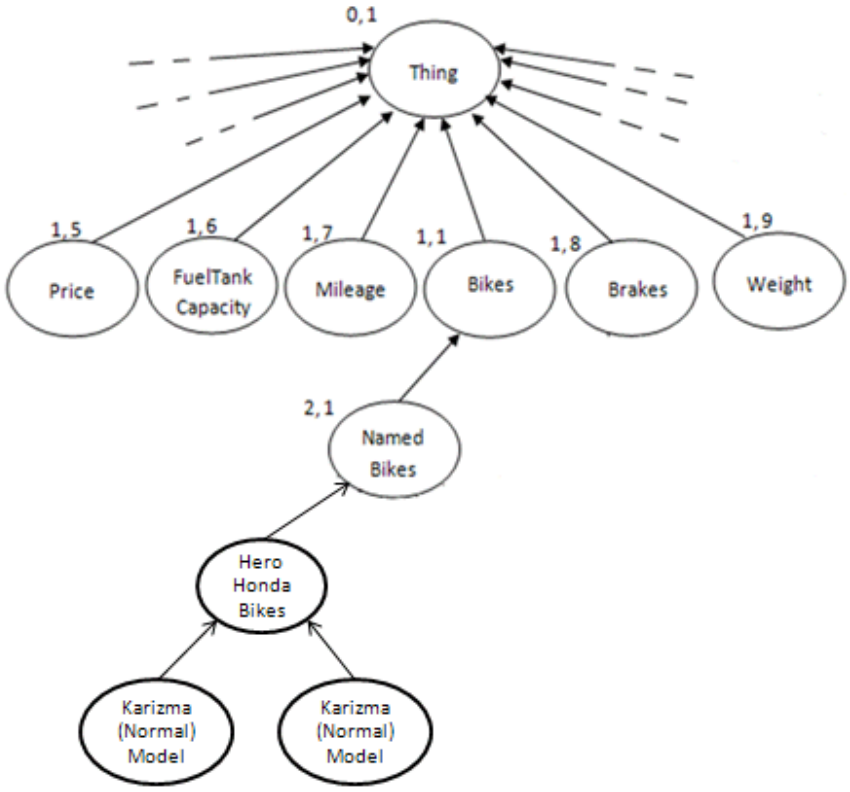


Table25: Modified Location map

CONCEPT ADDRESS	CONCEPT FEATURE
Thing (0, 1)	X
Bikes (1, 1)	f^+1 • f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12
Make (1, 2)	f^+2
EngineCapacity (1, 3)	f^+3
Power (1, 4)	f^+4
Price (1, 5)	f^+5
FuelTankCapacity (1, 6)	f^+6
Mileage (1, 7)	f^+7
Brakes (1, 8)	f^+8
Weight (1, 9)	f^+9
WheelType (1, 10)	f^+10
Ignition (1,11)	f^+11
Gears (1 ,12)	f^+12
NamedBikes (2, 1)	f^+1 • f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12
HeroHonda (2, 2)	f^+2 • f^+13
Hero (2, 3)	f^+2 • f^+14
Bajaj (2, 4)	f^+2 • f^+15
RoyalEnfield (2, 5)	f^+2 • f^+16

Yamaha (2, 6)	$f^+2 \bullet f^+17$
TVS (2, 7)	$f^+2 \bullet f^+18$
Honda (2, 8)	$f^+2 \bullet f^+19$
HeroHondaBikes (3, 1)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13$
HeroBikes (3, 2)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+14$
BajajBikes (3, 3)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15$
RoyalEnfieldBikes (3, 4)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+16$
YamahaBikes (3, 5)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+17$
TVSBikes (3, 6)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+18$
HondaBikes (3, 7)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+19$
Combo (2, 9)	$f^+8 \bullet f^+20$
DiskBrakes (2, 10)	$f^+8 \bullet f^+21$
DrumBrakes (2, 11)	$f^+8 \bullet f^+22$
Alloy (2, 12)	$f^+10 \bullet f^+23$
WireSpoke (2, 13)	$f^+10 \bullet f^+24$
Self (2, 14)	$f^+11 \bullet f^+25$
Kick (2, 15)	$f^+11 \bullet f^+26$
6 (2, 16)	$f^+12 \bullet f^+27$
5 (2, 17)	$f^+12 \bullet f^+28$
4 (2, 18)	$f^+12 \bullet f^+29$
Karizma(Normal)Model (4, 1)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+28$
Karizma(ZMR)Model (4, 2)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+21 \bullet f^+23 \bullet f^+25 \bullet f^+28$

15. Following the above methodology, we develop the ontology structure for the following subsequent concepts:

Table26: List of subsequent concepts

CONCEPT NAME	CONCEPT FEATURE	BOOLEAN EQUATION
Splendor(Plus)Model	$\{ hasMake^+ \} \rightarrow f^+2$ $\{ hasEngineCapacity^+ \} \rightarrow f^+3$ $\{ hasPower^+ \} \rightarrow f^+4$ $\{ hasPrice^+ \} \rightarrow f^+5$ $\{ hasFuelTankCapacity^+ \} \rightarrow f^+6$ $\{ hasMileage^+ \} \rightarrow f^+7$ $\{ hasBrakes^+ \} \rightarrow f^+8$ $\{ hasWeight^+ \} \rightarrow f^+9$ $\{ hasWheelType^+ \} \rightarrow f^+10$ $\{ hasIgnition^+ \} \rightarrow f^+11$ $\{ hasGears^+ \} \rightarrow f^+12$ $\{ hasMake^+ = HeroHonda \} \rightarrow f^+13$ $\{ hasBrakes^+ = DrumBrakes \} \rightarrow f^+22$ $\{ hasWheelType^+ = WireSpoke \} \rightarrow f^+24$ $\{ hasIgnition^+ = Kick \} \rightarrow f^+26$	C (Splendor(Plus)Model): $\{ f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+22 \bullet f^+24 \bullet f^+26 \bullet f^+29 \}$

	{ hasGears^{+} =4} → f^{+} 29	
Splendor(NXG)Model	{ hasMake^{+} } → f^{+} 2 { hasEngineCapacity^{+} } → f^{+} 3 { hasPower^{+} } → f^{+} 4 { hasPrice^{+} } → f^{+} 5 { hasFuelTankCapacity^{+} } → f^{+} 6 { hasMileage^{+} } → f^{+} 7 { hasBrakes^{+} } → f^{+} 8 { hasWeight^{+} } → f^{+} 9 { hasWheelType^{+} } → f^{+} 10 { hasIgnition^{+} } → f^{+} 11 { hasGears^{+} } → f^{+} 12 { hasMake^{+} = HeroHonda } → f^{+} 13 { hasBrakes^{+} =DrumBrakes} → f^{+} 22 { hasWheelType^{+} =Alloy} → f^{+} 23 { hasIgnition^{+} =Kick} → f^{+} 26 { hasGears^{+} =4} → f^{+} 29	C (Splendor(NXG)Model): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 13 • f^{+} 23 • f^{+} 23 • f^{+} 26 • f^{+} 29 }
Splendor(Super)Model	{ hasMake^{+} } → f^{+} 2 { hasEngineCapacity^{+} } → f^{+} 3 { hasPower^{+} } → f^{+} 4 { hasPrice^{+} } → f^{+} 5 { hasFuelTankCapacity^{+} } → f^{+} 6 { hasMileage^{+} } → f^{+} 7 { hasBrakes^{+} } → f^{+} 8 { hasWeight^{+} } → f^{+} 9 { hasWheelType^{+} } → f^{+} 10 { hasIgnition^{+} } → f^{+} 11 { hasGears^{+} } → f^{+} 12 { hasMake^{+} = HeroHonda } → f^{+} 13 { hasBrakes^{+} =DrumBrakes} → f^{+} 22 { hasWheelType^{+} =Alloy} → f^{+} 23 { hasIgnition^{+} =Self} → f^{+} 25 { hasGears^{+} =4} → f^{+} 29	C (Splendor(Super)Model): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 13 • f^{+} 22 • f^{+} 23 • f^{+} 25 • f^{+} 29 }
Splendor(Pro)Model	{ hasMake^{+} } → f^{+} 2 { hasEngineCapacity^{+} } → f^{+} 3 { hasPower^{+} } → f^{+} 4 { hasPrice^{+} } → f^{+} 5 { hasFuelTankCapacity^{+} } → f^{+} 6 { hasMileage^{+} } → f^{+} 7 { hasBrakes^{+} } → f^{+} 8 { hasWeight^{+} } → f^{+} 9 { hasWheelType^{+} } → f^{+} 10 { hasIgnition^{+} } → f^{+} 11 { hasGears^{+} } → f^{+} 12 { hasMake^{+} = HeroHonda } → f^{+} 13 { hasBrakes^{+} = DrumBrakes} → f^{+} 22 { hasWheelType^{+} = Alloy} → f^{+} 23 { hasIgnition^{+} = Self} → f^{+} 25 { hasGears^{+} = 4} → f^{+} 29	C (Splendor(Pro)Model): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 13 • f^{+} 22 • f^{+} 23 • f^{+} 25 • f^{+} 29 }
PassionProModel	{ hasMake^{+} } → f^{+} 2	C (PassionProModel):

	{ hasEngineCapacity$^+$ } → f^+3 { hasPower$^+$ } → f^+4 { hasPrice$^+$ } → f^+5 { hasFuelTankCapacity$^+$ } → f^+6 { hasMileage$^+$ } → f^+7 { hasBrakes$^+$ } → f^+8 { hasWeight$^+$ } → f^+9 { hasWheelType$^+$ } → f^+10 { hasIgnition$^+$ } → f^+11 { hasGears$^+$ } → f^+12 { hasMake$^+$ = HeroHonda } → f^+13 { hasBrakes$^+$ = Combo} → f^+20 { hasWheelType$^+$ = Alloy} → f^+23 { hasIgnition$^+$ = Self} → f^+25 { hasGears$^+$ = 4} → f^+29	{ f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12 • f^+13 • f^+20 • f^+23 • f^+25 • f^+29 }
CD-DawnModel	{ hasMake$^+$ } → f^+2 { hasEngineCapacity$^+$ } → f^+3 { hasPower$^+$ } → f^+4 { hasPrice$^+$ } → f^+5 { hasFuelTankCapacity$^+$ } → f^+6 { hasMileage$^+$ } → f^+7 { hasBrakes$^+$ } → f^+8 { hasWeight$^+$ } → f^+9 { hasWheelType$^+$ } → f^+10 { hasIgnition$^+$ } → f^+11 { hasGears$^+$ } → f^+12 { hasMake$^+$ = HeroHonda } → f^+13 { hasBrakes$^+$ = DrumBrakes} → f^+22 { hasWheelType$^+$ = WireSpoke} → f^+24 { hasIgnition$^+$ = Kick} → f^+26 { hasGears$^+$ =4} → f^+29	C (CD-DawnModel): { f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12 • f^+13 • f^+22 • f^+24 • f^+26 • f^+29 }
CD-DeluxeModel	{ hasMake$^+$ } → f^+2 { hasEngineCapacity$^+$ } → f^+3 { hasPower$^+$ } → f^+4 { hasPrice$^+$ } → f^+5 { hasFuelTankCapacity$^+$ } → f^+6 { hasMileage$^+$ } → f^+7 { hasBrakes$^+$ } → f^+8 { hasWeight$^+$ } → f^+9 { hasWheelType$^+$ } → f^+10 { hasIgnition$^+$ } → f^+11 { hasGears$^+$ } → f^+12 { hasMake$^+$ = HeroHonda } → f^+13 { hasBrakes$^+$ = DrumBrakes} → f^+22 { hasWheelType$^+$ = Alloy} → f^+23 { hasIgnition$^+$ =Kick} → f^+26 { hasGears$^+$ = 4} → f^+29	C (CD-DeluxeModel): { f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 • f^+8 • f^+9 • f^+10 • f^+11 • f^+12 • f^+13 • f^+22 • f^+23 • f^+26 • f^+29 }
Glamour(Normal)Model	{ hasMake$^+$ } → f^+2 { hasEngineCapacity$^+$ } → f^+3	C (Glamour(Normal)Model): { f^+2 • f^+3 • f^+4 • f^+5 • f^+6 • f^+7 •

	{ hasPower$^+$ } → f^+ 4 { hasPrice$^+$ } → f^+ 5 { hasFuelTankCapacity$^+$ } → f^+ 6 { hasMileage$^+$ } → f^+ 7 { hasBrakes$^+$ } → f^+ 8 { hasWeight$^+$ } → f^+ 9 { hasWheelType$^+$ } → f^+ 10 { hasIgnition$^+$ } → f^+ 11 { hasGears$^+$ } → f^+ 12 { hasMake$^+$ = HeroHonda } → f^+ 13 { hasBrakes$^+$ =Combo} → f^+ 20 { hasWheelType$^+$ =Alloy} → f^+ 23 { hasIgnition$^+$ = Self} → f^+ 25 { hasGears$^+$ = 4} → f^+ 29	f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 13 • f^+ 20 • f^+ 23 • f^+ 25 • f^+ 29 }
Glamour(PGMFi)Model	{ hasMake$^+$ } → f^+ 2 { hasEngineCapacity$^+$ } → f^+ 3 { hasPower$^+$ } → f^+ 4 { hasPrice$^+$ } → f^+ 5 { hasFuelTankCapacity$^+$ } → f^+ 6 { hasMileage$^+$ } → f^+ 7 { hasBrakes$^+$ } → f^+ 8 { hasWeight$^+$ } → f^+ 9 { hasWheelType$^+$ } → f^+ 10 { hasIgnition$^+$ } → f^+ 11 { hasGears$^+$ } → f^+ 12 { hasMake$^+$ = HeroHonda } → f^+ 13 { hasBrakes$^+$ =Combo} → f^+ 20 { hasWheelType$^+$ =Alloy} → f^+ 23 { hasIgnition$^+$ = Self} → f^+ 25 { hasGears$^+$ = 4} → f^+ 29	C (Glamour(PGMFi)Model): { f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 13 • f^+ 20 • f^+ 23 • f^+ 25 • f^+ 29 }
AchieverModel	{ hasMake$^+$ } → f^+ 2 { hasEngineCapacity$^+$ } → f^+ 3 { hasPower$^+$ } → f^+ 4 { hasPrice$^+$ } → f^+ 5 { hasFuelTankCapacity$^+$ } → f^+ 6 { hasMileage$^+$ } → f^+ 7 { hasBrakes$^+$ } → f^+ 8 { hasWeight$^+$ } → f^+ 9 { hasWheelType$^+$ } → f^+ 10 { hasIgnition$^+$ } → f^+ 11 { hasGears$^+$ } → f^+ 12 { hasMake$^+$ = HeroHonda } → f^+ 13 { hasBrakes$^+$ =Combo} → f^+ 20 { hasWheelType$^+$ =Alloy} → f^+ 23 { hasIgnition$^+$ = Self} → f^+ 25 { hasGears$^+$ = 5} → f^+ 28	C (AchieverModel): { f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 13 • f^+ 20 • f^+ 23 • f^+ 25 • f^+ 28 }
CBZXtremeModel	{ hasMake$^+$ } → f^+ 2 { hasEngineCapacity$^+$ } → f^+ 3 { hasPower$^+$ } → f^+ 4 { hasPrice$^+$ } → f^+ 5	C (CNZXtremeModel): { f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 13 • f^+ 20 • f^+ 23 • f^+ 25 • f^+ 28 }

	{ hasFuelTankCapacity$^+$ } → f^+ 6 { hasMileage$^+$ } → f^+ 7 { hasBrakes$^+$ } → f^+ 8 { hasWeight$^+$ } → f^+ 9 { hasWheelType$^+$ } → f^+ 10 { hasIgnition$^+$ } → f^+ 11 { hasGears$^+$ } → f^+ 12 { hasMake$^+$ = HeroHonda } → f^+ 13 { hasBrakes$^+$ = Combo} → f^+ 20 { hasWheelType$^+$ = Alloy} → f^+ 23 { hasIgnition$^+$ = Self} → f^+ 25 { hasGears$^+$ = 5} → f^+ 28	
HunkModel	{ hasMake$^+$ } → f^+ 2 { hasEngineCapacity$^+$ } → f^+ 3 { hasPower$^+$ } → f^+ 4 { hasPrice$^+$ } → f^+ 5 { hasFuelTankCapacity$^+$ } → f^+ 6 { hasMileage$^+$ } → f^+ 7 { hasBrakes$^+$ } → f^+ 8 { hasWeight$^+$ } → f^+ 9 { hasWheelType$^+$ } → f^+ 10 { hasIgnition$^+$ } → f^+ 11 { hasGears$^+$ } → f^+ 12 { hasMake$^+$ = HeroHonda } → f^+ 13 { hasBrakes$^+$ = Disk} → f^+ 21 { hasWheelType$^+$ = Alloy} → f^+ 23 { hasIgnition$^+$ =Self} → f^+ 25 { hasGears$^+$ =5} → f^+ 28	C (HunkModel): { f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 13 • f^+ 21 • f^+ 23 • f^+ 25 • f^+ 28 }

Fig. 54: Modified Ontology structure

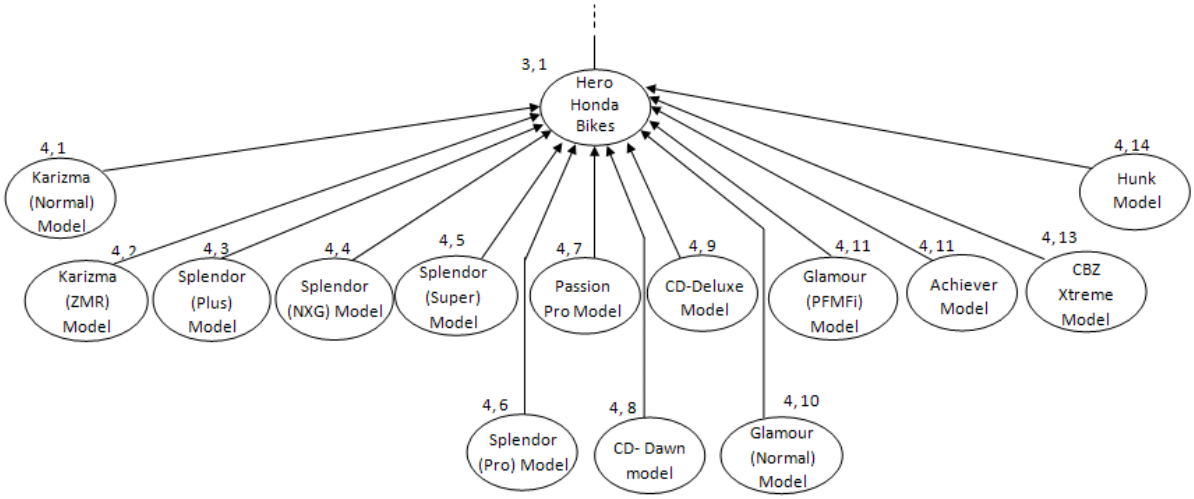


Table27: Modified Location map

CONCEPT ADDRESS	CONCEPT FEATURE
Thing (0, 1)	X
Bikes (1, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
Make (1, 2)	f^+2
EngineCapacity (1, 3)	f^+3
Power (1, 4)	f^+4
Price (1, 5)	f^+5
FuelTankCapacity (1, 6)	f^+6
Mileage (1, 7)	f^+7
Brakes (1, 8)	f^+8
Weight (1, 9)	f^+9
WheelType (1, 10)	f^+10
Ignition (1,11)	f^+11
Gears (1 ,12)	f^+12
NamedBikes (2, 1)	$f^+1 \bullet f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12$
HeroHonda (2, 2)	$f^+2 \bullet f^+13$
Hero (2, 3)	$f^+2 \bullet f^+14$
Bajaj (2, 4)	$f^+2 \bullet f^+15$
RoyalEnfield (2, 5)	$f^+2 \bullet f^+16$
Yamaha (2, 6)	$f^+2 \bullet f^+17$
TVS (2, 7)	$f^+2 \bullet f^+18$
Honda (2, 8)	$f^+2 \bullet f^+19$
HeroHondaBikes (3, 1)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13$
HeroBikes (3, 2)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+14$
BajajBikes (3, 3)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15$
RoyalEnfieldBikes (3, 4)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+16$
YamahaBikes (3, 5)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+17$

TVSBikes (3, 6)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+18$
HondaBikes (3, 7)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+19$
Combo (2, 9)	$f^+8 \bullet f^+20$
DiskBrakes (2, 10)	$f^+8 \bullet f^+21$
DrumBrakes (2, 11)	$f^+8 \bullet f^+22$
Alloy (2, 12)	$f^+10 \bullet f^+23$
WireSpoke (2, 13)	$f^+10 \bullet f^+24$
Self (2, 14)	$f^+11 \bullet f^+25$
Kick (2, 15)	$f^+11 \bullet f^+26$
6 (2, 16)	$f^+12 \bullet f^+27$
5 (2, 17)	$f^+12 \bullet f^+28$
4 (2, 18)	$f^+12 \bullet f^+29$
Karizma(Normal)Model (4, 1)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+28$
Karizma(ZMR)Model (4, 2)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+21 \bullet f^+23 \bullet f^+25 \bullet f^+28$
Splendor(Plus)Model (4, 3)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+22 \bullet f^+24 \bullet f^+26 \bullet f^+29$
Splendor(NXG)Model (4, 4)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+23 \bullet f^+23 \bullet f^+26 \bullet f^+29$
Splendor(Super)Model (4, 5)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+22 \bullet f^+23 \bullet f^+25 \bullet f^+29$
Splendor(Pro)Model (4, 6)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+22 \bullet f^+23 \bullet f^+25 \bullet f^+29$
PassionProModel (4, 7)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+29$
CD-DawnModel (4, 8)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+22 \bullet f^+24 \bullet f^+26 \bullet f^+29$
CD-DeluxeModel (4, 9)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+22 \bullet f^+23 \bullet f^+26 \bullet f^+29$
Glamour(Normal)Model (4, 10)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+29$
Glamour(PGMFi)Model (4, 11)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+29$
AchieverModel (4, 12)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+28$
CBZXtremeModel (4, 13)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+28$
HunkModel (4, 14)	$f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+13 \bullet f^+21 \bullet f^+23 \bullet f^+25 \bullet f^+28$

16. So building the ontology structure for all remaining concepts, based on the above methodology:

Table28: List of subsequent concepts

CONCEPT NAME	CONCEPT FEATURE	BOOLEAN EQUATION
ImpulseModel	{ $hasMake^+$ = Hero} → f^+ 14 { $hasBrakes^+$ =Combo} → f^+ 20 { $hasWheelType^+$ = Alloy} → f^+ 23 { $hasIgnition^+$ = Self} → f^+ 25 { $hasGears^+$ = 5} → f^+ 28	C (ImpulseModel): { $f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+14 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+28$ }
CT100Model	{ $hasMake^+$ = Bajaj} → f^+ 15 { $hasBrakes^+$ = DrumBrakes} → f^+ 22 { $hasWheelType^+$ = WireSpoke} → f^+ 24 { $hasIgnition^+$ = Kick} → f^+ 26 { $hasGears^+$ = 4} → f^+ 29	C (CT100Model): { $f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15 \bullet f^+22 \bullet f^+24 \bullet f^+26 \bullet f^+29$ }
Pulsar135LSModel	{ $hasMake^+$ = Bajaj} → f^+ 15 { $hasBrakes^+$ =Combo} → f^+ 20 { $hasWheelType^+$ = Alloy} → f^+ 23 { $hasIgnition^+$ = Self} → f^+ 25 { $hasGears^+$ = 5} → f^+ 28	C (Pulsar135LSModel): { $f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+28$ }
Pulsar150DTS-iModel	{ $hasMake^+$ = Bajaj} → f^+ 15 { $hasBrakes^+$ =Combo} → f^+ 20 { $hasWheelType^+$ = Alloy} → f^+ 23 { $hasIgnition^+$ = Self} → f^+ 25 { $hasGears^+$ = 5} → f^+ 28	C (Pulsar150DTS-iModel): { $f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+28$ }
Pulsar180DTS-iModel	{ $hasMake^+$ = Bajaj} → f^+ 15 { $hasBrakes^+$ =Combo} → f^+ 20 { $hasWheelType^+$ = Alloy} → f^+ 23 { $hasIgnition^+$ = Self} → f^+ 25 { $hasGears^+$ = 5} → f^+ 28	C (Pulsar180DTS-iModel): { $f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+28$ }
Pulsar220DTS-iModel	{ $hasMake^+$ = Bajaj} → f^+ 15 { $hasBrakes^+$ =DiskBrakes} → f^+ 21 { $hasWheelType^+$ = Alloy} → f^+ 23 { $hasIgnition^+$ = Self} → f^+ 25 { $hasGears^+$ = 5} → f^+ 28	C (Pulsar220DTS-iModel): { $f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15 \bullet f^+21 \bullet f^+23 \bullet f^+25 \bullet f^+28$ }
Avenger220DTS-iModel	{ $hasMake^+$ = Bajaj} → f^+ 15 { $hasBrakes^+$ =Combo} → f^+ 20 { $hasWheelType^+$ = Alloy} → f^+ 23 { $hasIgnition^+$ = Self} → f^+ 25 { $hasGears^+$ = 5} → f^+ 28	C (Pulsar220DTS-iModel): { $f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet f^+7 \bullet f^+8 \bullet f^+9 \bullet f^+10 \bullet f^+11 \bullet f^+12 \bullet f^+15 \bullet f^+20 \bullet f^+23 \bullet f^+25 \bullet f^+28$ }
Discover135Model	{ $hasMake^+$ = Bajaj} → f^+ 15 { $hasBrakes^+$ =Combo} → f^+ 20	C (Pulsar220DTS-iModel): { $f^+2 \bullet f^+3 \bullet f^+4 \bullet f^+5 \bullet f^+6 \bullet$

	{ hasWheelType^{+} = Alloy} → f^{+} 23 { hasIgnition^{+} = Self} → f^{+} 25 { hasGears^{+} = 4} → f^{+} 29	f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 15 • f^{+} 20 • f^{+} 23 • f^{+} 25 • f^{+} 29 }
Discover125Model	{ hasMake^{+} = Bajaj} → f^{+} 15 { hasBrakes^{+} = DrumBrakes} → f^{+} 22 { hasWheelType^{+} = Alloy} → f^{+} 23 { hasIgnition^{+} = Self} → f^{+} 25 { hasGears^{+} = 5} → f^{+} 28	C (Pulsar220DTS-iModel): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 15 • f^{+} 22 • f^{+} 23 • f^{+} 25 • f^{+} 28 }
Discover100Model	{ hasMake^{+} = Bajaj} → f^{+} 15 { hasBrakes^{+} = DrumBrakes} → f^{+} 22 { hasWheelType^{+} = Alloy} → f^{+} 23 { hasIgnition^{+} = Self} → f^{+} 25 { hasGears^{+} = 4} → f^{+} 29	C (Pulsar220DTS-iModel): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 15 • f^{+} 22 • f^{+} 23 • f^{+} 25 • f^{+} 29 }
Platina100Model	{ hasMake^{+} = Bajaj} → f^{+} 15 { hasBrakes^{+} = DrumBrakes} → f^{+} 22 { hasWheelType^{+} = Alloy} → f^{+} 23 { hasIgnition^{+} = Kick} → f^{+} 26 { hasGears^{+} = 4} → f^{+} 29	C (Pulsar220DTS-iModel): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 15 • f^{+} 22 • f^{+} 23 • f^{+} 26 • f^{+} 29 }
Duke200Model	{ hasMake^{+} = Bajaj} → f^{+} 15 { hasBrakes^{+} =DiskBrakes} → f^{+} 21 { hasWheelType^{+} = Alloy} → f^{+} 23 { hasIgnition^{+} = Self} → f^{+} 25 { hasGears^{+} = 6} → f^{+} 27	C (Pulsar220DTS-iModel): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 15 • f^{+} 21 • f^{+} 23 • f^{+} 25 • f^{+} 27 }
BulletElectraTwinspark Model	{ hasMake^{+} = RoyalEnfield} → f^{+} 16 { hasBrakes^{+} = Combo} → f^{+} 20 { hasWheelType^{+} = Wirespoke} → f^{+} 24 { hasIgnition^{+} = Self} → f^{+} 25 { hasGears^{+} = 5} → f^{+} 28	C (BulletElectraTwinsparkModel): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 16 • f^{+} 20 • f^{+} 24 • f^{+} 25 • f^{+} 28 }
Bullet 350 Twinspark Model	{ hasMake^{+} = RoyalEnfield} → f^{+} 16 { hasBrakes^{+} = Combo} → f^{+} 20 { hasWheelType^{+} = Wirespoke} → f^{+} 24 { hasIgnition^{+} = Self} → f^{+} 25 { hasGears^{+} = 5} → f^{+} 28	C (Bullet350TwinsparkModel): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 16 • f^{+} 20 • f^{+} 24 • f^{+} 25 • f^{+} 28 }
Bullet Electra EFI Model	{ hasMake^{+} = RoyalEnfield} → f^{+} 16 { hasBrakes^{+} = Combo} → f^{+} 20 { hasWheelType^{+} = Wirespoke} → f^{+} 24 { hasIgnition^{+} = Self} → f^{+} 25 { hasGears^{+} = 5} → f^{+} 28	C (BulletElectraEFIModel): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 16 • f^{+} 20 • f^{+} 24 • f^{+} 25 • f^{+} 28 }
Bullet Electra Deluxe Model	{ hasMake^{+} = RoyalEnfield} → f^{+} 16 { hasBrakes^{+} = Combo} → f^{+} 20 { hasWheelType^{+} = Wirespoke} → f^{+} 24	C (BulletElectraDeluxeModel): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 16 • f^{+} 20 • f^{+} 24 • f^{+}

	{ hasIgnition^{+} = Self} → f^{+} 25 { hasGears^{+} = 5} → f^{+} 28	25 • f^{+} 28 }
Royal Enfield Classic 500 Model	{ hasMake^{+} = RoyalEnfield} → f^{+} 16 { hasBrakes^{+} = Combo} → f^{+} 20 { hasWheelType^{+} = Wirespoke} → f^{+} 24 { hasIgnition^{+} = Self} → f^{+} 25 { hasGears^{+} = 5} → f^{+} 28	C (RoyalEnfieldClassic500Model): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 16 • f^{+} 20 • f^{+} 24 • f^{+} 25 • f^{+} 28 }
Royal Enfield Classic 350 Model	{ hasMake^{+} = RoyalEnfield} → f^{+} 16 { hasBrakes^{+} = Combo} → f^{+} 20 { hasWheelType^{+} = Wirespoke} → f^{+} 24 { hasIgnition^{+} = Self} → f^{+} 25 { hasGears^{+} = 5} → f^{+} 28	C (RoyalEnfieldClassic350Model): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 16 • f^{+} 20 • f^{+} 24 • f^{+} 25 • f^{+} 28 }
Thunderbird Twinspark Model	{ hasMake^{+} = RoyalEnfield} → f^{+} 16 { hasBrakes^{+} = Combo} → f^{+} 20 { hasWheelType^{+} = Wirespoke} → f^{+} 24 { hasIgnition^{+} = Self} → f^{+} 25 { hasGears^{+} = 5} → f^{+} 28	C (ThunderbirdTwinsparkModel): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 16 • f^{+} 20 • f^{+} 24 • f^{+} 25 • f^{+} 28 }
R15Model	{ hasMake^{+} = Yamaha} → f^{+} 17 { hasBrakes^{+} = DiskBrakes} → f^{+} 21 { hasWheelType^{+} = Alloy} → f^{+} 23 { hasIgnition^{+} = Self} → f^{+} 25 { hasGears^{+} = 6} → f^{+} 27	C (R15Model): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 17 • f^{+} 21 • f^{+} 23 • f^{+} 25 • f^{+} 27 }
FZModel	{ hasMake^{+} = Yamaha} → f^{+} 17 { hasBrakes^{+} = Combo} → f^{+} 20 { hasWheelType^{+} = Alloy} → f^{+} 23 { hasIgnition^{+} = Self} → f^{+} 25 { hasGears^{+} = 5} → f^{+} 28	C (FZModel): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 17 • f^{+} 20 • f^{+} 23 • f^{+} 25 • f^{+} 28 }
VictorModel	{ hasMake^{+} = TVS} → f^{+} 18 { hasBrakes^{+} = DrumBrakes} → f^{+} 22 { hasWheelType^{+} = Wirespoke} → f^{+} 24 { hasIgnition^{+} = Kick} → f^{+} 26 { hasGears^{+} = 4} → f^{+} 29	C (VictorModel): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 18 • f^{+} 22 • f^{+} 24 • f^{+} 26 • f^{+} 29 }
CBRModel	{ hasMake^{+} = Honda} → f^{+} 19 { hasBrakes^{+} = DiskBrakes} → f^{+} 21 { hasWheelType^{+} = Alloy} → f^{+} 23 { hasIgnition^{+} = Self} → f^{+} 25 { hasGears^{+} = 6} → f^{+} 27	C (CBRModel): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 19 • f^{+} 21 • f^{+} 23 • f^{+} 25 • f^{+} 27 }
ShineModel	{ hasMake^{+} = Honda} → f^{+} 19 { hasBrakes^{+} = DrumBrakes} → f^{+} 22 { hasWheelType^{+} = Alloy} → f^{+} 23 { hasIgnition^{+} = Self} → f^{+} 25 { hasGears^{+} = 4} → f^{+} 29	C (ShineModel): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 19 • f^{+} 22 • f^{+} 23 • f^{+} 25 • f^{+} 29 }
UnicornModel	{ hasMake^{+} = Honda} → f^{+} 19 { hasBrakes^{+} = Combo} → f^{+} 20 { hasWheelType^{+} = Alloy} → f^{+} 23 { hasIgnition^{+} = Self} → f^{+} 25	C (UnicornModel): { f^{+} 2 • f^{+} 3 • f^{+} 4 • f^{+} 5 • f^{+} 6 • f^{+} 7 • f^{+} 8 • f^{+} 9 • f^{+} 10 • f^{+} 11 • f^{+} 12 • f^{+} 19 • f^{+} 20 • f^{+} 23 • f^{+}

	{ $hasGears^{+} = 5$} → $f^{+}28$	25 • $f^{+}28$ }

Table29: Final Location map

CONCEPT ADDRESS	CONCEPT FEATURE
Thing (0, 1)	X
Bikes (1, 1)	$f^{+}1$ • $f^{+}2$ • $f^{+}3$ • $f^{+}4$ • $f^{+}5$ • $f^{+}6$ • $f^{+}7$ • $f^{+}8$ • $f^{+}9$ • $f^{+}10$ • $f^{+}11$ • $f^{+}12$
Make (1, 2)	$f^{+}2$
EngineCapacity (1, 3)	$f^{+}3$
Power (1, 4)	$f^{+}4$
Price (1, 5)	$f^{+}5$
FuelTankCapacity (1, 6)	$f^{+}6$
Mileage (1, 7)	$f^{+}7$
Brakes (1, 8)	$f^{+}8$
Weight (1, 9)	$f^{+}9$
WheelType (1, 10)	$f^{+}10$
Ignition (1,11)	$f^{+}11$
Gears (1 ,12)	$f^{+}12$
NamedBikes (2, 1)	$f^{+}1$ • $f^{+}2$ • $f^{+}3$ • $f^{+}4$ • $f^{+}5$ • $f^{+}6$ • $f^{+}7$ • $f^{+}8$ • $f^{+}9$ • $f^{+}10$ • $f^{+}11$ • $f^{+}12$
HeroHonda (2, 2)	$f^{+}2$ • $f^{+}13$
Hero (2, 3)	$f^{+}2$ • $f^{+}14$
Bajaj (2, 4)	$f^{+}2$ • $f^{+}15$
RoyalEnfield (2, 5)	$f^{+}2$ • $f^{+}16$
Yamaha (2, 6)	$f^{+}2$ • $f^{+}17$
TVS (2, 7)	$f^{+}2$ • $f^{+}18$
Honda (2, 8)	$f^{+}2$ • $f^{+}19$
HeroHondaBikes (3, 1)	$f^{+}2$ • $f^{+}3$ • $f^{+}4$ • $f^{+}5$ • $f^{+}6$ • $f^{+}7$ • $f^{+}8$ • $f^{+}9$ • $f^{+}10$ • $f^{+}11$ • $f^{+}12$ • $f^{+}13$
HeroBikes (3, 2)	$f^{+}2$ • $f^{+}3$ • $f^{+}4$ • $f^{+}5$ • $f^{+}6$ • $f^{+}7$ • $f^{+}8$ • $f^{+}9$ • $f^{+}10$ • $f^{+}11$ • $f^{+}12$ • $f^{+}14$
BajajBikes (3, 3)	$f^{+}2$ • $f^{+}3$ • $f^{+}4$ • $f^{+}5$ • $f^{+}6$ • $f^{+}7$ • $f^{+}8$ • $f^{+}9$ • $f^{+}10$ • $f^{+}11$ • $f^{+}12$ • $f^{+}15$
RoyalEnfieldBikes (3, 4)	$f^{+}2$ • $f^{+}3$ • $f^{+}4$ • $f^{+}5$ • $f^{+}6$ • $f^{+}7$ • $f^{+}8$ • $f^{+}9$ • $f^{+}10$ • $f^{+}11$ • $f^{+}12$ • $f^{+}16$
YamahaBikes (3, 5)	$f^{+}2$ • $f^{+}3$ • $f^{+}4$ • $f^{+}5$ • $f^{+}6$ • $f^{+}7$ • $f^{+}8$ • $f^{+}9$ • $f^{+}10$ • $f^{+}11$ • $f^{+}12$ • $f^{+}17$
TVSBikes (3, 6)	$f^{+}2$ • $f^{+}3$ • $f^{+}4$ • $f^{+}5$ • $f^{+}6$ • $f^{+}7$ • $f^{+}8$ • $f^{+}9$ • $f^{+}10$ • $f^{+}11$ • $f^{+}12$ • $f^{+}18$
HondaBikes (3, 7)	$f^{+}2$ • $f^{+}3$ • $f^{+}4$ • $f^{+}5$ • $f^{+}6$ • $f^{+}7$ • $f^{+}8$ • $f^{+}9$ • $f^{+}10$ • $f^{+}11$ • $f^{+}12$ • $f^{+}19$
Combo (2, 9)	$f^{+}8$ • $f^{+}20$
DiskBrakes (2, 10)	$f^{+}8$ • $f^{+}21$
DrumBrakes (2, 11)	$f^{+}8$ • $f^{+}22$
Alloy (2, 12)	$f^{+}10$ • $f^{+}23$
WireSpoke (2, 13)	$f^{+}10$ • $f^{+}24$
Self (2, 14)	$f^{+}11$ • $f^{+}25$
Kick (2, 15)	$f^{+}11$ • $f^{+}26$
6 (2, 16)	$f^{+}12$ • $f^{+}27$
5 (2, 17)	$f^{+}12$ • $f^{+}28$
4 (2, 18)	$f^{+}12$ • $f^{+}29$
Karizma(Normal)Model (4, 1)	$f^{+}2$ • $f^{+}3$ • $f^{+}4$ • $f^{+}5$ • $f^{+}6$ • $f^{+}7$ • $f^{+}8$ • $f^{+}9$ • $f^{+}10$ • $f^{+}11$ • $f^{+}12$ • $f^{+}13$ • $f^{+}20$ • $f^{+}23$ • $f^{+}25$ • $f^{+}28$
Karizma(ZMR)Model	$f^{+}2$ • $f^{+}3$ • $f^{+}4$ • $f^{+}5$ • $f^{+}6$ • $f^{+}7$ • $f^{+}8$ • $f^{+}9$ • $f^{+}10$ • $f^{+}11$ • $f^{+}12$ • $f^{+}13$

(4, 2)	• f^+ 21 • f^+ 23 • f^+ 25 • f^+ 28
Splendor(Plus)Model (4, 3)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 13 • f^+ 22 • f^+ 24 • f^+ 26 • f^+ 29
Splendor(NXG)Model (4, 4)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 13 • f^+ 23 • f^+ 23 • f^+ 26 • f^+ 29
Splendor(Super)Model (4, 5)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 13 • f^+ 22 • f^+ 23 • f^+ 25 • f^+ 29
Splendor(Pro)Model (4, 6)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 13 • f^+ 22 • f^+ 23 • f^+ 25 • f^+ 29
PassionProModel (4, 7)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 13 • f^+ 20 • f^+ 23 • f^+ 25 • f^+ 29
CD-DawnModel (4, 8)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 13 • f^+ 22 • f^+ 24 • f^+ 26 • f^+ 29
CD-DeluxeModel (4, 9)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 13 • f^+ 22 • f^+ 23 • f^+ 26 • f^+ 29
Glamour(Normal)Model (4, 10)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 13 • f^+ 20 • f^+ 23 • f^+ 25 • f^+ 29
Glamour(PGMFi)Model (4, 11)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 13 • f^+ 20 • f^+ 23 • f^+ 25 • f^+ 29
AchieverModel (4, 12)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 13 • f^+ 20 • f^+ 23 • f^+ 25 • f^+ 28
CBZXtremeModel (4, 13)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 13 • f^+ 20 • f^+ 23 • f^+ 25 • f^+ 28
HunkModel (4, 14)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 13 • f^+ 21 • f^+ 23 • f^+ 25 • f^+ 28
ImpulseModel (4, 15)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 14 • f^+ 20 • f^+ 23 • f^+ 25 • f^+ 28
CT100Model (4, 16)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 15 • f^+ 22 • f^+ 24 • f^+ 26 • f^+ 29
Pulsar135LSModel (4, 17)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 15 • f^+ 20 • f^+ 23 • f^+ 25 • f^+ 28
Pulsar150DTS-iModel (4, 18)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 15 • f^+ 20 • f^+ 23 • f^+ 25 • f^+ 28
Pulsar180DTS-iModel (4, 19)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 15 • f^+ 20 • f^+ 23 • f^+ 25 • f^+ 28
Pulsar220DTS-iModel (4, 20)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 15 • f^+ 21 • f^+ 23 • f^+ 25 • f^+ 28
Avenger220DTS-iModel (4, 21)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 15 • f^+ 20 • f^+ 23 • f^+ 25 • f^+ 28
Discover135Model (4, 22)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 15 • f^+ 20 • f^+ 23 • f^+ 25 • f^+ 29
Discover125Model (4, 23)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 15 • f^+ 22 • f^+ 23 • f^+ 25 • f^+ 28
Discover100Model (4, 24)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 15 • f^+ 22 • f^+ 23 • f^+ 25 • f^+ 29
Platina100Model (4, 25)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 15 • f^+ 22 • f^+ 23 • f^+ 26 • f^+ 29
Duke200Model	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+

(4, 26)	15 • f^+ 21 • f^+ 23 • f^+ 25 • f^+ 27
BulletElectraTwinspark Model (4, 27)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 16 • f^+ 20 • f^+ 24 • f^+ 25 • f^+ 28
Bullet 350 Twinspark Model (4, 28)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 16 • f^+ 20 • f^+ 24 • f^+ 25 • f^+ 28
Bullet Electra EFI Model (4, 29)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 16 • f^+ 20 • f^+ 24 • f^+ 25 • f^+ 28
Bullet Electra Deluxe Model (4, 30)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 16 • f^+ 20 • f^+ 24 • f^+ 25 • f^+ 28
Royal Enfield Classic 500 Model (4, 31)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 16 • f^+ 20 • f^+ 24 • f^+ 25 • f^+ 28
Royal Enfield Classic 350 Model (4, 32)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 16 • f^+ 20 • f^+ 24 • f^+ 25 • f^+ 28
Thunderbird Twinspark Model (4, 33)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 16 • f^+ 20 • f^+ 24 • f^+ 25 • f^+ 28
R15Model (4, 34)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 17 • f^+ 21 • f^+ 23 • f^+ 25 • f^+ 27
FZModel (4, 35)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 17 • f^+ 20 • f^+ 23 • f^+ 25 • f^+ 28
VictorModel (4, 36)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 18 • f^+ 22 • f^+ 24 • f^+ 26 • f^+ 29
CBRModel (4, 37)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 19 • f^+ 21 • f^+ 23 • f^+ 25 • f^+ 27
ShineModel (4, 38)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 19 • f^+ 22 • f^+ 23 • f^+ 25 • f^+ 29
UnicornModel (4, 39)	f^+ 2 • f^+ 3 • f^+ 4 • f^+ 5 • f^+ 6 • f^+ 7 • f^+ 8 • f^+ 9 • f^+ 10 • f^+ 11 • f^+ 12 • f^+ 19 • f^+ 20 • f^+ 23 • f^+ 25 • f^+ 28

Fig. 55: Bike Ontology Structure

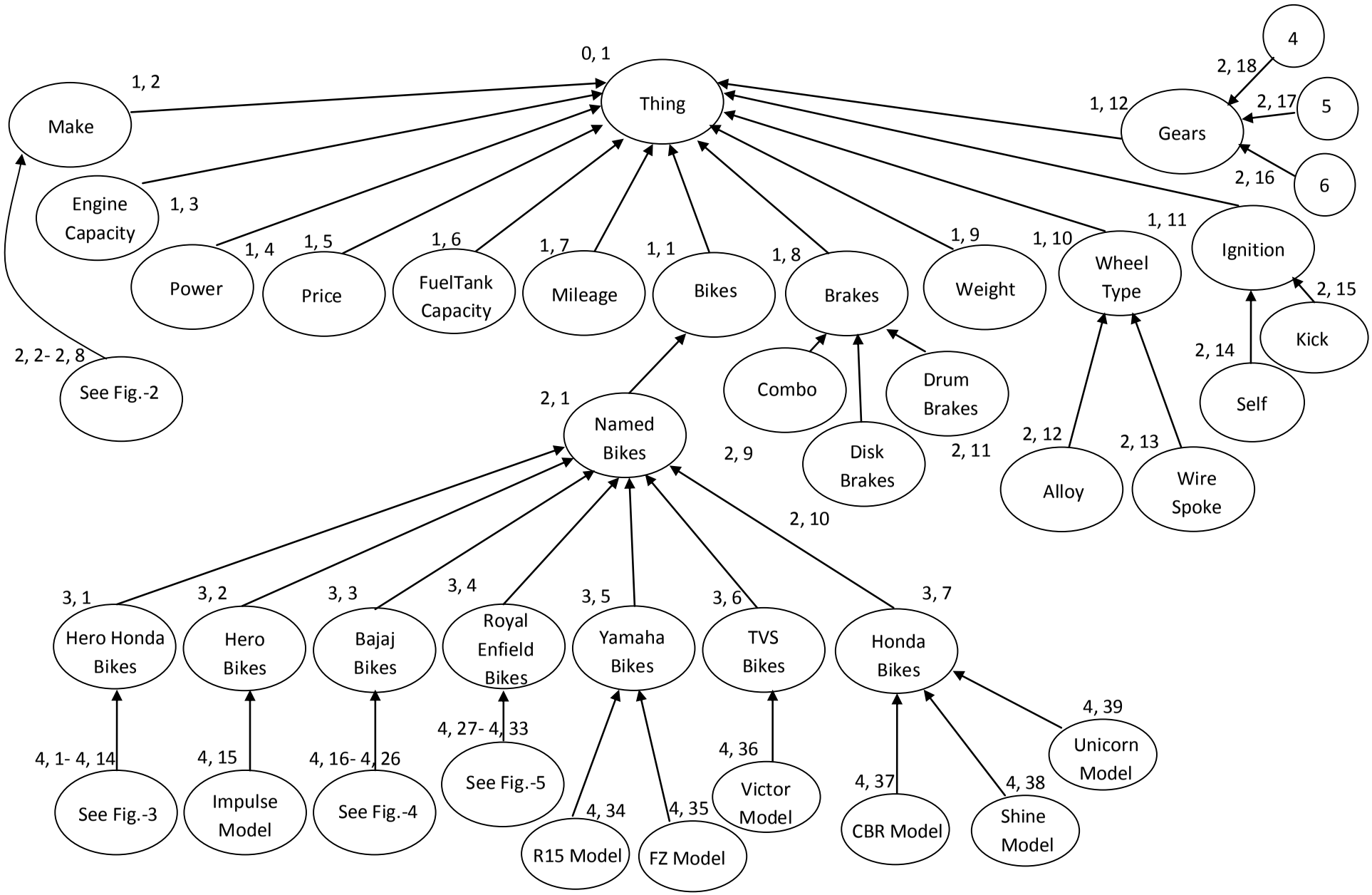


Fig. 56: Sub Ontology Structure of *HeroHondaBikes*

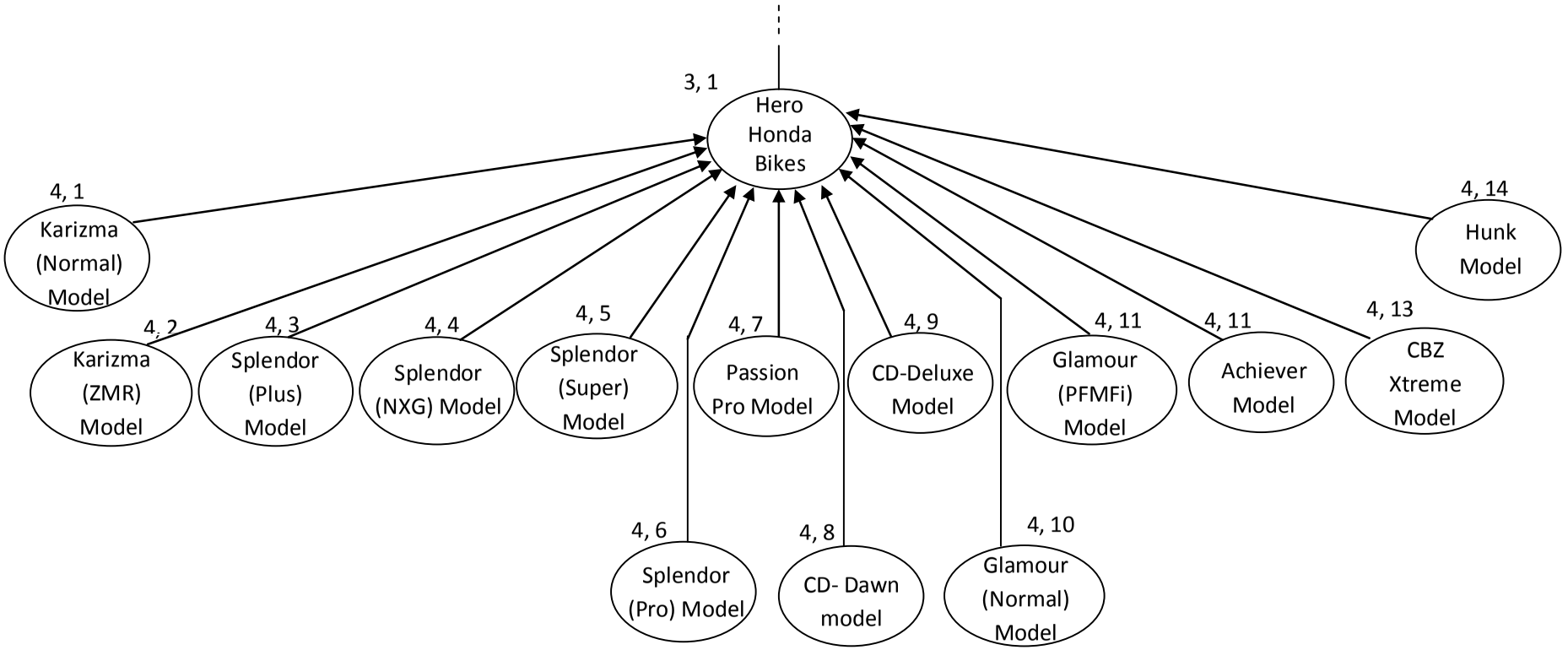


Fig. 57: Sub Ontology Structure of *BajajBikes*

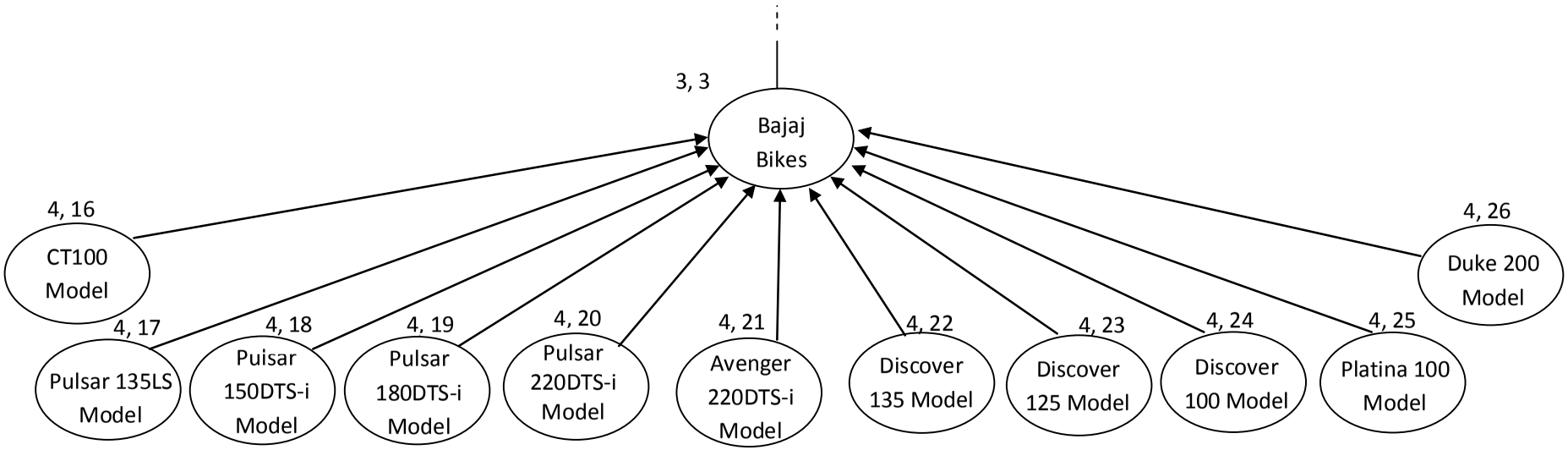


Fig. 58: Sub Ontology Structure of *Make*

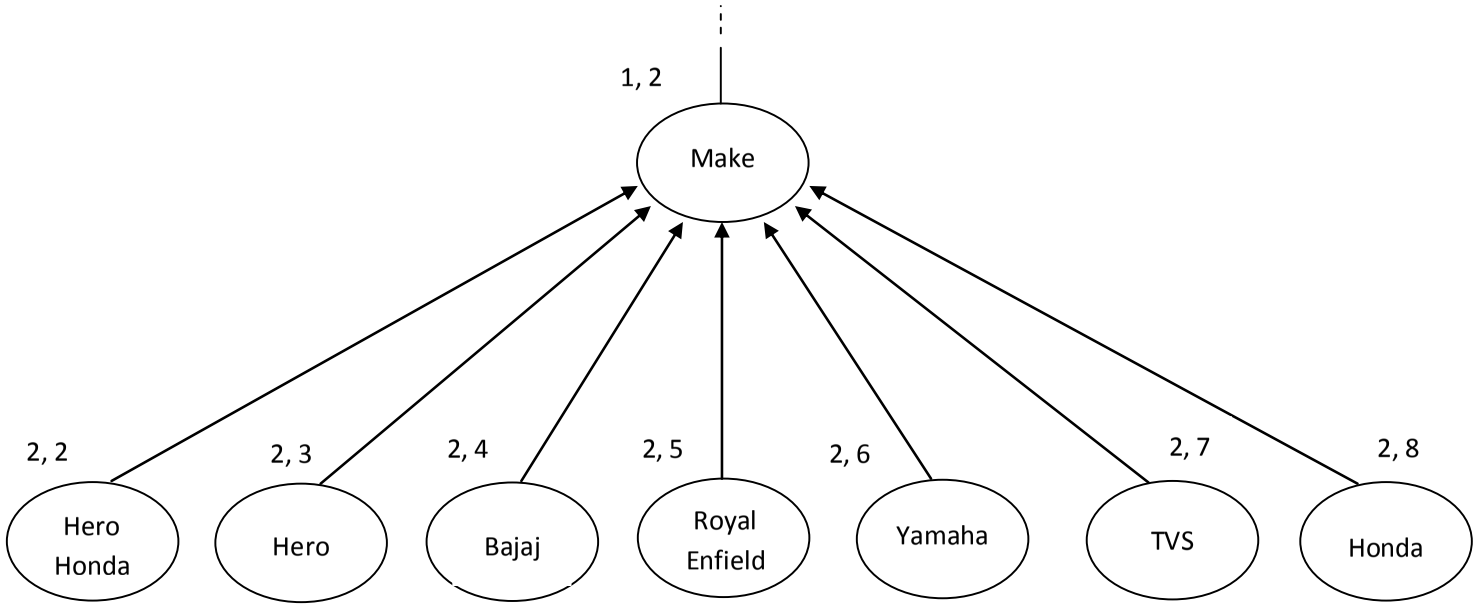


Fig. 59: Sub Ontology Structure of *RoyalEnfieldBikes*