ISSN : 0976-8491 (Online) | ISSN : 2229-4333 (Print)

The Concept of Data Management in Peer 2 Peer Networks

Anil Lamba
Dept. of Computer Engineering, HEC, Jagadhri, Kurukshetra University, HR, India

Abstract
The current growth in Peer-to-Peer (P2P) systems, now also requires the presence of the concept of data management in P2P systems. Data management in P2P computing covers a wide range of issues starting from consistency and redundancy, security and privacy and indexing, replication. In this paper we propose a model for data management in P2P systems that divides the functional components, also enables the implementation of various P2P services with specific quality of service requirements using common infrastructure besides improving the data management in the P2P system.

Keywords
Redundancy, Peer-to-peer, Indexing, Storage, Security

I. Introduction

Today on the Internet, there are many applications that benefit from the cooperation between peers. P2P systems are often classified as structured or unstructured depending on the network overlay and how the peers interact in the overall system. However, in some literature, P2P systems have been classified according to the applications or services they offer, i.e., instant messaging, file sharing, grid computing, and collaboration [1]. These applications range from simple chatting to file-sharing to robust Internet-based storage and processor cycle management to digital library applications. Each of these applications imposes different requirements on the underlying P2P infrastructure. For example, file-sharing applications need equality and keyword search capabilities, but do not need sophisticated fault-tolerance. On the other hand, storage and CPU cycle management require not only simple querying, but also robust fault-tolerance. Digital library applications require both complex queries, including equality, keyword search, range queries, and sophisticated fault-tolerance. Other applications, such as P2P television (P2PTV) and service discovery on the Grid, impose their own specific requirements on the underlying P2P infrastructure. One solution to this problem is to devise a special-purpose P2P infrastructure for each application. Clearly, this is quite wasteful, and does not leverage the common capabilities required across many applications [2].

We propose a modularized P2P system architecture that separates different functional components, and allows the reuse of existing algorithms tailored to the needs of the application. In this paper we focus more on the data management aspect of the architecture. Our proposed model has ten components mapped on six layers which are: The basic P2P Connectivity layer, Data Management layer, Replication layer, Lookup layer, Properties layer and Applications layer. Table 1, summarizes the functions of each layer.

Existing algorithms proposed in the literature can be used for the various system components shown in Table 1, to devise an overall architecture whose functionality is far more expressive than any existing P2P system that we are aware of. Crainiceanu et al. [3], proposes a similar approach for P2P storage and indexing, which uses the Chord algorithms [2], for the Connection Control and Replication Manager; the PePeR algorithm [4], for the Data Store, and the Skip Graph algorithm [3], for the Content Router. Hence such approach creates a system that is fault-tolerant, supports both equality and range queries, provides logarithmic search performance and supports possibly large sets of items per peer.

Table 1: P2P Data Management Layers

Layer	Functions
Basic P2P Connectivity	Provides fault-tolerant connectivity among peers.
Data Management	Stores actual data items and provides methods for reliably exchanging items between peers.
Replication	Ensures that copies of data items are stored even in the face of peer failures.
Lookup	Allows efficient location of data items, peers with required resource routing of information and content over the network.
Properties	Provides service to Application layer keeping track of properties of a current machine to assess its suitability for participation.
Applications	Provides configuration parameters for various P2P applications, such as file sharing, storage management, etc.

However, we note that their proposal focused more on efficient data retrieval and data availability at the expense of security and privacy, application management, and task scheduling. Hence, our model seeks to improve the P2P storage and indexing framework proposed in [4], by provide more features covering security, application management, task scheduling, and entity search. The rest of this paper is organized as follows: In Section II, we present our proposed P2P data management model, in Section III, we discuss the application areas, and in Section IV, we present the future research directions and some conclusions of our work.

II. The Model

In the following Subsections, we describe the various components that make up our model as illustrate by fig. 1, below.

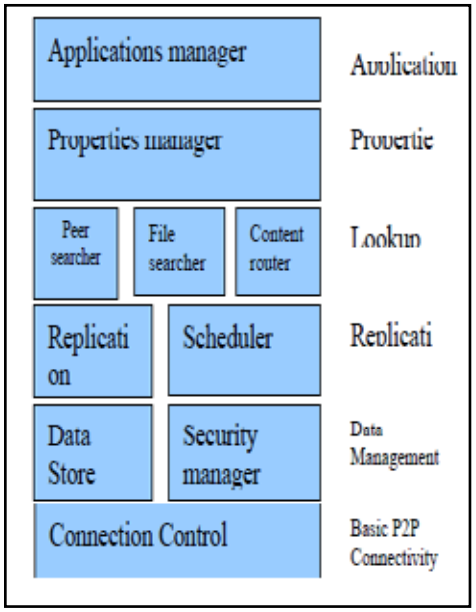

Fig. 1: Data Management Model for P2P

A. Connection Control

The primary goal of the Connection Control (CC) is to maintain the connectivity of the peers in the system. Conceptually, the CC manages the basic operations in a peer network. Issues concerning the joining and leaving of network by peers are handled by this component. For example, a peer efficiently joining and leaving the network consequently poses a structural change in the overlay network. This requires all updates to be sent to peers so as to maintain and update active routes within the system. This component can be instantiated for exchange and use of presence information. Presence information plays very important role in respect to P2P applications. It is decisive in self-organization of P2P networks because it provides information about which peers and which resources are available in the network. It enables peers to establish direct contact with other peers and inquire about the resources such as which resources are available for sharing.

B. The Data Store

The Data Store is a component that maintains data items on the local system and distributes them to peers. Ideally, each peer should store approximately the same number of items, achieving storage balance. The Data Store maintains a mapping of data items and peer indices within the P2P system, and stores the item at the peer responsible for a particular data. If a peer ends up with too many data items (due to insertions) or too few data items (due to deletions), it will have to re-balance the assignment of data items to peers. Exactly how this re-balancing is done depends on the specific instantiation of this component. The equivalent of the Data Store in Chord [5], is implemented using a hash-based scheme. Data items are hashed to values on the ring, and assigned to the first peer with Identity (ID) following the value in the ring (note that since hashing destroys the ordering of the values, Chord cannot process range queries). PePeR [5], does not use hashing, but maps data items to the peers responsible for the respective ranges in the value space. Re-balancing is done at peer insertion or deletion/failure, when some ranges are split.

C. Security Manager

Besides providing efficient storage and retrieval of data items, our proposed model seeks to improve the security and privacy of the stored data items. This task is accomplished by the security manager. Security in this context refers to authenticity of entities, confidentiality and integrity of data items. The specific use of this component depends on its instantiation. E.g. for backup or recovery applications, this component has to maintain the confidentiality and integrity of data items so that the backed up data is not only visible to authorized peers but also its integrity is not compromised. For file sharing applications, authentication of peers and confidentiality of data is very important and in our model is performed by this module.

D. Replication Manager

The role of the Replication Manager is to ensure that all the data items inserted into the system are reliably (under reasonable failure assumptions) stored at some peer until the items are explicitly deleted. The component keeps track of the number of copies of each data item currently available in the system. This aids the lookup service and load balancing of the P2P system. Different applications can use this component in different ways. For example, a backup/recovery application uses this component for backup, where as file sharing application uses replication for efficient search as discussed in [6]. In certain P2P systems like Gnutella, only the requester stores a copy of the requested object (owner replication). To aid fast search more proactive replication strategies can be implemented such as: path replication (replicate along the path from the requester to the provider); and random replication (same number of replicas as in path replication, but replicas are placed randomly among the sites probed). This is managed by replication module in our model.

E. Scheduler

The Schedule Manager is responsible for scheduling tasks in the system. Tasks include, data item replication, backup, recovery, peer re-balancing among others. The specific implementation depends on the requirements of specific applications. The operations of this module are identical to those of a scheduler in ordinary personal computer or in a distribute grid computing system which is a well researched area.

F. Peer Searcher

The Peer Searcher is used to locate a peer in the system that would be a suitable candidate for maintaining replica or backups of a specific data item on the local peer. When a data item needs to be replicated, the local peer searcher component sends a broadcast query to all the peers, looking for peers that would be able to store a backup copy of the item. The peer searcher components on other peers contact their local properties manger to provide a picture of their current resource statistics which they use to respond to peer search queries. The requester peer searcher module would then select a subset of the responding peers based on their resource statistics as suitable repository for replicated or backup copies.

G. File Searcher

The File Searcher is used by lookup service to search for peers that are holding a copy of a given data item and are currently operational. It very common in P2P computing for a peer holding a given data item to abruptly go offline, since peer join and leave at will especially in unstructured P2P systems. Thus, the process of locating a data item involves the search from the index table the potential holders of the copies and their availability. The use of this module depends on the requirements of a particular application. E.g., for file sharing, this component helps is locating a file to be transferred to the query source while for backup/restore application, it is used to find the peer holding the backup copy needed for restoration.

H. Content Router

The Content Router is responsible for efficiently routing packets to their destination in the P2P system. This module implements the search primitives, such as equality and/or range queries. Unstructured P2P search algorithms such as I-walks [7] could be used to instantiate content router together with file and peer searcher components. I-Walk demonstrates how simple but efficient search technique can greatly reduce extra traffic on the overlay whilst delivering good search results. For structured P2P, the Content Router component could also be instantiated using the Chord finger tables, the CAN neighborhood table, or Skip Graphs [7].

I. Properties Manager

The Properties Manager is a module that keeps track of the current computation resource statistics of the peer. Properties tracked by the properties manager includes peer connectivity status, the disk utilization on the system, processor idle time, memory

utilization, number of threads running among others. Depending on the specific data management requirement of an application, this module can be tailored to provide peer assessment platform for participation on the system.

J. Application Manager

This module maps the application layer of our model. Like the application layer of OS1 model of internetworking, the Application manager provides protocols for P2P applications and also manages the connections between cooperating applications. It defines an interface for the implementation of various P2P services with specific requirements without modifying the core infrastructure. In the following Section, we discuss some applications supported by our model.

III. Applications of The Model

The main significance of this model is to tailor the system to the needs of the P2P application. There are many applications that can use our model such as instant messaging systems, P2P backup or restore system, P2P file sharing system, Internet storage management, P2P digital Library systems, among others.

A. Instant Messaging

Instant messaging service† allows you to maintain a list of people that you wish to interact with. You can send messages to any of the people in your list, often called a buddy list or contact list, as long as that person is online. These and similar systems offer peers to pass on information via the network, such as whether or not they are available for communication processes. Instant messaging and similar systems can use mostly the Connection Control of our model for establishing direct contact between peers. The Instant Messaging server does not store the Instant Messenger end-user authentication information so it doesn't need the use of Data Store and Security manager component but requires Peer and File searcher components to search for end-user and group information respectively. However, depending on the components of the instant messaging application installed, some components may be instantiated. For example if you install Access Manager and Access Manager SDK‡ (for Sun Java System Instant Messenger) to provide end user and service management, authentication and single sign on services, security manager may be instantiated.

B. P2P Backup and Recovery

Dinesh et al. [5], proposes an application for managing data backup and recovery in P2P. Such application can use our model as follows: It can implement the Peer Searcher to search for a peer that would be a suitable candidate for maintaining backups of a specific data item on the local peer; Implement a File Searcher to locate peers which are current online and holding a specific data item; The Properties Manager to keep track of the current resource utilization statistics. The Data Store to maintain copies of backup files on the local system; The Security Manager to maintain the security and privacy of data items so that the backed up data are only visible to authorized users of the system; The Schedule Manager to schedule tasks like backup; The Content Router to deliver the files from source to destination and the replication manager to backup data items.

C. File Sharing

Currently file sharing is probably the most common application of P2P technology. It is estimated that as much as 70% of the network traffic in the Internet can be attributed to the exchange of files, in particular music files [6]. In [4] it is estimated that more than one billion downloads of music file can be listed each week. File Sharing allows peers that have downloaded files in the role of client subsequently make them available to other peers in the role of a server. An Internet-based P2P file sharing system could use the Connection Control for connectivity, i.e. to joining and remaining in the overlay; the Data Store and Security manager for managing files, and a somewhat sophisticated Content router, peer and file searcher for supporting quality lookups and keyword search queries, but may have no need for the Scheduler and Properties manager. However for more efficient search and replication is may be used as suggested in [3]. In this case, the Replication Manager may be used to create and manage replica of files that are to be shared.

D. Storage Management

Storage management applications permit shared storage, management and use of data. A more interesting example to note is a Serverless Network Storage (SNS) proposed in [2] that runs over the Internet. For SNS, whose main function is to allow files to be saved or retrieved from any of the peers on the virtual network, can use our model as follows: Connection Control for creating and maintaining the underlying peer-to-peer network as well as service discovery, self-organization, and inter-peer communication. Security manager for securing all the data content in SNS, e.g., the XML messages and the file replicas, and provides the user and peer authentication. The Properties manager can be used to implement File Information Protocol (FIP), which provides an efficient and lightweight mechanism to maintain and transfer application-level information such as the file properties, the storage properties, and the query/response messages. FIP uses XML-formatted messages to exchange information with other peers [3].

E. P2P Digital Library

P2P digital library systems, however, need a very sophisticated File and Peer searcher and Content Router (that supports equality, keyword search, and range queries), a robust Replication Manager and so on. The main benefit of the model is that it allows us to plug in the appropriate instantiation of the relevant modules for each application, without having to redesign the entire system from scratch. As described in the introduction, the other benefit of our Model is that it enables us to put together a system that has more functionality than any existing system, solely by using existing components. Such a system is ideally suited for the digital library application described above.

IV. Conclusions and Future Work

The aim of our work was to put up a framework that enhances data management capability for P2P systems. We have proposed a modularized data management Model for P2P systems. The model we have proposed allows the use of existing algorithms for various modules with differing requirements depending on the requirements of the P2P application. The model also allows us to put together novel systems using existing components. We are going to implement this model with different applications including backup and recovery, file sharing, and storage management systems.

References

[1] Aspnes J., Shah, G.,"Skip graphs. In Fourteenth Annual Acm-Slam Symposium on Discrete Algorithms", January 2003, pp. 384–393, 2003.

[2] Cohen, E., Shenker, S.,"Replication Strategies in Unstructured Peer-to-Peer Networks". In Proceedings of ACM SIGCOMM, August 2002.

[3] Dabek, F., Kaashoek, Karge, D., Morriss, R., Stoica, I. "Wide-area Coperative storage with CFS". In Proceedings of the eighteenth ACM symposium on Operating systems principles, pp. 202 – 215, October 21-24, 2001, Banff, Alberta, Canada, 2001.

[4] Dinesh, C. Verma,"Using Peer-to-Peer Systems for Data Management; Peer-to-Peer Computing". The Evolution of Disruptive Technology, Idea Group Inc, pp. 66-78, 2005.

[5] Doskos,A., Xinghua, A.,"Peper: A distributed range addressing space for P2P systems". In LNCS, Springer Berlin, Vol. 2944/2004, pp. 200-218, 2004.

[6] Obeholzer, F., Stumpf, K.,(2004),"The effect of File Sharing on Record Sale – An Empirical Analysis", [Online] Available: http://www.unc.edu/~cigar/papers/FileSharing_March2004.pdf, 2004.

[7] Ottp, F., Ouyang, S.,"Improving Search in Unstructured P2P Systems", Intelligent Walks (I-Walks); IDEAL Sept, 2006, Burgos, Spain; Springer, LNCS 4224, pp. 1312-1319, 2006.

[8] Qin, L., Cao, P., Cohen, Shenkar, S.,"Search and Replication in Unstructured Peer-to-Peer Networks". Proceedings of the 16th international conference on supercomputing, NY, USA pp. 84 – 95, 2002.