ONTOLOGY BASED INTRUSION DETECTION SYSTEM FOR WEB APPLICATION SECURITY

Harshal Karande, Prof.Shyam Gupta.
Department Of Computer Engineering , Siddhant College Of Engineering,

Savitribai Phule University Pune, India.

***Abstract*- Web application security is the major security concern for e-business and information sharing community. The use of e-business and information sharing community are exponentially increased and due to this cyber threats also increased. Earlier detection techniques try to keep up with the inherent complexity of web design so increasing variety of attacks that can exploit it. Security system modeled using an ontology propose new class of solution that can be highly effective in detecting zero day and sophisticated web application attacks by capturing the context of the contents of information such as HTML pages or in-lilne scripts. Proposed approach has the ability to filter these contents by taking into consideration their consequences to the target applications. A detailed ontological model describes caters working of web applications, the used communication protocols and attacks. Proposed ontological model not only detect HTTP protocol specification attacks but also helps focus only on specific portions of the request and response where a malicious script is possible. Also model captures the context of important attacks, the various technologies used by the hackers, source, target and vulnerabilities exploited by the attack, impact on system components and controls for mitigation**

***Index Terms:* - Cyber security, Information security, Ontology based intelligent system, Semantic security, Web application security.**

I. INTRODUCTION

Different generic security controls like as signature-based firewalls, intrusion detection and prevention systems and encryption devices have been deployed, however their effectiveness against web-based threats is restricted, because of their extreme rigidness. For efficient mitigation of web based attacks the system should understand the context of the information contents to be processed and have the ability to filter the contents on the basis of their effect on the target application. Since it is difficult to capture the actual attack context in a traditional security solution, it has been previously proposed that a security framework modeled using an ontological approach can be more effective in detecting zero day and sophisticated web application attacks.

Ontology refers to the explicit specification of the conceptualization of a domain which captures its context (Interpretation of words in specific domain/situation).Mostly non-ontological approaches are based upon attack signatures. Normally the attack signatures are not generic in nature, using specific languages related to particular domains and depend upon specific environments and systems. They lack extensibility and are also not suitable for communication in heterogeneous systems. Attack signatures often carry vague semantic knowledge and lack solid ground in any formal logic. Small variations in the business logic invalidate the signatures. Constantly updating of the signature is also a very tedious job.

The Web applications security has become increasingly important in the last decade and becomes hottest issue due to exponentially increase in electronic communication to millions of users globally through diverse range of applications. Intrusion Detection Systems have become a critical technology to help protect these systems. Traditional security solution like web scanners provides the first line of defense against web attacks and detects the "well-known" security flaws those have signatures. Scanners lack semantics thus unable to make intelligent decision upon data leakage or business logic flaws, result in false alarm, and fail to detect novel and critical vulnerabilities. Signature base solution usually maintains the white list (positive security model) and black list (negative security model) which contains the signature of benign inputs and signature of malicious attack vectors respectively. These lists needed updating of signature, lack of detection zero day attacks and generate the false positive and false negative alarms. Both positive and negative security models have limitations in terms of configuration, and tuning, learning capabilities. Furthermore, many of the network solution ignore the payload and scan only the header of request. So due to the lack of contextual nature, network solution are ineffective to mitigate the application level attacks. So there is need of semantic system which can intelligently understand the application's context, the data and contextual nature of attacks. System should validate the input syntactically and semantically. Syntax based validation provide the size or content restrictions whereas semantic based validation may focus on specific

data type, specific format and understanding potentially dangerous and malicious commands or content with respect to their context and consequences.

Our system is addressing all these issued through automatically updating of knowledge base. Our system mitigate the web application attacks effectively and efficiently and capable of defeating the current strategy of novel manipulation of attacks by the hackers. We are proposing an advanced defensive mechanism for producing a proper automatically input validation through semantic knowledge base populated with OWL- DL ontologies.

II. PROBLEMS WITH EXISTING SYSTEMS

Most existing intrusion detection systems suffer from the following problems:

1. Current IDS are usually tuned to detect known service level network attacks. This leaves them vulnerable to original and novel malicious attacks. Current IDS Show low Accuracy and Detection rate, which is important problem in Existing IDS.
2. Data overload: Another aspect which does not relate directly to misuse detection but is extremely important is how much data an analyst can efficiently analyze. That amount of data he needs to look at seems to be growing rapidly. Depending on the intrusion detection tools employed by a company and its size there is the possibility for logs to reach millions of records per day.
3. False positives: A common complaint is the amount of false positives an IDS will generate. A false positive occurs when normal attack is mistakenly classified as malicious and treated accordingly.
4. False negatives: This is the case where an IDS does not generate an alert when an intrusion is actually taking place.

III. CONTRIBUTION

The proposed mechanism is a novel approach for employing the use of semantics in
application layer security contrary to tradition signature based approaches. It has the following key contributions:

1. Ontological model based on communication protocol.
2. This model captures the context of important web application attacks, various technologies used by the hackers, source and target of an attack, impact on the system components affected by the attack, vulnerabilities exploited by the attack and control in terms of policies for mitigating these attacks.
3. A comprehensive and best metrics suite for ontology evaluation has been used for assessing the quality of proposed ontology models.

Objectives

1. To provide low false positive rate and low false alarm rate.
2. To increases accuracy and efficiency.
3. To Study of Ontology Framework for Intrusion Detection System.
4. To Detection of Cyber attacks.

IV. RELATED WORK

The system expresses this with respect to the end users' domain and allows non-experts to model it easily by using the terminologies and concepts of intrusion detection. A major problem which is common for all the above systems is that the ontology is only used to depict a simple model of attack attributes. These systems also lack the necessary reasoning ability due to their taxonomical structure and their focus is mainly on the network layer missing attacks at the application layer. Some partially address these issues but still rely on signature based approach. Consequently these systems miss the most important web application attacks. The proposed ontological model of attack detection, allows us to successfully capture the context of user input, which is an essential requirement in designing and implementing effective defense mechanisms against web attacks. The power and utility of the ontological model is exploited to the fullest by applying the inference process upon concepts and properties of the model to establish the fact that an inferred result characterizes a particular type of attack.

V. ONTOLOGY ENGINEERING METHEDOLOGY

Ontology is specifies uses a formal data structure with common depiction of domain concepts. Ontology design is an iterative process to determine the scope and define the concepts (classes), properties (relations), axioms, constraints and instances. Our system ontology has been developed keeping in view.

The following objectives:

1. Detailed description of all important security concepts related to target application resources
2. Ability to manage security aspects at various levels ranging from abstract to more specific.
3. Provide a generic solution for various environments.
4. Enable the reusability of ontology.

VI. DEVELOPMENT OF ONTOLOGICAL MODELS FOR WEB APPLICATION ATTACKS

The Methontology framework has been used for ontology construction, which follows the
primary activities outlined by the software development process and knowledge engineering methodologies. This framework facilitates construction of ontology in a systematic way and is also compatible with RUP, the process for the system development. For the technical evaluation, with respect to a frame of reference, ontology verification and validation has been carried out using Onto Clean which is a community best practice in this regard. This methodology used to check the consistency, completeness and conciseness of the ontology.

METHONTOLOGY is among the more comprehensive ontology engineering methodologies as it is one for building ontologies either from scratch, reusing other ontologies as they are, or by a process of re-engineering them. The framework enables the construction of ontologies at the knowledge level, *i.e.*, the conceptual level, as opposed to the implementation level. The framework consists of: identification of the ontology development process with the identification of the main activities, such as, evaluation, configuration, management, conceptualization, integration implementation; a life cycle based on evolving prototypes; and the methodology itself specifying the steps for performing the activities, the techniques used, the outcomes and their evaluation. They describe very detailed the process to build ontology for centralized ontology based systems

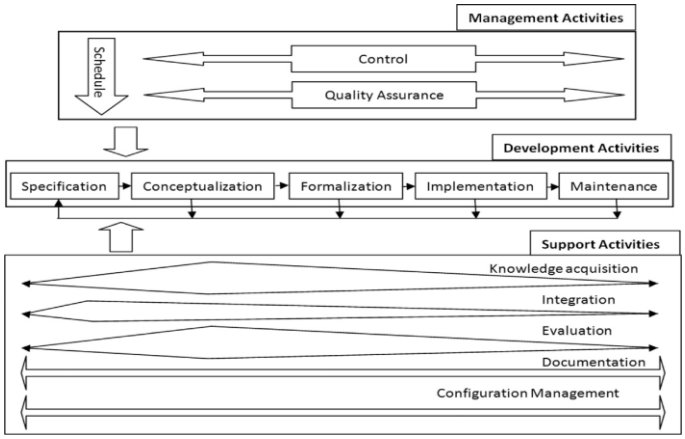

Fig 1. Life cycle of ontology model.

Life cycle of this methodology is based on evolving prototypes (Fernandez-Lopez et al., 2000); starting with the identification of ontology development process along with techniques to carry out each activity. Life cycle permits an incremental design and development of the ontology that allows early verification and adjustment. Fig. 1 (Corcho et al., 2005), highlights iterative process of ontology development. The life cycle activities (Fernandez, 1999; Fernandez-Lopez et al., 1997b, 1999; Gomez-Perez, 1998; Tsoukas, 1996) of Methontology are composed of specification, conceptualization, formalization, implementation and maintenance activities. The management activities comprises of control and quality assurance. The parallel support activities are knowledge acquisition, integration, evaluation, configuration management and documentation.

A layered approach has been used for ontology design in the knowledge base that depicts the tools and technologies which have been used, as shown in below figure The bottom-most layer represents the conceptualization in which concepts of our domain in question i.e. web security, are defined in the form of classes, properties and individuals either dynamically or through interaction with the tool's Graphical User Interface

The ontology layer facilitates the expression of classes in logical form (i.e. predicate logic) and defines restriction upon the classes by using OWL (web ontology language) through interaction with OWL GUI. The inference (top-most) layer drives additional knowledge from the defined ontology and provides the ability to check the uniformity and arrangement of concepts (classes).

The layer representing Rules uses Semantic Web Rule Language (SWRL) (Horrocks et al., 2004) and Jena API for parsing, reasoning and rule generation. In addition the Jena API allows query construction for retrieving information via simple protocol and resource query language (SPARQL).

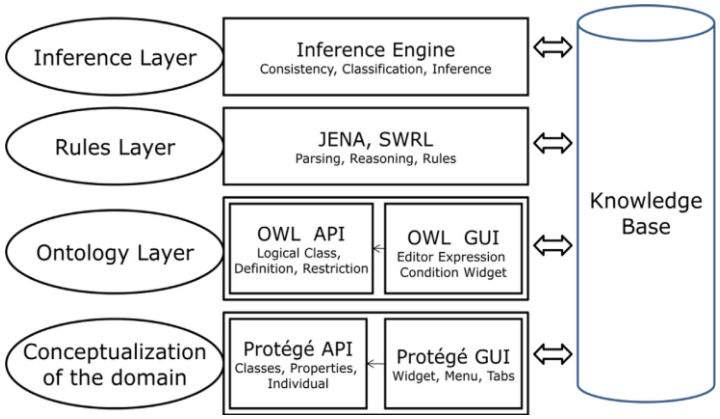

Fig.2 Layered approach of ontology

VII. FORMAL REPRESENTATION OF ONTOLOGY MODEL

Ontologies are not a just combination of terminologies or vocabularies rather they can attain a semantic level that can capture the context of a specific domain. The basic aim of the proposed ontology is to express the complex knowledge of web application security domain in a way that it can be computationally traceable, machine processable and structure of information facilitates communication among software agents. The formal designing of ontological model can be represented as:

O =Ontology system =(C; P; A; I)

Where C: conceptualization for each possible world that includes concepts and instances; P: properties and relationships; A: axioms and rules; and I: interpretation of model.

These notations can be further elaborated as:

$$C = (\cup_{t \in Type} C_t) \cup (\cup_{i \in Type} I_i)$$

$$P = (\cup_{p \in Type} P_p) \cup (\cup_{e \in Type} Rel_e)$$

$$A : (\cup_{a \in Type} A_a) \cup (\cup_{r \in Type} R_r)$$

Where C: is a set of all concepts/classes with type t, I: is a set of all instances with type i and P: is a set of all properties of type p, including data type and object. The symbol Rel: represent

the set of all relationships of type e, including equivalence, subsume and disjoint. Similarly A: is a set of all axioms and R: is a set of all rules. Properties and relationships can be formally represented as:

P =(D : datatype; O : object; T : transitive; F : functional)

Rel =(≡: equivalence; 3: subsumed; ∩: disjoint)

The system knowledge base consists of all concepts, instances, assertions, properties and relationships in ontology model. Formally represented as:

$$[C(i)]; i \in I; C \in C \quad C(a) \in KB$$

$$[p(i1;i2)]; i1; i2 \in I; p \in P \quad r(a1; a2) \in KB$$

$$\text{Domain of property} : [\forall a; b : p(a; b) \rightarrow \text{Domain}(a)]; p \in P; \text{Domain} \in C$$

$$\text{Range of property} : [\forall a; b : p(a; b) \rightarrow \text{Range}(b)]; p \in P; \text{Range} \in C$$

$$[\forall a : C1(a) \rightarrow C2(a)]; C1; C2 \in C \in C \quad C1 \sqsubset C2 \in KB;$$

$$[\forall a : P1(a) \rightarrow P2(a)]; P1; P2 \in P \in C \quad P1 \sqsubset P2 \in KB$$

$$\left[\forall a : C1(a) \hat{C}2(a) \rightarrow \perp\right]; C1; C2 \in C \text{and} \exists p \cdot \top \sqsubseteq C \in KB; \top \sqsubseteq \forall p \cdot C \in KB;$$

$$C1 \sqcap C2 \cdot \perp \in KB$$

The interpretation I of ontology O is known as the model of O. For $\forall c, c' \in C \in O \rightarrow I(c) \neq \phi$ means, every concept is associated with some object. The expressions I (c) ≡ I (c'), I (c) $\sqsubset I(c')$ and $I(c) \sqcap I(c') = \phi$ are the interpretation of equivalence ,subsuming and disjoint relationships respectively. The Equivalence (≡) relationship possessed the properties of reflexive,

symmetric and transitive. Conceptually subsume $(\sqsubset)$ relation is irreflexive, transitive, and asymmetric. Conceptually disjoint $(\sqcap)$ relation is reflexive and symmetric and transitive.

VIII. ATTACK ONTOLOGY MODEL

In order to obtain semantic information about attacks and their impact, ontology that describes the entities and relations in the model is used. Figure 3 describes the structure of the attack ontology where the term Attack is depicted as the top level concept in the model. The model includes semantic annotations of Source, Technology, Vulnerability, Target and Policies. There are numerous sub-classes of each of these concepts such as Source (IP, Port, MAC) and Target (IP, Port, MAC) of an attack. The subclasses of Technology and Vulnerability are (SQL,PHP, Java Script, ASP), and (Physical, Technical, Logic and Administrator Weakness) respectively. The ontology also defines various attributes like received From, directed To, used By, exploited By, and mitigated By. The attributes in

the Attack ontology models inter-linking concepts such as the assertion "Attack mitigated By Policy." Policies are defined using these concepts as ontological model to mitigate attacks. A set of a number of semantic rules give rise to a mitigation policy. Similarly, class Vulnerability, represents all vulnerability instances in the web application that are exploited by class Attack.

IX. PROTOCOL ONTOLOGY MODEL

Application layer communication protocols are used as semantic networks. Every protocol is termed as a conceptual category in this model. The initial point in the model is the "Protocol" concept. From conceptual modeling point of view, it is a parent of all the specific application layer protocol classes such as HTTP, HTTPS, FTP, and SMTP. The model further includes three concepts: Message, Request, and Response. Model presents specific details for each protocol. For instance, the HTTP concepts in the model contain all the associated parameters and their relationships with relevant concepts by using RFC 2616 .In order to elaborate the HTTP ontology model, we present specific examples. HTTP Request is a part of HTTP Message Structure, which is further classified in to Request Line, Entity Header, Request Header, General Header and Payload. HTTP-Request concept further relates to Request Line, which
has three parts: Version, Method, and Request URI. Each HTTP Message can include headers, modeled as Header class in the ontology and is sub classified into Request Header, Entity Header
and General Header following figure depicts the inter connection of these concepts.

X. PROPOSED SYSTEM

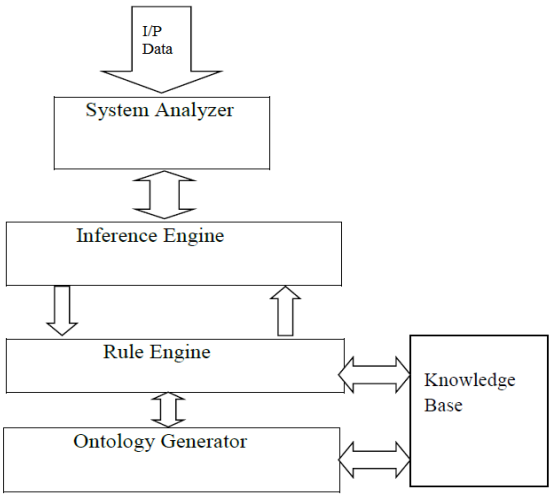

Fig 5. System Architecture

1. System Analyzer

This is first layer of our architecture and most important component of our System. Raw input data is given as input to this Analyzer which analyzes the user input for suspected malicious contents based on information stored in the knowledge base.

System Analyzer will perform the following task:-

1. Analysis the raw input data.
2. Filter out received HTTP packets.
3. Input HTTP packets are parsed.
4. Perform Message Header checking

i. Read URL and referrer field.

ii. Check Query string portion.

iii. Analyze its value for malicious script and produce alert in case of malicious value.

5. Perform Message-Body checking

i. Read URL and HTML tags

ii. Check Query string portion

iii. Produce alert in case of malicious value

6. After analyzing input if no malicious activity found, packet would be

Delivered to web application for retrial of requested data. Analyzer accepts the network stream, captured each HTTP messages, it filter out the HTTP protocols on the basis of HTTP ontology stored in knowledge base .HTTP messages are checked at two points-header and payload. The header contains information regarding the payload and control information. The Analyzer interacts with the inference engine in order to analyze input packet. If packet will found malicious script the packet is blocked otherwise it will given as input to inference engine.

System Analyzer checks input semantically and syntactically as mentioned in ontology and it also checks hidden field values of URI of each page .It also checks for Method attribute like GET or POST, Action attribute and Input field for all the information given in the input
parameters tags like size, name, max Length, their type like checkbox, radio button, hidden field etc.

2. Inference Engine:

The Inference Engine observes the inconsistency indifferent ontologies in knowledge base and makes the knowledge base consistence and updated.

3. Rule Engine:

Rule engine layer generates appropriate rules after inference upon the Knowledge base for specific request. It processes the assertions, rules, deduces new rules and

statements. For implementation purpose we are using the Java base Jena rule engine.

k

4. Ontology Generator:

This module consists of two layers

a. Ontology Layer

b. Layer of conceptualization of Domain.

All classes are defined using Protégé API using Protégé GUI. Concepts are represented in Resource description format. Using the Ontology web Language plug-in the RDF format. Ontology file is stored with .owl extension which is accessible in Java platform through Jena API.

XI. ATTACK DETECTION

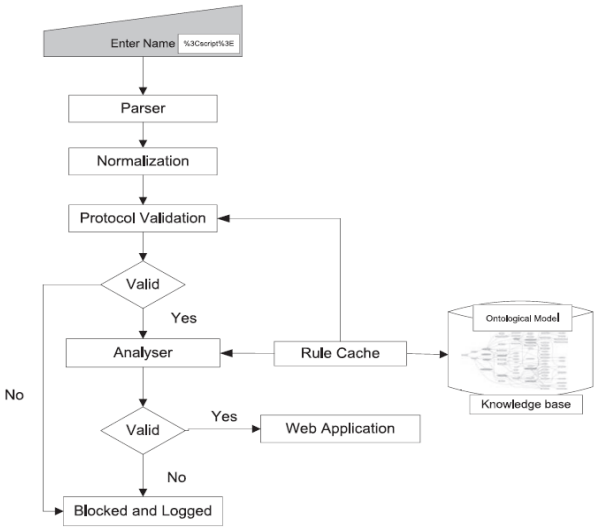

Fig Web Attack Detection.

In the proposed system, HTTP request parsed according to ontological model is stored in the Knowledge base. Input is checked for encoding. In case it is encoded it would be first decoded and then would be checked for anomaly After the normalization module the request is passed on to the Protocol Validation and Analyzer module where it is matched against the semantic rules that are generated by ontological models in the knowledge base for identifying malicious content in input validation. Protocol Validation module caters to the violation of protocol specification whereas the Analyzer handles all other Web application attacks. If the input content matches any of the rules the request is blocked and a log is made for the said attack.

XII. CONSIDERED ATTACKS

1. Analyzer Detection Mechanism (XSS attack) :

In Cross Site Scripting (XSS) attack the analysis is normally done on the input fields in an application. For example, in a web page where the user is asked to enter his name, the user can inject a script in encoded form.

When the input field passes through the analysis engine, it is checked for encoding. In case it is encoded it would be first decoded and then would be checked for anomaly. The above given string when decoded would look like:

"<script> alert ("This is cross site script") </script>".

After the normalization module the request is passed on to the Protocol Validation and Analyzer module where it is matched against the semantic rules that are generated by ontological models in the knowledge base for identifying malicious content in input validation. Protocol Validation module caters to the violation of protocol specification whereas the Analyzer handles all other web application attacks. If the input content matches any of the rules the request is blocked and a log is made for the said attack as shown in Fig

[rule1234: (?x rdf:type ex:HTTPRequest) (?y rdf:type ex:ResponseHeaders)

(?z rdf:type ex:ResponseSplitting) (?x ex:hasRequestHeaders ?y) -> (?x ex:hasAttackCarrier ?z)]

2. Protocol Validation Attack:

In protocol validation attacks, an attacker tries to send an abnormal request that does not follow the RFC 2616 (Hypertext, 2014) standards. HTTP response splitting and HTTP request smuggling (Testing, 2014; OWASP, 2014) are common and complex examples of these attacks. To mitigate these types of attacks, in the proposed system, HTTP request parsed according to ontological model is stored in the Knowledge base. This ontological model is used by the

Analyzer to analyze it for protocol validation attacks. If the request is valid then it is further checked for other attacks, otherwise it is blocked and stored in the log with the attack type that is found in that request.

3. Request / Response Smuggling attack:

In HTTP request smuggling attack, a malicious request contains multiple start lines that are not allowed in one HTTP request (violation of RFC 2616). When the hacker tries to assign multiple start lines to one request, it will be detected by the system analyzer and reported as an HTTP request smuggling attack

4. Buffer Overflow:

A buffer overflow is the computing equivalent of trying to pour two liters of water into a one-liter pitcher: Some water is going to spill out and make a mess. A buffer (or array or

string) is a space in which data can be held. A buffer resides in memory. Because memory is finite, a buffer's capacity is finite. For this reason, in many programming languages the programmer must declare the buffer's maximum size so that the compiler can set aside that amount of space.

Let us look at an example to see how buffer overflows can happen. Suppose a C language program contains the declaration:

char sample[10];

The compiler sets aside 10 bytes to store this buffer, one byte for each of the ten elements of the array, sample[0] through sample[9]. Now we execute the statement:

sample[10] = 'A';

The subscript is out of bounds (that is, it does not fall between 0 and 9), so we have a problem. The nicest outcome (from a security perspective) is for the compiler to detect the problem and mark the error during compilation. However, if the statement were

sample[i] = 'A';

we could not identify the problem until i was set during execution to a too-big subscript. It would be useful if, during execution, the system produced an error message warning of a subscript out of bounds. Unfortunately, in some languages, buffer sizes do not have to be predefined, so there is no way to detect an out-of-bounds error. More importantly, the code needed to check each subscript against its potential maximum value takes time and space during execution, and the resources are applied to catch a problem that occurs relatively infrequently. Even if the compiler were careful in analyzing the buffer declaration and use, this same problem can be caused with pointers, for which there is no reasonable way to define a proper limit. Thus, some compilers do not generate the code to check for exceeding bounds.

XIII. CONCLUSION

Ontology of attacks and communication protocols provide a powerful construct for improving the detection capability of application level attacks. The proposed system is a novel approach for application of Semantic technologies in Web application security. It creates a synergy between the two emerging technologies, web semantics and information security. The system captures parses and validates incoming HTTP requests based on the HTTP protocol ontology model. If the request is valid then it is forwarded to the next component for application data extraction process. The extracted contents are passed to the Analyzer for context analysis and attack detection through inference upon the rules and ontology stored in the knowledge base. Due to ontological approach, the proof-of-concept system has shown better attack detection rate especially in protocol validation.

XIV. ACKNOWLEDGMENT

I would like to express my sincere thanks to my guide Prof. Shyam Gupta for his motivation and useful suggestions which truly helped me in improving the quality of this paper. I take this opportunity to express my thanks to my teacher, family and friends for their encouragement and support

REFERENCES

[1] Gomez-Perez A. Knowledge sharing and reuse. Journal: Handbook of Applied expert systems 1998:10-1.

[2] Sierra J. Building a chemical ontology using methontology and the ontology design environment .In Proc.workshop on ontologies and problem solving methods. IEEE intelligent systems and their applications;1999.pp.37-46.

[3] Fernandz Lopez M.,Gomez-Perez A, Rojas MD. Ontologies crossed life cycles. In Proc International Conference in knowledge engineering and management. LNAI, vol .1937. Springer-Verlag;2000.pp.65-79.

[4] Grosof Benjamin, Dean Mike,SWRL: a semantic web rule language combining OWL and RuleML.Journal W3C member submission 2004;21:70.

[5]Gorcho O, Fernandz M, Gomez-Perez A, Lopez-cima A, Building legal ontologies with methodology and webODE. In Benjamins R, Casanovas P, Breuker J, Gangemi A, editors Law And The Semantic Web. Springer-Verlag;2005.pp.142-57.LNAI No 3369. ISBN: 3-540-25063-8.