# Fault Classification in P2P Semantic Mapping

**Abdul-Rahman Mawlood, Michael Weiss and Nicola Santoro**
Carleton University, Ottawa, Ont. K1S 4B6, Canada,
{armyunis,weiss,santoro}@scs.carleton.ca

## Abstract

An alternative to global ontology use in the semantic interoperability among heterogeneous information sources is the reaching of a consensus among distributed ontologies using local mapping and translation exchange. We investigate the construction of a flexible consensus system from distributed ontoliges in a peer-to-peer (P2P) network. We believe that such a flexible consensus system requires a semantic query processing with a fault-tolerance capability. Fault-tolerance capability refers to the ability to differentiate between permanent and non-permanent mapping faults. As first step in building such a flexible consensus system, this work provides a classification along the temporal dimension for the different types of faults that could arise in the context of semantic mapping in a P2P network.

## 1 Introduction

The success of the Semantic Web initiative and Web Services depends heavily on enabling semantic interoperability between distributed and heterogeneous information sources. The need for semantic interoperability between ontologies in a P2P environment is even more imperative. This is because, by definition participants in P2P environment are equal, autonomous and distributed. For example, the synthesis of concepts developed independently by different academic researchers, different research labs, various emergency service departments and hospitals and pharmacies, just to mention a few, are an assertive request for cooperation and collaboration among these independent peers [Bernstein *et al.*, 2002; Kementsietsidis *et al.*, 2003; Haase *et al.*, 2004a].

There has been considerable work on semantic interoperability, i.e. the mapping between different concepts from different ontologies. Some of this work suggests achieving interoperability through a global ontology mediator [Gomez-Perez *et al.*, 2003], while others suggest building a consensus incrementally with translation exchange and local mapping [Aberer *et al.*, 2003; Bonifacio *et al.*, 2003; Williams *et al.*, 2005]. We favor the latter approach.[1]

We are working on endowing the latter approach, i.e. semantic query processing using local mapping, with fault-tolerance capability[2]. A fault-tolerance capability denotes the ability to differentiate between permanent and non-permanent mapping faults during semantic query processing. As an initial step for endowing semantic query processing with fault-tolerance capability, this work provides a classification along the temporal dimension for the different types of faults that could arise in the context of semantic mapping in a P2P network. The temporal dimension refers to transient, intermittent and permanent fault types. Knowledge about different fault types is the enabling mechanism which facilitates the differentiation between permanently and temporarily uncooperative peers.

The rest of this paper is organized as follow: In Section 2, we provide arguments for why we need to add fault-tolerance to query processing. In Section 3, we introduce the definition of concepts used in this article. In Section 4, we provides several motivative examples. In Section 5 fault classification is provided. In Section 6, we review some existing semantic interoperability systems and methods and identify their lack of the fault-tolerance capability. Finally in Section 7 we conclude the work and describe the direction of our future research.

## 2 ON the Need for Fault-tolerance Endowing

The key substance in the current bottom-up construction of semantic inter-operable systems, i.e. reaching a consensus incrementally from the interaction of the local ontologies, is that every-time a peer P encounters another peer $\bar{P}$ that could handle its request, i.e. a peer with similar semantic knowledge representation, that peer $\bar{P}$ will be added to the list of related peers to peer P. This knowledge will be used for future collaboration, for example, when answering a query. However, if a peer P meets another peer $\hat{P}$ with a different semantic knowledge representation, that peer $\hat{P}$ will not be considered for subsequent tasks ( *Chatty Web* [Aberer *et al.*, 2003],

[1] For theoretical arguments on local ontology vs. global ontology, readers are encouraged to look into [Bouquet *et al.*, 2002; Bonifacio *et al.*, 2003]. We share the authors opinion on using local ontology instead of global ontology.

[2] A description of our work could be found in [Mawlood *et al.*, 2006]

*REMINDIN* [Tempich *et al.*, 2004], *KEx* [Bonifacio *et al.*, 2003] and *Local Consensus* [Williams *et al.*, 2005]).

One shortcoming with the above described bottom-up construction of a semantic inter-operable system is that, once a peer is unable to fulfill a particular request, for example answering a query, it will not be considered for the subsequent tasks. In other words, the described method sees the peers' inability to answer a query as a permanent fault - permanent non-cooperation. The described method does not make any distinction between permanent and temporary faults. We see this as a deficiency in the existing bottom-up construction of inter-operable system because peers' inability to answer a query could be a result of temporary disconnection, noise or incompetency to answer a particular request. This deficiency would result in the erroneous labeling of peers with the incompatible knowledge representation, excluding the labeled peers from *teaming up* [Kementsietsidis *et al.*, 2003] with the other knowledge comparable peers.

Accepting partial query results used by both Piazza and OBSERVER systems [Mena *et al.*, 2000a; Halevy *et al.*, 2003a] could be considered as an acknowledgment to the viability of reducing the number of expelled peers in consensus formation. It is worthwhile to mention that both OBSERVER and Piazza do not use the partial query result technique for preventing peers from being expelled.

To be able to extract the most consensus possible among related peers, we need to construct a consensus system that has a semantic query processing mechanism with a fault-tolerance capability. In other words, we should focus not only on the cooperative peers, which most of the existing works do, but also on uncooperative peers as well. Consensus formation should meet two objectives. 1. it should strive to identify the greatest possible common knowledge between all of the peers in a P2P network. 2. it should focus on cooperative peers but should not expel temporarily uncooperative peers from future consideration in the consensus formation.

The differentiation between different types of faults is particularly important in critical applications such as security and business applications. This particularity arises from the fact that excluding a useful source of information or a valuable business partner just for a transient type error will have severe consequences on the level of accuracy of the collected information and could jeopardize potential financial gain for the peers.

## 3 Fault and Fault Type Definitions

In this section, the definition of fault and mapping fault types is provided. We adapt the fault type definitions provided by the software and hardware fault-tolerance discipline to our context - the semantic mapping context. Reviewing this discipline we found that there are at least three different types of faults. These faults are different from each other based on the *duration* of the fault, i.e., the length of time faults stays in an active state. These faults are: i. permanent fault, ii. transient fault, and iii. intermittent fault [Paradhan, 1996]. In order to apply the described notion to our semantic mapping context, we will replace the duration which a system stays in non-operational mode by the duration in which a peer will be uncooperative. In other words, the duration in which peers were unable to perform semantic mapping or unable to perform correct semantic mapping. Characteristic of all three fault types is represented in Figure 1.

The existing works on fault in the semantic mapping considers fault as a semantic conflict [Naiman and Ouskel , 2002] or semantic incompatibility [Ram and Park , 2004]. In the works related to translation exchange and local mapping, the focus is on the *incorrect mapping* or *information losses* during transitive mapping process [Aberer *et al.*, 2003; Mena *et al.*, 2000b]. we are concentrating on the temporal issue of the faults, we define fault concept as follows:

**Definition 1**

Fault *is incorrect mapping or inabilities to do mapping between concepts with or without existence of semantic correspondence between concepts.*

More precisely, we say that a fault occurs when i. a concept in one ontology is incorrectly mapped to the concept of another ontology. ii. a concept in one ontology cannot be mapped to the concept of another ontology. Both situations occur regardless of the existence or non-existence of the corresponding concepts in the other ontologies.

An example of fault according to our definition will be as follow: lets assume that we have two ontologies O1 and O2 where $\alpha$ concept belongs to the ontology O1 and $\beta$ concept belongs to ontology O2. Let us also assume that the mapping from $\alpha$ to $\beta$ does exist and it is provided. Now, if we were unable to map $\alpha$ to $\beta$, or the result of mapping from $\alpha$ to $\beta$ is incorrect, we say that a fault has occurred.

**Definition 2**

Permanent mapping fault is a type of fault that continues to exist unless some outside action takes place to remove the cause of the fault [Paradhan, 1996]. For example, any attempt to map between two concepts from two unrelated ontologies where mapping correspondence does not exist between concepts results in an error. This situation will continue forever unless some outside action takes place, i.e., mapping bridge or mapping table between concepts been constructed and provided to the mapping process.

The permanent fault curve in Figure 1 is an illustration of mapping characteristic where waving lines are indications that the mappings performed correctly and, the flat line is an indication that a system no longer produce correct results.

**Definition 3**

Transient mapping fault is a type of fault that appears only once and stays for a short period of the time. A transient fault could damage the data but the system would remain in operational mode. It is a statistical fault and it is hard to predict when it will happen. For example, a symbol change for a company on the stock market could result in a transient error if: i. the propagation of the change notification to the related peers or applications is delayed and, ii. the related peers or application were unable to capture the change immediately. In other words, the change and the action of the source (peer) which is responsible for the change and/or the change and the action of the recipient of the change could be the cause of the transient error. The transient fault curve in Figure 1 illustrates a mapping characteristic where transient error represented by a sudden change in the curve for a short period of time [Dupont *et al.*, 2002].

**Definition 4**

Intermittent mapping fault is a type of fault that appears for a short period of time, disappears, and then reappears repeatedly. For example, it is very common for peers in P2P environment to disappear for a short period of time, reappear and then disappear again. Each disappearance of peer in P2P could result in a transient error. The multiple disappearances and reappearances of peers is what we call the intermittent errors. The intermittent curve in Figure 1 illustrates the intermittent behavior of a peer. Each sudden change in the curve shape represents a short error in the peer cooperation. Multiple error existence in the system is an illustration of intermittent behavior of the peer.

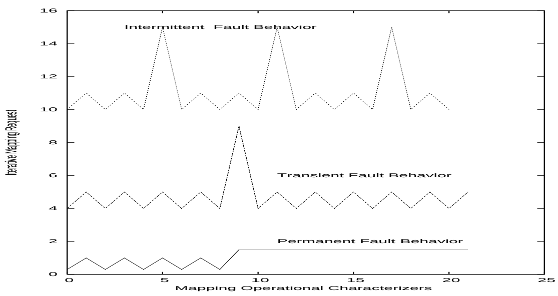


Figure 1: Fault Types

## 4 Motivation Examples

In this section we will provide several examples to illustrate the need for augmenting the fault-tolerance capability to the semantic query processing. These examples are situations under witch different types of faults could arise, and they need a proper handling to prevent its further consequences on the consensus formation.

Please note that the list of enumerated examples is not comprehensive. There are other situations such as the fuzzy concept representation, differences in the structure representation, etc which could cause faults similar to the one's listed below. Further more, there could be scenarios that we are not even aware of it yet that could appear in the future and cause fautls.

**Example 1** (Faults Caused by Temporal Concepts).

To avoid concept conflict in the mapping process among distributed and heterogeneous ontologies, an important issue which has to be accounted for during mapping is the change of data sources or ontologies over time. In other words, semantic interoperability between autonomous and heterogeneous ontologies is *time dependent*. This issue is found in situations where data/info changes continuously such as in stock prices and weather temperature.

The *price* in the *stock* ontology and the *temperature* in the *weather* ontology are properties of concepts with the special characteristic, their values change with the time.

A mapping procedure which compares the temperature values or compares stock prices from two different ontologies will have different results with time. Not accounting for time

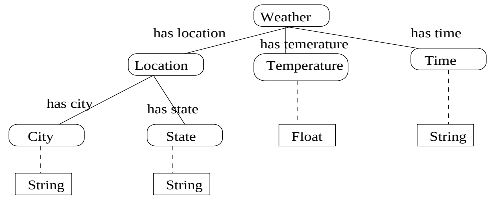


Figure 2: Partial Weather Ontology

dependency characteristics of mapping procedure could result in mapping failure. This failure could result in expelling peers for further considering in teaming up with other knowledge comparable peers.

Lets assume that there a network of cities each with a weather ontology similar to the one depicted in Fig 2. [3] Lets also assume that we want to find out the coldest city among them. One way to achieve this is by posing a query similar to the following one, written in XPath notation, on all the related cities and comparing the results.

$//location[/@Temperature < x]$ , x is coldest temperature found so far.

If the query processing experienced some time delay for some reason, or the queries were posed at different times on each peer, the result will be incorrect. This incorrectness is not the result of semantic representation differences, all uses same ontology, but rather the result of the temporal nature of temperature concept. This incorrectness could be temporary or permanent based on weather the temporal values are accounted for or not. Similar thing could be said about the query which tries to find out the cheapest stock price. Other examples related to the temporal change of the ontology concepts presented in [Zhu *et al.*, 2004].

**Example 2** ( Faults Caused by Ontology Modification )

Versioning and evolution issues are very well known in the software engineering and database design [Roddick, 1995]. The same issue also appears in ontology building. As in database and software engineering, replacing an existing component, an ontology in our context, with new version of the component or the modification to the existing component will have an impact on the overall system behavior. The issue of versioning and evolving of the ontology on the Web is even more dramatic because of the distributed and dynamic nature of the Web [Klein *et al.*, 2002].[4] Examples of ontology modification include:

- *adding new concepts* to the existing ontology. For example, adding a newly discovered class or type of drugs, proteins or diseases to the existing relevant ontologies.
- *deleting concepts* from existing ontologies. For reasons such as outdated concepts, no longer used or useful concepts, concepts may be deleted from the ontology structure.

---

[3] please note that this a partial diagram of a weather ontology created by http://refapp.semwebcentral.org/tutorial/ontology-knowledgebase/ontology-decomposition.html in the OWL syntax.

[4] Noy [Noy and Klein , 2004] argues that the issue of versioning and evolving are the same in the context of ontology mapping. What we see as an important matter is that, both versioning and evolving introduce modifications to the existing ontology.

• *change in meaning, conceptualization,* of the existing concept. Change in meaning could be done by removing/adding concept relation or property. Attaching hydrogen fuel type to the car concept, the fuel type which does not exist previously, is an example of change in concept conceptualization by adding new property. Removing disc drive from personal computer (PC)concept because the company no longer build PCs with disc drive build-in is another example of the change in the ontology concept by removing the concept property.

In circumstances where ontology modification is not a complete substitution to the previous ontologies, there is a possibility for related peers or application to continue working. In the described scenario, there are possibilities for fault occurrence -intermittent type of fault. Faults will occur because there are situations where related peers are unable to interpret the meaning of concepts in modified ontologies. Other fault types caused by ontology modification will be further elaborated on in the section 5.

**Example 3** (Faults Caused by Context and Static Mapping ) Static mapping is a type of mapping that does not consider context (the relations and properties of a concept) during the mapping process when the concept in a certain ontology is about to be mapped to another concept in a different ontology. Static mapping looks at the concepts as an *isolated single term* and mapping procedure as a *term to term* comparison. Once the mapping procedure concludes that a concept A for example is equivalent to another concept B, it will produce the same results, A is equivalent to B, every time mapping procedure is applied. This happens regardless of discovering new evidence that contradict the initial mapping conclusion. Static mapping creates situations such that using the *same existing concept correspondence* between ontologies result in a different mapping outcome when applied to different queries, i.e. incorrect mapping.

The notion of static mapping could be explained more in the following example. Lets assume we have two ontologies. The first ontology represents information about University student and the second ontology represents information about Research Institute members as represented in Fig.3. Let us also assume that some form of relations exists between the two ontologies. For example, some of the Research Center members are University students and Employer concept represents these information, i.e. the domain of Employer is Universities and Institutes. Also, both the University concept from the first ontology and the Institute concept from the second ontology are synonymous since both concepts could be mapped to a common concept the Institute. Now lets consider the following query: Q1. list the name of all Institutes in the area. This query could be posed on both ontologies and the relation (University, Institute) be asserted. But if the query Q2: list the name of all educational Institutes is posed instead, the synonymous relation between (University, Institute) no longer holds and its assumption will result in error. In other words, while the semantic correspondence between concepts produces a perfect outcome for one query, it results in an error for the next one [Ouksel, 1999].

The above scenario could be a perfect example of what we call intermittent type of fault. This is because every time an existing correspondence between two concepts, i.e., existing mapping, is used in mapping contexts other than contexts were the relation defined for, an error will occur. Other works, such as the work which is been done by Bouquet [Bouquet *et al.*, 2003], could be used for further elaboration on the effect of context and static mapping on the fault type.

| | |
|---|---|
| DTD for | DTD for |
| Source1.xml | Source2.xml |
| Student | Member |
| StudentID | MemberID |
| University | Institute |
| | Employer |

Figure 3: DTD for Source XMLs

**Example 4** (Faults Caused by Unavailability of Data Sources ). It has been pointed out by Gal [Gal, 1999] that the design of the conceptual schema for information services possesses special properties. These special properties includes: i. repaid change of *data source and meta-data*, and ii. instability, since there is no control over the information sources. The *availability* of information sources is solely dependent upon the information source's decision. A possible scenario that could arise is the temporary unavailability of information when such information is needed. This possibility is particularly acute during query execution.

**Example 5** ( Faults Caused by Peers' Misbehavior) The semantic mapping correctness in the P2P environment depends on the honest conduct of peers. A peer could be dishonest or biased in his interaction with other peers during the mapping process for reasons such as selfishness and greed. There are various ways through which a peer could influence the mapping process. These ways include: i. not forwarding a query to other peers during transitive mapping process or, ii. not forwarding answers to the other peers during mapping and, iii. altering/delaying the queries/results before forwarding to other peers. In all the above situations the mapping process yields incorrect mapping [Aberer *et al.*, 2004].

Working in hostile or uncooperative environment, gives rise to situations where peers are permanently hostile or uncooperative. This is leads to permanent faults. Other situations could arise from unintentional misinterpretation or misimplementation of mappings. In the latter case, since the fault(s) will be produced from the noise like type of acts, it will be correct to assume that the fault(s) will be non-permanent type of fault.

In all above scenarios we need to differentiated between permanent and temporary mapping fault. the knowledge about different type of faults along the temporal dimension will help us to determine when peers should be expelled for further interaction. This helps in better consensus formation which in turn helps in solving the semantic interoperability problem.

## 5 Classification of Faults

It should be evident by now that, in order to build a a flexible consensus system we need a semantic query processing mechanism with a fault-tolerance capability. As the first step to solve this problem, we provide a classification for the different types of faults along the temporal dimension. We have identified three types of faults and several sources for fault. The classification is based on the (fault sources, fault type) association. Please note, since we assume that local mapping between ontologies preexist, our classification emphasizes on the type of faults that may occur during *mapping execution* rather than on those faults that may occur because of *mapping design logic*, for example substituting a concept by its hypernyms or hyponyms. Hence, the mapping faults caused by meaning and representation of concept are not included in this classification. For this type of fault we refer reader to the [Naiman and Ouskel , 2002; Ram and Park , 2004; Glushko, 2005].

In order to make the analysis simple we distinguish between two cases: i. semantic mapping where support for handling fault source (FS, here after) is provided, and ii. semantic mapping without support for handling FS. Also, for simplicity and to minimize confusion some time we refer to both intermittent and transient type errors as non-permanent faults. Table 1 summarizes this classification.

### 5.1 Fault Types when FS is not handled

Here we list the situations under which both permanent and non-permanent mapping faults could occur because no fault handling mechanism been provided for removing the source of the fault.

**1. Permanent Faults**:

• mapping temporal concepts without representing time constraints in the ontology leads to permanent faults. This is because temporal ontology concepts are continuously changing with time. Even if the mapping process produces some correct mapping without considering for the time constraints, they are random mapping and eventually the system will be in total failure state.

• the *level* of ontology modification (versioning and evolution) and whether or not the modified concepts will be *used* in the mapping process will determine the mapping result. The high level of modification and the repeated use of the modified concepts could give rise to inability of related applications or peers to work with the modified ontology.

• if the system is *unavailable* (Example 4), in other words, the unavailability time $= \infty$, the mapping process cannot be performed. The unavailability could be the result of network failure or peer failure.

• working in hostile or uncooperative environment, gives rise to situations where peers are permanently hostile or uncooperative. This is leads to permanent faults.[5]

We would like to bring to the attention that the *static mapping and context* (Example 3) will not lead to permanent faults. This is because it does not make sense to use an existing mapping which contradicts the context all the time. If this is the situation, it means that the existing mapping is not complete. Hence, a better concept mapping is required. Also, as indicated above the modification of ontology will not lead to faults all the time.

[5]if multiple peers cooperate and misbehave intensionally, this will create different type of faults known as *byzantine* fault, not considered in this work.

**2. Non-permanent Faults**. Except from those situations identified in the first case all other situations will result in non-permanent faults. These situations are:

• change in query context (Example 3) could give raise to intermittent type of fault. This is because every time an existing correspondence between two concepts, i.e., existing mapping, is used in mapping contexts other than contexts were the relation defined for, an error will occur.

• in circumstances where ontology modification (Example 2) is not a complete substitution to the previous ontologies, there is a possibility for related peers or application to continue working. In the described scenario, there are possibilities for fault occurrence -intermittent type of fault. Faults will occur because there are situations where related peers are unable to interpret the meaning of concepts in modified ontologies.

• unintentional misinterpretation or misimplementation of mappings (Example 5) gives rise to a an incorrect mapping. Since the fault(s) will be produced from the noise like type of acts, it will be correct to assume that the fault(s) will be non-permanent type of fault.

### 5.2 Fault Types when FS is handled

There are considerable efforts underway to solve each individual issues causing the fault. Hence, it is interesting to see what type of faults could arise regardless of incorporating the exiting solutions for each individual fault source. Before going any further in detail, we would like to make the following two observations:

**i.** once proper solutions for each individual faults have been provided, *permanent faults will not occur*. This is because, as indicated in the 5.1 subsection, permanent fault will become an issue only when participating peers are either totally unavailable or misbehave permanently. With the assumption about the existence of some sort of solution, the two mentioned cases will become invalid. Hence, the permanent fault will not happen when FS is handled.

**ii.** if peers' unintentional misinterpretation or misimplementation for queries or mappings are handled properly, there is no reason to believe that a *transient or intermittent* fault type will occur because of peers' misbehavior. Similarly, once a proper solution for handling context and static mapping available, query context no longer represents a threat to incorrect mapping.

In the rest of this subsection we will concentrate our discussion on the *non-permanent* faults:

• Even if the time constraints represented in temporal ontology concepts, there is situation that could give raise to a transient type fault. For example, if *delay* is experienced in query propagation during transitive mapping. In other word, if *Query starting time*[6] + *query delivery time*[7] $>$ *information display time*[8], mapping yields to incorrect mapping or null value. Depending on the frequency of query propagation delays we will have intermittent or transient type faults.

[6]a time when query submitted to the other peers.

[7]a length of time that a query takes to be propagate from peer A to peer B.

[8]a point in time where the information on the remote site is correct.

• We all experienced in one form or another denial of service request because of temporary server crashes, disappearance and reappearance of peers in P2P environment. If these situations happen during mapping process, we will experience non-permanent faults.
• Each modification to ontology could result in one of the following two case: i. *unavailability* for short period of time, if the ontology is blocked while the modification is performed or, ii. a *race* situation between information source and information users, if the ontology user is informed about the change *before* or *after* the modification is made. In other words, the modification problem is an instance of unavailability or temporal problems described before. Hence we could have a non-permanent fault type every time an ontology is modified.

The following characteristics about the ontology modification, unavailability and temporal concept could be generalized (extracted):

- The effect of ontology modification is not as dramatic as the effect of unavailability. This is because we assume that modification to ontologies should be less frequent than information source unavailabilities.

- the probability for transient type fault could be higher than intermittent fault type. This is again because of the expectation about less frequent ontology modifications.

It is important to note that we looked at the sources of faults one at a time. For example, we studied the effect of context, temporal aspects, modification etc separately. One can foresee some interesting questions here. Hence, we pose the following research questions: *is there a possibility for a fault to be a result of multiple causes? is it important to distinguish between different sources of fault? or is there a solution which could capture the fault regardless of the source of the fault?*

Thesummary of fault classification is presented in Table 1

| | **Transient Fault** | **Intermittent Fault** | **Permanent Fault** |
|---|---|---|---|
| **Temporal Semantic Conflict** | One time message delay | Frequent message delays | unsupported time constraint |
| **Ontology Versioning and Evolution** | During changes | During changes | unsupported change management |
| **Query Context and Static Mapping** | Unsupported Query Context | Unsupported Query Context | Disqualify |
| **Unavailability of Data Source** | unavailability $\geq$ Timeout | Frequent unavailability $\geq$ Timeout | unavailability = $\infty$ |
| **Peers' Misbehavior** | one time misbehavior | repeated misbehavior | Permanent misbehavior |

Table 1: Fault Classification in Semantic Mapping Process

## 6 Existing Semantic Mapping Approaches

In this section we choose five different systems and methods for scrutiny. These systems are: Chatty Web, OBSERVER, Piazza, H-Match and KEx. The common theme among selected approaches is the use of local mapping to achieve some form of knowledge sharing and cooperation. When applicable, we will pin point to the strategy(ies) used by these systems to tolerate fault. In examining these approaches we look at five different aspects: purpose, environment, mapping strategy, future interaction and fault-tolerance ability. In the following we will go through these aspects and provide descriptions to the selected system. Table 2 summarizes how the selected approaches address the chosen aspects. Please note that the purpose of this section is neither to provide a full survey on existing systems and methods, nor to examine each of these individual systems or methods thoroughly.

**1. Purpose of the approach.** The objective here is to highlight the exact goal of each approach.

**2. Environment used.** The approach which could be applied to a certain network topology is highlighted here.

**3. Mapping strategy.** Here we make the distinction between the approaches based on their way to perform mapping. For the systems surveyed, the following mapping modes were identified: i. mapping pre-exists - mapping is a separate activity that could be plugged in to the query processing. ii. built-in mapping - mapping is part of the system and is used during system operation. iii. provided as a separate component, the system has a separate component where the relation between concepts from various ontologies can be declared manually.

**4. Future interaction.** Here we determine how different approaches use the collected information from the interaction with other peers. Some approaches make use of the interaction history for future query forwarding (i.e., builds and updates a routing table), while other approaches cache answers for better performance.

**5. Fault-tolerance.** Considers peers' reaction to those interactions that did result in incorrect answers in various approaches.

A short description of selected systems are:

**Chatty Web**. [Aberer *et al.*, 2003] describes a method for building a *common ontology* or semantic global agreement from local interaction between peers. Each peer has its own schema that is different from schema's of other peers. Using the XQuery language, a query imposed on one peer could be translated to another query and imposed on a different peer. During their life time, peers will be able to find the other related schemes. Hence, a directed graph of the related peers will be constructed. The Chatty Web method does not distinguish between permanent and temporary faults. All faults are considered to be permanent.

**OBSERVER**. [Mena *et al.*, 2000a] is a *query processing* approach in the global information system. This system uses multiple preexisting ontologies. For locating information, the system navigates ontologies using inter-ontology relations between them based on the *synonyms, hypernyms* and *hyponyms* relation between terms or concepts used in each ontology. The relation between ontologies have to be determined by human experts and provided to the system manually; as such, the relation between ontologies (query processing) is constrained by the the defined relations between ontology terms. This is a disadvantage if compared to consensus emergence in [Aberer *et al.*, 2003].

**Piazza**. [Halevy *et al.*, 2003a] develop the infrastructure and *mapping language* for semantic mapping and data management in P2P environment which takes into account both the domain and the document structure. The infrastructure, called Piazza, and mapping language use features of the XQuery language. In Piazza, queries posed on one peer can be reformulated or rewritten to other peers, thus the transitive closure of these translations or mappings could be used to answer queries using both *local* peer-to-peer mapping and *mediated* mapping methods. In Piazza, information loss during query translation is not evaluated and, faults and tolerance to faults are not accounted for.

**H-MATCH**. [Castano *et al.*, 2003] is an algorithm for matching ontology concepts. The purpose of the algorithm is to enable *knowledge sharing* and *knowledge evolution* in P2P environment. H-MATCH uses the meaning of terms used as a name and properties, and the context of a concept (i.e., the set of properties and the set of adjacent concept) to perform matching. The matching processes uses WordNet thesaurus to conclude the relation between concepts. H-MATCH provides three different ways of matching: shallow matching, intermediate matching and deep matching.

**KEx** is an architecture for semantic search in a P2P network based on the principles that the heterogeneity of knowledge representation should not be seen as an obstacle in front of knowledge management but rather as an opportunity for promoting innovation. KEx facilitates building a community of sharing within a network of autonomous peers such that the knowledge within the space will be available for other peers and the search for interested information will be possible. Each peer could either request information from other peers or provide information to other peers or group of peers. A document repository and a context repository, ontology, is associated with each peer. A document repository is the place where the structured document is been saved and context repository is the place where the semantics of concepts will be clarified during query request and answers. Different kind of knowledge could be saved at document repositories including references to experts in some domain, links to other peers, to external resources etc. KEx do not tolerates the non-permanent faults.

Table 2 summarizes some aspect of translation exchange and semantic mapping for selected systems and methods.

| | **Purpose** | **Environment** | **Mapping** | **Future interaction** | **Fault-tolerance** |
|---|---|---|---|---|---|
| **Chatty Web** | Building global consensus | Orthogonal | local and preexist | Routing table built | Not handled |
| **OBSERVER** | query processing in global information system | Network of related ontologies | terminological relationship | Not handled | Not applicable |
| **Piazza** | Query answering using mapping | Network of sites or peers (P2P) | local and preexist | Query answers cached | Not handled |
| **H-MATCH** | Knowledge sharing and evolution | P2P | Built-in | Routing table built | Not handled |
| **KEx** | knowledge discovery and exchange | P2P | Built-in | Query answers cached & Routing table built | Not handled |

Table 2: Some Aspects of the Selected Systems and Methods

## 7 Conclusion and Future Works

The lack of semantic interoperability among heterogeneous and distributed ontologies impedes the Semantic Web initiative. One way to tackle this problem is by reaching consensus among distributed ontologies. There are several methods for reaching consensus including: negotiation [Moor, 2005], argumentation [Tempich *et al.*, 2005] and translation exchange [Aberer *et al.*, 2003; Bonifacio *et al.*, 2003]. This work focused on improving the latter approach by endowing it with fault-tolerance capability. A fault-tolerance capability denoted as the ability to differentiate between permanent and non-permanent mapping faults during semantic query processing. This work also emphasized that the failure to distinguish between permanently and temporarily mapping faults could result in the erroneous labeling of peers with incompatible knowledge representation which could have further consequence of preventing labeled peers from teaming up with other knowledge-comparable peers. As first step in solving this problem, we provided a classification along the temporal dimension for the different types of faults that could arise in the context of semantic query processing in a P2P network. The temporal dimension referred to transient, intermittent and permanent fault types. Future research directions will center on two research strategies: i. algorithm development, and ii. system implementation. Algorithm development will focus on the incorporation of fault-tolerance capabilities in semantic consensus building. System implementation will focus on validating the effectiveness of flexible consensus building in real world applications.

## References


[Aberer *et al.*, 2003] K. Aberer, P. Cudr-Mauroux, M. Hauswirth. Start making sense: The Chatty Web approach for global semantic agreements. In *Journal of Web Semantics*, 1(1): 89-114, 2003.

[Aberer *et al.*, 2004] K. Aberer, T. Catarci, P. Cudre-Mauroux, T. Dillon, S. Grimm, M.-S. Hacid, A. Illarramendi, M. Jarrar, V. Kashyap, M. Mecella, E. Mena, E.J Neuhold, A.M. Ouksel,T. Risse, M. Scannapieco, F. Saltor, L. de Santis, S. Spaccapietra, S. Staab, R. Studer and O. De Troyer. Emergent semantics systems. In *Semantics of a Networked World. Semantics for Grid Databases. First International IFIP Conference, ICSNW*, P: 14-43, 2004.

[Bernstein *et al.*, 2002] P. Bernstein, F. Giunchiglia, A. Kementsietsidis, J. Mylopoulos, L. Serafini and I. Zaihrayeu. Data management for peer-to-peer computing: A vision. In *Workshop on the Web and Databases, WebDB*, 2002.

[Bonifacio *et al.*, 2003] M. Bonifacio, P. Bouquet, G. Mameli and M. Nori. Peer-Mediated Distributed Knowledge Management. In *proc. of International Symposium AMKM* , pp. 31-47, 2003.

[Bouquet *et al.*, 2002] P. Bouquet, A. Donk, L. Scrafini, and S. Zanobini. ConTcXtualizcdlocal ontology specification via ctxml. In *AAAI-02 workshop on Meaning Negotiation*. AAAI Press, 2002.

[Bouquet *et al.*, 2003] P. Bouquet, F. Giunchiglia, F. v. Harmelen, L. Serafini and H. Stuckenschmidt. C-OWL: Contextualizing Ontologies. In *Proceedings of the 2nd International Semantic Web Conference*. Springer, pp. 164-179, 2003.

[Castano *et al.*, 2003] S. Castano, A. Ferrara, S. Montanelli. H-Match: an Algorithm for Dynamically Matching Ontologies in Peer-based Systems. In *Proc. of the 1st VLDB Int. Workshop on Semantic Web and Databases (SWDB)*, P: 231-250, 2003.

[Dupont *et al.*, 2002] E. Dupont, M. Nicolaidis and P. Rohr. Embedded Robustness IPs for Transient-Error-Free ICs. In *IEEE Design & Test* 19(3): 56 - 70, 2002.

[Fergus *et al.*, 2003] P. Fergus, A. Mingkhwan, M. Merabti,and M. Hanneghan. Distributed emergent semantics in P2P networks. In *Proc. of the Second IASTED International Conference on Information and Knowledge Sharing*, P: 75-82, 2003.

[Gal, 1999] A. Gal. Semantic interoperability in information services: experiencing with CoopWARE. In *SIGMOD Record*, 28(1):68-75, 1999.

[Glushko, 2005] R. J. Glushko and T. McGrath. Document Engineering. The MIT Press,2005.

[Gomez-Perez *et al.*, 2003] A. Gomez-Perez, M. Fernandez-Lopez and and O. Corcho. Ontological Engineering. Springer publishing, 2003.

[Haase *et al.*, 2004a] P. Haase, J. Broekstra, M. Ehrig, M. Menken, P. Mika, M. Olko, M. Plechawski, P. Pyszlak, B. Schnizler, R. Siebes, S. Staab, C. Tempich. Bibster - a semantics-based bibliographic peer-to-peer system. In *Proc. Third International Semantic Web Conference (ISWC)* , P: 122-136, 2004.

[Haase *et al.*, 2004b] P. Haase, R. Siebes and F. van Harmelen. Peer selection in peer-to-peer networks with semantic topologies. In *Semantics of a Networked World. Semantics for Grid Databases. First International IFIP Conference, ICSNW* , P: 108-125, 2004.

[Halevy *et al.*, 2003a] A. Halevy, Z. Ives, P. Mork, and I. Tatarinov. Piazza: Mediation and integration infrastructure for semantic web data. In *proceedings of the International World-Wide Web Conference WWW-03*, 2003.

[Halevy *et al.*, 2003b] A. Halevy, Z. Ives, D. Suciu and I. Tatarinov. Schema mediation in peer data management systems. In *Proc. 19th International Conference on Data Engineering*, P: 505-516, 2003.

[Kementsietsidis *et al.*, 2003] A. Kementsietsidis, M. Arenas and R. Miller. Managing data mappings in the hyperion project. In *n Proc. of the 19th International Conference on Data Engineering (ICDE)*,732-734, 2003.

[Klein *et al.*, 2002] M. Klein,A. Kiryakov, D. Ognyanov,and D. Fensel. Finding and characterizing changes in ontologies In *Proc. 21st International Conference on Conceptual Modeling*, P: 79-89, 2002.

[Mawlood *et al.*, 2006] A-R. Mawlood-Yunis, M. Weiss, N. Santoro. Issues for Robust Consensus Building in P2P Networks. In *International Workshop on Ontology Content and Evaluation in Enterprise (OnToContent), LNCS 4278*, P: 1020-1028, 2006.

[Mena *et al.*, 2000a] E. Mena, A. Illarramendi, V. Kashyap and A. Sheth. OBSERVER: an approach for query processing in global information systems based on interpretation across pre-existing ontologies. In*Distributed and Parallel Databases*, 8(2):223-71, 2000.

[Mena *et al.*, 2000b] E. Mena, V. Kashyap, A. Illarramendi, and A. Sheth . Imprecise answers in distributed environments: estimation of information loss for multi-ontology based query processing. In *International Journal of Cooperative Information Systems*, 9(4): 403-25, 2000.

[Moor, 2005] A. de Moor. Ontology-Guided Meaning Negotiation in Communities of Practice. In *Proc. of the Workshop on the Design for Large-Scale Digital Communities at the 2nd International Conference on Communities and Technologies* pp. 21-28, 2005.

[Naiman and Ouskel , 2002] C.F. Naiman and A.M. Ouskel. A classification of semantic conflicts in heterogeneous database systems. In *Journal of Organizational Computing, 5(2):167-193*, 1995.

[Noy and Klein , 2004] N. F. Noy and M. Klein. Ontology Evolution: Not the Same as Schema Evolution. In *knowledge and Information Systems*, 6(4):428-440, 2004.

[Noy and Musen , 2004] N. F. Noy and M. M.A. Musen. Ontology versioning in an Ontology management framework . In *IEEE Intelligent Systems*, 19(4):6-13, 2004.

[Ouksel, 1999] A. M. Ouksel. Ontologies are not the panacea in data integration: a flexible coordinator to mediate context construction. In *Distributed and Parallel Databases*, 15(1):7-35,1999.

[Paradhan, 1996] D. K. Paradhan. Fault-Tolerant Computing System Design. *Prentice-Hall PTR* publication, 1996.

[Ram and Park , 2004] S. Ram and J. Park. Semantic Conflict Resolution Ontology (SCROL): An Ontology for Detecting and Resolving Data and Schema-Level Semantic Conflicts. *16( 2):189-202*, 2004.

[Roddick, 1995] J. F. Roddick. A survey of schema versioning issues for database systems. In *Information and Software Technology*,37(7):383-393,1995.

[Stephens and Huhns , 2005] L.M. Stephens, M.N. Huhns. Consensus ontologies. Reconciling the semantics of Web pages and agents. In *IEEE Internet Computing*, 5(5): 92-95, 2001.

[Stojanovic *et al.*, 2003] L. Stojanovic, A. Maedche, N. Stojanovic and, R. Studer. Ontology evolution as reconfiguration-design problem solving. In *Proc. of the 2nd international conference on Knowledge capture*, P: 162-171, 2003

[Studer *et al.*, 1998] R. Studer, VR. Benjamins and D. Fensel. Knowledge Engineering: Principle and Methods. In *IEEE Transactions on Data and Knowledge Engineering*, 25(1-2):161-197,1998.

[Tempich *et al.*, 2004] C. Tempich, S. Staab, A. Wranik. Distributed semantic query: Remindin': semantic query routing in peer-to-peer networks based on social metaphors In *Proc. of the 13th international conference on World Wide Web*, P: 640-649,2004.

[Tempich *et al.*, 2005] C. Tempich, H.S. Pinto, Y. Sure, and S. Staab. An argumentation ontology for distributed, loosely-controlled and evolving engineering processes of ontologies (DILIGENT). In *Proceedings of Second European Semantic Web Conference ESWC 2005* (LNCS 3532), P: 241-56, 2005.

[Williams *et al.*, 2005] A. B. Williams, A.Padmanabhan and M. B. Blake. Experimentation with Local Consensus Ontologies with Implications for Automated Service Composition. In *IEEE Transactions on knowledge and data engineering*, 17(7):969-981, 2005.

[Zhu *et al.*, 2004] H. Zhu, S.E Madnick and M.D. Siegel. Effective data integration in the presence of temporal semantic conflicts. In *Proc. 11th International Symposium on Temporal Representation and Reasoning*, P: 109-114, 2004.