SEMANTIC-BASED PUBLISH/SUBSCRIBE SYSTEM IN SOCIAL NETWORK

A Paper
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Farzana Jahan

In Partial Fulfillment
for the Degree of
MASTER OF SCIENCE

Major Department:
Computer Science

December 2013

Fargo, North Dakota

North Dakota State University
Graduate School

Title

Semantic-based Publish/Subscribe System in Social Network

By

Farzana Jahan

The Supervisory Committee certifies that this ***disquisition*** complies with North Dakota State University's regulations and meets the accepted standards for the degree of

MASTER OF SCIENCE

SUPERVISORY COMMITTEE:

Dr. Juan(Jen) Li
Chair

Dr. Jun Kong

Dr. Yarong Yang

Approved:

01/08/2014	Dr. Brian M. Slator
Date	Department Chair

ABSTRACT

The publish/subscribe model has become a prevalent paradigm for building distributed notification services by decoupling the publishers and the subscribers from each other. The semantics-based publish/subscribe system allows highly expressive descriptions of subscriptions and publications and thus is more appropriate for content dissemination when a finer level of granularity is necessary.

In this paper we have designed and implemented a semantic-based publish/subscribe system that can be adapted into social networks where thousands of people can share their common interests through publications and subscriptions. We have described our ontology, defined publishers' and subscribers' data semantics or schema, provided a matching algorithm, portrayed the implementation and shown the result of an implemented publish/subscribe system that allows the users to publish and subscribe different kinds of news in a social network platform. Our experience shows that the semantic-based publish/subscribe system can enhance the current social networks by providing an effective content dissemination mechanism.

ACKNOWLEDGEMENTS

I would like to express my deepest appreciation to my advisor Dr. Juan Jen Li who has been an excellent mentor during my Master's paper study. Without her guidance, support and help, I would never been able to come up with this paper. I feel very fortunate to be her student. Finally I would like to thank faculty of the Computer Science Department and member of my supervisory committee for their tremendous support during my study.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

1.1. Motivation

In recent years, the demand for social-networking sites has increased dramatically and has changed many of the social aspects of communication on the internet. Popular social-networking communities, such as MySpace, Facebook, Friendster, Tribe, and Twitter provide an avenue for friends and strangers to share common interests by providing a universal standard of their minute–to-minute activities [1]. A Publish/Subscribe system is another popular paradigm in which users can subscribe or publish enormous amounts of information based on their interests. These two emerging phenomenon of social networks and publish/subscribe systems have motivated us to present an extensive proposal of enhancing social-networking systems with the integration of a publish/subscribe system in order to make both more desirable and attractive to the users. In this paper, we intend to show how semantic-based publish/subscribe functionality can be adapted into social networks based on user's interest in subscription and publication.

This paper explores the idea of the integration of a publish/subscribe system based on semantic matching between the subscribers' interests and publishers' publication with a large distributed social network system by decoupling the publishers and the subscribers from each other. This semantic-based publish/subscribe is an appealing standard for the distributed and selective content delivery system. It enables subscribers to express their interests within datasets by registering subscriptions with the system in order to notify matching subscribers about any forthcoming events issued by publishers. The existing publish/subscribe systems are primarily based on person-to-person communications, such as Instant Messaging and E-mail, or group communication systems, which allow subscriptions to relatively a few groups of people as well as a low volume of information. In the existing publish/subscribe system a single publisher originates every message, but in social networks the message can be originated from many users and must be delivered to the destination in a timely manner. This is because in social networks, people may have hundreds of friends with thousands of interests. This unique nature of the social networking demands more from the underlying communication system [1]. Moreover, the current social-networking systems are centralized and each network is controlled by a single entity. But as the demand for social-networking systems grows, both privacy and scalability concerns associated with centralized systems make them undesirable to the end-users. Major disadvantages of centralized systems are the lack of quality and fault tolerance. Also, in current social networks, the accessibility of the publisher's posting is dependent on the publisher's central privacy setting. Therefore, this posting may either go to everybody in the system or to the friends that the publisher is associated with. The system also doesn't pay regard to the interest of the people who are receiving this posting. This is because the system doesn't provide the matching of the publishers' posting and peoples' interest. To alleviate these issues of social networking, we are proposing a semantic-based distributed publish/subscribe system. Our design uses highly descriptive subscription criteria which allow proper dissemination of information with a finer level of granularity. This design is asynchronous in nature and privacy of the publisher and subscriber is protected as the design decouples publisher from the subscriber.

The proposed semantic-based publish/subscribe system's design is more flexible and useful since subscribers can specify their interests more accurately using a set of predicates.

The matching of subscriptions and publications is based on content and no prior knowledge is needed (e.g. the set of available topics). Our publish/subscribe system can be implemented centrally or in a distributed manner. This approach uses structured semantic knowledge of subscribers' interests of different kinds to calculate the similarity between the published topics and the subscribers' interest and it is also shown that new interest can be handled by taking domain specific ontology into consideration. Therefore, the design not only allows us to automatically match the users with common interests but also extends the existing publish/subscribe systems by integrating new users interest to the interest profile hierarchy. Since semantic-based ontology supports a formal as well as conceptual annotation for an effective inference query, intelligent users interest matching is possible.

1.2. Problem Statement

In this paper, we designed a semantic-based publish/subscribe system for social-network. In our design, we considered a publish/subscribe system for news on music where publishers publish various types of music news and subscribers get notification on news types based on their interest. For example, a publisher publishes a news link on pop music; this publication is notified to the subscriber whose subscription interest matches with pop music.

As a part of the system design, we have described our ontology, defined publishers' and subscribers' data semantics or schema and provided a matching algorithm. At the end of this paper we have portrayed the implementation and result of an implemented publish/subscribe system that allows users to publish and subscribe the news in a social network platform and discussed different scenarios that reflects the mechanism of our system. A popular semantic framework known as Resource Description Framework (RDF) [4] has been used for designing schemas for semantic-based publish/subscribe capabilities in a social network. The rest of the paper is organized as follows:

- Chapter 2 provides the background of semantic web and its key component, explains the publish/subscribe systems, RDF [4], OWL [5] and ontology design
- Chapter 3 provides the overall design framework of the semantic-based publish/subscribe system
- Chapter 4 provides the result with examples and detail analysis on those examples
- Chapter 5 concludes the paper with a brief discussion on summary, limitations and future work of the proposed design

CHAPTER 2: BACKGROUND STUDY & RELATED WORK

2.1. Background Study

2.1.1. Publish/Subscribe System

The publish/subscribe system is a well-established communication pattern where publishers publish a set of information using a set of publications and subscribers specify their interests using a set of subscriptions. Upon receiving a publication, the system accepts publication and relays the publication to the matching subscribers. The publish/subscribe model decouples time, space, and flow between publishers and subscribers and thus reduces program complexity and resource consumption [2].

In the publish/subscribe model, subscribers typically receive only a subset of the total messages published. The process of selecting messages for reception and processing is called filtering. There are two common forms of filtering: topic-based and content-based [3]. In topic-based systems, subscribers join a group containing a topic of interest [2]. Publications that belong to the topic are broadcasted to all members of the group. Therefore, publishers and subscribers must explicitly specify the group they wish to join. Topic-based systems are similar to the earlier group communication and event-notification systems.

In content-based publish/subscribe systems, the matching of subscriptions and publications is based on content and no prior knowledge is needed; therefore systems are more flexible and useful since the subscribers can specify their interests more accurately using a set of predicates. Building this type of system requires extensive and efficient matching between millions of publications and subscriptions.

There are some systems that support a hybrid of topic-based and content-based system. Publishers post messages to a topic while subscribers register content-based subscriptions to one or more topics. In many publish/subscribe systems, publishers post messages to an intermediary message broker or event bus, and subscribers register subscriptions with that broker, letting the broker perform the filtering [3]. The broker normally performs a store and forward function to route messages from publishers to subscribers. In addition, the broker may prioritize messages in a queue before routing. Subscribers may register for specific messages at build time, initialization time or runtime.

The main advantages of using the publish/subscribe systems are loose coupling and scalability. Publishers are loosely coupled to subscribers; that means, subscriber need not even know of their existence. With the topic being the focus, publishers and subscribers are allowed to remain ignorant of the system topology. Each can continue to operate normally regardless of the other. The publish/subscribe system provides the opportunity for better scalability than traditional client–server through parallel operation, message caching, tree-based or network-based routing, etc. The major disadvantage with the publish/subscribe systems is a side-effect of their main advantage: the decoupling of publisher from subscriber. A publish/subscribe system must be designed carefully enough to be able to provide stronger and efficient system properties and matching algorithm that offers assured delivery.

2.1.2. Semantic Web

The semantic web is the extension of the World Wide Web that provides the people an easier way to find, share, reuse and combine information beyond the boundaries of applications and websites. The semantic web is a globally linked mesh of ontological information that can be understood programmatically by the machine. The semantic web uses ontological schemas to define data in web documents and creates instances of those objects defined in the schemas as metadata added to web documents. This metadata can then be polled by artificially intelligent agents to produce meaningful search results or provide more details about web resources [6].

The current World Wide Web is based mainly on documents written in Hypertext Markup Language (HTML) [7]. HTML is a markup convention that is used for coding a body of text consisting of tags enclosed in angle brackets (such as <html>) within the web page content. Humans are capable of using the web to carry out different tasks. For example: <H1> indicates a major heading. Semantically we know the words surrounded by <H1> tags are more important to the reader than the other text because of the meaning of H1. Some web pages add Meta tags to categorize the content of web pages. But still this option lacks to provide a more meaningful context since Meta tags are isolated with the keywords. The semantic web on the other hand, as its name suggests, provides the meaning of underlying data to the machines so that machines can perform more of the tedious work involved in finding, combining, and acting upon information on the web. The semantic web gives keywords useful meaning through establishment of relationships. It uses a common language for recording how data relates to real world objects. It involves publishing in languages specifically designed for data: Resource Description Framework (RDF), Web Ontology Language (OWL), and Extensible Markup Language (XML). HTML describes the documents and the links between them. RDF, OWL, and XML, by contrast, can describe arbitrary things such as people, meetings, airplane parts etc. [7]. These technologies are combined together in order to provide meaningful descriptions that supplement or replace the content of web documents. Thus, content may manifest itself as descriptive data stored in web-accessible databases, or as markup within documents combined with XML, or, more often, purely in XML, with layout or rendering cues stored separately. The machine-readable descriptions enable content managers to add meaning to the content. In this way, a machine can process data, thereby obtaining more meaningful results and helping computers to perform automated information gathering [7].

Although the semantic web promotes the current World Wide Web by converting unstructured and semi-structured documents into a structured "web of data"; it still faces a few challenges like vastness, vagueness, uncertainty, inconsistency etc. Proper reasoning of underlying logic and relationship is needed in order to overcome these issues.

A semantic web application consists of several discrete components. The main components consist of semantic web statements, a Uniform Resource Identifier (URI), semantic web languages, an ontology and instance of data. Some of the components are described in the next sections.

2.1.3. URI

A uniform resource identifier (URI) is a string of characters used to identify a unique name of a web resource across the entire internet. Thus each component of a statement--subject, predicate, and object-contains a URI to support its identity throughout the entire internet. This removes naming conflicts, ensures the duplicate item or not and also provides a path to additional information. The most common form of URI is the Web page address, which is a particular form or subset of URI called a Uniform Resource Locator (URL). A URI typically describes:

- The mechanism to access the resource

- The specific computer that the resource is contained in
- The unique name of the resource (a file name) on the computer

2.1.4. Semantic Web Ontology

Ontology is a set of representational primitives which is used to model domain knowledge. The representational primitives provide a set of concepts within a specific domain denoting types, properties and inter-relationships of those concepts. The definitions of the representational primitives include information about their meaning and constraints on their logically consistent application. In other words, ontology design helps to create explicit formal specifications in terms of the domain and relationships among them. The main reason of designing ontology is to provide a common structured vocabulary for researchers, to improve the sharing, reusability and to make explicit domain assumption of knowledge about facts that are perceptible by senses, with regard to world of interest.

Ontology can be compared to a database schema or an object-oriented class diagram. In the context of database systems, ontology can be viewed as a level of abstraction of data models, analogous to hierarchical and relational models, but intended for modeling knowledge about individuals, their attributes, and their relationships to other individuals [8]. Ontologies are typically specified in languages that allow abstraction away from data structures and implementation strategies; in practice, the languages of ontologies are closer in expressive power to first-order logic than languages used to model databases. For this reason, ontologies are said to be at the "semantic" level, whereas database schema are models of data at the "logical" or "physical" level [9].

The main components which are used to design the ontology are: classes that represent the collection of concepts, individuals which define instances of concepts, attributes like features, properties, characteristics that the classes have and the relation between the classes. Ontologies can be modeled with different modeling techniques and can be implemented with different kinds of languages [22]. The ontology can be expressed in highly informal, semi informal, semiformal and rigorously formal manner depending on the clarification of the Metadata of the domain. One language that is commonly used to design ontology is OWL (Web Ontology design). Web Ontology Language (OWL) is an ontology language for the web; it can process the content of information instead of just presenting the information. OWL can be used to explicitly represent the meaning of terms in vocabularies and the relationships between those terms [5].

The goal of ontology is to achieve a common and shared knowledge that can be transmitted between people and between the application systems [22]. Thus over the years, ontology has become a popular research topic in building specific knowledge based domain such as the Semantic Web technology, the Semantic Web Service Discovery (such as E-business), Artificial Intelligence, Multi Agent etc. in a range of disciplines and formalization .

2.1.5. Resource Description Framework (RDF)

Resource Description Framework is a data model. It is a universal framework, to describe the resource of the websites with a variety of syntax notations and data serialization formats. RDF is a domain-independent concept, through which users can define their own terminology on different domain concept. RDF has the capability to describe the author of the resource, date of creation, date of modification, keywords for search engines, categories of subject, organization of pages in the web site and so on, which is referred to as metadata or data about data. RDF is based upon the idea of making statements about resources in the form of subject-predicate-object expressions which is called triples in RDF terminology. The subject denotes the resource and the predicate represents the relationship between the subject and the object. For example, “The singer has an album named18 till I die”. According to the term triple “The singer” denotes the subject, “has an album named” denotes the predicate and “18 Till I die” denotes the object. The RDF description statements, enclosed as part of an Extensible Markup Language (XML) section, could be integrated within a Web page with today’s HTML (Hyper Text Markup Language) tag or could be in separate files.

Main benefits of using RDF are [23]:

- RDF provides metadata about Internet resources by providing a consistent framework
- As RDF includes a standard syntax for describing resources and underlying data, software that exploits metadata will be easier and faster to produce
- The standard syntax and query capability of RDF allows applications to exchange information more easily
- Searchers gets more precise results from searching, based on metadata rather than on indexes derived from full text gathering
- Intelligent software agents have more precise data to work with

2.1.6. Web Ontology Language (OWL)

OWL stands for The Web Ontology Language. OWL is a semantic web standard which provides a framework for the excellence management of web information and also to increase the sharing and reusability of the web information. It is designed for use by applications that need to process the content of information instead of just presenting information to humans. It sufficiently supports the well-defined syntax, the effective formal semantics, the efficient reasoning and the convenience of expression of the web information. Basically, OWL is an extension of the RDF Schema. But OWL facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDF-S) by providing additional vocabulary along with a formal semantics. It also describes the relationships among the web information. OWL is basically designed to strengthen the foundations of the semantic web, which is able to carry out richer integration and interoperability of data across the domain of different concerns [5]. OWL has three increasingly-expressive sublanguages: OWL Lite, OWL DL, and OWL Full. OWL Lite has limited facility and it is intended to support the small group of users with simple hierarchical organization and simple constraint. OWL DL was designed to provide maximum expressiveness and computational efficiency. And finally OWL lite uses OWL language primitives that is compatible with RDF and allows ontology to evolve the meaning of predefined vocabulary.

2.2. Related Work

There are various types of publish/subscribe systems have been developed over the years. Most research on publish/subscribe systems have been done in the area of either heterogeneous database integration or network centric integration or both. Researchers utilized various data models based on topics and contents. Topic-based systems are very limited to a group of people having interest in a similar topic. Content-based systems, on the other hand, are flexible enough to express the content of messages in various applications. These type of systems adopt various filtering algorithms on a distributed publish/subscribe architecture [2]

[3]. SIENA [11] is one of the most well-known examples of the publish/subscribe system to be developed in this area. SIENA uses a P2P model of interaction among servers (super-peers terminology) and adopts a language based on attribute-value pairs in order to express notifications, subscriptions and advertisement. SIENA adopted a conventional network based algorithm on shortest paths and minimum weight spanning trees for routing messages.

Another distributed content-based publish/subscribe system [2] was approached with the advent of distributed hash-tables. In this approach topics are automatically generated from the content of subscriptions and publications through the schema, which is a set of guidelines for selecting topics [2]. The schema is application-specific and can be provided by the application designer after some statistical analysis. The schemas that have been used are very much similar to the schemas of RDBMS (Relational Database Management System). With this approach, they have increased the expressiveness of subscriptions compared to the purely topic-based systems. But their schema does not fully provide the query semantics of a traditional content-based system. Moreover, issues with fault tolerance in subscription are not explored yet.

Another hash based publish subscriber model [12] adopts publish/subscribe communication paradigm to WMNs (Wireless Mesh Network). This system dynamically determines brokers that perform a store and forward function to route messages from publisher to subscriber. Notifications are delivered to subscribers once publish events/services match the subscriptions. This approach overcomes the fault tolerance and scalability issues by using a new routing metric for publish/subscribe communication. The main problem with the systems [2] [11] [12] is enabling database integration of heterogeneous information so that subscriber can access multiple information in a uniform and efficient manner. Since the publish/subscribe systems are very loosely coupled within space, time and synchronization, these systems need to provide a scalable and distributed infrastructure for information exchange. The publish/subscribe systems are also firmly coupled to the semantics of the underlying event schema and values of event subscription and pattern. One way of solving that problem is using semantic ontology based system and semantic-based matching between the publication and subscription. But it is very difficult to develop and maintain a semantic-based publish/subscribe system because of the high degree of semantic heterogeneity of events in large and open deployments. In order to address semantic coupling within the publish/subscribe systems, the use of approximate semantic matching of events is an active area of research [3]. Several researchers [13] [14] have made remarkable progress on this matter. The semantic Toronto publish/subscribe system [13] described three approaches to add more extensive semantic capability to the matching algorithm. In the first approach they talk about a matching algorithm to match events and subscriptions which are expressed in semantically equivalent attributes-synonyms. The second approach describes the relationships (more precisely specialization and generation) between attributes and values to allow additional matches. The third approach uses mapping functions which allow definition of arbitrary relationships between schemas and attribute values. Although their approaches are very innovative and intuitive, the paper lacks detailed architecture, precise content hierarchy, efficient and effective matching algorithm.

CHAPTER 3: DESIGN

3.1. Designing Publish/Subscribe System for Social Network

In this section, we have proposed our design for publish/subscribe system. First we would like to provide an overview.

3.1.1. Publish/Subscribe System Overview

The hierarchical structure of the proposed publish/subscribe system is depicted in Figure 1 and the overall system architecture is displayed in Figure 2. In order to build the publish/subscribe model, several components have been considered. The following discussion portrays how those components are organized and correlate to each other.

User: The User component is the top most component of the model which has two sub components: Subscriber and Publisher. The User component stores both publishers' and subscribers' information like name, address, state, zip, email etc.

Subscriber: The Subscriber component inherits attributes name, address, state, zip, email etc. from the parent component User. It has the property "has Interest" to make an association with the component Interest. Based on subscriber's various levels of interest publish/subscribe model notifies the subscriber of a recent publication that matches with any of his/her interest.

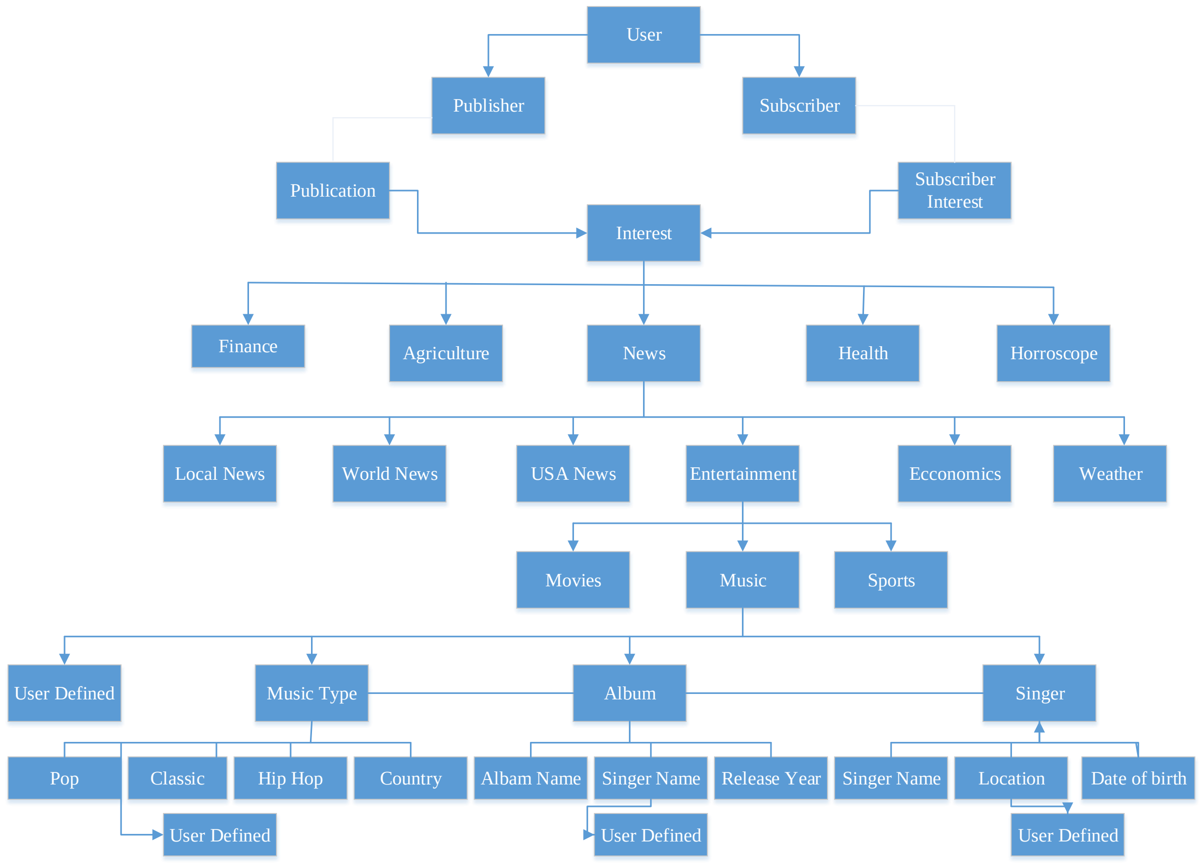


Figure 1. Hierarchical publish/subscribe system structure

Publisher: Like the Subscriber, the Publisher component inherits the attributes name, address, state, zip, email etc. from the parent component User. It has the property "publishes" to make an association with the Publication component. For example, whenever a Publisher publishes any type of news (for example, news on music) the Subscriber whose interest matches with this news, gets a notification.

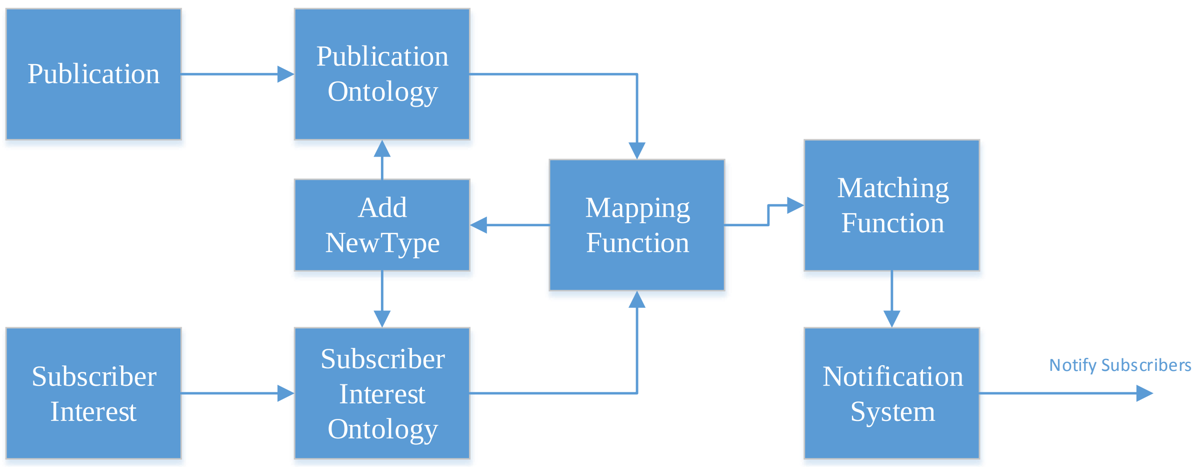


Figure 2. Publish/Subscribe system architecture

Publication/Interest: The Publication/Interest is the most complicated and important component(s) in Publish/Subscribe model. A Publisher publishes different types of publications on news, finance, agriculture, weather etc. or the subtype of these components. A Subscriber also has interests on these components or the sub-type of these components. Therefore, Publication/Interest component is associated with different components which can also be further subdivided into other components. One important factor of our system is that it is capable of adopting user's new interest or publication types. This new type can be added at any place of the system hierarchy and the system updates the hierarchy to incorporate this new type of publication or interest. We will explain this aspect in more detail when we will discuss about news on music component. This news on music component is the target example that we use for explaining our concepts.

News: The News component can be subdivided into local, weather, foreign, USA, entertainment, finance, economics etc.

Entertainment: The Entertainment component is sub-divided into components music, videos, sports etc.

News on Music Components: The Music component has different sub-components: Music type, Album, Singer, etc. The Music type component is extended to Classical, Pop, Hip hop, Country etc. User (Publisher or Subscriber) might have interest in Album or Singer; therefore, these components are also added under the Music component. Music type, Album and Singer components are also related to each other. This relationship is shown in the figure 1 by adding a connecting line between them. One thing to note is that all these components have a component named “User defined”. These user defined components are added to portray the dynamic nature of this publish/subscribe system. A Publisher should be able to publish or a Subscriber should be able to subscribe on an interest that is not known to this system yet. When this new type component appears, the system adopts to incorporate this new component. For example, in figure 1, music type Jazz is not added as a component of music type. If a publisher publishes something on Jazz or a subscriber shows interest on Jazz, this new music component is going to be added to the system graph and the ontology dictionary gets updated.

3.1.2. Ontology Design

As we mentioned before, in this system, publisher publishes information and that information gets notified to the subscriber who has interest in that information. Therefore, determining a mechanism for matching publishers’ publication and subscribers’ interest is a crucial element of the publish/subscribe infrastructure. This matching algorithm depends on developing sematic based profile schemes for subscriber and publishers which are detailed enough for expressing interests and publishing information. These profile schemes should also be structured enough for the system to efficiently locate the relevant subscribers' interest or publishers' resources. Moreover, the design of profiles requires shared representations of knowledge as a basic dictionary from which profiles can be asserted. An ontology, "a shared and common understanding of a domain" [24], is precisely intended to convey this kind of shared understanding. Therefore, ontology is being used to represent publisher's publication and subscribers' interest profiles. There are a number of advantages for using ontology based systems. The ontology-based representation is more expressive and less confusing than a keyword-based representation [25]. The ontology provides formal, machine-executable meaning on the concepts, which supports inference mechanisms that can be used to enhance semantic matching capabilities. It also provides an adequate foundation for the representation of coarse or fine-grained user publication/interests and is also able to deal with the fragilities of user's new preferences.

In order to manage the openness and extensibility requirements, we adopt W3C recommendations: the Resource Description Framework (RDF) [4]. In this section we formalize the basic concepts of subscribers' interest and publishers' publication using RDF XML serialization. Figure 3 shows examples of Subscriber's and Publisher's profile schemes. These schemes (Figure 3) stem from the discussion of the detailed ontology of the publish/subscribe system for the music news portrayed in Figure 1.

Figure 3 defines common understanding for all the concepts and their relationship for publishing news on music and subscribing interest on music news. Here we adopted top-down design. On the top we have generalized concepts/components like users, publishers, subscriber, music, album and singer. If we go down the graph we see specialized components like pop, music, etc. The figure also shows the relationship between components. For example, User is a Publisher, Publisher publishes a publication.

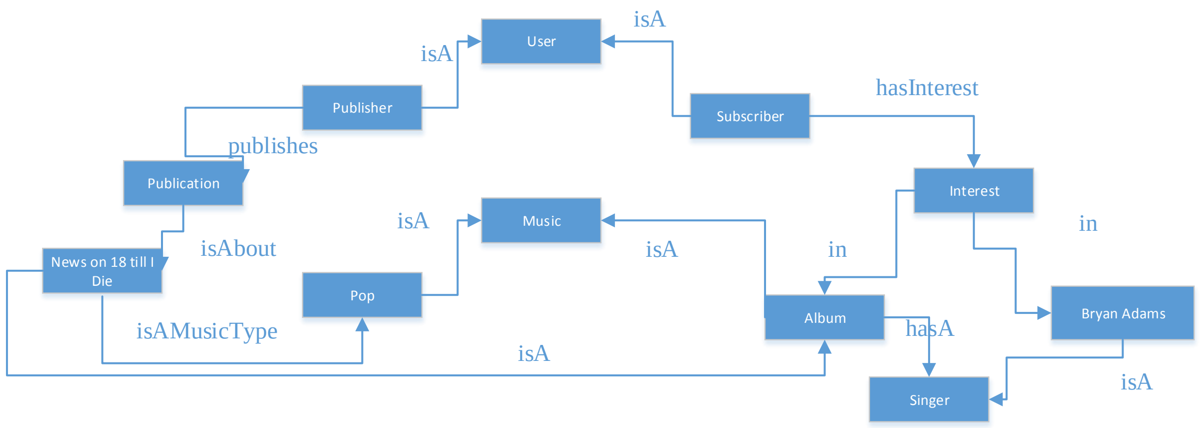
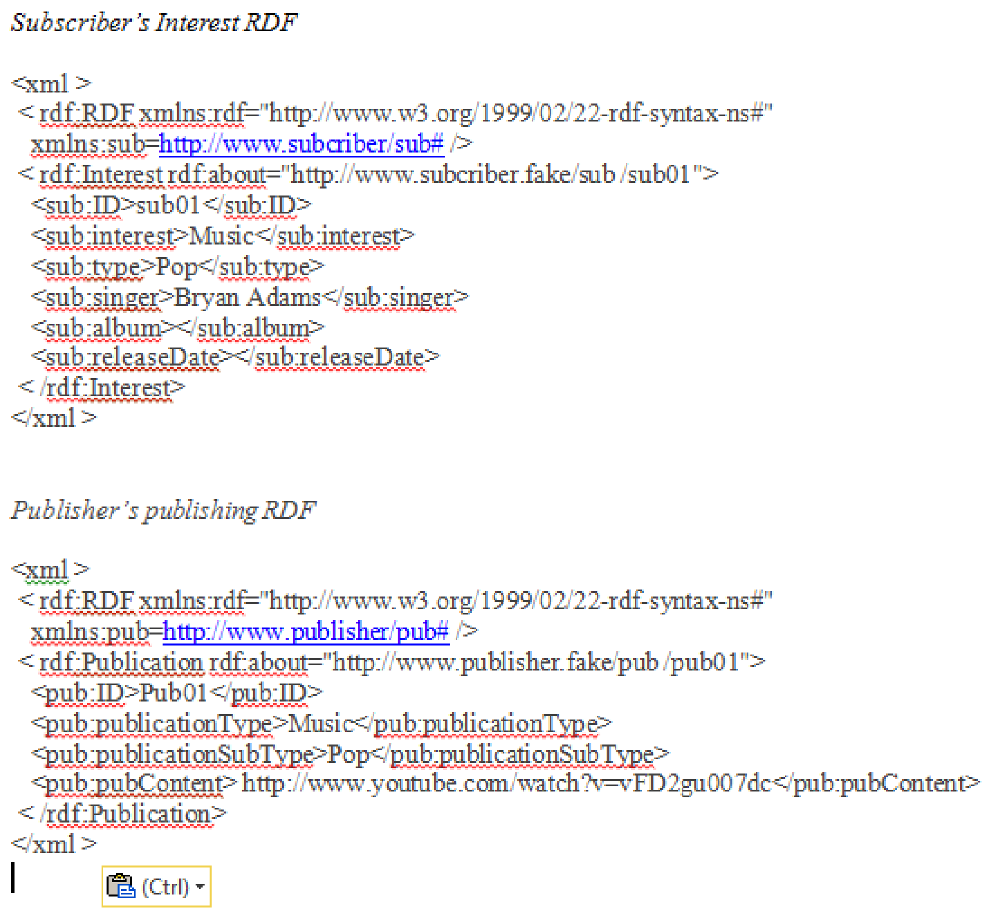


Figure 3. Relation ontology of publish/subscribe system news on music

To represent the interest and publication profile schemes we use a form of <subject, predicate and object> expression called triples in RDF terminology. The subject represents a resource which has a URI; the predicate represents the facets of the resource and expresses a relationship between subject and object. The object is the actual value. Figure 4 portrays these representations.



Subscriber's Interest RDF

```
<xml >
 < rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:sub=http://www.subcriber/sub# />
 < rdf:Interest rdf:about="http://www.subcriber.fake/sub /sub01 ">
   <sub:ID>sub01</sub:ID>
   <sub:interest>Music</sub:interest>
   <sub:type>Pop</sub:type>
   <sub:singer>Bryan Adams</sub:singer>
   <sub:album></sub:album>
   <sub:releaseDate></sub:releaseDate>
 < /rdf:Interest>
</xml >
```

Publisher's publishing RDF

```
<xml >
 < rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
   xmlns:pub=http://www.publisher/pub# />
 < rdf:Publication rdf:about="http://www.publisher.fake/pub /pub01">
   <pub:ID>Pub01</pub:ID>
   <pub:publicationType>Music</pub:publicationType>
   <pub:publicationSubType>Pop</pub:publicationSubType>
   <pub:pubContent> http://www.youtube.com/watch?v=vFD2gu007dc</pub:pubContent>
 < /rdf:Publication>
</xml >
```

(Ctrl)

Figure 4. RDF notation of an instance of publish/subscribe system

3.2. Matching Algorithm

In order to match between the subscribers' interest and the publishers' publication we have adopted the mechanism used by Juan Li et al [18] by calculating the similarity between the subscribers' interest and publishers' publication. Consider a case where a subscriber has interest in album "Stars Dance" which is a pop type of music; another subscriber has interest in any pop music; another subscriber has interest in all type of music. Now a publisher published a link of a pop music. Our publish/subscribe system will notify this link to the two subscribers; one with interest in pop music and another with interest in all music.

This subscriber interest RDF (mentioned in Figure 4) is basically a vector which represents the interest (music, music type, singer, album, release date etc.) of that subscriber:

$$S_u = (I_1, I_2, \ldots\ldots.. I_n) \tag{1}$$

This publisher's publishing RDF (mentioned in Figure 4) is basically a vector which represents the publication (publication type, publication subtype etc.) of that publisher:

$$P_u = (N_1, N_2, \ldots\ldots.. N_n) \tag{2}$$

Next, the semantic distance between two concepts has been calculated using the following definition [19]:

$$dis(I_a, N_b) = \frac{1}{2}\left(\frac{\sum_{i \in path(I_a, C_p)} w_i dis(I_i, I_{i+1})}{\sum_{i \in path(I_a, C_{root})} w_i dis(I_i, I_{i+1})} + \frac{\sum_{j \in path(Nb, C_p)} w_j dis(Nj, N_{j+1})}{\sum_{i \in path(N_b, C_{root})} w_i dis(Nj, N_{j+1})} \right) \tag{3}$$

Here, C_p is the nearest common parent of I_a and N_b. To calculate the edge weight we used the following equation introduced by Jike Ge et al. [21]:

$$w(I_a, N_b) = 1 + \frac{1}{k^{depth(N_b)}} \tag{4}$$

After calculating the semantic distance between two concepts I_a and N_b, we can calculate the similarity between them using the following equation [18]:

$$sim(I_a, N_b) = 1 - dis(I_a, N_b) \tag{5}$$

Now, the similarity between their profiles can be measured using the following equation [18]:

$$sim(S_u, P_u) = \frac{\sum_1^n \max_{j \in [1,m]} sim(Ia_i, Nb_j)}{n} \tag{6}$$

Here n is the number of concepts in profile S_u and m is the number of concepts in P_u. If sim (S_u, P_u) is larger than a user defined similarity threshold, then we find a match.

Now let's discuss the scenarios where subscribers' interest profile doesn't match with the publisher's publication. This may happen due to the following reasons:

1. No subscriber has particular interest for the publisher's publication
2. The publisher defined keyword doesn't fully match with the subscriber's keyword. For example, publisher mentioned publication type as word *song* but subscriber mentioned interest as word *music*

Issue number 2 is of particular interest to us. To solve this issue we propose to develop a Semantic Synonym Dictionary (SSD) where each node in the ontology will have an ordered set of similar words like the following:

$$NodeName \; => \{S_1, S_2, S_3\ldots\ldots\} \qquad (7)$$

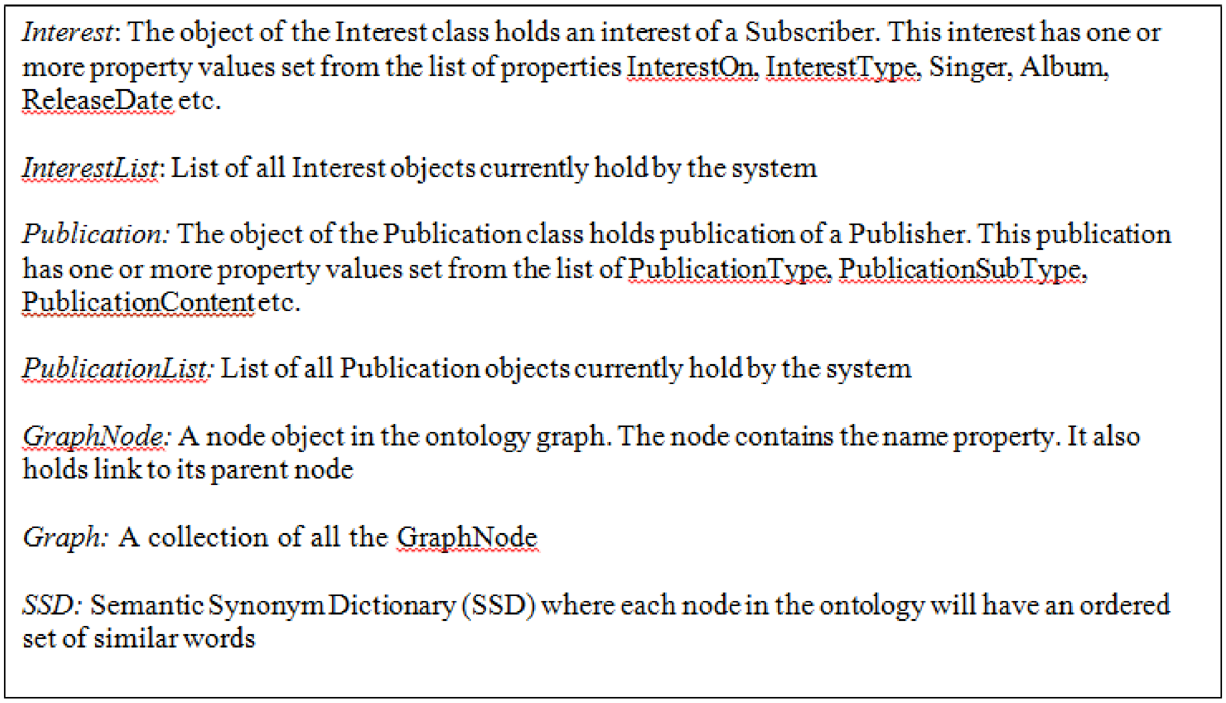
In the above mentioned equation S_i stands for the synonym *i* for the ontology node name. For the SSD, node name is used as the key. For example, for Ontology node Music, we might have an entry like *Music => {Fusion, Hymn, Melody, Tune……}*

One other scenario that might happen is publisher's publication or subscriber's interest is totally new for our ontology graph. One example might be if publisher publishes something on Jazz music or subscriber would like to subscribe to something on Jazz. In this scenario we would have to enrich our ontology graph for this new concept as soon as they are entered either by subscriber or publisher. This can easily be done by using any existing algorithm.

Now we will discuss briefly the abstract matching algorithm that can be used by the publish/subscribe systems. This algorithm is based on equations mentioned above. The algorithm has the following assumptions:

- The system adds Subscribers' interest to interest collection by calling procedure *AddInterest* mentioned below
- The system adds Publishers' publication to publication collection by calling procedure *AddPublications* mentioned below
- As soon as a publication gets added the matching algorithm triggers

The data structures used by this algorithm are mentioned in Figure 5 below.



Interest: The object of the Interest class holds an interest of a Subscriber. This interest has one or more property values set from the list of properties InterestOn, InterestType, Singer, Album, ReleaseDate etc.

InterestList: List of all Interest objects currently hold by the system

Publication: The object of the Publication class holds publication of a Publisher. This publication has one or more property values set from the list of PublicationType, PublicationSubType, PublicationContent etc.

PublicationList: List of all Publication objects currently hold by the system

GraphNode: A node object in the ontology graph. The node contains the name property. It also holds link to its parent node

Graph: A collection of all the GraphNode

SSD: Semantic Synonym Dictionary (SSD) where each node in the ontology will have an ordered set of similar words

Figure 5. Data structure used by the algorithm

```
1. for each Publication in the PublicationList do
    1.1. for each Interest in the InterestList do
        1.1.1. for each InterestProperty in InterestProperties
            1.1.1.1. Find the appropriate node in the GraphNode and assign it to _node1
            1.1.1.2. for each PublicationProperty in PublicationProperties
                1.1.1.2.1. Find the approriate node in the GraphNode and assign it to _node2
                1.1.1.2.2. Find common ancestor node _commonNode by calling procedure GetCommonAncestor(_node1, _node2)
                1.1.1.2.3. Calculate the distance between _node1 and _node2
                           Dist(_node1, _node2) =
                           ½[((GetDistance(commonNode, _node1)/(GetDistance(rootNode, _node1))) +
                           ½[((GetDistance(commonNode, _node2)/(GetDistance(rootNode, _node2)))]
                1.1.1.2.4. Calculate Similarity between _node1 and _node2 by
                           Sim(_node1, _node2) = 1- Dist(_node1, _node2)
                1.1.1.2.5. Add the Sim(_node1, _node1) to SimilarityCollection
            1.1.1.3. Find the Maximum Value from the SimilarityCollection and add it to MaxValue (MaxValue = MaxValue+ Max(SimilarityCollection)
            1.1.1.4. Clear SimilarityCollection
        1.1.2.  now *SimilarityValue between Interest and Publication =
                    MaxValue/Number of interestProperties*
        1.1.3.  If *SimilarityValue > threshold, Notify*
```

```
*Procedure AddInterest (Interest _interest)*
*{*
1. If (_interest Ɇ Graph)
    1.1. if ( FindKey (SSD, _interest.InterstType) != Empty)
        1.1.1.  Replace _interest.InterestType with FindKey (SSD, _interest.InterstType)
    1.2. else
        1.2.1. Update SSD either by adding a new Key or by adding value to a key
        1.2.2. Update Graph with adding the new _interest
2. Add _interest to the InterestList
}

*Procedure AddPublication (Publication _publication)*
{
1. If (_publication Ɇ Graph)
    1.1. if ( FindKey (SSD, _ publication.PublicationType) != Empty)
        1.1.1. Replace _ publication.PublicationType with FindKey (SSD, _
               publication.PublicationType)
    1.2. else
        1.2.1. Update SSD either by adding a new Key or by adding value to a key
        1.2.2. Update Graph with adding the new _publication
2. Add _ publication to the PublicationList
}
```

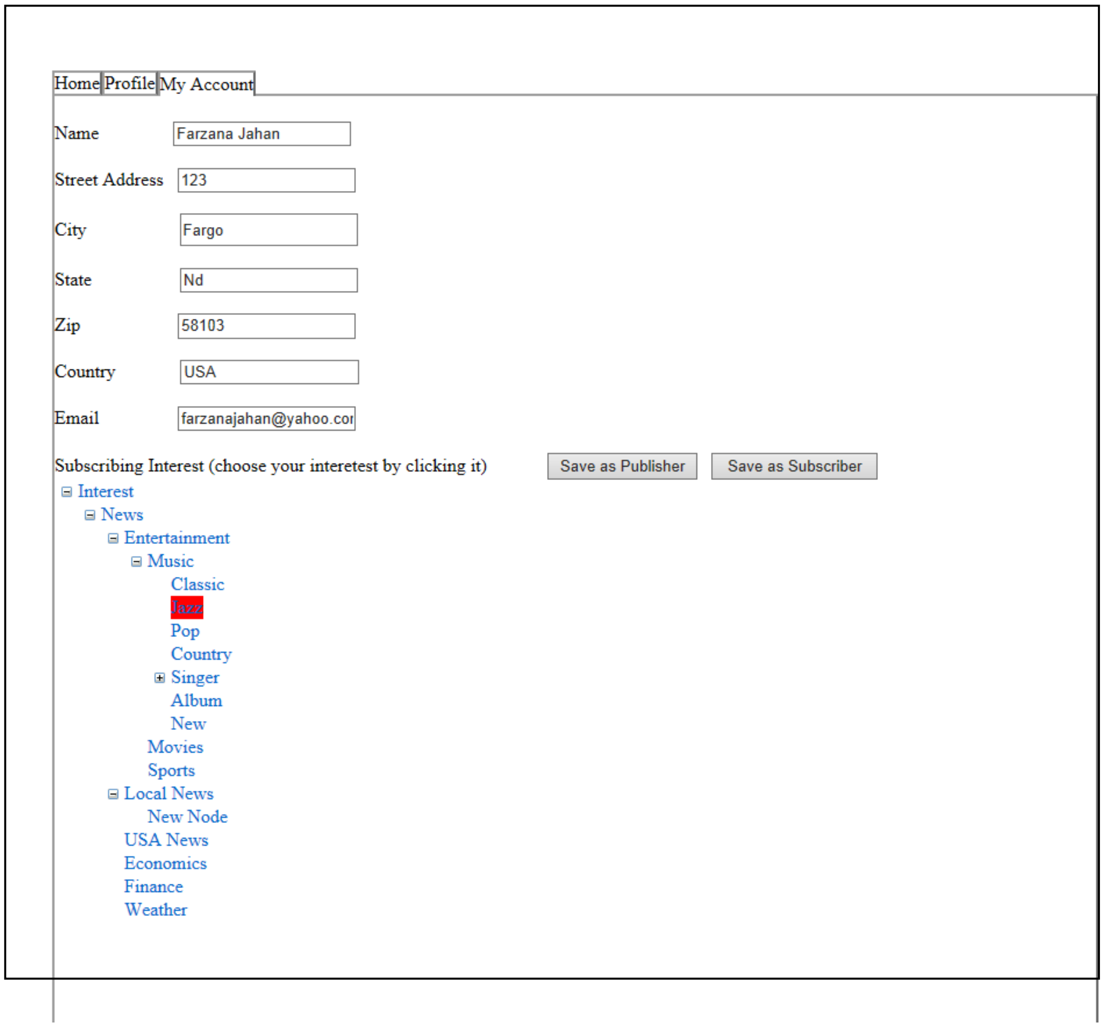
```
Procedure GraphNode  GetCommonAncestor (GraphNode  _node1, GraphNode  _node2)
{
    1.  Traverse from _node1 to the root node and add all these nodes to NodeCollection1
    2.  Traverse from _node2 to the root node and add all these nodes to NodeCollection2
    3.  for each item1 in the NodeCollection1 do
            3.1.    for each item2 in the NodeCollection2 do
                        3.1.1.  if the item2 == item1, then
                            3.1.1.1.  return Item2
    4.  return item1
}
```

```
Procedure decimal GetDistance (GraphNode  _fromNode, GraphNode  _toNode)
{
    1.  Traverse from _ fromNode to  _toNode and add all these nodes to NodeCollection
    2.  decimal distance = 0.0
    3.  For (int i = 0; i< NodeCollection.Lenght -1;  i++)
        3.1.  Calculate w using Equation 4
        3.2.  distance = distance + w * distance_between (NodeCollection[i+1],
              NodeCollection[i])
    4.  return distance
}
```

CHAPTER 4: IMPLEMENTATION AND ANALYSIS

4.1. Implementation

Our proposed publish/subscribe system can have the similar interface of subscription and publication mentioned in figure 6 and figure 7.



Name Farzana Jahan

Street Address 123

City Fargo

State Nd

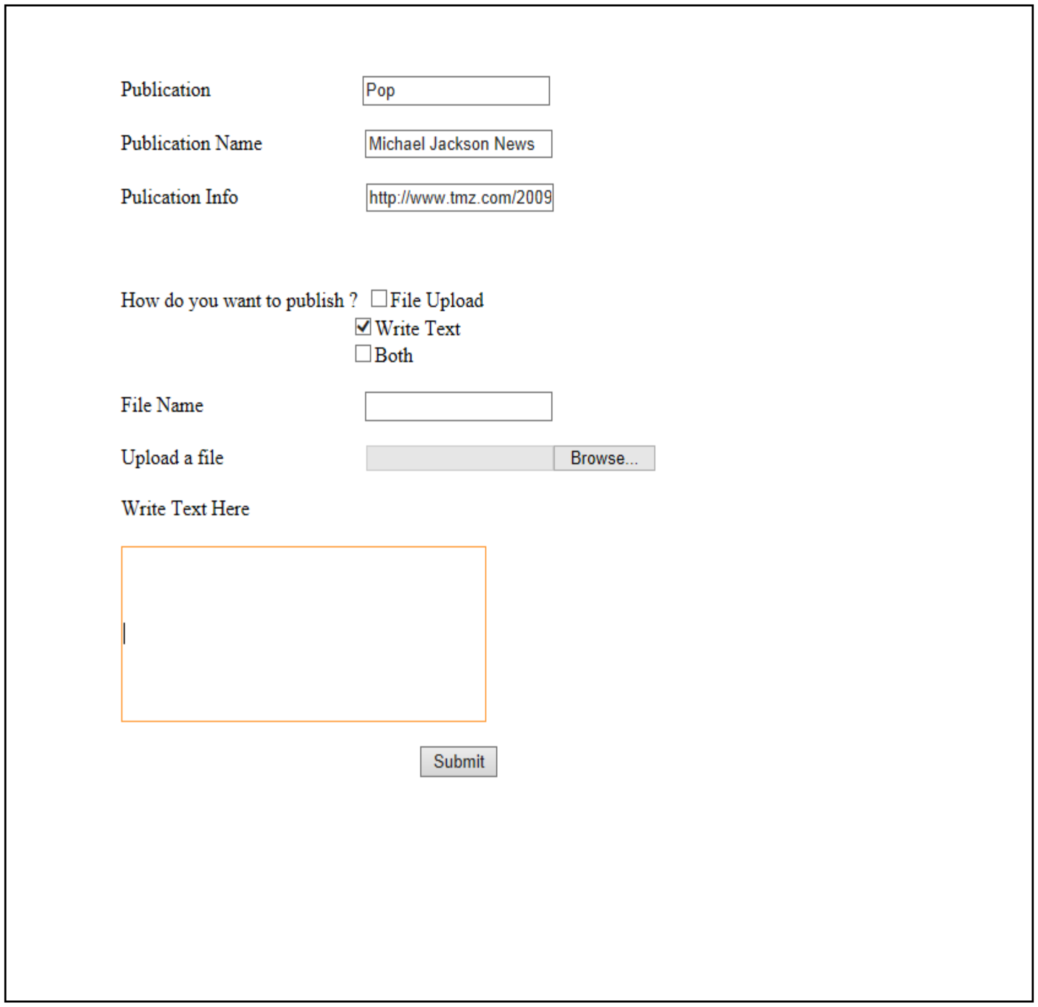
Zip 58103

Country USA

Email farzanajahan@yahoo.cor

Subscribing Interest (choose your interetest by clicking it)

Figure 6. Interface of a subscriber profile in a publish/subscribe system



Publication	Pop
Publication Name	Michael Jackson News
Pulication Info	http://www.tmz.com/2009

How do you want to publish ? ☐ File Upload [✓ Write Text] ☐ Both

File Name	
Upload a file	

Write Text Here

Figure 7. Interface of a publication in a publish/subscribe system

4.2. Experimental Study

We have conducted a set of scenarios of publications and subscriptions in order to evaluate our publish/subscribe system. Due to privacy issues we could not gather real time data; instead, we populated some random data of publications and subscriptions in our test database.

4.2.1. Case Study 1

In our publish/subscribe system, we have users Lorie, Janet, Kristi, Rebecca, Arnie, John and Jeff. Lorie is interested in news on pop music. She placed a subscription request for pop music; meaning that she would like to get notified on any news on pop music. Janet is interested in any news on music and she placed a subscription for music. Kristi is a fan of Michael Jackson. She placed a subscription for news on Michael Jackson. Jeff is interested in local news. He placed a subscription request on local news. Therefore, the publish/subscribe system has the following total subscription request:

Table 1. Case 1 subscribers

Subscriber	**Interest**	**Type**
Lori	Music	Pop
Janet	Music	
Kristi	Michael Jackson	Singer
Jeff	News	Local News

It should be noted that as soon as these subscription requests are entered, the system looks into the ontology graph and maps the subscription request metadata with the ontology graph node.

Like Kristi, Arnie is a big fan of Michael Jackson. He heard the terrible news that Michael Jackson passed away. He looked on the internet and found a link on that news and published that link in the publish/subscribe system. Therefore, our publish/subscribe system has the following publication:

Table 2. Case 1 publisher

Publisher	**PublicationType**	**PublicationSubType**	**PublicationContent**
Arnie	Michael Jackson	Pop	Michael Jackson passed away (http://www.tmz.com/2009/06/25/michael-jackson-dies-death-dead-cardiac-arrest/)

As soon as a publication entered into the system, the matching algorithm kicked in. Like subscription request, the system makes sure that the metadata placed with the publication matches with the ontology graph node and maps those nodes with the publication metadata. After that the system matches the publication with the subscription requests. In this scenario, this publication matches with three subscription requests: Lori’s pop music request, as Michael Jackson was a pop musician, Janet’s music request as Michael Jackson was a musician and Kristi’s request, as she wanted to know about Michael Jackson. Based on this matching, Lori, Janet and Kristi receive notifications about this link.

4.2.2. Case Study 2

In our publish/subscribe system, we have users Lorie, Janet, Jenney, Kristi, Rebecca, Arnie, John, Jack and Jill. Lorie is interested in news on singer Pink. She placed a subscription request for news on pop singer Pink; meaning that she would like to get notified on any news on Pink. John is interested in Fargo news. He placed a subscription request for news on Fargo. Janet is big concert goer. She is interested in any news on local concert events and she placed a subscription for concerts in Fargo. Jenney is interested in any entertainment news and she placed a subscription for Entertainment related news. Kristi is interested to know about events in Fargo. She placed a subscription for events in Fargo. Jack likes to know the current news of North Dakota and he placed a subscription request for news on North Dakota. Therefore, the publish/subscribe system has the following total subscription request:

Table 3. Case 2 subscribers

Subscriber	**Interest**	**Type**	**Singer**	**Album**	**ReleaseDate**
Lori	Music	Pop	Pink		
John	News	Fargo			
Janet	Concert	Fargo			
Jenney	News	Entertainment			
Kristi	Events	Fargo			
Jack	News	North Dakota			

It should be noted that as soon as these subscription requests are entered, the system looks into the ontology graph and maps the subscription request metadata with the ontology graph node. If there is new node, the system enriches the ontology graph by including that new node. For example, in the subscription request mentioned above, type is mentioned as Concert. But in the ontology graph there is no node named Event under music. The system adds this new node in the ontology graph as a child of music node. It also adds a new key-value pair in the Semantic Synonym dictionary with key as the Event and the entire synonym for it as the value set.

Arnie bought ticket to go to the Pink concert. He heard the news on the radio that the Pink concert has been rescheduled. He looked on the internet and found a link on that news and published that link in the publish/subscribe system. Therefore, our publish/subscribe system has the following publication:

Table 4. Case 2 publisher

Publisher	**PublicationType**	**PublicationSubType**	**PublicationContent**
Arnie	Concert	Local	Pink Fargo dome concert has been rescheduled (http://www.valleynewslive.com/story/23768834/pink-concert-cancelled-at-fargodome)

As soon as a publication entered into the system, the matching algorithm kicked in. Like subscription request, the system makes sure that the metadata placed with the publication matches with the ontology graph node and maps those nodes with the publication metadata. After that the system matches the publication with the subscription requests. In this scenario, this publication matches with all subscription requests. Based on this matching, Lori, John, Janet and Jenney, Kristi and Jack receive notifications about this link.

4.2.3. Case Study 3

In our publish/subscribe system, we have users Lorie, Janet, Jenney, Kristi, Rebecca, Arnie, John, Jack and Jill. Lorie is interested in news on movies. She placed a subscription request for news on movies; meaning that she would like to get notified on any news on movies. John is interested in cinema news. He placed a subscription request for news on cinema. Janet is a fan of Jennifer Lopez. She is interested in any news on Jennifer Lopez and she placed a subscription for Jennifer Lopez. Jenney is interested in pop music and she placed a subscription for news in pop music. Kristi is interested in romantic movies. She placed a subscription for romantic movies. Jack is interested to know about the release of upcoming movies and he placed a subscription request for news on upcoming movies. Therefore, the publish/subscribe system has the following total subscription request:

Table 5. Case 3 subscribers

Subscriber	Interest	Type	Actor	Moviename	ReleaseDate
Lori	Movie				
John	Cinema				
Janet	Movie star		Jennifer Lopez		
Jenney	Music	Pop			
Kristi	Movie	Romantic			
Jack	News	Upcoming movie			

It should be noted that as soon as these subscription requests are entered, the system looks into the ontology graph and maps the subscription request the metadata with the ontology graph node. For example, John placed a request for cinema. The system maps the word to movie by using the Semantic Synonym Dictionary. If there is a new node, the system enriches the ontology graph by including that new node. For example, in the subscription request mentioned above, type is mentioned as romantic. But in the ontology graph there is no node named romantic under movie. The system adds this new node in the ontology graph as a child of the movie node. It also adds a new key-value pair in the Semantic Synonym dictionary with key as the Event and the entire synonym for it as the value set.

Arnie heard the news that the movie Maid in Manhattan is released. He looked on the internet and found a link on that news and published that link in the publish/subscribe system. Therefore, our publish/subscribe system has the following publication:

Table 6. Case 3 publisher

Publisher	**PublicationType**	**PublicationSubType**	**PublicationContent**
Arnie	Movie	Romantic	Maid in Manhattan (http://en.wikipedia.org/wiki/Maid_in_Manhattan)

As soon as a publication entered into the system, the matching algorithm kicked in. Like subscription request, the system makes sure that the metadata placed with the publication matches with the ontology graph node and maps those nodes with the publication metadata. After that the system matches the publication with the subscription requests. In this scenario, this publication matches with all subscription requests. Based on this matching, Lori, John, Janet and Jenney, Kristi and Jack receive notifications about this link.

CHAPTER 5: CONCLUSION

5.1. Summary

Since most of the existing social networks are centralized and the common content based publisher/subscribe systems primarily rely on peer to peer communication and heterogeneous database integration, it is difficult to integrate and relay heterogeneous information between millions of publications and subscriptions. Semantic-based publish/subscribe systems can overcome this difficulty. But the establishment of this type of systems in social networks needs extensive matching between the publications and subscriptions in a distributed manner. In this paper we proposed and designed an ontology based publish/subscribe system and described the detailed interest profile hierarchy using Resource Definition Framework (RDF) which provides a structured way of organizing the publishers' publications and subscribers' interests. We also provided an efficient and extendable matching algorithm in order to calculate the semantic similarity between the publications and subscribers' interests. Our system is extendable to adapt to new users' interest by updating the interest profile at any point of hierarchy.

5.2. Limitations and Future Work

There are few limitations in our proposed publish/subscribe system. Scalability has become an issue due to the extensive matching and delivering inherent in semantics-based events in social networks. In order to prove scalability, we needed to get real time users data from different social networks and implement another existing algorithm and compare with our system. Due to privacy issues we could not test our matching algorithm with real time data. Another limitation is that, we are not matching and notifying any users (friend of subscribers) who are not in the subscription list even if they have similar taste of publication interest. We could think of integrating FOAF (friend of a friend) concept with our proposed semantic-based publish/subscribe system as future work. Another interesting future integration could be combining semantic-based publish/subscribe systems with mobile social networks. Since these days a large number of people are using more mobile devices like tablets, smart phones etc., building sematic based publish/subscribe system in mobile social network is a promising area of research.

REFERENCES

[1] Bernard Wong and Saikat Guha, “Quasar: A Probabilistic Publish-Subscribe System for Social Networks”.

[2] D. Tam, R.Azmi and H. Jaconsen “Building Content-Based Publish/Subscribe Systems with Distributed Hash Tables, *H.-A., International Workshop On Databases, Information Systems and Peer-to-Peer Computin*g, 2003.

[3] Wikipedia, http://en.wikipedia.org/wiki/Publish%E2%80%93subscribe_pattern/.

[4] Resource Definition Framework, http://www.w3.org/RDF/.

[5] Web Ontology Language, http://www.w3.org/TR/owl-features/.

[6] Yufei Li, Yuan Wang, and Xiaotao Huang, “A Relation-Based Search Engine in Semantic Web”, *IEEE* February 2007.

[7] Wikipedia, http://en.wikipedia.org/wiki/Semantic_Web/.

[8] Ling Liu and M. Tamer Özsu (Eds.),”Ontology to appear in the Encyclopedia of Database Systems”, *Springer-Verlag*, 2009.

[9] Boanerges Aleman-Meza, Phillip Burns, Matthew Eavenson, Devan, and Palaniswami, Amit Sheth, “An Ontological Approach to the Document Access Problem of Insider Threat”, *IEEE Intl. Conference on Intelligence and Security Informatics (ISI-2005)*, May 19–20, 2005.

[10] Halaschek, C., Aleman-Meza, B., Arpinar, I. B., and Sheth, “Discovering and Ranking Semantic Associations over a Large RDF Metabase”, *A. P. 30th International Conference on Very Large Data Bases*, Toronto, Canada, 2004.

[11] Antonio Carzaniga, David S. Rosenblum, Irvine and Alexander L. Wolf, “Design and Evaluation of a Wide-Area Event Notification Service”.

[12] Juan Li, Wei Zhang and Ziojun Xia,“Scalable Publish/Subscribe Service in Wireless Mesh Networks”, *IEEE Globeco*, 2010.

[13] Milenko Ptrovic, Iona Burcea, Hans-Arno Jacobsen, “S-ToPSS: Semantic Toronto Publish/Subscribe System”, November 2003.

[14] QT.Laliwala, Dhirubhai Ambani, Gandhinagar, Sorathia, V., and Chaudhary, S, “Semantics based Event-driven Publish/Subscribe Service-Oriented Architecture”.

[15] Chris Halaschek, Boanerges Aleman-Meza, I. Budak Arpinar, and Amit P. Sheth “Discovering and Ranking Semantic Associations over a Large RDF Metabase”, *30th International Conference on Very Large Data Bases*, Toronto, Canada, 2004.

[16] Yufei Li, Yuan Wang, and Xiaotao Huang, “A Relation-Based Search Engine in Semantic Web”, *IEEE Transactions on Knowledge and Data Engineering*, VOL. 19, NO. 2, February 2007.

[17] Juan Li, Hui Wang, and Samee Ullah Khan, “A Semantic-based Approach to Large-Scale Mobile Social Networking”, 2012.

[18] Juan Li and Samee Ullah Khan, “MobiSN: Semantic-Based Mobile Ad Hoc Social Network Framework”, 2009.

[19] Yufei Li, Yuan Wang, and Xiaotao Huang, “A Relation-Based Search Engine in Semantic Web”, *IEEE Transactions on Knowledge and Data Engineering*, VOL. 19, NO. 2, February 2007.

[20] Kemafor Anyanwu et al. “.ρ-Queries: Enabling Querying for semantic Associations on the Semantic Web”, 2003.

[21] Jike Ge and Yuhui Qui “Concept similarity matching based on semantic distance. In Semantics, Knowledge and Grid”, *IEEE*, 2008.

[22] Mohammad Mustafa Taye, “Understanding Semantic Web and Ontologies: Theory and Applications”, *Department of Software Engineering Faculty of Information Technology Philadelphia University, Amman, Jordan, Journal of Computing*, Volume 2, Issue 6, June 2010.

[23] RDF, http://searchsoa.techtarget.com/definition/Resource-Description-Framework/.

[24] London Gruver and WA Boudreaux, “Intelligent manufacturing: programming environments for CIM”, *Springer-Verlag*, 1993.

[25] Li JVuong S, “Grid resource discovery based on P2P communities”, *In Proc. Of the 21st ACM SAC*, 2006.