

AdaPtP - a Framework for Building Adaptable Peer-to-Peer Systems

Danny Hughes¹, Geoff Coulson¹ and Ian Warren²

¹ Computing, InfoLab 21, South Drive Lancaster University, Lancaster, UK. LA1 4WA

² Department of Computer Science, University of Auckland, Private Bag 92019, Auckland, New Zealand

{danny, geoff} @comp.lancs.ac.uk | warreni@cs.auckland.ac.nz

Abstract: *Peer-to-peer systems have gained widespread popularity for supporting resource location and distribution applications. To response to this, a number of peer-to-peer network architectures have been developed, each supporting a subset of application types, though no unifying P2P network architecture has emerged that is suitable for supporting today's diverse peer-to-peer systems. The heterogeneous and dynamic environments in which peer-to-peer applications operate has also driven the development of a number of adaptation schemes, though current peer-to-peer platforms offer no specific support for adaptive mechanisms. This paper describes AdPtP, a peer-to-peer framework that provides support for diverse applications and inherent support for adaptation.*

1. Introduction

Peer-to-Peer (P2P) systems are realised using a family of distributed programming approaches that emphasise co-operation between entities known as peers that are essentially equal and can both require and provide services. Two key factors have driven the increasing importance of P2P: The growing pool of untapped resources available at the edge of the network, and increasing Internet connection speeds, which facilitate access to such resources. P2P network architectures may be categorised as belonging to one of four classes:

- **Semi-centralised networks** such as Napster [1] use servers to mediate the resource discovery phase of interaction. Scalability and legal problems have today rendered these networks obsolete.
- **Unstructured decentralised networks** such as Gnutella 0.4 [2] connect nodes in an undifferentiated and random fashion, providing support for search through broadcast flooding. Due to poor scalability, unstructured networks have largely been replaced by super node networks.
- **Super-node Networks** such as Gnutella 0.6 [3] connect more capable nodes with a higher degree and assign these nodes greater routing responsibility. As less capable nodes form the leaves of the network graph and do not route search messages, the scalability of flooding is significantly improved.
- **Structured decentralised networks** such as Chord [4] connect nodes according to strict rules, usually based on a consistent hash function. The resulting networks are capable of providing reliable and efficient object lookup, though they do not provide support for complex (e.g. plain-text) search.

Each class of P2P network has advantages for specific application domains. Super node schemes are considered state-of-the-art for supporting file sharing, while structured decentralised networks are considered state of the art for supporting distributed storage. However, this is problematic for systems that require both *complex search* and *efficient object location*. Such systems must implement multiple classes of P2P network, leading to significant duplication of effort and increased resource usage.

Multi-layer P2P networks [5] have recently emerged as a promising platform to support diverse P2P applications that require both complex search and efficient object lookup. The framework presented and evaluated in this paper implements such a network and improves upon existing multi-layer schemes such as Structella [5] by providing a generic abstraction that allows multiple P2P protocols to be used to implement each layer of network functionality.

Adaptation is particularly critical in P2P environments that are typically highly heterogeneous and dynamic. Furthermore, if multiple applications are to successfully co-exist on the same P2P network, adaptation may be necessary to tailor network performance to meet diverse application requirements. While multiple adaptation schemes, [8], [9], [10], have been developed to address emergent issues on P2P networks, specific support for adaptation is conspicuously absent from P2P frameworks. This paper presents a classification scheme for adaptation in P2P networks (Section 3.1) and the AdaPtP adaptation model (Section 3.2) which is capable of supporting each class of adaptation..

The remainder of this paper is structured as follows: section 2 presents the AdaPtP network architecture. Section 3 presents the AdaPtP adaptation model. Section 4 evaluates AdaPtP's network and adaptation support and concludes.

2. AdaPtP Network Architecture

As previously described AdaPtP uses a multi-layer network architecture to support scalable object-lookup and complex search. Specifically, AdaPtP supports a two-layer network architecture. At the lower layer (*layer-0*), a structured routing substrate is used to support efficient object-lookup. At the upper layer (*layer-1*), an unstructured decentralised or super-node network is used to support search.

Layer-0, the Object-Lookup/Router Layer: Any structured network, which provides reliable and scalable object lookup may be used to implement layer-0, for example Chord [4] or Pastry [6]. Ring based distributed hash tables (DHTs), such as Chord and Pastry are particularly suitable for supporting this layer as these networks are highly scalable. Whatever overlay is used to implement layer-0, it should expose a standard point-to-point routing method to other system elements.

Layer-1, the Search Layer: Any unstructured network, which provides support for complex search may be used to implement layer-1, for example Gnutella [2]. As the search layer operates over the layer-0 object lookup layer it is likely that some cross layer optimisation will be required for efficient operation. For example Structella [6] layers the Gnutella resource discovery network over a Pastry [4] routing substrate and uses the locality-aware Pastry routing table to optimise the Gnutella overlay. Whatever overlay is used to implement layer-1, it should expose a standard broadcast method to other system elements.

2.1 AdaPtP Network Model

In order to increase the re-use of network elements and decrease development effort, P2P network functionality is encapsulated in standard elements with consistent interfaces, which can thus be easily replaced or re-used. Existing component models for overlay networks, such as Open Overlays [7] demonstrate that systematically decomposing P2P functionality is a promising approach to facilitating the development of new P2P networks and encouraging re-use. A routing logic abstraction is provided at each network layer. At the object-lookup level, the structured routing layer provides the *RoutingManager* element, which must expose a single point-to-point routing method to other systems elements: **Route (Message, Key)**. At the search layer, the structured routing layer provides the *SearchManager* element, which must expose a single broadcast method to other systems elements: **Broadcast (Message)**.

AdaPtP also provides a consistent abstraction for connection logic abstraction at each network layer. At the object-lookup level, the structured routing layer provides the *RoutingConnection* element and at the search layer AdaPtP provides the *SearchConnection* element. Both of these system elements expose the following connection functionality: **BootStrapConnection ()** which initiates an initial connection to the network, **ConnectToPeer (PeerID)** which initiates a new network connection to the specified peer and **DisconnectNeighbour (PeerID)** which removes a specified peer from this peer's routing tables.

2.2 Considering the Requirements of Adaptation in the Network Model

Adaptation mechanisms often need to call routing functionality. For example, content based routing schemes such as APS [8] call this functionality in order to efficiently direct resource discovery traffic, while anti-freeriding schemes such as [9] selectively call routing functionality in order to punish non-contributing peers. Adaptation mechanisms may also need to call connection functionality. For example APT [10] selectively calls connection functionality in order to build an overlay wherein peers which 'trust' each other are located close on the application level network. Thus, it can be seen that the specification of consistent connection and routing interfaces at both network layers facilitates the separation of adaptation concerns from those of the underlying network. Adaptation mechanisms are also dependent upon events generated by the P2P network architecture. For example, APS' routing behaviour adaptation mechanism [8] is triggered by the reception of search-response messages, while APS' resource awareness mechanism is triggered upon reception of a particular message. As it is not possible to know in advance what events may be of interest to adaptation policies, AdaPtP provides a generic event model that publishes all connection and network events [11]. The suitability of the AdaPtP network model for supporting diverse routing and search protocols along with the appropriateness of the AdaPtP event model is discussed and evaluated in detail in [11]. Due to space limitations, this is not reproduced here.

2.3 Initial Implementation of AdaPtP

The initial implementation of AdaPtP uses a Chord [4] layer-0 routing protocol, which is overlaid by a cross-layer optimised version of the Gnutella [2] protocol at layer-1. A description of cross-layer optimised search follows.

The cross-layer optimised Gnutella search scheme selects neighbours based upon their relative position in the Chord overlay. Specifically, each peer adds its immediate predecessor 'P', the predecessor of P (P-1), its immediate successor 'S' and the successor of S (S+1). Thus, as in standard Gnutella [2], each peer maintains four neighbours distributed across the physical network. In order to perform broadcast resource discovery two new message types are defined: RING_SKIP_LEFT and RING_SKIP_RIGHT, each of which contains an object payload and a TTL value. A peer wishing to perform resource discovery will generate one of each message. The RING_SKIP_LEFT message will be disseminated through the Broadcast method of P and P-1 while the RING_SKIP_RIGHT message will be disseminated through the Broadcast method of S and S+1. When the Broadcast() method of a peer's SearchManager is called with a RING_SKIP_LEFT message as its argument, the receiving peer will route the message to neighbour P-1 in its search-layer

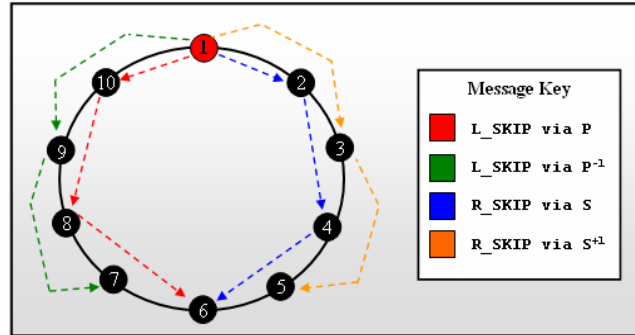


Figure 1 – Cross Layer Optimized Gnutella

neighbour set. When the Broadcast method of a peer's SearchManager is called with a RING_SKIP_RIGHT message as its argument, the receiving peer will route the message to neighbour S+1 in its search-layer neighbour set. The emergent effect of this broadcast scheme is that resource discovery mechanisms 'skip' around the Chord ring. This is illustrated in Figure 1 using a simplified network structure. As can be seen from Figure 1, due to the structure of the underlying network, the optimised search scheme demonstrates 1(N) efficiency, where N is the size of the network. The Chord routing layer, which is not described here, demonstrates Log (N) efficiency for object lookup.

3. Adaptation Support in AdaPtP

Adaptation in P2P systems can be classified into three distinct classes; i) network restructuring adaptation, ii) routing behaviour adaptation and iii) service selection adaptation:

Network restructuring adaptation adapts the relative position of nodes on the network through selective (re)connection. For example, adaptive super-node schemes such as SG-1 [12] adapt each peer's role dynamically based upon changing resource availability. Network restructuring has also been used in incentive schemes for combating the problem of free riding [9].

Routing behaviour adaptation adapts the routing behaviour of nodes on the network. AGnuS [13] uses routing behaviour adaptation to support content based routing and load balancing on Gnutella [2]. Furthermore, this form of adaptation has been suggested as a mechanism to control the effects of free riding, wherein traffic originating from contributing peers is treated preferentially to traffic originating from free-riding peers as in [9].

Service selection adaptation consists of adapting which service a peer selects following the resource discovery phase. For example, Gnutella [2] supports users in performing service selection adaptation by providing them with information about the connection speed of peers offering files. Other systems such as IHD, the interactive help-desk system [14], take this a step further, by automatically selecting the peer with the most appropriate knowledge matching a given query.

3.1 The AdaPtP Adaptation Model

As with networking functionality, AdaPtP provides a consistent abstraction for implementing *adaptive mechanisms*. Based upon systematic analysis of existing adaptation schemes from each class, adaptive mechanisms were designed with a *condition / action / resource awareness* abstraction. A *condition* must be met in order to trigger an adaptation sequence and conditions may be either event or time based. An *action* encapsulates functionality that will be called when the *condition* is met and *resource awareness* encapsulates the data-gathering functionality required to inform adaptation. Evaluation has shown this *condition / action / resource awareness* abstraction is suitable for diverse schemes and promotes re-use. This is discussed in more detail in section 4.2.

The *resource awareness* element is itself broken down into a *trigger*, *monitoring function*, *state* abstraction where the *trigger* is either time or event based, a *monitoring function* encapsulates resource

awareness functionality and writes appropriate *state* data to the framework. As with adaptive mechanisms this abstraction has been shown to facilitate development and promote re-use [11].

Adaptive mechanisms are bundled together into network-wide *adaptation policies*, which each peer downloads at boot-strapping time. A complete adaptation policy may include several adaptive mechanisms and associated resource awareness components.

4. Evaluation

The complete architecture of AdaPtP including adaptation policies, event model and multi-layer network is shown in Figure 2 opposite. This section now provides an evaluation of the current implementation. The system was evaluated using a test-bed of three nodes: a 2.8GHz P4 with 512MB of RAM, a 1.1GHz P4-M with 768MB of RAM and a 2GHz Celeron D with 1GB of RAM connected via a 100Mbps network and running Windows XP SP 2. Section 4.1 discusses the resilience and efficiency of AdaPtP’s network support and Section 4.2 evaluates AdaPtP’s adaptation support.

4.1 AdaPtP P2P Network Support

As AdaPtP uses Chord to implement *layer-0* routing and object lookup functionality, this layer demonstrates the same $\text{Log}(N)$ object location efficiency and high resilience as the Chord protocol.

The optimized *layer-1* search protocol has been compared to a standard Gnutella network in the context of efficiency and resilience using networks of between 10 and 1000 nodes. Figure 3 shows that search in AdaPtP (shown in red) is significantly more efficient than Gnutella: $1(N)$ efficiency versus $4(N)$ for Gnutella (shown in black).

Figure 4 shows that search in AdaPtP (shown in red) is also significantly more resilient than Gnutella (shown in black). This is because of the additional redundancy offered by the structured routing layer which contains references to 16 predecessors and successors rather than the 4 neighbours maintained by Gnutella peers.

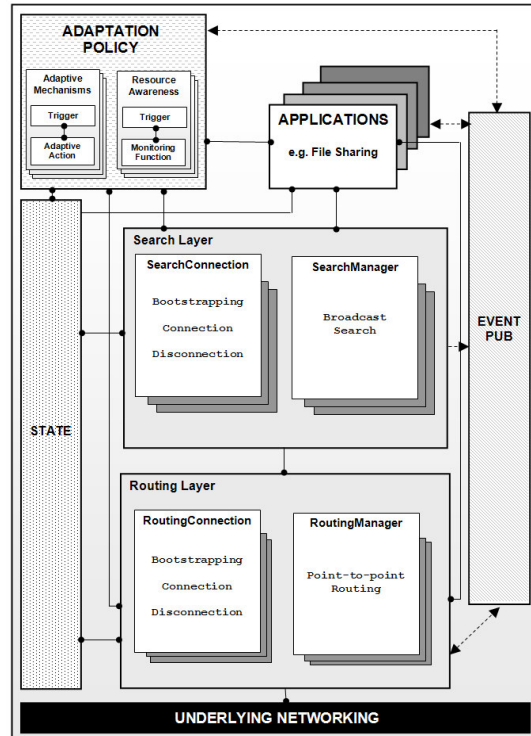


Figure 2 – AdaPtP Architecture

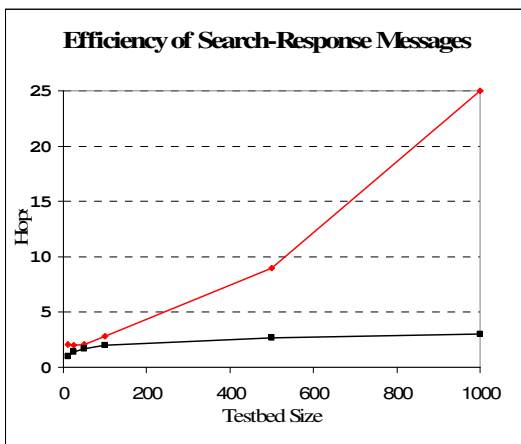


Figure 3 – Efficiency of Search

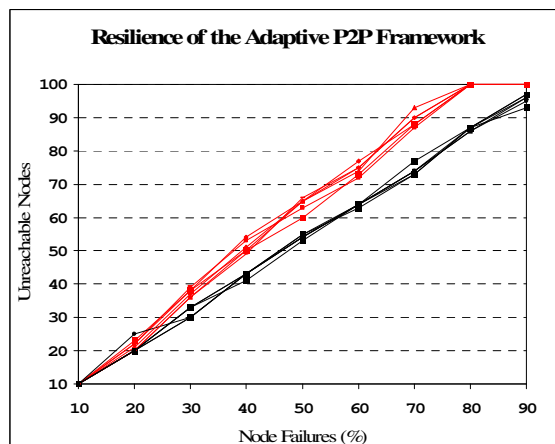


Figure 4 – Resiliency of Search

The brief evaluation of network functionality provided in this section shows that AdaPtP’s multi-layer approach can offer support for diverse P2P applications; offering resilient and efficient support for both object lookup and complex search at minimal additional overhead.

4.2 Evaluation of Support for Adaptation

In order to evaluate the generality of the AdaPtP's support for adaptation, adaptive mechanisms corresponding to each class of adaptation have been implemented; including [8], [9] and [10]. Due to space constraints, these case-study implementations are not described here, however, they may be found in [11]. This evaluation firstly found that the AdaPtP abstraction is suitable for modelling diverse schemes and secondly that this abstraction results in significant re-use of system elements.

AdaPtP's support for diverse network requirements (i.e. efficient routing and complex search) together with inherent support for adaptation are a promising platform for supporting truly diverse P2P systems.

5. References

- [1] Merriden T., *Irresistible Forces: the Business Legacy of Napster and the Growth of the Underground Internet*, Capstone Publishing, 2001, pages 1-11.
- [2] The Gnutella Protocol Specification v0.4 (Document Revision 1.2), http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf, June 2001.
- [3] The Gnutella Protocol Specification v0.6 (draft), http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html, June 2002.
- [4] Stoica I., Morris R., Karger D., Kaashoek F., Balakrishnan H., Chord: A scalable peer-to-peer lookup service for internet applications, in the proceedings of ACM SIGCOMM 2001, ACM Press, vol. 31, issue 4, San Diego, California, USA, September 2001, pages 149–160.
- [5] Castro M., Costa M., Rowstron A., Should we build Gnutella on a Structured Overlay, in the proceedings of the 2nd Workshop on Hot Topics in Networks (HotNets-II), Cambridge, MA USA, November 2003, pages 131-136.
- [6] Rowstron A., Druschel P., Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems, in the proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms (Middleware'01), Heidelberg, Germany, November, 2001, pages 329-350.
- [7] Grace P., Coulson G., Blair G., Mathy L., Yeung W. K., Cai W., Duce D., Cooper, C., GRIDKIT: Pluggable Overlay Networks for Grid Computing, in the proceedings of the International Symposium on Distributed Objects and Applications (DOA'04), Springer LNCS, VOL 3291, Larnaca, Cyprus, October 2004, pages 1463-1481.
- [8] Tsoumakos D., Roussopoulos N., Adaptive Probabilistic Search for Peer-to-Peer Networks, published in the 3rd IEEE International Conference on Peer-to-Peer computing (P2P'03), Linkoping, Sweden, September, 2003, pages 102-109.
- [9] Karakaya M., Korpeoglu I., Ulusoy O., A Distributed and Measurement-Based Framework against Freeriding in Peer-to-Peer Networks, in the proceedings of the 4th IEEE International Conference on Peer-to-Peer computing (P2P'04), Zurich, Switzerland, August, 2004, pages 276-277.
- [10] Condie T., Kamvar S. D., Garcia-Molina H., Adaptive Peer-to-Peer Topologies, published in the proceedings of the 4th IEEE International Conference on Peer-to-Peer computing (P2P'04), Zurich, Switzerland, August, 2004, pages 53–62.
- [11] Hughes D., *AdaPtP - a Framework for Building Adaptable Peer-to-Peer Systems*, PhD Thesis in Computer Science, Lancaster University, Lancaster, UK, September 2007.
- [12] Montresor A., A Robust Protocol for Building Superpeer Overlay Topologies, published in the proceedings of the 4th IEEE International Conference on Peer-to-Peer computing (P2P'04), Zurich, Switzerland, August, 2004, pages 202-210.
- [13] Hughes D., Warren I., Coulson G., Improving QoS for Peer-to-Peer Applications through Adaptation, in the proceedings of the 10th IEEE International Workshop on Future Trends of Distributed Computing Systems (FTDCS'04), Suzhou, China, May 2004, pages 178-183.
- [14] IHD – The Interactive Help Desk System, <http://polo.lancs.ac.uk/p2p/>