Abstract— For the interoperation among enterprises in eBusiness, there are many information inconsistencies such as polysem and synonymy which blocking the ambient Semantic collaboration. Ontology is an important tool to overcome serious of syntax and semantic misunderstanding. In this paper, the relations among entities were described by OWL, and then these relations could be defined as a kind of linking-entity added in ontology. Based on the comprehensive description of linking, a similarity rating arithmetic was defined according to the architecture of WordNet. Finally matching and filtering of semantic entities was realized in customization of user to find the entities correlative with user’s requirements and to filter the entities having inconsistencies and conflicts information.

Keywords-semantic; ontology; entity; STASIS, matching

I. INTRODUCTION

In the process of interoperation, every organization had met the challenge of exchanging information from disparate systems in business-business collaboration seamlessly. There are many information inconsistencies such as language difference, wording differences, and structure differences. The same concepts have different kinds of representations and been described in different way with different levels of definitions.

For enterprise semantic interoperation, each enterprise wants to share relational data within an environment which provides a common framework for the integration of information from heterogeneous sources. One of the ways to deal with the problems of polysemy (the existence of several meanings for a single word or phrase) and synonymy (the equivalence of meaning) is the technology of Ontology.

STASIS (Software for Ambient Semantic Interoperable Services) is a research and development project sponsored under the Europeans Commission’s 6th Framework program as well as its projects members. Its objective is for research, development and validation of open, webServices based, distributed semantic services for SME empowerment within the automotive, furniture and other sectors. Partners are spread across Europe and China [1].

II. STASIS PROJECT

The STASIS project will provide a research and development of semantic interoperability services based on webService and associated ebXML technologies. These will help remove complexities of electronic trade for organization, particularly for SMEs.

The first step in STASIS is to identify all of the semantics of data. All kinds of data from many enterprises should be extracted and transformed into the normal format of STASIS through serious of rules. After extracting data into the starting formats and specifications, the implicit semantic has to be made explicit. This could be done by either referencing or describing the own concepts by using an existing ontology or expressing the semantics by an individually developed ontology which could be used by others. After the format of semantic entities is defined, the annotation process becomes emergent to describe the relation among entities.

Figure 1 illustrates how STASIS works in order to obtain the link specification reflecting the semantic correspondences of two business documents in a simple case were two organizations conducting electronic business with each other.

There are some concepts should be described firstly:

- **SSE**: STASIS Semantic Entity, an INSTANCE of some concept recognized as such in the STASIS environment by means of some semantic extraction or annotation;
- **SSE_T (or SSE_T)**: (Terminological) STASIS Semantic Entity, a CONCEPT recognized as such in the STASIS environment by means of some semantic extraction or annotation
- **SNF**: STASIS Neutral Format, a representational (hence, in particular, non semantic) encoding of new elements entering the STASIS environment as acquisition of corresponding items of an external (source or target) schema.
III. ANNOTATION PROCESS

The object of annotation is to build the semantic relations between SSE and the available ontology. There are seven requirements for semantic annotation such as standard formats, user centered/collaborative design, ontology support (multiple ontologies and evolution), support of heterogeneous document formats, document evolution (document and annotation consistency) and annotation storage.

In the past years, many researchers have done for the semi-automated and automated semantic annotation [2, 3].

- Annota [4] project: uses RDF-based format but can only annotate the HTML or XML-based documents.
- Simple HTML Ontology Extensions (SHOE) [6]: can allow users to mark-up HTML pages in SHOE guided by ontologies available locally or via a URL As one of the earliest systems.

Compared with the semi-automatic semantic annotation which provides suggestions for annotations but still requires intervention by knowledge workers, automated systems can acquire annotations automatically on a large scale. SemTag [7] focuses on automatic mark-up and realizes the automated semantic annotation of large corpora based on IBM’s text analysis platform Seeker. However, it is intended as a tool for specialists rather than one for knowledge workers. The next phase of development for annotation will be multimedia annotation which expands the range files types that can be marked up into images, video and audio.

Annotation is mainly expected to find links among annotated SSEs that are expressed in OWL. We can use the reasoning capability of this language to find the relation between SSE of SNF and the concepts (classes and properties) of existing ontology.

A. Simple Annotation

In STASIS, when a schema is represented by means of the SNF, annotation consists of formulas expressing the relationship between SSEs of the SNF and the concepts (classes and properties) of an existing ontology.

The following description of annotation processes and methods is to introduce how this annotation is represented for a SSE. The abstract syntax [8] will be used to represent relations involving individual

- individual(SSE type(c)) : SSE is an instance of the class C
- individual(SSE1 value(p SSE1)) : SSE1 is related to SSE2 through the property p.

B. Ranked Annotation

This kind of annotation permits to express that an SSE is annotated by a class C of an external ontology EO with a certain rank; to represent this kind of annotation.

- Import the external ontology EO in the STASIS framework.
- Since the purpose of this part is to represent ranked binary relationship, a class called rank is defined through stasis:Ranked_SSE with a data property. Then the object property can be defined as stasis:annotation_for with range stasis:Ranked_SSE and domain stasis:SSE.
- To represent the annotation of an individual SSE w.r.t. the class eo:C with a rank r an instance, say a1, of the class stasis:Ranked_SSE must be created and related to SSE; the following assertions are introduced:
  - individual(a1 type(stasis:Ranked_SSE))
  - individual(a1 value(stasis:annotation_for SSE))
  - individual(a1 value(stasis:rank 10))
  - individual(a1 type(eo:C))

Figure 2 shows the format of the above-mentioned ranked annotation.
C. Description of Annotation

In STASIS, the annotation process uses classes directly as annotation property values for some simple annotations and creating special instances of the class to be used as property values for most complex situations of had been ranked, such as annotation with the relation of disjoint. The instance to be used as property values is simple enough to be created and clear enough to describe the relations [9, 10].

In order to get the description of annotation relation between the source entity and the target entity, an object property hasAnnotation is created to get the correspondence. The annotation format can be described as a linking-entity named SLS to give the multi-relation information:

\[
\text{hasAnnotation(Source,Target,Rank,Relation,Method)}
\]

Source and Target are the entities been related. Rank is the number been gotten in ranked annotation. Relation is the description in OWL. Method is the approach taken in the process of rank calculating.

IV. SEMANTIC MATCHING IN STASIS

Based on the annotation, user can find the entities they want by giving out words they are interested. This process can be produced by the automatic ranked search methods proposed following:

A. Terminological ranking using WordNet.

WordNet presently contains approximately 95,600 different word forms (51,500 simple words and 44,100 collocations) organized into some 70,100 word meanings, or sets of synonyms, and only the most robust hypotheses have survived. The most obvious difference between WordNet and a standard dictionary is that WordNet divides the lexicon into five categories: nouns, verbs, adjectives, adverbs, and function words. Assign distance for every relationship in the WordNet, for each pair Terminological entities, Calculate and choose the shortest path as the similarity distance [11].

B. Entities ranking using arithmetic.

The Class Match Measure (CMM): it is meant to evaluate the coverage of an ontology for the given search terms. Similar metrics have been used in the past as part of measures to estimate similarity of software descriptions.

Let \( C[O] \) be a set of classes in ontology \( O \), and \( T \) is the set of search terms.

\[
\begin{align*}
E(O,T) &= \sum_{c \in C[O]} \sum_{t \in T} I = (c, t) \\
I(c,t) &= \begin{cases} 
1 : \text{label}(c) = t \\
0 : \text{label}(c) \neq t 
\end{cases} \\
P(O,T) &= \sum_{c \in C[O]} \sum_{t \in T} J(c,t) \\
J(c,t) &= \begin{cases} 
1 : t \in \text{label}(c) \\
0 : t \notin \text{label}(c) 
\end{cases} 
\end{align*}
\]

Where \( E(O,T) \) and \( P(O,T) \) are the number of classes of ontology \( O \) that have labels that match any of the search terms \( t \) exactly or partially, respectively.

\[
CMM(o, \tau) = \alpha E(O,T) + \beta P(O,T)
\]

Where \( CMM(o, \tau) \) is the Class Match Measure for ontology \( O \) with respect to search terms \( \tau \). \( \alpha \) and \( \beta \) are the exact matching and partial matching weight factors respectively.

The similarity match measure calculates how close the concepts of interest laid out in the ontology structure are. If the concepts are positioned relatively far from each others, then it becomes unlikely for those concepts to be represented in a compact manner, rendering their extraction and reuse more difficult.

V. SYSTEM IMPLEMENTATION

The STASIS Purifier traverses the STASIS network to filter inconsistencies in the information and find the entities which close to user’s requirements according the value of rate-degree. It uses the Search and View services for retrieving that information, and uses the Notify service to report on the located inconsistencies to the information owners.
A. Functions and Composites

Technically, STASIS Purifier will be implemented with three functionalities as shown in Figure 2, including configuration, crawling, as well notification and Subscription. Additionally, a timer component is separated from the three parts to trigger the crawling process and notification service.

1) Configuration.
This part consists of two sub components, the run-time configuration and the filtering configuration. The run-time configuration is to set the parameters for triggering crawling threads and notification service. The filtering configuration is to set criteria for the crawling process.

In this part, user will give out the reference attributes and data they prefer including running frequency of purifier, the domain of entities what they wants, the threshold of rank value and other description which are important and correlative with their purpose.

2) Crawling
This part is the most important part in the Purifier. There are several threads in these parts. These threads are running in the background to create the indexes of the elements (SSE/SSET) in STASIS, to collect the SLS-errors reported from users, to scan the users’ subscriptions and configurations.

3) Notification and Subscription
This part is responsible for the creation and saving the filtering results to XML files, and reporting the filtering results to relevant users.

On the one hand users can give their requirements on the Purifier through a configuration dialog. A real-time thread may continuously detect changes in the background and may be used to immediately update the purifier perspective in case of any changes.

Once crawling is complete, matching elements are displayed in the Results Dialog. This component allows users to do search and sort of the search results based on any of displayed properties, in either ascending or descending order.

B. Platform of Purifier Software

Based on the “one user one configuration” model, users can give out their different requirements on the Purifier through a configuration dialog. A real-time thread may continuously detect changes in the background and may be used to immediately update the purifier perspective in case of any changes.

VI. CONCLUSION AND FUTURE WORK

After the crawling is completed, matching elements are displayed in the Results Dialog. This component allows users to do search and sort of the search results intelligently based on any of displayed properties of requirements given by user or the description information about user. All results can be divided and saved into the file as useful entities or inconsistencies according to the value of rating. Figure 5 is the backup file to save all filtering results. It validates the architecture and matching arithmetic is effective.

On another side, the filtering function will be controlled by a timer function of the Purifier. It will step out according to the required timer to show the elements coincident based on all users’ configuration data and the filtering arithmetic.

In this paper, we describe the relation between entities in STASIS using OWL and record the relation with a kind of linking entities. Based on the comprehensive description of linking, we can realize a similarity rating arithmetic under the function of purifier in STASIS. Finally purifier report all the
entities which correlative with user’s requirements. On the other side, the information inconsistencies and conflicts are found and notified to STASIS.

Our arithmetic is not a full automotive function to deal with a semantic matching. In the future we will commit to improve the arithmetic to get more comprehensive and exactly results based on the platform.

ACKNOWLEDGMENT

The research work is obtained financial support from the Europeans Commission’s 6th Framework programme (FP6-2005-IST-5-034980), the National 973 Project (2006CB705407), the Country’s Natural Science Fund of China (60674080), and the 863 Plan of China (2007AA04Z150). The interested readers are able to access http://www.stasis-project.net for information.

REFERENCES

[1] www.stasis-project.net