Simulated Self-Organization of a Peer To Peer Awareness Network

David Nutter
University Of Lincoln
dnutter@hemswell.lincoln.ac.uk

Introduction

Software engineers working in distributed teams are deprived of much "out of band" communication with their colleagues, with consequent effects on the coherence of team work [3]. Distribution itself has deleterious effects on team work due to problems rarely suffered by colocated teams including a language barrier, time differences and cultural differences. Consequently, to help alleviate these problems, software tools instead of colocation must provide context awareness for software engineers working in distributed teams.

Many awareness systems are real time (synchronous awareness), facilitating tasks such as collaborative editing of documents [4, 1] by highlighting changes made by other users. While useful at times they do not provide a developmental history of the collaborative object In distributed software engineering domains, particularly Libre software, this knowledge is often incompletely contained in mailing list archives and the log files of configuration management repositories.

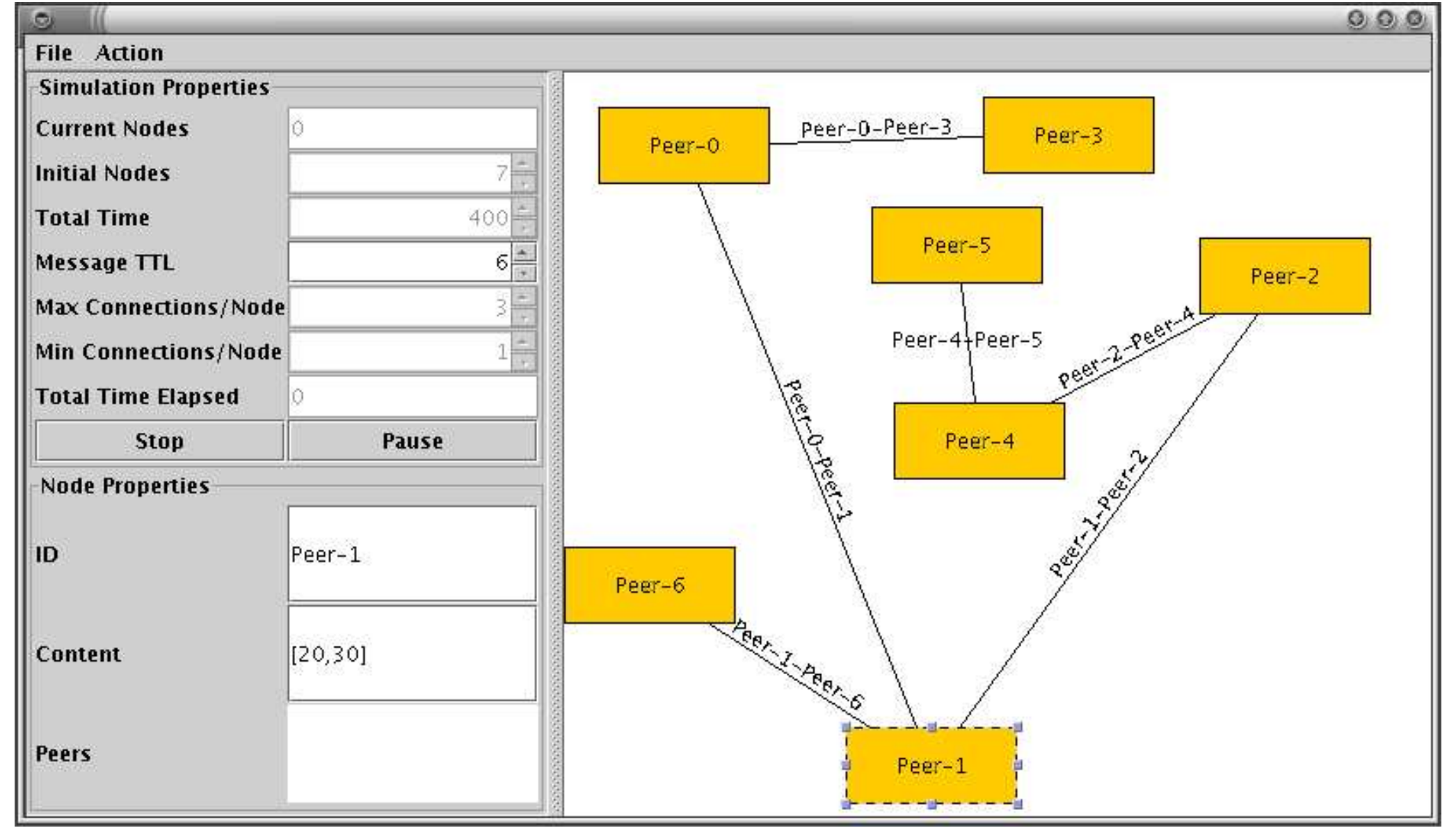
Therefore the concept of *historical awareness* has been defined[5]:

> The complete context of an artefact's creation, derived from a collection of heterogenous artefacts (source code, design etc) rather than a contextless view of a single artefact's evolution.

Certain elements of such information should be presented to the user for a specific artefact or collection of artefacts.

- Details of previous conflicts
- "Proximity" of current user activities
- Artefacts related to current artefact

Tool In Use



Simulation Properties
Current Nodes: 0
Initial Nodes: 7
Total Time: 400
Message TTL: 6
Max Connections/Node: 3
Min Connections/Node: 1
Total Time Elapsed: 0

Node Properties
ID: Peer-1
Content: [20,30]
Peers:

Awareness Distribution Model

In order to provide ad-hoc context awareness capabilities for distributed teams without complex setup the Peer To Peer model of distribution has been selected. Rather than all awareness information passing through a central reflector, with attendant scalability and robustness problems, users' clients exchange awareness information with a limited number of peers.

As an additional benefit, given a suitably organised network, peers will be "close enough" to all other peers working on artefacts relevant to them that awareness information may be given a Time To Live (TTL) in hops from the originating peer. Once information has travelled this "distance" it will no longer be relevant and may be discarded, keeping network load down.

Therefore, the remaining problem is the organisation of the network. Much research has been carried out in the area of Semantic Overlay Networks[2] though most implementations rely on the concept of a Super Peer or Edge Rendezvous Peer to incorporate new nodes into the network. Whilst appropriate for large networks (such as filesharing or messaging) Super Peers are not required for small, ad hoc awareness networks.

The simulation tool

Prior to deployment of the awareness toolset in a controlled experiment, the algorithms used to organise the network will be tested in a simulation environment. This environment will allow:

- Algorithms to be refined without costly, complex experiments
- Algorithm performance evaluation in multiple network environments that would be difficult or disruptive to simulate for real.
- Reproducible experiments and test cases, with total parameter control. Other activities on a "real" network may disrupt experiments unpredictably.
- Simple demonstration of the self-organization concept.

Architecture & algorithms

The tool is based on an Open Source graph framework (JGraph) and associated layout algorithms. Consequently, operations on the network (connection/disconnection of peers, routing of information) may be expressed as operations on the graph.

The simulation packages offers the experimenter various parameters including:

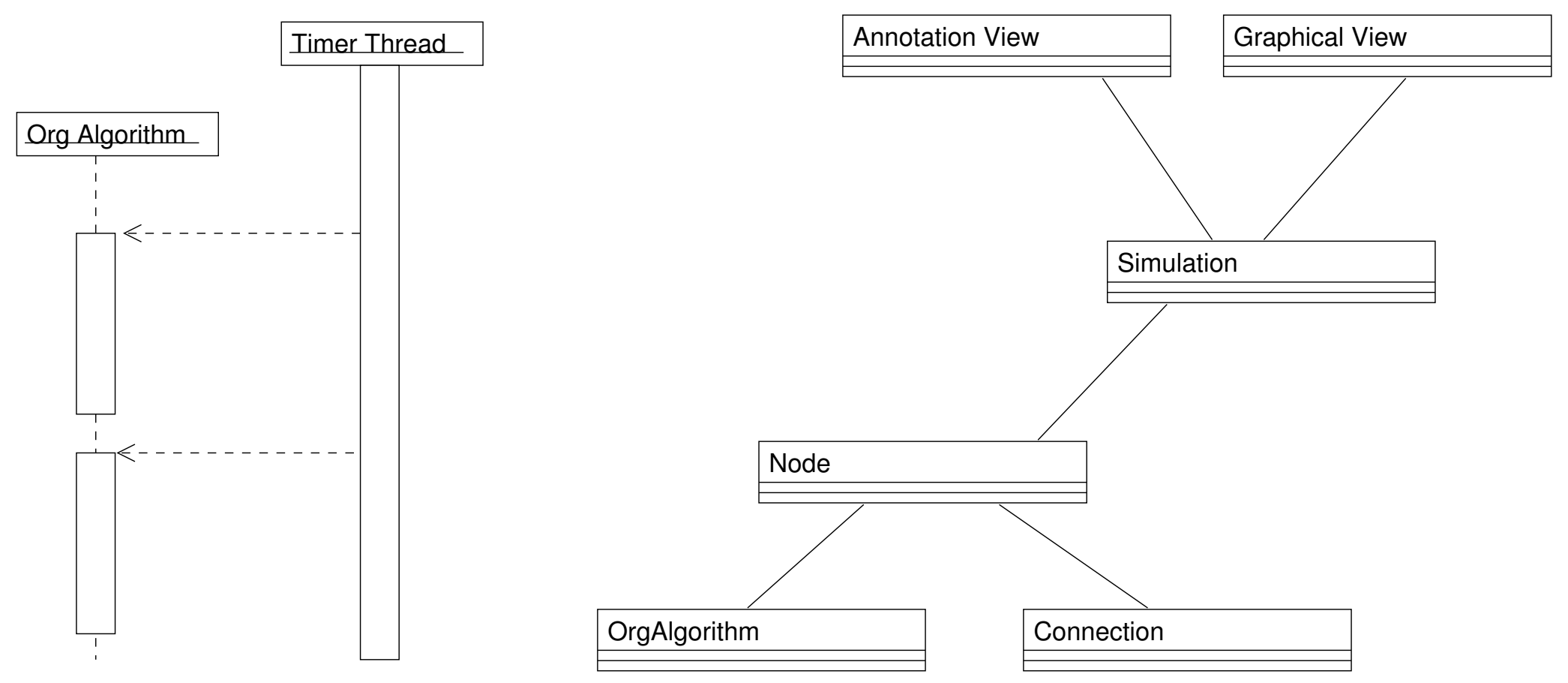
- The Message Time To Live (MsgTTL)
- Max and Min connections permitted per node.
- Event frequency (tick)
- Automatic content generation, within bounds
- Length of the simulation run
- Number of seed nodes to generate

At simulation launch, nodes will be randomly connected to one other node. No guarantee is made that the network will be fully connected. Subsequently, algorithms are applied by *each node* to organise the network into an optimum state.

An example of the current algorithm operating on four distinct peers may be shown as follows:

$$
\begin{array}{l}
Peer\ A \xrightarrow{connect} Peer\ B \\
Peer\ A \xrightarrow{send\ details\ of\ A} Peer\ B \\
Peer\ B \xrightarrow{send\ details\ of\ A} Peer\ C \\
Peer\ B \xrightarrow{send\ details\ of\ A} Peer\ D \\
Peer\ B \xrightarrow{reconnect\ D} Peer\ A \\
Peer\ A \xleftarrow{disconnect} Peer\ B \\
Peer\ A \xrightarrow{connect} Peer\ D
\end{array}
$$

The tool architecture is as follows:



Evaluation Testbed

By using the simulation, a refined algorithm for network organisation will be developed. Subsequently, the algorithm will be ported to the Sun JXTA framework and the awareness timeline view implemented as an Eclipse plugin. The eclipse plugin will also start the JXTA back end to connect to other clients and exchange awareness information.

The items being edited in Eclipse will indicate to the awareness system what items are currently "relevant" to the user and consequently what nodes the network organisation algorithm decides to connect to.

The current experimental plan[5] is to validate the testbed by deploying it with existing Eclipse users in the research group. Once successfully validated, a controlled experiment based on simple collaborative programming exercises is planned, with awareness and without.

Awareness project goals

This loose requirement for historical awareness may be decomposed into some high-level project goals for implementing historical awareness for distributed teams:

- Contextual view of an artefact's development, displayed using the *timeline visualisation*
- Reflector-free distribution model, for ad-hoc construction of awareness networks. By contrast, the most common method of CSCW application development relies on a centralised reflector to distribute information
- Integration with existing development environment for evaluation purposes.
- Evaluation driven development of the prototype implementaiton.

References

[1] 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.

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

[3] J. A. Espinosa, R. E. Kraut, S. A. Slaughter, J. F. Lerch, J. D. Herbsleb, and A. Mockus. Shared mental models, familiarity, and coordination: A multi-method study of distributed software teams. In *Proceedings of the 23rd International Conference in Information Systems (ICIS)*, pages 425–433, Barcelona, Spain, 2002.

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

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