roar @UEL
research open access repository

University of East London Institutional Repository: http://roar.uel.ac.uk

This paper is made available online in accordance with publisher policies. Please scroll down to view the document itself. Please refer to the repository record for this item and our policy information available from the repository home page for further information.

To see the final version of this paper please visit the publisher's website.
Access to the published version may require a subscription.

Author(s): Nutter, David; Boldyreff, Cornelia.
Article title: Evaluation of an awareness distribution mechanism: a simulation approach
Year of publication: 2005
Citation: Nutter, D. & Boldyreff, C. (2005) 'Evaluation of an awareness distribution mechanism: a simulation approach' in: *14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 2005),* 13-15 June 2005, Pittsburgh, PA, USA. IEEE Computer Society 2005, pp. 235 - 240
Link to published version: http://dx.doi.org/10.1109/WETICE.2005.37
DOI: 10.1109/WETICE.2005.37

Evaluation of an awareness distribution mechanism: A simulation approach

David Nutter and Cornelia Boldyreff
University of Lincoln
{*dnutter,cboldyreff*}@*hemswell.lincoln.ac.uk*

Abstract

In distributed software engineering, the role of informal communication is frequently overlooked. Participants simply employ their own ad-hoc methods of informal communication. Consequently such communication is haphazard, irregular, and rarely recorded as part of the project documentation. Thus, a need for tool support to facilitate more systematic informal communication via awareness has been identified. The tool proposed is based on the provision of awareness support that recognises the complete context of the evolution of software artefacts rather than single events.

Peer-to-Peer (P2P) networking has been successfully employed to develop various distributed software engineering support tools. However, there are scalability problems inherent in naive P2P networks. To this end a semantic overlay network organisation algorithm has been developed and tested in simulation prior to deployment as part of a forthcoming awareness extension to the Eclipse environment.

The simulation verified that the self-organisation algorithm was suitable for arranging a P2P network, but several unexpected behaviours were observed. These included wandering nodes, starved nodes, and local maxima. Each of these problems required modification of the original algorithm design to solve or ameliorate them.

1 Background

Tool support to facilitate informal communication ungoverned by any documented process may improve the success of Distributed Software Engineering efforts. One such form of communication is that which results in awareness[7, 9, 5] of the activities of others in the team, whether in the form of synchronous, presence-type awareness, or asynchronous awareness such as change histories. Earlier work discusses a new type of asynchronous awareness: *historical awareness*[13], suggests a framework for implementing this awareness support within a software engineering development environment and describes its evaluation implications.

The framework in question proposed that all awareness producers and consumers belong to a peer-to-peer (P2P) network and exchange awareness information between nodes in the network. This avoids the problems associated with centralised CSCW systems discussed in [13]. To address the scalability problems inherent in naive peer-to-peer network designs, a semantic overlay network[8, 6] approach has been chosen, wherein a node's "nearest" neighbours would contain the information of greatest interest to that node, albeit one slightly different from commonly deployed semantic overlay systems like Herald[4] and PROST[16]. Theoretically in P2P networks all peers are equal, yet in deployment "some are more equal than others". For example, studies[2] have shown that in most file sharing networks a minority of the peers provide the majority of the files; the rest of the peers are "free riders". This finding has implications for the stability of the network. Many semantic overlay networks rely on this existing behaviour and introduce *super peers*[12, 17, 1] which take care of integrating new peers with the established peers (rendezvous) and making complex routing decisions.

While this functionality is desirable for file sharing networks, a P2P network designed for distributing awareness information has differing properties: information is pushed from each node, and nodes usually have similar levels of connectivity. Table 1 compares the properties of these two network types. Consequently, it has been decided to omit super peers from the overlay network requirements for reasons of simplicity.

Without super peers, a new method of organising the network is required. A local iterative connection optimisation algorithm has been loosely specified in earlier work and has subsequently been implemented albeit only in a form suitable for testing.

Initially, the algorithm design was as follows:

$$
\begin{aligned}
&Peer\ A \xrightarrow{connect} Peer\ B\\
&Peer\ A \xrightarrow{send\ details\ of\ A} Peer\ B\\
&Peer\ B \xrightarrow{send\ details\ of\ A} Peer\ C\\
&Peer\ B \xrightarrow{send\ details\ of\ A} Peer\ D
\end{aligned}
$$

Property	Awareness Net	File-sharing Network
Information retrieval	Pushed from each node	Pull (search results)
Node resources	Largely equal	Many consumer peers with few resources, a few super-peers with more
Node connectivity	Largely equal	Many weakly connected edge nodes, some super-peers
State management	Local	Partially global (discovery services, super peers etc)

Table 1. Comparative properties of Awareness and File sharing-type P2P networks

$$Peer\ B \xrightarrow{reconnect\ D} Peer\ A$$
$$Peer\ A \xleftarrow{disconnect} Peer\ B$$
$$Peer\ A \xrightarrow{connect} Peer\ D$$

Peer A connects to Peer B and receives various details of the nodes known about by Peer B. In return it would send its own details to B for further distribution. From the results received from B, A knows that Peer D is more interesting, so it breaks its connection to B and reconnects to D. This process repeats until the network enters a stable configuration. Aside from organisational traffic like this, each node will also transmit awareness messages to its immediate peers. Since all interested peers should in theory be close to the originating node by virtue of the self-organisation process, after the message has been retransmitted a number of hops (the Time To Live (TTL)) it may be discarded, reducing overall network load and retaining system scalability without requiring super-peers for routing.

Furthermore, a requirement was identified for a more lightweight form of evaluation[14] to be performed during the prototyping phase of research projects which attempts to provide some of the benefits of full, user-centric evaluation without the attendant costs in time, resources and user patience. This process was devised to support the evolution of an existing research prototype, OSCAR[15, 3] but is equally applicable to any project without sufficient resources or the desire to perform a "full strength" evaluation.

This paper discusses a simulation approach adopted to drive the subsequent evolution and evaluation of the self-organisation algorithm and to address some of the issues with using more conventional network organisation techniques.

2 Evaluation of this tool

The use of a P2P network and the design of this tool in particular raises several issues pertinent to evaluation. Firstly, a P2P network has no central node which may be instrumented to assist in data collection for evaluation. At any one time it is impossible to definitively know either the size of the network or even the full state of known peers, since they may well be interacting with additional peers beyond the network's boundaries. As knowledge of *global state* is lacking, the evaluator can never know these things except in the most contrived deployment circumstances. Obviously individual peers may be instrumented, but then the issue arises of establishing the true order of all events without global accurate time.

Secondly, debugging any semantic overlay system in a deployment environment is extremely difficult. The lack of global state knowledge described previously makes it difficult for the evaluator to determine the different effects of various factors (e.g. network reliability, user behaviour/experience etc) on the performance of the algorithm.

Thirdly any controlled experiments to validate the assumptions made during the design process of any semantic overlay are extremely difficult and would almost certainly require a specialised testing cluster to be built, wherein all relevant factors could be tightly monitored and adjusted. Though such a test system is desirable, insufficient resources are available to build it. Consequently a different approach must be taken to solving these problems.

Work conducted by Microsoft Research indicates in passing that simulation is useful throughout the development life-cycle of a P2P project. "Don't throw away your simulation environment" is the precise quote[11].

In the context of the awareness project, simulation will perform two main tasks. Firstly the controlled nature of the simulation environment will allow assumptions inherent in the design to be checked without lengthy setup of a complex physical test environment. Where such assumptions are found to be inaccurate, the simulation may be modified easily whereas changing a deployed test environment is likely to be much more difficult. While the simulation may not be perfect at identifying such issues, it still provides an ideal environment for debugging, even if the experience in the subsequent deployed environment does not initially match the expected results from simulation. Secondly, in the context of evaluation the simulation may be used as a sort of "hypothesis generator" to predict behaviour before users become involved. In this way problems with the design and performance of the algorithm may be identi-

fied and the structure of any future user involvement in the evaluation process developed before limited resources are committed.

2.1 Desired Results

The role of simulation in the awareness project is to answer several questions, detailed below:

1. What is the relevance of all the messages received by particular nodes?
2. What are the optimum settings for any simulation parameters, such as the message Time To Live? Are such apparently optimum settings desirable or likely in a deployed environment? Ideally, optimum settings should match the most common situations and if they do not, attempts should be made to improve the algorithm to make this possible.
3. What is the tolerance of the algorithm to unusual situations such as random link failure or delays in message transmission? A rather subjective interpretation may be required here.
4. How does network self-organisation compare to the standard naive P2P model? Later on, a comparison to a Super Peer architecture may also be interesting.

An initial set of requirements for the simulation tool was derived from this list of questions and the personal requirements of the tool's designer.

2.2 Simulation Tool Design

Firstly, some sort of visualisation was required. Since physical networks may be represented as an undirected graph which permits cycles, the JGraph[1] toolkit was chosen both to represent the network under simulation, display the intermediate and final results as a diagram and provide drag-and-drop editing capabilities for the operator to allow them to add and remove nodes at will.

Secondly a facility should be provided to start, stop, step-through and replay the simulation so interesting situations may be examined and if necessary adjusted to probe the properties of the self-organisation system. Additionally, a self timer should be provided so the simulation stops running after a preset number of steps.

In order to help generate such interesting situations, an optional events generator should be provided. The generator should be able to create and inject random events into the running simulation, including new peers joining the network, links failing, awareness information being transmitted from a particular node etc.

[1] http://www.jgraph.org

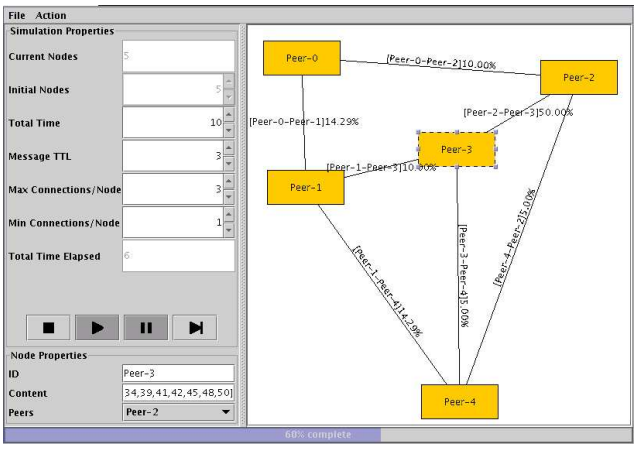

Figure 1. Simulation Tool Screen-shot

Thirdly, a facility to record all occurrences within the simulation, whether originated by the self-organisation actions or the event generator, is necessary to answer the question about message pertinence.

Fourthly, to ensure the design of the tool remains simple, each node in the simulation will be permitted to use a random discovery service to obtain an initial connection to the network. Thereafter, self-organisation or operator intervention must take place to make or break connections.

Fifthly, several interesting or particular taxing simulations should be saved and provided as "test cases". When evolving the algorithm, the comparative performance of these known cases may serve as a guide to see if any given modification is an improvement or a retrograde step.

Finally, the self-organisation algorithm should be pluggable to permit the interesting possibility of pitting several versions of the same algorithm against each other to see which one performs the best. Additionally control cases such as a Super Peer overlay architecture may be implemented as pluggable algorithms later in the research.

3 Results from Simulation

A tool (screen-shot in figure 1) was written conforming to the above design[2] and several interesting cases were examined to see how the initial design for a self-organisation algorithm behaved. During this testing, various unexpected algorithm behaviours were observed and documented; some of which were deemed inconvenient and consequently eliminated by modifications made to the algorithm.

The evaluation does not employ statistical techniques, for though the simulation could easily be used to gather the requisite information, statistical evaluation instruments can-

[2] At time of writing some advanced functionality such as replay is not available but the tool is usable despite this.

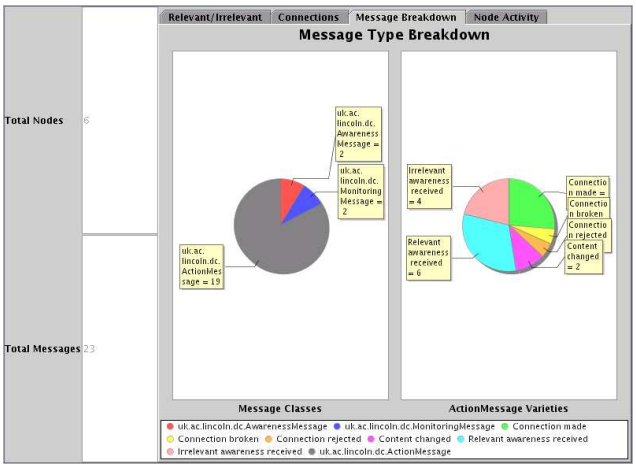

Figure 2. Live Graphing of Simulation Results

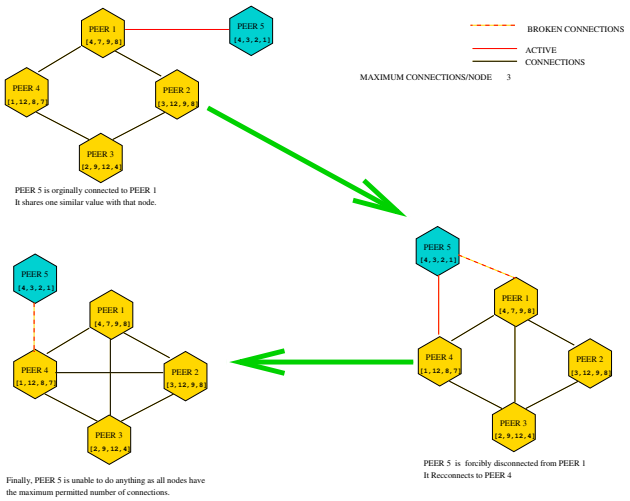

Figure 3. Wandering Nodes

not be used to evaluate the awareness tool. The issues identified earlier of missing global state knowledge and the difficulty of establishing total event order are to blame. Consequently a case study approach aimed at answering some of the questions posed earlier was used, and any other observations noted. At first, no reporting functionality was available in the tool but this made studying the simulation difficult. Consequently, some simple live graphing of results was implemented, shown in figure 2.

3.1 Control Case: No Self-Organisation

Initially, as an artefact of developing the simulation system the control case where no self-organisation at all was performed was examined. In this case nodes were allowed to randomly connect to each other, albeit within the bounds of the simulation parameters such as Maximum and Minimum connections-per-node. Each node contains some fake "data", that in the deployed version of the system will correspond to the output of metrics (e.g. fuzzy MD5) run on the resources the node possesses. By comparing this data with that of other nodes, a measure of similarity between nodes may be obtained.

When awareness messages are received by a node, they are deemed relevant if they relate to at least one resource the node owns. Predictably enough, the self-organisation method was shown to result in greater numbers of relevant messages being received by each node. With larger networks of nodes under the naive model, many did not receive any relevant information during the lifetime of the simulation. However, it does not follow from this result that the existing self-organisation algorithm is adequate simply because it is better than the obviously inadequate naive unaugmented P2P architecture, especially since the characteristics of the networks resulting from the no-organisation strategy and the organisation strategy were quite different. In particular, the latter strategy resulted in many more connections per node.

3.2 Link Failure

To determine what happened when links between nodes randomly failed, various simulations were set up and allowed to enter a reasonably stable state. Once stabilised, connections were broken at random at first by the operator and (once written) the event generator. In many cases the connection immediately reformed and network became stable once more. However, in certain cases a node that was completely disconnected from the network for whatever reason (see "Starved Nodes" below) would reconnect at this point and result in a large change in the network's organisation.

In the real world, the failure of underlying network infrastructure is likely to be the main cause of a dropped link. Such a failure may not allow a lost connection to reform immediately as observed in the simulation. At present the effects of this type of failure cannot be simulated by the tool.

3.3 Observation: "Wandering" Nodes

Whilst testing the effect of randomly breaking links in the network, it became apparent that certain simulation configurations resulted in one node "wandering" around the network and never entering a stable configuration. Figure 3 shows a sequence where this can occur for a time. Since the overhead of making a new connection is likely to be high in a production environment, eliminating this behaviour is

quite important. One possible solution is to maintain some state in each of the nodes to influence the outcome of the self organisation algorithm. At present each node is fairly dumb and maintains a record only of its currently connected peers. By maintaining a history of past peers, the algorithm will be able to make a more informed decision about whether to (re)make a particular connection, reducing this type of behaviour. In any case, maintenance of such a history is required in a production environment where no "random discovery service" exists to provide nodes with their initial connections[3].

3.4 Observation: "Starved" Nodes

In certain instances of the simulation, one or more nodes would be so dissimilar from the rest of the nodes in the network that their initial connection provided by the random discovery service would be immediately broken before the node had sufficient opportunity to migrate to a more suitable position, resulting in nodes that were completely disconnected from the network. A combination of two modifications to the simulation resolved this problem. Firstly, nodes without any connections were permitted to use the discovery service at any time, rather than just once. Secondly, nodes could decide whether to accept or reject an incoming connection and even disconnect an existing peer if a more suitable connection attempt arrives.

A further improvement to this process would require peers who break a connection to provide a list of nodes for the just-disconnected peer to try and connect to. Obviously this functionality relies on some degree of trust between peers.

3.5 Observation: Network Fragmentation

If two or more nodes become starved or a particularly vital connection between nodes is broken, the network may fragment into two distinct subgraphs. In this situation, the two subgraphs cannot at present reconnect themselves into one graph since no node in either graph can "know" of nodes in the other graph and none may use the random discovery service since they all have connections. Obviously this is not ideal. A partial solution is the history of connections mentioned above; as a consequence of this each of the nodes in each subgraph has a reasonable chance of obtaining a connection to a node within one of the other subgraphs.

However, two disconnected networks that are unaware of one another's existence is possible in a production environment since they could evolve independently. In this case, little is lost by having separate networks, though obviously connecting them together may be beneficial. Since this tool is likely to be deployed as part of structured projects rather than ad-hoc like many P2P applications, network fragmentation and initial peer discovery do not present the same challenges.

[3]Other possibilities for initial discovery in production environments include implementing a discovery service or using local broadcast to find peers already running locally. In any case this is a digression as the goal of this research is the development and evaluation of effective awareness support, not the creation of a novel method for "bootstrapping" P2P networks.

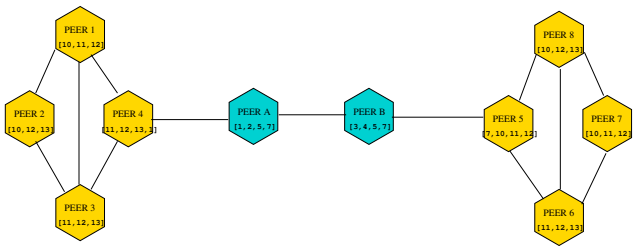

Figure 4. Local Maxima

3.6 Observation: Local Maxima

Due to the lack of a global state, it is possible for local maxima to evolve such as those shown in Figure 4. As far as the nodes are concerned, they are in the "correct" places, but obviously a better network structure is possible and would arise if Peer 5 and Peer 4 could communicate directly. This problem is somewhat similar to the network fragmentation problem described earlier and at present no solution exists. If more interesting nodes are close enough for awareness messages originating from them to bridge the gap, then potentially such information ("there is a more interesting node somewhere nearby") could be used by the algorithm when deciding which connections to make. However, if the maxima are too far apart, they may be considered to be disconnected subgraphs.

It is not anticipated that such maxima will be a problem in deployment for the same reasons that fragmentation and initial discovery present less of a challenge than they would in P2P applications such as file-sharing. Indeed, the situation in Figure 4 would only arise with a TTL of 2 or less; a very unlikely parameter.

4 Conclusion

A simulation approach has been used to evaluate the behaviour a local optimisation algorithm used for preparing a P2P semantic overlay network suitable for distributing awareness information between nodes prior to deploying the algorithm in a historical awareness tool for distributed software engineering. Several problems were identified with the algorithm including a tendency to starve certain nodes of resources and the algorithm evolved from the original design to a new, more complex but more capable

variant. Further evolution is expected as the evaluation continues.

A number of outstanding issues have been identified as a result of the evaluation to date via simulation and these remain to be addressed by further research. Firstly, control of the simulation must be improved. While the existing event generator is useful, a more controlled method such as scripting must be provided for event injection; otherwise, duplicating interesting circumstances is difficult. Earlier in the research the issue of results reporting was identified; introduction of the live results graphing shown in figure 2 resolved this to the researcher's satisfaction.

Secondly, the "small world" properties of the network structures created by the algorithms should be examined and compared to the properties of existing collaboration networks that are not necessarily technologically mediated. Intuition suggests that in order to be successful the properties of the awareness P2P network should closely mirror the properties of the existing collaboration networks, but this is a large assumption to make without further confirming evidence. Modelling existing collaborations presents a challenge; however, there is sufficient literature available for a preliminary review at least.

For user-facing evaluation a P2P awareness module and associated visualisation will be added to the Eclipse platform as a plug-in. Use of an IDE like Eclipse makes gathering information about what a user is doing considerably easier than a mere stand-alone tool monitoring the file-system. Information can be gathered from what resources a user has open and the sorts of operations they perform on them. The algorithm implementation used will not actually differ from that of the simulation, since at present the algorithm runs in a type of container within the simulation and a similar container could be provided within the Eclipse plug-in.

Further development and evaluation of this tool will form the next major step in this research. The aim of this phase is to address further questions such as:

1. What is the effect that awareness has on the development process?
2. How does this new tool affect the way collaboration is performed in a distributed software engineering environment in comparison to normal face-to-face collaboration?

References

[1] K. Aberer, P. Cudré-Maroux, A. Datta, Z. Despotovic, M. Hauswitrth, M. Punceva, and R. Schmidt. P-grid: A self-organizing structured p2p system. *SIGMOD Record*, 32(3), September 2003. P2P Special Issue.

[2] E. Adar and B. A. Huberman. Free riding on gnutella. *First Monday*, August 2000. Technical Report, Xerox PARC.

[3] C. Boldyreff, D. Nutter, and S. Rank. Communication and conflict issues in collaborative software research projects. In *Proceedings of the 4rd Open Source Software Workshop, ICSE 2004*, March 2004.

[4] L. F. Cabrera, M. B. Jones, and M. Theimer. Herald: Achieving a global event notification service. In *Proceedings of the 8th Workshop on Hot Topics in Operating Systems*. IEEE, May 2001.

[5] A. Crespo and H. Garcia-Molina. Awareness services for digital libraries. In *Proceedings of the First European Conference on Research and Advanced Technology for Digital Libraries*, Pisa, September 1997.

[6] A. Crespo and H. G. Molina. Semantic overlay networks for p2p systems. Technical report, Google Technologies Inc. Stanford University, 2002.

[7] P. Dourish and V. Belotti. Awareness and coordination in shared workspaces. In *ACM Conference on Computer Supported Cooperative Work (CSCW'92)*, pages 107–114, Toronto, Ontario, November 1992. ACM Press, New York City.

[8] D. Doval and D. O'Mahony. Overlay networks: A scalabale alternative for P2P. *IEEE Internet Computing*, August 2003.

[9] C. Gutwin, S. Greenberg, and M. Roseman. Workspace awareness support with radar views. In *CHI Conference Companion*, pages 210–211, 1996.

[10] IEEE Computer Society. *The Fourth IEEE International Conference on Peer To Peer Computing*, Zurich, Switzerland, August 2004. IEEE. http://femto.org/p2p2004/.

[11] M. B. Jones and J. Dunagan. Engineering realities of building a working peer-to-peer system. Technical Report MSR-TR-2004-54, Microsoft Research, June 2004.

[12] A. Montresor. A robust protocol for building superpeer topologies. In *Proceedings of the Fourth IEEE International Conference on Peer To Peer Computing* [10]. http://femto.org/p2p2004/.

[13] D. Nutter and C. Boldyreff. Historical awareness support and its evaluation in collaborative so ftware engineering. In *Proceedings of WETICE 2003*, pages 171–176. IEEE Computer Society, June 2003.

[14] D. Nutter, C. Boldyreff, and S. Rank. An evaluation framework to drive future evolution of a research prototype. In *Proceedings of the 5th workshop on Evaluating Collaborative Enterprises (ECE)*, Modena, Italy, June 2004. IEEE Computer Society. http://hemswell.lincoln.ac.uk/wetice04/.

[15] D. Nutter, S. Rank, and C. Boldyreff. Architectural requirements for an Open Source Component and Artefact Repository System within GENESIS. In *Proc. of the Open Source Software Development Workshop*, pages 176–196. University Of Newcastle, February 2002.

[16] M. Portmann, S. Ardon, P. Senae, and A. Seneviratne. PROST: A programmable structured peer-to-peer overlay network. In *Proceedings of the Fourth IEEE International Conference on Peer To Peer Computing* [10]. http://femto.org/p2p2004/.

[17] Y. J. Pyun and D. S. Reeves. Constructing a balanced (log(n)/loglog(n))-diameter super peer topology for scalable p2p systems. In *Proceedings of the Fourth IEEE International Conference on Peer To Peer Computing* [10]. http://femto.org/p2p2004/.