

A P2P Network with inherent Support for Adaptation

Daniel Hughes
Computing Department
Lancaster University
Lancaster, UK.
+44 (0) 1524 594117
d.r.hughes@lancaster.ac.uk

Geoff Coulson
Computing Department
Lancaster University
Lancaster, UK.
+44 (0) 1524 593054
geoff@comp.lancs.ac.uk

Ian Warren
Computing Department
Lancaster University
Lancaster, UK.
+44 (0) 1524 593793
iw@comp.lancs.ac.uk

Abstract

There has been a significant body of research conducted into various structured and unstructured overlay network protocols. Both paradigms have advantages for specific application domains and researchers are beginning to examine the benefits of using hybridized systems. We hypothesize that resource awareness and adaptation are essential to the efficient exploitation of the resources available on the diverse nodes which compose peer-to-peer networks. To support this, we propose a hybrid peer-to-peer model which uses an unstructured decentralised network layered on top of a structured overlay to provide support for multiple levels of adaptation. This model will improve quality of service for traditional peer-to-peer systems and provide support for novel next generation peer-to-peer applications.

1. Introduction

In heterogeneous peer-to-peer environments, where nodes have very different capabilities and requirements, we believe that resource-awareness and adaptation are essential. Adaptation can be used to maximize the contribution that peers make to the network by selecting the most appropriate role for each node. It can also be used to increase the benefit that nodes accrue from participating in the network by adapting network services based on the requirements of each node. The potential of such techniques to improve network performance has been shown by projects such as AGnuS [1].

The use of resource awareness and adaptation necessitates a departure from the purist's approach to peer-to-peer, wherein all nodes are considered equal. In reality, nodes are far from equal. For example, the number of nodes accessing the Internet via fast broadband connections is increasing rapidly, though the number of nodes accessing the Internet via slow mobile connections is also increasing

due to the proliferation of mobile internet-access technologies.

These two groups of peers have very different capabilities and requirements, making it difficult for them to coexist efficiently on one network where all nodes are treated as equals. Our solution is to have the network adapt its behaviour to make better use of each node's capabilities and better meet its requirements.

The resources made available to the network from each node will change over time. This is due in part to competing processes running on these nodes and changing patterns of use. This is a more of an issue in peer-to-peer systems than in traditional routing infrastructures, as the vast majority of the nodes which compose peer-to-peer systems are general-purpose workstations running multiple applications.

The Reflective and Dynamic P2P Framework (RaDP2P) makes use of resource awareness in both a static and dynamic context. Static resource awareness represents the capabilities of a node, while dynamic resource awareness represents the resources this node is currently making available to the network.

Recent research has shown the potential of using hybridized network structures as in Structella[2]. We suggest that such hybridized models can be used to support a range of adaptation techniques which can enhance network performance. RaDP2P provides support for three levels of adaptation:

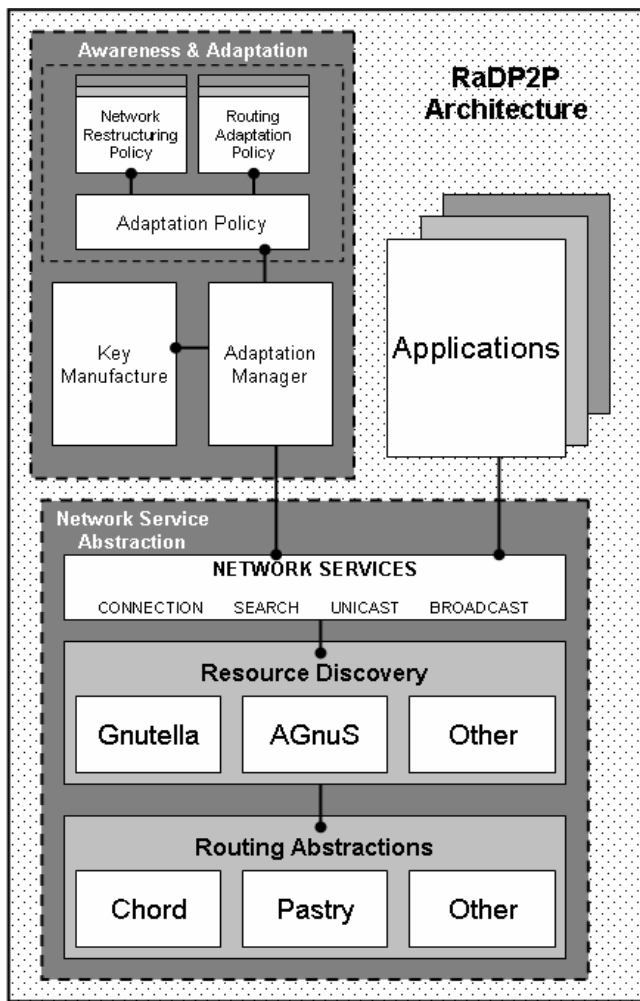
- **Network Restructuring Adaptation:** Based on reflected information about a node and knowledge of the network structure, a node's relative position in the overlay network may be modified.
- **Routing Behaviour Adaptation:** Based upon a node's changing state, or the state of its immediately connected peers, a node may adapt its routing policy.

- **Peer Selection Adaptation:** Following the resource discovery stage, multiple peers may be able to provide a desired service. Meta data about each node is used to inform better peer-selection.

RaDP2P is a combination of two proven models. A structured decentralised model provides an efficient routing substrate upon which an unstructured decentralised resource discovery layer is overlaid.

By implementing our resource discovery network over a structured overlay, rather than directly over TCP/IP, it is possible to restructure the network according to a number of optimization strategies. Examples of these strategies are discussed in detail in section 4.

2. RaDP2P



[Figure 1 – RaDP2P Model]

The base layer of our system is formed by a Key Based Routing (KBR) protocol such as Pastry [3] or Chord [4]. These allow for efficient routing of messages and look-up of nodes.

Resource discovery services are provided using unstructured decentralised networks such as Gnutella [5] or AGnuS [1]. These are overlaid on top of the base routing layer.

Key allocation in RaDP2P differs from the mechanisms used in most structured overlays, in that the value assigned to RaDP2P keys is used to reflect information about each node.

This information is used for network-level adaptation and routing behaviour adaptation, as described previously in section 1 and explored further in section 4.

Network Restructuring

A globally defined network restructuring policy is used together with meta-information harvested from each node to generate the most significant bits of each node's key. As KBR overlays are ordered by key value, and the most significant bits of the key are derived from meta-information, the network restructuring policy defines each nodes position in the overlay.

Routing Adaptation

A globally defined routing behaviour adaptation policy is used together with meta-information to generate the least significant bits of each node's key. In this case, the goal is not to modify the relative position of the node on the network, but simply to mark nodes for differential treatment by their peers.

Following the resource discovery phase, application interaction may occur via the structured decentralised routing substrate, or outside of the RaDP2P model (for example a direct TCP connection). The latter provides a fall-back communication method so that direct peer-to-peer communication remains available during periods of reconnection. In this way, network services remain optimized, while existing interactions between applications on remote nodes are unaffected by network restructuring.

A complete policy component must define a network restructuring policy, a routing adaptation policy and a general adaptation policy which contain the supporting meta-data harvesting methods.

As the state of each node changes over time, the nodes key will be remanufactured and it will reconnect to the most appropriate area of the network. In this way, the structure of the network is dynamically maintained.

2.1 Adaptation Policy Implementation

Adaptation policy components are implemented by developers based on a set of interfaces provided by RaDP2P. These interfaces are:

▪ Network Adaptation Component

This component should implement a **monitor** method that, when called returns the latest meta-data information harvested from the system and a **getTiming** method which returns the rate of key-regeneration required by this policy.

▪ Network Restructuring Policy

As with the network adaptation component, this component requires the implementation of a **monitor** method, together with a **getTiming** method, which defines how often the node's reflective key should be regenerated.

▪ Adaptation Policy

The adaptation policy component requires the methods **getNetworkRestructuringPolicy** which returns the network restructuring policy component, **getRoutingAdaptationPolicy**, which returns the routing adaptation policy component, **getName** which returns the policy name and **getDescription** which returns a plain text description of this adaptation policy. This component is used by the Adaptation Manager to determine how to structure and maintain the network.

2.2 Adaptation Manager

The adaptation manager loads the policy implementation at run-time from a known directory using Java's reflection API.

The adaptation manager calls the **getTiming** method of the network restructuring and routing adaptation policy components and initiates an adaptation sequence accordingly, polling the **monitor** method of each policy at the interval defined by the **getTiming** method.

The result of the **monitor** method is passed to the key manufacture component which returns a new key based upon the latest monitoring information. In this way, the adaptation manager dynamically adjusts the node's position in the routing overlay and its message routing behaviour.

2.3 Key Manufacture Component

The key manufacture component takes a pair of integer values representing the result of the network restructuring and routing adaptation policies and casts these into the most and least significant bits of a key respectively. The remaining body of the key is populated randomly to maintain the key uniqueness.

2.4 Network Services Layer

The network services layer provides a common interface to the potentially diverse resource discovery and routing protocols that may be used. This component is still under heavy development, however it currently provides for the following common peer-to-peer services:

- Connecting to the underlying networks.
- Sending messages directly to peers.
- Sending messages to all peers (broadcast).
- Plain text search.
- Remote search (described in section 4).

3. Implementation

This section provides a brief overview of the implementation of RaDP2P. RaDP2P is in the early stages of development.

An initial prototype has been developed and is currently being tested in a number of different scenarios. We anticipate that following this period of testing, the framework will go through a period of redevelopment, incorporating features that developers found lacking and fixing any bugs that may have become evident.

The Framework's core modules are written entirely in Java and policy components must also currently be defined in Java. Policy components are loaded at run time from a policy subdirectory using the Java reflection API.

It was considered important that RaDP2P be able to use different resource-discovery and routing substrates. This serves two purposes; it allows the developer to select the most appropriate substrate for any given environment and makes it easy to perform performance comparisons on the different substrates which may be used to underpin a RaDP2P network.

The current release of RaDP2P supports each of the case studies discussed in Section 4. However, the performance of the system has not yet been comprehensively evaluated.

Applications interact with the system through the API of the network services layer, which abstracts over the specific complexities of the underlying peer-to-peer substrates and provides a simple set of generic functions. We hope that this will allow the rapid development of novel applications and adaptation strategies.

Alongside the core system model described in section 2, RaDP2P contains many supporting utility components providing facilities including standard meta-information harvesting tools such as CPU benchmarking, network bandwidth measurement and disk performance testing.

All RaDP2P components extend the RaDP2PComponent class which provides common system-wide functionality such as activity logging, special exceptions and other utility functions.

4. Case Studies

In this section we described three applications which are used as cases-studies to illustrate the potential of the RaDP2P framework. The case-studies used are ad-hoc mobile chat, scalable ad-hoc file sharing and geographically-aware service location.

- The ad-hoc mobile chat application illustrates how routing behaviour adaptation can be used to compensate for the highly variable capabilities of nodes, maximizing the performance of the network as a whole and also the benefit accrued by individual nodes.
- The scalable ad-hoc file sharing application shows how network restructuring adaptation can be used to dramatically improve the scalability of unstructured decentralised resource-sharing networks.
- The geographically aware service location system is an example of a novel peer-to-peer application which makes use of network restructuring functionality.

4.1 Ad-Hoc Mobile Chat

Consider an ad-hoc peer-to-peer chat application which operates over an unstructured decentralised network infrastructure (similar to Gnutella). While such networks are excellent for forming ad-hoc groups and simple resource discovery, the bandwidth consumed due to message passing can be prohibitively high for mobile nodes.

In unstructured decentralised networks, all message-passing is handled by the peer-nodes themselves. As peers must route all network messages, participating in this kind of community on expensive, narrowband mobile

connections, such as GPRS and GSM, would be extremely expensive and consume a significant fraction of a node's available bandwidth. This makes participation for such nodes unfeasible.

In order to reduce the cost of participation in such networks for mobile nodes, a routing adaptation policy could be defined which tags nodes based on their connection type (mobile or fixed). Tagging nodes is accomplished by manufacturing the least-significant bits of their key from meta-information which reflects their connection type.

If any node has a directly connected peer possessing such a tag, it will not use it to route messages that are destined for other peers, instead making the next best selection from its routing table. In this way, mobile nodes will receive all messages intended for them, but will not participate in routing messages destined for other nodes.

As messages are not routed through mobile nodes, the typical hop-count between origin and destination nodes will increase, potentially reducing performance, however, the performance decrease caused by an extra few hops when delivering messages may well be counterbalanced by the increased reliability of message delivery, as less reliable mobile nodes are no longer taking part in the message-routing process.

In situations where a very large number of nodes on the network are tagged as mobile, a situation could potentially arise where some nodes become unreachable or message delivery times unacceptably long. In cases where there is no viable next-hop in a node's routing table, mobile nodes can be forced to route messages, ensuring they will still be delivered.

The effect of this kind of routing adaptation is to allow those nodes on mobile low-bandwidth connections to use the ad-hoc chat service cost effectively and without having their highly limited bandwidth flooded by messages they are routing to other nodes. Furthermore, the network may well benefit from the exclusion of slow, unreliable peers from the routing process.

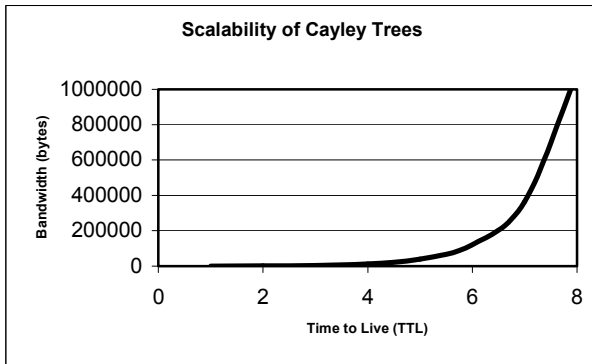
4.2 Scalable Ad-Hoc File Sharing

Consider an ad-hoc peer-to-peer file-sharing application which operates over an unstructured decentralised network infrastructure similar to the Cayley tree used in Gnutella [5]. Networks such as these are subject to scalability constraints.

The broadcast search mechanism employed in Gnutellaesque systems causes the bandwidth consumed by searches to rise dramatically as the numbers of users on the network grows.

‘Why Gnutella Can’t Scale’ [6] explores the relationship between network coverage and the bandwidth consumption caused by the generation of search terms.

Figure 2 shows how the bandwidth consumed by an 83 byte textual search term "grateful dead live" rises dramatically with network coverage on a Cayley tree network where each node maintains four connections to its peers (the default connection number used in Gnutella).



[Figure 2 – Poor scalability of Cayley tree Networks]

The graph shown in Figure 2 clearly illustrates that bandwidth consumption rises dramatically with network coverage on Gnutellaesque networks.

For this reason, all messages in such networks are assigned a time to live (TTL) value which limits the bandwidth consumption to a level which is friendly to the underlying network infrastructure.

The use of TTL values effectively limits how far messages propagate through the network. The default TTL value used in Gnutella is 7, resulting in an accessible pool of approximately 10,000 nodes. Considering that Napster [7] was said to have serviced up to 1,000,000 users, it is clear a query may only reach a fraction of the available nodes. This effect is known as a ‘Search Horizon’ [8] and makes Cayley tree networks inherently unsuitable for supporting very large communities of users.

Using the resource awareness and adaptation mechanisms of RaDP2P, it is possible to target broadcast searches directly to the most appropriate region of the network.

While the search horizon limitation is still present in RaDP2P’s resource discovery layer, using a network restructuring policy that orders the network based on whether nodes are sharing resources, we can direct queries to the most appropriate area of the network, ensuring that, while a query may not reach every node on the network, it does reach every node sharing relevant resources.

To accomplish this, we define a network adaptation policy wherein network restructuring information (and hence the most significant bits of a node’s key) are generated based on the kind of files that node is sharing.

When a node connects to the network, the routing layer will join it to the overlay in a region that contains nodes with similar keys (and hence nodes that are sharing similar files).

As the network is now ordered by file-type, it’s a simple matter to direct queries to the most appropriate area of the network using RaDP2P’s remote broadcast query.

Remote Broadcast Queries are used to perform a broadcast search from a given start point (key) on the network.

In this case, the start-point key is generated based upon the file-type that the search is targeted at and a random seed. The use of a random seed, rather than a fixed key-value prevents congestion, as remote queries for a given file type are not always directed to the same position in the overlay.

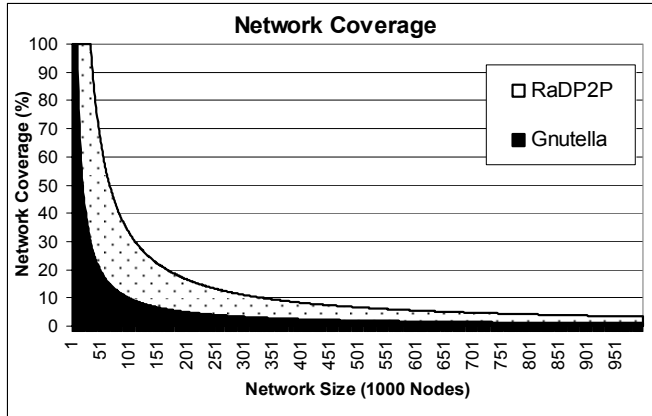
The remote broadcast query is routed to the peer whose key is closest to the given start point. This peer then initiates a broadcast search over the resource discovery network from that position. As the overlay is arranged in order of file type, all 10,000 nodes within the available search horizon will be sharing the type of file desired. The syntax of a remote broadcast query is show below:

```
remoteBroadcastQuery(key StartPoint, String Query)
```

Research by Xerox Parc [8] reveals that 70% of Gnutella nodes share no files at all, while 50% of all files are shared by just 1% of nodes, making the actual architecture of the Gnutella network closer to a client-server rather than peer-to-peer paradigm. This can be considered typical for ad-hoc resource sharing networks.

With a search horizon of 10,000 nodes (i.e. 10,000 reachable nodes), the study suggests that only 3,000 of the reachable nodes will actually be sharing files of any kind.

Consider a network restructuring policy which orders nodes based simply on whether they are sharing files or not. Using the network restructuring scheme described above and remote broadcast queries, it is possible to ensure that the entire search horizon is populated with nodes that are sharing files. This can lead to dramatic scalability improvements over unordered networks. The effect of which is shown in Figure 3.



[Figure 3 – Network Coverage Example 1]

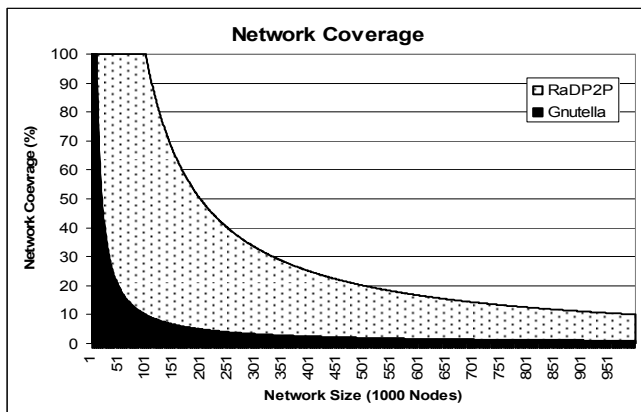
Figure 3 shows that by using network restructuring to avoid routing Query messages through peers which are not sharing files, network coverage (of relevant nodes) can be very significantly increased.

It is possible to achieve greater improvements still by structuring the network based on more fine-grained meta-information. Studies have shown [9] that more than 50% of query traffic on current file-sharing networks is dedicated to searches for music files.

If we assume that query traffic is a reasonable representation of the presence of resources on the network, then this would equate to 50% of the nodes that share files distributing music files.

Now consider the effect of ordering the network based on the file type that users are sharing. Once again, the network is ordered in such a manner that nodes sharing similar kinds of files are collocated.

Figure 4 shows the resultant coverage of relevant nodes when searching for video files on this kind of ordered network compared to the coverage on an unordered overlay.



[Figure 4 – Network Coverage Example 2]

As the two graphs clearly indicate, the more fine-grained the network restructuring policy, the more accurately queries can be channelled and the greater the network coverage possible using standard unstructured resource discovery networks. We envisage three levels of granularity:

1. Ordering by participation level (sharing / not sharing).
2. Ordering by primary resource type.
3. Ordering by resource type and genre.

Increasing the fineness of the network restructuring component's granularity will increase potential network coverage for resource discovery significantly. The result of this is that a standard unstructured decentralised network using the same architecture as Gnutella can be used as an efficient resource discovery service for networks of far greater size.

Furthermore, where the pool of users is known to be smaller than the available search horizon of 10,000 nodes, messages can be tagged with a smaller TTL value, reducing unnecessary message passing load.

4.3 Geographically Aware Service Location

The ability to adapt network structure based on a wide range of meta-information could lead to some novel group-interaction/organization policies.

Consider the example of a peer-to-peer communications network designed to support mobile emergency workers. This system uses mobile devices participating on an ad-hoc peer-to-peer network.

A network restructuring policy may be defined, wherein meta information harvested about the nodes geographical location (for example from GPS hardware), is used to restructure the network such that nodes which are geographically closest are located close to each other on the routing overlay network.

By maintaining the network structure so that it reflects the geographical position of nodes, queries can be efficiently directed to those peers who are geographically closest to the sender (and thus able to assist most rapidly in the case of an emergency). This network could be used as an efficient, low bandwidth ad-hoc substrate for locating nearby help in an emergency.

Any node that requires assistance generates a broadcast request, in which the TTL value represents an acceptable response time in which help must arrive. As nodes are ordered geographically, the size of the TTL value is proportional to the maximum acceptable response time.

For example, in a case of acute injury where assistance is required very rapidly, a very small TTL value may be used – alerting only those workers who are geographically very close and hence able to respond quickly. Using such an ordered network has two key advantages over traditional decentralised networks and semi-centralised systems:

- Queries are only broadcast to those peers who are able to respond within a helpful time-frame, which reduces wasted bandwidth due to needless message passing. This is particularly important in mobile environments.
- The decentralised architecture of this system would have a number of advantages over a centralised paradigm, including the lack of a single point of failure and the ability to form ad-hoc networks anywhere where there are participating mobile nodes without the need for additional infrastructure

5. Related Work

Current work relating to this project includes adaptive P2P systems such as AGnuS [1], hybrid P2P systems such as Structella [2] and protocol abstractions such as MIT's Common API for structured overlays [11] and the GridKit project [12].

AGnuS [1] is an enhanced Gnutella peer which layers resource awareness and adaptation mechanisms above the core Gnutella protocol. It employs four mechanisms which are used to improve performance across the network: Load balancing, content based routing, caching and file filtering. AGnuS demonstrates that awareness and adaptation can significantly improve the performance of peer-to-peer resource sharing networks, though its performance is inherently limited by the underlying protocol. RaDP2P is purpose-built to support resource awareness and adaptation.

Structella [2] demonstrates that it is possible to layer unstructured resource discovery networks on top of structured decentralised routing abstractions. Structella implements a variant of Gnutella over the Pastry [3] routing substrate, successfully demonstrating that decentralised resource discovery networks can be used to add complex query support to structured overlays. However, Structella does not use the inherently structured nature of its routing substrate to organize the content of the network. We believe by adapting the network in this way, significant performance improvements can be brought to existing peer-to-peer systems and new classes of resource aware and adaptive applications can be supported (See section 4).

There is a lot of current research activity directed towards the development of standard models for peer-to-peer systems including The Common API for Structured Overlay Networks [11] and the wider-ranging GridKit project [12]. By comparison, RaDP2P's abstraction model is simple, however, alternative models are currently still immature. We envisage that RaDP2P will eventually adopt the GridKit overlay abstraction which is also being developed at Lancaster University.

6. Summary

This paper briefly discusses structured and unstructured peer-to-peer overlays and emerging research into hybrid schemes.

We contend that peer-to-peer networks in general need to become more resource-aware and adaptive in order to efficiently exploit the growing pool of resources available on nodes around the edge of the network, which tend to have highly variable connection speeds, connection costs and available resources.

We propose that needs of such nodes are diverse, requiring tailored interaction with the network so that the network can best meet their requirements and so that they are able to provide the best possible service to the network. This involves a departure from the purist's approach to peer-to-peer, wherein all nodes are considered equal.

We describe a generic framework for creating adaptive peer-to-peer applications that operates over a hybridized peer to peer network. This model provides support for meta-data harvesting and adaptation.

We describe three distinct levels of adaptation: Network restructuring adaptation, routing behaviour adaptation and peer-selection adaptation and use three case-studies to illustrate their benefits.

▪ Ad-Hoc Mobile Chat

We present a mobile peer-to-peer chat application that uses our adaptation layer to significantly reduce the financial cost and bandwidth load associated with participating in decentralised peer-to-peer networks using mobile nodes. We then discuss potential improvements in performance for the network as a whole resulting from the differential treatment of mobile nodes.

▪ Scalable Ad-Hoc File Sharing

We present a scalable ad-hoc file sharing system which uses our adaptation framework to improve the network coverage that can be achieved using unstructured decentralised overlays. We compare the network coverage

achieved using ordered and unordered overlays, clearly illustrating the benefits of using a network restructuring policy to maintain a content-ordered network.

▪ Geographically Aware Service Location

We present a novel application which makes use of geographical network reordering in order to provide an efficient communication service for mobile emergency workers. This is made possible by the network restructuring facilities of the RaDP2P framework.

7. Future Work

The RaDP2P framework is currently at an early prototype stage. Further development and testing are required before the framework will be ready for release and evaluation.

Currently, the main focus of our development work is the substrate abstraction layer. This layer currently supports only the Pastry [3] routing substrate, though we are working towards support of Chord [4] and Lancaster University's peer-to-peer application framework [10]. The latter is a particular challenge due to key differences in its architecture.

We anticipate that a full release of RaDP2P will be made available by the third quarter of 2004. Further details are available at Lancaster's P2P site:

<http://polo.lancs.ac.uk/p2p/>

8. References

- [1] D. Hughes, I. Warren, G. Coulson. "Improving QoS for Peer-to-Peer Applications through Adaptation."- 10th International Workshop on Future Trends in Distributed Computing Systems. Suzhou, China. May 26-28, 2004.
- [2] M. Castro, M. Costa, A. Rowstron. "Should we build Gnutella on a structured overlay?" - 2nd Workshop on Hot Topics in Networks. Cambridge, MA USA. November 20-21, 2003
- [3] A. Rowstron, P. Druschel. "Pastry: Scalable, Decentralised Object Location and Routing for LargeScale Peer-to-Peer Systems" – Conference on Distributed Systems Platforms, Heidelberg, Germany 2001.
- [4] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek,, H. Balakrishnan. "Chord: A scalable peer-to-peer lookup

service for Internet applications". Technical Report TR-819, MIT, March 2001.

[5] Gnutella Community. Gnutella Protocol Specification v0.4. dss.clip2.com/GnutellaProtocol04.pdf.

[6] J. Ritter. "Why Gnutella can't scale, no really". <http://www.tch.org/gnutella.html> - 2001.

[7] S. Fanning et al. Napster (Press Room), <http://www.napster.com/press.html>, 2000.

[8] Eytan Adar, Bernardo A. Huberman. "Free riding on Gnutella." Technical report, Xerox PARC, 10 Aug. 2000

[9] K. Sripanidkulchai "The popularity of Gnutella queries and its implications on scalability". <http://www-2.cs.cmu.edu/~kunwadee/research/> - 2004

[10] Walkerdine, J., Melville, I., Sommerville, I., A Framework for P2P Application Development, Technical Report COMP-004-2004, Computing Department, Lancaster University, 2004.

[11] F. Dabek, B. Zhao, P. Druschel, J. Kubiatowicz, I. Stoica. "Towards a Common API for Structured Peer-to-Peer Overlays" - Berkeley, CA, USA. 20-21 February 2003.

[12] G. Coulson, P. Grace, G. Blair, L. Mathy, D. Duce, C. Cooper, W. Yeung, W. Cai. "Towards a component-based middleware framework for configurable and reconfigurable grid computing". Workshop on Emerging Technologies for Next generation GRID (ETNGRID-2004) - Emilia, Italy, June 14-16, 2004

9. Acknowledgements

We would like to thank James Walkerdine for his comments and suggestions on this work.