Analyzing the Scalability of a Social Network of Agents

Mohammad Moshirpour[1], Shimaa M. El-Sherif [1], Reda Alhajj[2], Behrouz H. Far[1]

[1] *University of Calgary, Department of Electrical and Computer Engineering, 2500 University Dr NW, Calgary, AB, Canada, T2N 1N4*

[2] *University of Calgary, Department of Computer Science, 2500 University Dr NW, Calgary, AB, Canada, T2N 1N4*

{mmoshirp, smmelshe, alhajj, far}@ucalgary.ca

Abstract

Social networks are ever-growing systems by inheritance. The increase in the number nodes in these systems often brings forth the need to add additional functionalities. However due to the distributed nature of social networks, system growth can be a challenging task. Therefore scalability of the system is of vital importance in the design of social networks. This research attempts to establish a comprehensive framework for analysis and validation of requirements and design documents for software systems. In previous work, we applied this framework to analyze the requirements of a social network of agents; expressed using scenario-based specifications. Scenarios are appealing because of their expressive power and simplicity. Moreover due to the clear and concise notation of scenarios, they can be used to analyze the system requirements for general validity, lack of deadlock, and existence of emergent behavior. In this paper a methodology to analyze the scalability of social networks is presented. This methodology is devised to indicate whether or not the new requirements of the system are consistent with the current requirements in place. A larger prototype of a social network of MSA for semantic search is utilized to illustrate the developed methodology.

Keywords: Scalability of social networks; Software requirements; Scenario-based software engineering; Emergent behavior; Multi-agent systems

1. Introduction

Modifying the scale of software applications is in general a non-trivial endeavor especially in social networks due to their rapid growth. As a social network increase in size, the need for additional functionality becomes evident. Therefore these systems face a serious scalability challenge due to their lack of central control as well as their rapid growth which results in increased complexity of the system. To avoid introducing bugs into the system, it is highly beneficial to ensure the correctness and integrity of the system will be preserved after scaling. Research suggests that detection of failures and removal of faults during field use of a system is about 20 times more expensive than detection and removal in the requirement and design phase [1].

Due to the integrated nature of social networks, it can be challenging to gather comprehensive and correct requirements for these software systems. Scenario-based specification is an effective and efficient way to describe the behavior of a variety of software systems such as multi-agent systems and distributed systems. Scenarios enable engineers and designers to describe system's functionality using the partial interactions of the system elements. There are several advantages of using scenarios such as expressive power and simplicity. In [2] we demonstrated that scenario-based software engineering (SBSE) can be used to effectively represent the requirements of social networks.

There are two main ways of representing scenarios, namely, Sequence Diagrams (SD) developed by the object management group (OMG) [3] and Message Sequence Charts (MSC) which were developed by the International Telecommunications Union (ITU) [4]. In this research MSCs are used to represent scenarios.

In [2] a methodology to analyze the requirements of social networks, expressed using scenarios, to detect emergent behavior was introduced. Emergent behavior, also known as implied scenario is a specification of behavior that is in the synthesized model of the system but is not explicitly specified in the set of scenarios. [5-8]. This usually happens when several autonomous components need to handle a joint task as a group in a

shared environment where control is also distributed. Although emergent behavior is not always unwanted, it is extremely useful for system designers and engineers to be aware of its existence.

In this paper the methodology presented in [2] is extended to verify the correctness and integrity of the design of social networks after applying the changes in system requirements. To illustrate this methodology, it is applied to the extended case study of semantic search engine which is a social network of multi-agent systems. Furthermore a software verification tool which is developed based on the devised methodology is introduced in this paper.

The structure of this paper is as follows: in Section 2 some background on the scalability of social networks as well as the time management amongst the nodes of such systems is presented. Section 3 contains the case study of the social networks of multi-agent systems (MAS). The verification methodology is demonstrated in section 4. In section 5 the software verification tool is introduced and conclusions and future work are presented in section 6.

2. Background

Some background knowledge with regards to scalability of social networks as well as the tie management amongst nodes in social networks is provided in this section.

2.1. Scalability of Social Networks

A simple paradigm to avoid the scalability challenge in social networks is with a fully distributed architecture of the network. It is not always possible to achieve this paradigm due to resource scarcity. There is always a tradeoff between functionality and future scalability.

There is several recent works in the literature which aim at solving the problem of social networks scalability. In [9], the authors propose a system to scale up a centralized social networks design without undergoing a costly transition to a fully distributed system. They take advantages of the structural properties of social networks to propose their paradigm. They call it One-Hop Replication (OHR). OHR utilizes some of structural characteristics of social networks. For instance most of information is one-hop away, and the topology of the network of connections among nodes displays a strong community structure. This system is composed of two components. The first component is the controller which is responsible for assigning users to servers. The second component is the middleware which ensures replication consistency.

Social Partition And Replication (SPAR) in another paradigm which is implemented in [9]. SPAR is a middleware that leverages the social graph structure to achieve data locality and minimize replication at the same time. SPAR constricts all relevant data for a user on a server and guarantees that for all users' direct neighbors, data is co-located on the same server. In this paradigm, scalability is achieved by adding commodity servers with low memory and network I/O requirements.

On the other hand, [10] introduces a decentralized scalable social network (eXO) to solve the problem of centralization in order to face the scalability challenge. eXO offers a fully decentralized social network with the ability to efficiently index and search globally for top-k users and content based on metadata information. The architecture of eXO provides also content replication as in P2P networks. eXO is based on Distributed Hashed Table (DHT) and it adds methods on top of it for efficient indexing and search and retrieval of users and content in the social networks scenarios.

2.2. Managing the Ties between Nodes

The strength of a tie is affected by several factors. Granovetter [11] proposed four dimensions that may affect tie strength: the duration of the relationship; the intimacy between the two actors participating in the relationship; the intensity of their communication with each other; and the reciprocal services they provide to each other. In social networks of humans other factors such as socioeconomic status, educational level, political affiliation, race and gender are also considered to affect the strength of ties [12].

Structural factors, such as network topology and information about social circles, may affect the tie strength [13]. The work in [14] suggests quantitative measures (variables) for tie strength including intensity variable, days passed since the last communication and duration. Another variable that may affect the strength of the tie is the neighborhood overlap variable [15] which refers to the number of common friends the two actors have. The work in [16] introduced mutual confidence between the actors of social networks. In [17] we propose a new methodology to calculate the strength of ties between agents in a social network using Hidden Markov Models (HMM) [18].

We showed that tie strength depends on several factors: Closeness factor: by measuring how close two agents are to each other (i.e. the degree of similarity between the two ontologies used by the two agents participating in the relationship); Time-related factor: combines all time factors that affect the strength of the relationship (e.g. duration of the relationship, frequency of communication between the two agents, time passed since the last communication); Mutual confidence factor:

clarifying the nature of the relationship under measure, if it is a one-sided relationship or a mutual relationship. Then we built an HMM model to measure the strengths of ties between agents in a social network using those factors.

3. Case Study: Semantic Search Engine

A model for semantic search was presented in [2]. This model utilizes a spiral workflow to incorporate both search and concept learning in the semantic search process [19]. The spiral workflow and its suggested scenario are shown in Figure 1.

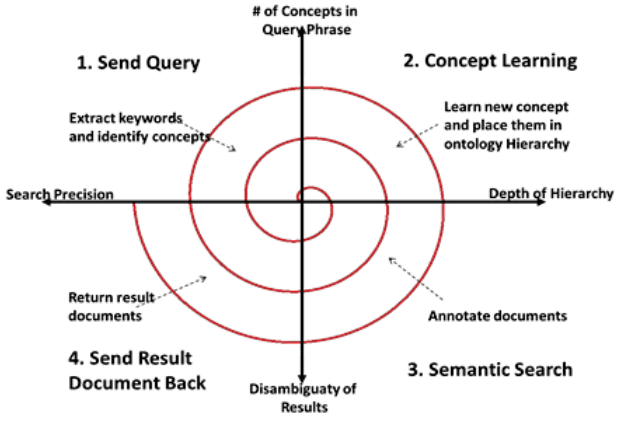

Figure 1. Spiral workflow between semantic search and concept learning

Semantic search depends on understanding the meaning of the concepts used in the context of other words. Thus it then tries to retrieve the related documents to these concepts. The foundation of semantic search is the semantic interoperability which is the main ingredient for notation extraction from the search phrase. Utilizing social networks in this system provides great flexibility; in particular when dealing with concepts in ontologies. It allows MAS to understand the meaning of the same concept even though its definition might be slightly different in each agent's ontology.

In our framework proposed in [2], we assume that in a society of n agents Ag_1, Ag_2… Ag_n, each agent Ag_i controls a repository R_i. Each repository uses an ontology that consists of a set of concepts and some documents to represent examples of these concepts. The architecture of our system is illustrated in Figure 2. Each concept C in our system possesses supporting examples. In addition, each agent has its own ontology (O_i) to represent the concept C.

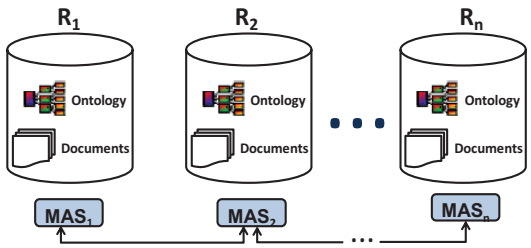

Figure 2. Semantic search system architecture

The key requirements for our proposed semantic search system are given below:

1. Software agents must be able to communicate with each other and be able to exchange information.
2. Agents must be responsible for organizing data in their own repository by annotating documents in their local repositories.
3. Agents must be able to reorganize their local repositories based on updates of concepts.
4. Agents must be able to cooperate with each other to learn and teach new concepts.
5. Agents should be able to hide the complexity of learning process and semantic search from the user.

Figure 3 shows different agent roles in each MAS in our prototype system [2]. In this system, agents of different MASs are the nodes in a social network. They connect with each other by a tie.

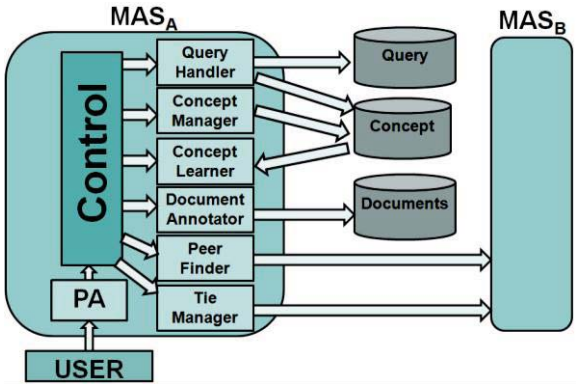

Figure 3. Agent roles within the semantic search engine

Here we need to apply the factors that affect tie strength in a normal social network as describe earlier in Section 2.2 of this paper. For the closeness factor, we can measure the closeness between the multi-agent systems by measuring similarity between their ontologies. The intensity variable can be represented in our system by the number of messages traded between two agents and how many concepts are learnt from each other which brings their ontologies closer to each other.

The dependency factor can indicate how much agents still depend on each other in learning new concepts or

searching for keywords. The neighbourhood overlap can be reflected as the overlap of neighbourhood circles (i.e. number of common neighbours) of two agents. In most social networks, the relationship between members is asymmetric. That means the strength of ties is not necessarily equal in both directions. The same consideration is valid in our system.

The original functionality for tie-management in this system is described using scenarios in MSCs 1 and 2, depicted in Figures 4 and 5 respectively [2]. In this case study, three arbitrary MAS of A, B and C are selected from the social network. It is assumed that ties already exist between A and C as well as between B and C; however there is no relations between A and B. The following scenarios expressed using MSCs, define the behavior of the network in identifying peers. It is important to note that these scenarios have been devised from the perspective of MAS_A.

For the initial requirements of this system, it was assumed that the strength of ties between each two nodes in the network (i.e. each MAS) depends on the following criteria:

a) Number of times they have been able to successfully cooperate
b) Number of peers they have in common

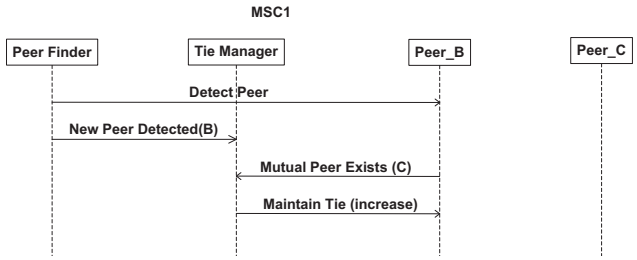

Figure 4. The tie between MAS A and B are established based on their mutual peers

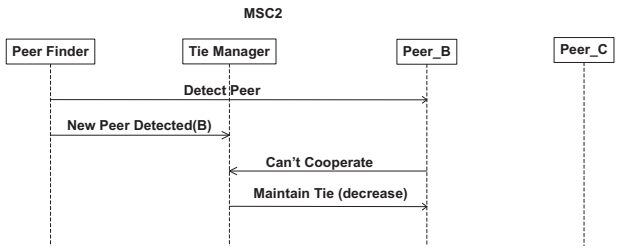

Figure 5. The relation between MAS A and B is decided based on their ability to cooperate on a given query

As the system was developed further, the need for more sophisticated tie-management criteria becomes evident. A number of existing different approaches were presented in Section 2.2 of this paper. This system was scaled up by increasing the functionalities of the tie-manager and adding criteria c:

c) Number of times nodes contact one another (Figure 6)

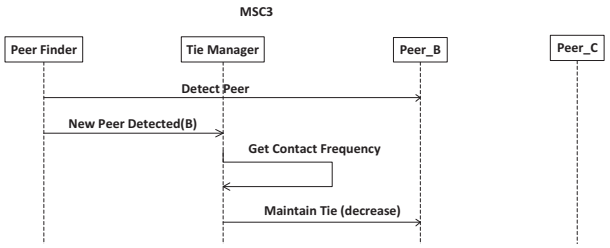

Figure 6. The tie between MAS A and B are established based on how frequently they contact each other

4. Verification Methodology

By scaling up the functionality of the social network of MAS, it is highly desirable to verify the system's behavior. The devised methodology to analyze software requirements is conducted in two steps of behavior modeling and detection of emergent behavior [20, 21].

4.1. Behavior Modeling

The model which describes the behavior of each system component (i.e. node for social networks) is usually called *behavioral model,* and the procedure of building the behavioral model from a scenario-based specification, is called *synthesis of behavioral models*, or simply, *behavior modeling* [20, 21]. State machines are commonly used to represent behavior models [22-27]. In the synthesis process, one state machine will be built for each node. The state machine includes all the interactions of a particular node based on the messages that it receives or sends. Theoretically, the behavior of the network can be described by the union (parallel execution) of all the state machines of the individual nodes. The behavior model for the initial tie-management mechanism, shown in MSCs 1 and 2 is depicted in Figure 7 [2].

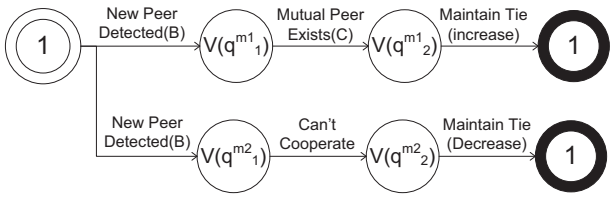

Figure 7. The union of state machines built from MSCs 1 and 2

To verify that the system's integrity is preserved after adding the new functionalities, the new behavior model for the system is constructed as shown in Figure 8.

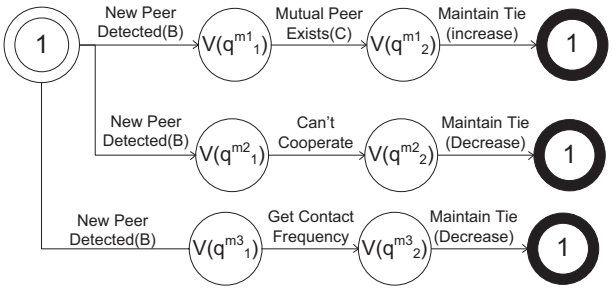

Figure 8. Behavior model of the system after scaling

4.2. Detection of Emergent Behavior

Implied scenarios or emergent behaviour appear when there is a state, in which the component becomes confused as to what course of action to take. This happens when identical states exist in the union of FSMs obtained through behavioural modelling. A definition for identical states is needed for detection of emergent behavior. To achieve this we must first have a clear procedure to assign values to the states of the eFSMs. This is a very important step and is performed differently in various works. For instance, [26] proposes the assignment of global variables to the states of eFSMs by the system engineer. However the outcome of this approach is not always consistent as the global variables chosen by different system engineers may vary. Therefore to achieve consistency in assigning state values, the approaches of [28, 29] which make use of an invariant property of the system called semantic causality is followed.

Definition 1 (Semantic causality): A message $m|_i[j]$ is a semantical cause for message $m|_i[k]$ and is denoted by $m|_i[j] \overset{se}{\rightarrow} m|_i[k]$, if agent *i* has to keep the result of the operation of $m|_i[j]$ in order to perform $m|_i[k]$.

By making use of semantic causality, the state values can be calculated as follows:

Definition 6 (State value): The state value $v_i|(q_k^m)$ for the state q_k^m in eFSM $A_i^m = (S^m, \Sigma^m, \delta^m, q_0^m, q_f^m)$ is a word over the alphabet Σ_i U {1} such that $v_i|(q_f^m) = m|_i[f-1]$, and for 0 < k < f is defined as follows:

i) $v_i|(q_k^m) = m|_i[k-1]v_i|(q_j^m)$, if there exist some j and l such that j is the maximum index that $m|_i[j-1] \overset{se}{\rightarrow} m|_i[l], 0 < j < k,\ k \leq l < f$

ii) $v_i|(q_k^m) = m|_i[k-1]$ if case i) does not hold but $m|_i[k-1] \overset{se}{\rightarrow} m|_i[l]$, for some k ≤ l < f

iii) $v_i|(q_k^m) = 1$, if none of the above cases hold

By calculating all state values, equivalent states are merged as shown in Figure 9 [2].

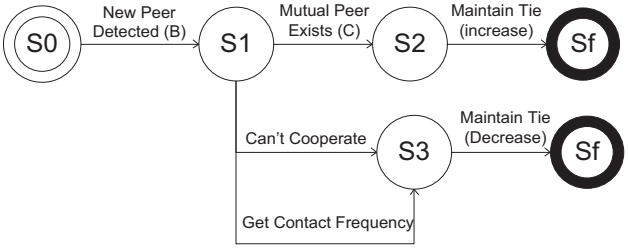

Figure 9. Resulted FSM after merging identical states

As it is shown in Figure 9, state S1 is where the tie manager of MAS_A falls into confusion. That is, as it is illustrated in MSC3 (Figure 6), TM will not be able to distinguish whether or not it should increase the tie with MAS_B or decrease it. Therefore as a result of the systematic approach in detecting emergent behavior in MAS, the system engineers is notified of such possible scenarios and is able to make modifications where necessary.

5. Software Analysis Tool

In order to enable the efficient and effective use of this methodology in real world projects, it needs to be automated in a software package. In this section, the analysis of system requirements using such a tool is presented.

As can be seen from the snapshot of the tool's graphical user interface shown in Figure 10, upon importing a design project from Microsoft Viso, the data boxes of the GUI are populated automatically with data. By clicking any of the imported MSCs, the components of that MSC will be shown in the *Component* subsection of the GUI and the actual MSC will be shown in the *Selected Diagram* area. Consequently, by selecting a component, the messages associated with that component will be shown in the *Message* subsection of the GUI. The synthesis of the behavior model, explained in Section 4 of this paper is conducted immediately upon importing related MSCs. The result of the built FSMs is shown in the *Constructed FSMs* subsection of the GUI. By clicking on the title of any of the FSMs the constructed figure will be shown in the Selected Diagram area as shown in Figure 11.

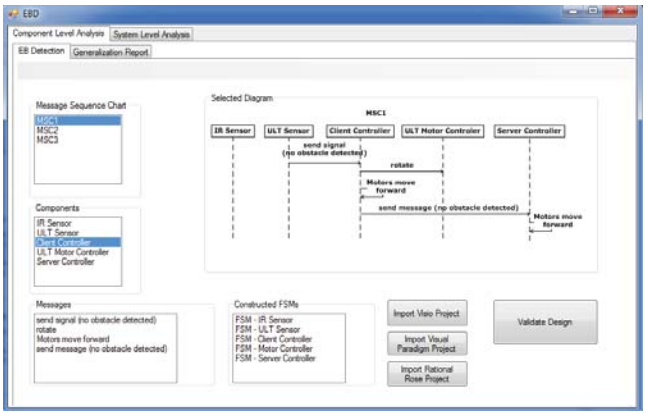

Figure 10. Screenshot of the tool's user interface; displaying an imported MSC

At this point, by clicking the Validate Design button, the methodology commences. Upon completion of the analysis, the user will be presented with a report outlining the areas in which unwanted behavior could occur.

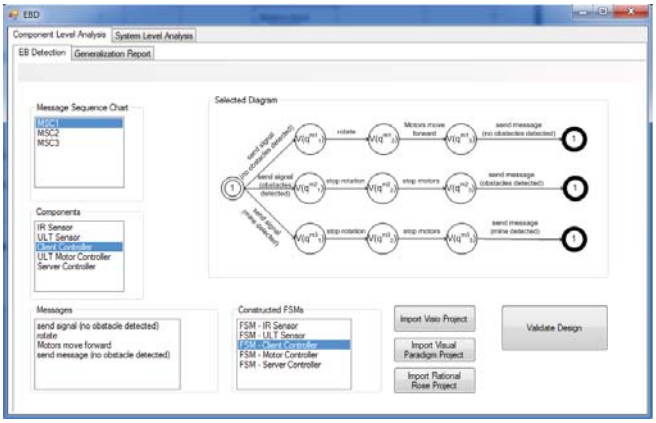

Figure 11. Screenshot of the tool's user interface; displaying a constructed FSM

6. Conclusions and Future Work

The constant growth of size and available functionalities in social networks makes scalability of the system to be of vital importance in the design of such systems. An effective and efficient approach to illustrate the requirements of software system is using scenario based specification. Scenarios are simple and expressive, and due to their specific notations can be used to analyze the requirements and design of software systems in an automated manner. Some of the failures in software systems can be directly attributed to their design. Research suggests that detection of failures and removal of faults during field use of a system is about 20 times more expensive than detection and removal in the requirement and design phase [1]. Unfortunately, manual review of the design documents may not efficiently detect all the design flaws due to the scale and complexity of the system. Therefore devising an automated and systematic methodology to analyze system requirements is greatly beneficial.

In the work presented in [2] a method to identify the exact cause of implied scenarios is provided, so that by capturing it, implied scenarios can be detected and removed. This method is novel in the sense of formalization of the cause of implied scenarios. This research indicates that this is the main reason for some shortcomings and conflicts in the current works, as they have been revealed in [29, 30]. In [2] we demonstrated our devised method to detect and remove design flaws that may lead to emergent behaviors in social networks.

In this paper the methodology presented in [2] is extended to verify the correctness and integrity of the design of social networks after applying the changes in system requirements. These techniques were illustrated using a prototype of a social network of multi-agent systems for semantic search. Due to the lack of central control in social networks, the requirement gathering and design of such systems can be difficult. Thus the presented methodologies can be used to systematically validate the requirements of social networks in terms of scalability. Furthermore this paper introduces a software tool which was developed based on the devised methodology.

In this research the requirements of social networks were analyzed with a component level perspective. For future work, the requirements of these systems can be analyzed with a system-level outlook. Furthermore since emergent behavior is not necessarily a negative quality of the system, the presented methodologies can be utilized to discover implied scenarios which do not cause problems for the system.

References

[1] D. R. Goldenson and D. L. Gibson, "Demonstrating the Impact and Benefits of CMMI: An Update and Preliminary Results," *CMU/SEI-2003-SR-009,* October 2003.

[2] M. Moshirpour, S. M. El-Sherif, R. Alhajj, and B. H. Far, "Detecting Emergent Behavior in a Social Network of Agents," in *The Influence of Technology on Social Network Analysis and Mining*, T. Özyer, J. Rokne, G. Wagner, and A. H. P. Reuser, Eds., ed: Springer, 2013, pp. 339 - 409.

[3] "Unified Modeling Language Specification. Version 2. Available from Rational Software Corporation," ed Cupertino, CA, 2006.

[4] "ITU: Message Sequence Charts. Recommendation, International Telecommunication Union. ," 1992.

[5] *Casual Closure for MSC Languages* 2005.

[6] R. Alur, K. Etessami, and M. Yannakakis, "Inference of Message Sequence Charts," *IEEE Transaction on Software Engineering,* pp. 623-633, July 2003.

[7] H. Muccini, "Detecting implied scenarios analyzing nonlocal branching choices," presented at the FASE 2003, Warsaw, Poland.

[8] S. Uchitel, J. Kramer, and J. Magee, "Negative scenarios for implied scenario elicitation," presented at the 10th ACM SIGSOFT International Symposium on the Foundations of Software Engineering (FSE 2002), Charleston.

[9] J. M. Pujol, G. Siganos, and V. Erramilli, "The little engine(s) that could: Scaling Online Social Networks without Pains," in *ACM SIGCOMM* New Dehli, India, 2010, pp. 375-386

[10] A. Loupasakis, N. Ntarmos, and P. Trianta_llou, "eXO: Decentralized Autonomous Scalable Social Networking," in *the 5th Biennial Conference on Innovative Data Systems Research (CIDR)*, Asilomar, California, USA, 2011, pp. 85 - 95.

[11] M. Granovetter, "The strength of weak ties: A network theory revisited," in *Sociological Theory*, ed, 1983.

[12] N. Lin, W. M. Ensel, and J. C. Vaughn, "Social resources and strength of ties: Structural factors in occupational status attainment," *American Sociological Review,* vol. 46, pp. 393 - 405, 1981.

[13] R. Burt, *Structural holes: The social structure of competition*: Harward University Press, 1995.

[14] E. Gilbert and K. Karahalios, "Predicting tie strength with social network," in *international conference on Human factors in computing systems*, Boston, MA, USA, 2009, pp. 211-220.

[15] J. P. Onnela, J. Saramaki, J. Hyvonen, G. Szabo, D. Lazer, K. Kaski, J. Kertesz, and A. L. Barabasi, "Structure and tie strengths in a mobile communication network," *National Academy of Science of the United States of America,* vol. 104, 2007.

[16] A. Petroczi, T. Nepusz, and F. Bazso, "Measuring tie-strength in virtual social networks," *the official journal of the International Network for Social Network Analysis* vol. 27, 2007.

[17] S. M. El-Sherif, B. Far, and A. Eberlein, "Calculating the strength of ties of a social network in a semantic search system using Hidden Markov Models," in *International Conference on Systems, Man and Cybernetics (SMC)*, Anchorage, Alaska, USA, 2011, pp. 2755 - 2760.

[18] O. Capp_e, E. Moulines, and T. Ryd_en, *Inference in Hidden Markov Models*: Springer, 2007.

[19] B. H. Far, C. Zhong, Z. Yang, and M. Afsharchi, "Realization of Semantic Search Using Concept Learning and Document Annotation Agents," in *Proceeding of Twenty-First International Conference on Software Engineering and Knowledge Engineering (SEKE)*, 2009, pp. 164-169.

[20] M. Moshirpour, "Model-Based Detection of Emergent Behavior In Distributed and Multi-Agent Systems from Component Level Perspective," Master of Science Department of Electrical and Computer Engineering, University of Calgary, Calgary, 2011.

[21] M. Moshirpour, A. Mousavi, and B. H. Far, "Detecting Emergent Behavior in Distributed Systems Using Scenario-Based Specifications," presented at the Proceedings of the International Conference on Software Engineering and Knowledge Engineering, San Francisco Bay, USA, 2010.

[22] D. Harel and H. Kugler, "Synthesizing state-based object systems from lsc specifications," *International Journal of Foundations of Computer Science,* 2002.

[23] I. Kruger, R. Grosu, P. Scholz, and M. Broy, "From mscs to statecharts," in *Franz j. rammig (ed.): Distributed and parallel embedded systems*, ed: Kluwer Academic Publis, 1999.

[24] E. Makinen and T. Systa, "MAS - an interactive synthesizer to support behavioral modeling in UML," presented at the ICSE 2001, Toronto, 2001.

[25] S. Uchitel, J. Kramer, and J. Magee, "Synthesis of behavioral models from scenarios," *IEEE Transaction on Software Engineering,* pp. 99-115, February 2003.

[26] J. Whittle and J. Schumann, "Generating statecharts designs from scenarios," presented at the ICSE, Limerick, Ireland, 2000.

[27] J. Whittle and J. Schumann, "Scenario-Based Engineering of Multi-Agent Systems," in *Agent Technology from a Formal Perspective*, Third ed London: Springer-Verlag, 2006.

[28] M. Moshirpour, A. Mousavi, and B. H. Far, "Detecting Emergent Behavior in Distributed Systems Using Scenario-Based Specifications," in *International Conference on Software Engineering and Knowledge Engineering*, San Francisco Bay, 2010.

[29] A. Mousavi, "Inference of Emergent Behaviours of Scenario-Based Specifications," PhD Thesis PhD Thesis, Department of Electrial and Computer Engineering, University of Calgary, 2009.

[30] A. Mousavi and B. Far, "Eliciting Scenarios from Scenarios," presented at the Proceedings of 20th International Conference on Software Engineering and Knowledge Engineering (SEKE 2008), San Francisco Bay, USA, 2008.