Semantic Interoperability of e-Health Systems using Dialog based Mapping of Ontologies in Diabetes Management

Pronab Ganguly

Integral Energy

51 Huntingwood Drive

Huntingwood, NSW 2148, Australia

Email: pronabganguly@gmail.com

Abstract

This paper aims to describe a framework that resolves some of the semantic interoperability issues in e-health systems. This framework combines dialogue game with model of rational ontology mapping decision mechanisms to generate automatic dialogue between software agents leading to resolution of certain types of ontological mismatches. The salient features of this framework are:

- Describes an argumentation mechanisms for resolving certain types of ontological mismatches
- As the mechanism is context driven in nature, it does not just find equivalent concepts , but instead finds concepts that can be substituted for each other (which is a different kind of matching) in the given context
- Offers potential for context specific mismatch resolution libraries for reuse in future
- Offers potential for dynamic modification of participating ontologies
- Provides explanation in context of diabetes management e-health systems.

1 Introduction

E-health involves the integration of information, human-machine and healthcare technologies. As different modalities of patient care require applications running inheterogenous computing environments, software interoperability is a major issue in e-health systems, specially in context of Semantic Web. Ontology can be defined as a set of concepts understood in a knowledge base. A formal ontology specifies a way of constructing a knowledge base about some part of the world and, thus, contains a set of allowed concepts, and rules which define the allowable relationships between concepts. In other words it is a structured repository for knowledge, consisting of a collection of knowledge elements such as rules and their associated data model. Ontology has emerged as a major technique for software interoperation in Semantic Web. Mapping between domain ontologies is an important research issue. Software agents may be deployed to resolve these complex issues. Again argumentation provides a rich mechanism by which software agents may engage in dialogue. Specific protocols have been developed to address the issues related to argumentation based interactions. Those protocols define how a participating software agent asserts statements and counter statements. The management of such protocols can be carried out in various ways - such as finite state machines, form-filling, dialogue game and plan-based. Though argumentation based protocol provides a communication language and associated syntax, it does not specify the rules under which specific locutions should be used by an agent. Thus a dialogue-game on its own is not able to govern the automatic discussion between software agents. A model for decision making needs to be coupled with the dialogue-game framework to conduct automated dialogue for mismatch resolution. This model for decision making composes of two modules – ontology reasoning using description logic and context reasoning using first-order logic.

This paper describes outline of a framework that combines dialogue game with model of rational ontology mapping decision mechanism to generate automatic dialogue between software agents leading to resolution of certain types of ontological mismatches in healthcare informatics. This involves:

- Adopting rules of dialogue games
- Adopting a decision mechanism based on ontology reasoning and context reasoning for ontology mapping
- Generating a dialogue model based on decision mechanism
- Defining the locutions in the dialogue game (syntax)
- Defining decision mechanisms that will invoke different types of locutions at different points
- Defining transitional rules governing linkage between dialogue locutions and decision making mechanism

Section 2 describes the various semantic mismatch issues citing examples from diabetes management domain. Section 3 describes the various semantic mismatch issues citing examples from diabetes management domain. Section 4 summarises the structure and major element of the framework. Section 3 explains a simple example from healthcare informatics domain. Section 4 outlines implementation specific outline and section 5 is the concluding section.

2. **Classification of Semantic Mismatch Issues**

There may be various types semantic mismatches in Semantic web, but for our study we will use the following classifications:

Semantically equivalent concepts:

- Different terms to refer to same concept in two models such as staff and employee;
- Different properties such as one model includes the color of the product and the other does not; and
- Property mismatch such as different units.

Semantically unrelated concepts:

- Conflicting terms—the same term may be chosen by two information systems to denote completely different concepts. For example, the word apple may be used to denote either a brand of hardware or a type of fruit.

Semantically related concepts:

- Generalization and specification—one system might have only the general concept of fruit, but the other has the concepts of mango, cherry, and so forth;
- Definable terms—a term may be missing from one information source, but it is defined in another information source;
- Overlapping concepts—for example, children one information source means persons aged between 5 to 12 years, but in another information source children may mean persons between the ages of 3 and 10 years, and in still another information source young persons may refer to persons aged between 10 and 30 years of age; and
- Different conceptualization—as described above, for example, one information source classifies a person as male or female. The other information source classifies the same person as employed or unemployed.

It is necessary to narrow down to an application area where one can identify above types of semantic heterogeneity. For illustrative purposes, we have chosen diabetes management, which exhibits the following semantically equivalent concepts:

- Property mismatch, such as different units for blood sugar measurement, for example, mmol/l (milimoles per liter…the world standard unit for measuring glucose in blood or mg/dl (milligrams per deciliter…the measure used in the United States).
- Different Properties - Glycohemoglobin (GHb) is formed by a non-enzymatic interaction between glucose and the amino groups of the valine and lysine residues in hemoglobin. Formation of glycohemoglobin is irreversible and the level in the red blood cell depends on the blood glucose concentration. Thus, measuring glycohemoglobin provides a measurement of glycemic control over time, and its use has been proven to evoke changes in diabetes treatment, resulting in improved metabolic control. First introduced in the 1970s, it is now accepted as a unique and important index of metabolic control. There are currently four principal glycohemoglobin assay techniques (ion-exchange chromatography, electrophoresis, affinity chromatography and immunoassay) and about 20 different methods that measure different glycated products and report different units. [6]
- Different terms such as intensive glycemic control or tight glycemic control may be used to refer to the same concept.

Similarly other types of semantic heterogeneity can be identified in this application scenario.

3. Ontology and Semantic Interoperation

In W3C has expressed the need for a Web Ontology language on top of the existing XML, and RDF. These can be summarized as follows: “Ontologies are critical for applications that want to search across or merge information from diverse

communities. Although XML DTDs and XML Schemas are sufficient for exchanging data between parties who have agreed to definitions beforehand, their lack of semantics prevent machines from reliably performing this task given new XML vocabularies. The same term may be used with (sometimes subtle) different meaning in different contexts, and different terms may be used for items that have the same meaning. RDF and RDF Schema begin to approach this problem by allowing simple semantics to be associated with identifiers. With RDF Schema, one can define classes that may have multiple subclasses and super classes, and can define properties, which may have sub properties, domains, and ranges. In this sense, RDF Schema is a simple ontology language. However, in order to achieve interoperation between numerous, autonomously developed and managed schemas, richer semantics are needed. For example, RDF Schema cannot specify that the Person and Car classes are disjoint, or that a string quartet has exactly four musicians as members."
The stack of W3C recommendations related to the Semantic Web can be briefed as below:

- The syntax of the document is provided by XML but not semantic constraints
- The restriction on the structure of the document is povided by XML schema
- RDF provides a simple semantics of the data model that can be structured in XML
- General semantics in terms of hierarchies is provided by RDF schema
- Detailed semantics can be provided by Ontologies expressed in Web Ontology Language(OWL).

A number of use cases have been identified where Web Ontology Language can be used in [OWL, 2004]. These can be summarized as follows:

- *Web Portals:*
- *Multimedia Collections:*
- *Corporate Web Site Management*:
- *Design Documentation:*
- *Agents and Services:*
- *Web services and Ubiquitous Computing*.

The healthcare informatics has different standards such as HL7, TC251, ISO TC215 and GEHR. The standards contain considerable domain knowledge in classifications, methodologies, terminologies and specific vocabularies. These standards can be effectively utilized to develop ontologies that can be used in web services. Though these standards offer to facilitate interoperability, they do no provide machine to machine interoperability. Thus representation of concepts as defined by different standards may result in disparate ontologies. These ontologies need to be mapped for machine level interoperation. Ontology is a formal, explicit specification of a shared conceptualization which provides a vocabulary of terms and relations with which to model the domain. It is suited to represent high-level information requirements to specify the context information in semantic web environment. An ontology generally provides a vocabulary of terms and relations with which to model the domain, whereas, domain ontology captures the knowledge valid for a particular type of domain. It includes abstract concepts, and specifics domain-level constraints that can be used for reasoning, and is especially suited to represent high-level information requirements. Within schemas and classes are data level concepts that are implementation platform dependent. They are designed to optimize procedural operations. Constraints at this level are operational constraints.

4. Telemedicine and Diabetes Management

Diabetes is a condition in which the level of blood glucose is consistently above the normal range. If left untreated the condition can lead to renal and ocular complications or damage to peripheral nerves . Worldwide, about 135 million people suffer from diabetes. The figure is projected to grow by 122% within the next 25 years. Hence, there is a growing need for an efficient and effective treatment plan for those who suffer from diabetes.

Currently, treatment of diabetes entails limited patient contact with multiple healthcare professionals including general practitioners, and specialists. The interaction might involve regular patient visits to a general practitioner or health provider who collect blood samples to send for on-site testing. The result is reviewed by the general practitioner to determine the appropriate treatment. Complicated cases are typically referred to a specialist.

For diabetes management, diet is an important issue along with medication and regular exercise. Dietician consults with the patient to formulate various meals for the patient. Calorie intake is dependent on body weight and other factors while patient's preference for food is a criterion. Let us consider a scenario where a dietician agent is engaged with a patient agent to formulate breakfast. For this particular patient (which is dependent on various factors such as height, body weight and preferences), it is recommended that the calorie intake should be as below:

- Vegetable - 25 Cal
- Milk products – 90 Cal
- Protein 35 - Cal
- Fruit 60 - Cal
- Starch - 80 Cal

Here the dietician and the patient are collaboratively formulating the patient's breakfast. Let dietician's ontology is represented by ontology S and patient's ontology is represented by Ontology R. For simplicity we will consider

- Part of the ontologies to highlight the issue
- They overlap to a large extent

It may be noted that dietician's ontology includes very lean protein but as the patient is vegetarian in nature it does not have that concept but contains kidney bin cooked under vegetable. Cooked kidney bin (1/3 cup) is equivalent to 1 small egg from protein content perspective. The high level problem is to plan the breakfast for the patient and the associated problem is the mapping between the ontologies. For the sake of simplicity, let us assume:

- The patient is vegetarian – does not take either meat or egg
- The dietician agent is the expert agent and carry the relevant features (such as type of the calories) of all the elements in the ontology. Dietician's ontology is represented by R and patient's ontology is represented by S.

3 Proposed Framework

This framework is primarily targeted for OWL-Lite type of domain ontologies. The major elements in this framework are:

- Decision mechanism consisting of :
  - Ontology Reasoner
  - Context Reasoner
- Locution Generator
- Transition Rules Generator
- Participating software agents

The interaction between the modules are below:

- Ontology reasoner reasons with ontology to generate relevant information
- Context reasoner interacts with ontology reasoner and locution generator to capture context
- Software agents exchange locutions as per Transition Rules that encompasses decision mechanism module and locution generator

The following subsection provides details of the each module.

2.1 Ontology Reasoner

For ontology reasoner OWL-lite and associated reasoning rules are used. With application of these rules, following details for a concept are generated [Harth et al,2004]. For details explanation of rules and enhanced formalism please refer to [Mei et al, 2004], [Harth et al,2004] & [Bechover S, 2003]

RDF Schema Features	Property Characteristics
RDF Schema Features 1. Class 2. rdfs:subClassOf 3. rdfs:domain 4. rdfs:range	Property Characteristics 1. ObjectProperty 2. TransitiveProperty 3. SymmetricProperty 4. inverseOf
Equivalence 1. equivalentProperty 2. equivalentClass	Property Restrictions 1. Restriction 2. onProperty 3. allVauesfrom

2.2 Context Reasoner

Depending on the task under considerations, mapping requirements may vary. The concept to be mapped will have a number of features (properties etc). The initiating agent (S) allocates a mapping logic for each feature. If a particular feature is required, mapping logic will be given a value of 1(yes) otherwise 0 (No). If the equivalent class (with the specified features) returned by the recipient agent (R) is not acceptable to the sending agent (S) for the task, the requesting agent (R) can reconfigure the mapping factors to the mapping features and get a different equivalent class[Wang et al,2004]. This process continues until an acceptable mapping is agreed. This is a novel feature of this dynamic framework.

2.3 High Level Dialogue Model

The high level models of dialogue between the agents are presented below [Mcburney M et al, 2003] [Beun R.j et al,2004]:

1. Open Dialogue: The dialogue commences.
2. Inform: The requesting agent Asi informs the need for a mapping .
3. Form Mapping Criteria: The agents Asi & Arj establishes the concept needs to be mapped and associated details criterion. The inform phase might be used for clarification.
4. Generate Mapping Options: Agent Asi & Arj assess mapping criterion based on the reasoning model in section 2.1 & 2.2. They might use inform phase for clarification.
5. Negotiate: Agents negotiate over preferred option.
6. Confirm: The agents confirm the preferred option.
7. Close Dialogue: The dialogue terminates.

2.4 Locutions Rule

The locutions are accompanied two stores shared by the agents – Information store and Confirmation store. During exchange of information, entries are inserted into an agent's information store by a locution uttered by that agent . Similarly the Commitment store records the confirmations by corresponding agents. Locution rules specifies following element:

- Precondition for the locution
- Meaning of the locution
- Response to that locution
- Effect on Information store
- Effect on Commitment store

Let the participating agents be denoted by Ax1, Ax2., … etc where X ϵ { Asi, Arj} represents the role of the agent as dietician or patient. We assume that each agent Axi has an information store denoted by IS(Axi) that contains entries such as (Ayj, ā) where Ayj is the other participant and ā is an option offered or under consideration. The locutions and associated rules in this context are given below:

Open Dialogue Phase:

L1: open_dialogue locution

Location: open_dialogue (Axi, α) where X ϵ { Asi, Arj }
Precondition: This locution should have not been used previously. The agent should utter this locution to initiate a dialogue.
Meaning: The speaker Axi suggests the opening of a dialogue on a concept α. A dialogue can only commence with this move.
Response: The other agent must respond with enter_dialogue (Axj, α)
Information Store Update: None
Commitment Store Update: None

Due to lack of space, unless there is a significant change, corresponding aspects of the locution will be omitted

L2: enter_dialogue
Location: open_dialogue (Axj, α) where X ϵ { Asi, Arj }
Precondition: Within the dialogue, an agent Axi, where i is not equal to j, must have uttered the locution open_dialogue.
Meaning: The agent Axj is willing to participate in the dialogue for α

Inform Phase:

For the following locations, there will be a general precondition that the participating agent has already uttered either open_dialogue() or enter_dialogue. Hence this common precondition is not included in the following locutions. We will mention additional precondition specific to that locution.

L3: seek_info
Location: seek_info (Axi, p) where X ϵ { Asi, Arj } where p is a proposition
Response: The recipient agent utter a provide_info(Axj, Z) where the elements of Z satisfy the constraint imposed by p.

L4: provide_info
Location: provide_info(Axj, Z) where the elements of Z satisfy the constraint imposed by p.
Precondition: The other participant must have previously uttered seek_info (Axi, p)
Information Store Update: for each a ϵ Z, (a) is inserted into IS(Axj), the information store for Axj

Negotiate Phase:

L5: willing_to_accept
Locution: agree_to_accept (Axi, Z) where Z is a non empty set of options.
Precondition: For each option a ϵ Z, a locution should have been uttered such that a ϵ K. In other words, the speaker can agree only to options that have been previously offered .
Meaning: The speaker commits to one of the options in set Z.
Commitment Store Update: for each a ϵ Z, (a) is inserted into commitment Store CS(Axj)

L6: refuse_to_accept
Locution: refuse_to_accept(Axj,K) where K is a set of options
Precondition: This locution cannot be uttered for agree_to_accept (Axi, Z) where Z ∩ K is non empty.
Meaning: The speaker refuses to accept any option in the set K.

Close Dialogue Phase:

L7: Withdraw_dialogue
Locution : withdraw_dialogue(Axi, α) for X ϵ { Adi, Apj }

2.5 Decision Making Mechanisms

For automated dialogue, on top of syntactical rules, the agents must be equipped with decision making mechanism to generate relevant type of locution at the point of need. This mechanism is internal to the agent and is based on the ontology reasoning and context reasoning rules . Thus each agent is equipped with its own mechanisms. These mechanisms will invoke particular locations at particular points in dialogue. Thus a generic function of the following mechanisms is to decide , if any, locution to utter taking the output of the corresponding mechanism or generate a null locution (do nothing). Before proceeding with individual mechanisms of each agent, let us describe following generic function that will be part of every mechanism.

Select Locution: A function to decide what locution will be uttered depending on the outcome of the associated decision mechanism.

The decision making steps are presented briefly below:

1. Sending (S) agent generates the details (as per ontology and configuration rules) of immediate parent (henceforth called ancestor class) of the class to be mapped.
2. Receiving (R) agent checks all the words in the supplied pointer class details with all words in its own ontology. If any unmatched word is found in ancestor class details, then it tries to get the equivalent word called (substitute) from Wordnet
3. If the substitute from Wordnet is found in R's ontology (as per ontology rules), suitable mapping factor is assigned to it. This information is stored in information store in R and notifies S for storage. If substitute word is acceptable to S, it notifies R. If the substitute is not acceptable S, R allocates a value of 0 (ignores) to it.
4. R tries to find an equivalent for ancestor class in its ontology as per ontology rule.
5. If equivalent class (called anchor class) is found, R processes it as per step 8 .
6. If no equivalent class for ancestor class is found in R, R gets details of the immediate parent class of ancestor class from S.
7. This process is continued until an anchor class is found in R.
8. All the subclasses under anchor class become potential candidates.
9. R evaluates all the candidates with associated grading and passes on to S.
10. S selects the best fit from the presented options.

The high level mechanisms associated with this process are described below:

M1: Recognize Need: Mechanism that enable P to identify the need for mapping a concept θ. The automated reasoner will be used for this purpose. The possible output for this mechanism there_is_need (θ) and there_is_no_need (θ)

M2: Seek Mapping Information: Mechanism to seek mapping information. Possible outcome seek_mapping_info (θ) or seek_no_mappinginfo

M3: Provide mapping Information: Mechanism to generate information to be provided for mapping [Step 1]. The possible outcome of this mechanism is provide_mapping_*info*(θ) or provide_no_mapping_info .

M4: Wordnet Substitution: Checks for word substitution from Wordnet. Possible outcomes are substitute(V (θ)) or reject(V(θ)) [Step 2 &3]

M5: Find Locking Class: Possible outcomes are anchoring(V (θ)) or no_anchoring [Step4]

M6: Find Parent of Anchor Class: Outcome is parent_anchor class(V(θ)) [Step 6 & 7)

M7: Generate Options : The possible outcome of this stage is to generate equivalent mapping options with associated grades options(V1 (θ1,… θn), … Vn (θ1,… θn)) {Step 8 & 9]

M8: Select Option: Select the best fit out of the options (V(θ)). The possible outcomes are accept or reject [Step 10]

M9: Consider Withdraw: A mechanism to consider withdrawal from the dialogue if it is not producing any effective outcome. The possible outcomes of this mechanisms are withdraw(θ) or not_withdraw.

2.6 Transition Rules

As the agents are equipped with decision mechanism, the outcomes of these mechanisms need to be associated with utterances of the locutions and vice versa. The locutions uttered will cause the transitions between the states of the mechanism . In order to define the links, ordered 3 { Sxi, K,s} is used to denote the mechanism with number K and with an output s of participant SXI. Where a transition is invoked by or invokes a particular output of a mechanism K this is denoted by the specific output s in the third place of the triple. Where no specific output is invoked, a "." is used in the third place. When transitions occur between mechanisms through locutions, they are denoted by relevant locution number from section . The transitions between decision mechanisms of the single agent are represented by unlabeled arrow. These transition rules are as below:

TR1: {S, MS1, there_is_need} > L1,L2 > {R,MR1,.}

This rule indicates that a patient agent with a current need for mapping a concept A will initiate a dialogue by means of a locution L1 i.e open_dialogue in the case where such a dialogue is not already initiated or will enter such a dialogue by means of locution l2 i.e enter_dialogue in the case where it has already been initiated.

TR2 {R, MR2, seek_mapping_info} > L3 > {S,MS3,.}

TR3 {S,MS3,provide_mapping_info} > L4 > {R,MR4,.)

TR4 {R, MR4,.} > {R, MR5,}

TR5 {R, MR5,no_anchoring } > {R,MR6,.}

TR6 {R, MR7, generate_options} > L4 > {S, MS8,select_option.}

TR7{S,MS8,selectoption} >L5 > (R,.,.}

TR8 {S, MS8, withdraw} >L7 > {R,.,.}

TR9 {S,MS8, reject} > L7 > {D,.,.}

We have provided the main high level transition rules. Depending on the size and complexity of the associated ontologies and type of mismatched, these high level rules can be decomposed into about 35 fine grained rules. Due to lack of space , all of them could not be presented here. Currently this issue is under further investigation A close of our framework will reveal that it supports generation of dialogue automatically [McBurney M et al, 2003]].

3. A Simple Example of Dialogue from Diabetes Management

For diabetes management, diet is an important issue along with medication and regular exercise. Dietician consults with the patient to formulate various meals for the patient. Calorie intake is dependent on body weight and other factors while patient's preference for food is an criterion. Let us consider a scenario where a dietician agent is engaged with a patient agent to formulate breakfast. For this particular patient (which is dependent on various factors such as height, body weight and preferences), it is recommended that the calorie intake should be as below:

- Vegetable - 25 Cal
- Milk products – 90 Cal
- Protein 35 - Cal
- Fruit 60 - Cal
- Starch - 80 Cal

Here the dietician and the patient are collaboratively formulating the patient's breakfast. Let dietician's ontology is represented by ontology D and patient's ontology is represented by Ontology P. For simplicity we will consider

- Part of the ontologies to highlight the issue
- They overlap to a large extent

It may be noted that dietician's ontology includes very lean protein but as the patient is vegetarian in nature it does not have that concept but contains bin under vegetable. Bin (1/3 cup) is equivalent to 1 small egg from protein content perspective. The high level problem is to plan the breakfast for the patient and the associated problem is the mapping between the ontologies. In our diabetes management ontologies, suppose the dietician has advised the patient to eggs for Protein. . But that concept of egg is missing from patient's ontology. Thus the mapping dialogue starts as below:

Step 1: As per TR1, the patient agent opens dialogue and identifies mapping requirement for egg.
Step 2: As per TR2, the dietician agent asks for related mapping information.
Step 3: As per TR3, the patent agent provides ancestor class details.
Step 4: As per TR4, 5 & 6, dietician agent generates option such as bin (a vegetable with equivalent protein content)
Step 5: As per TR7, the patient agent selects the best fit.
Step 6: As per T8 & 9,the dialogue closes.

Though the example is very simple in nature, it does highlight the essential features of the dialogue based framework.

4. Proof of Concept Implementation

For our implementation , we are using Altova Semanticworks for our ontology editor. This ontology is in advanced stage of formation. . TRIPLE is an efficient ontology reasoning system based rules. The term "TRIPLE" denotes both the language and the open source inferencing [Harth A et al, 2004]] engine which allows for processing programs expressed in the language. TRIPLE can process programs that consist of facts and rules from which conclusions for answering queries can be drawn. TRIPLE programs can be translated to Horn logic programs. The engine is integrated into the Semantic Web context by providing import facilities for RDF. TRIPLE can import facts encoded in RDF and OWL ontologies, since OWL is layered on top of RDF. Rules can be used to perform operations on the data available in TRIPLEs knowledge base .The basic operations that TRIPLE supports are add facts and rules, pose queries, and remove facts and rules from the knowledge base. Java programmes can access TRIPLE via a Java interface and integrate TRIPLE functionality. For a detailed description of the API , refer to [Harth A et al, 2004].

The Java interface offers the following methods.

- Adding operations can be performed on strings, files, or over the network on URIs. Facts and rules encoded in TRIPLE syntax can be added, as well as facts encoded in RDF.
- Queries are encoded in TRIPLE syntax, and return either variable bindings or a set of RDF statements.
- Individual facts and rules can be removed, as well as the content of models.

A high level schematic of the implementation framework can be represented by:

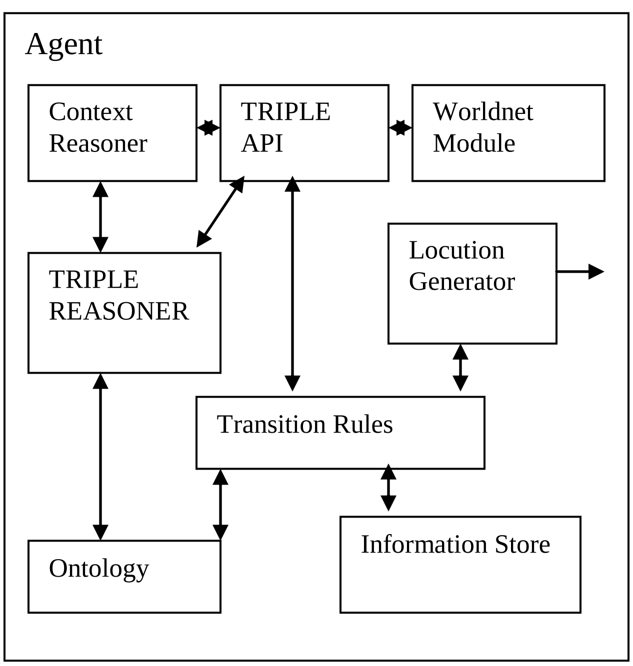

Figure 1: Schematic Of Implementation Framework

We are building our agents and other associated modules in Java as they can be supported by Java API of TRIPLE.

5. Conclusion

Ontology mapping, alignment, articulation and merging is n active research area. For a exhaustive survey to recent trends, refer to [Kalfoglou et al, 2003].We have drawn an outline of dialogue framework to resolve ontological mismatch in OWL Lite environment. There are similar works for ontology negotiation mechanisms such as extended ONP (ontology negotiation protocol) [Orgun B et al,2005] and game board rules [Beun R.J et al,2004]]. A brief comparison is given below:

Features	**Argumentation (SIS)**	**ONP**	**GameBoard**
Automatic generation of Dialogue	Explicit	Not Clear	Not Clear
Basis of Dialogue Model	Derived from rich human Dialogue	Simplistic	Simplistic
Dynamic upgradation of Ontologies	Possible	Not Possible	Not possible.
Semantic mismatch Classification	Clear and explicit	Not Clear	Not Clear

The following types of ontological mismatches are specifically targeted in our framework:

- ➢ *Semantically equivalent concepts:*
- ➢ *Semantically unrelated concepts:*
- ➢ *Semantically related concepts*
  - ✓ *Generalization and specification*
  - ✓ *Overlapping concepts*
  - ✓ *Different conceptualization*

As a varieties of OWL Lite ontologies are available in Semantic web environment, full implementation of our framework and associated benchmarking and validation will contribute to automated reasoning between agents. Further challenges include reasoning between various other versions of OWL type ontologies.

References

[Bechhover S ,2003] Bechhover S ; OWL Reasoning Examples; http://owl.man.ac.uk/2003/why/latest/ accessed on 29/06/2006

[Beun R.J et al, 2004] R.J. Beun and R.M. van Eijk; A Cooperative Dialogue Game for Resolving Ontological Discrepancies; In L. Breure, A. Dillon and H. van Oostendorp (eds.) Creation, Use and Deployment of Digital Information. London: Erlbaum, 2004

[Harth A et al,2004] Harth A & Decker S; OWL Lite Reasoning with Rules; http://www.wsmo.org/2004/d20/d20.2/v0.2/20041123/ accessed on 29/06/2006

[Kalfoglou et al, 2003] Kalfoglou, Y. and Schorlemmer, M. (2003) Ontology mapping: the state of the art. The Knowledge Engineering Review 18(1) pp. 1-31
[McBurney M et al,2003] P. McBurney, R.M. van Eijk, S. Parsons, and L. Amgoud; A Dialogue Game Protocol for Agent Purchase Negotiations; Autonomous Agents and Multi-Agent Systems. 7(3), pages 235-273, 2003
[Mei et al, 2004] Mei J & Bontas PE; Reasoning Paradigms for OWL Ontologies; Technical Report B-04-12; ftp://ftp.inf.fu-berlin.de/pub/reports/tr-b-04-12.pdf, Accessed on 29/06/06
[Orgun B et al,2005] Dialogue based Automation of semantic Interoperability in Multi Agent System ; Orgun B, Dras M, Cassidy S & Nayak A:; Proceeding of Australasian Ontology Workshop (AOW 2005)
[Wang et al,2004] Wang XH, Gu T, Zhang DQ Pung HK; Ontology-Based Context Modeling and Reasoning using OWL; Proceedings of Ontology-Based Context Modeling and Reasoning using OWL,Context Modeling and Reasoning Workshop at PerCom 2004