Distributed Video on Demand Services on Peer to Peer Basis

Chris Loeser Peter Altenbernd Michael Ditze Wolfgang Mueller

C-LAB
Fuerstenallee 11
Paderborn, Germany
Loeser@c-lab.de

Abstract

Within this paper we propose architecture ideas on a distributed Video on Demand network basing on peer to peer technology. I.e. each peer offers video streams to other peers and may receive a video stream from another peer simultaneously. This results in an optimization problem depending on different factors which we approximate with the help of an simulation environment. Our approach bases on a peer to peer framework by Sun called Project JXTA which provides a set of protocols.

1 Introduction

The base idea of this proposal is to build up a local video-on-demand service without the need of a central storage-server. Such a scenario could occur in hotels. On one hand a central video server stores a couple of movies and offers them to set-top boxes distributed in the hotel rooms. There are two obvious disadvantages: For just a few parallel video-streams the server needs significant performance which results in high costs building up the whole infrastructure. Furthermore scalability seems to be a problem. On the other hand an alternative is the usage of peer-to-peer techniques: Each set-top box in the rooms gets a (relatively) small harddisk and each peer offers and requests movies at the same time. The movies are individual and no multicasting streams.

Up to now our work is in an experimental status. So far we have built up a video streaming testbed shown in Fig. 2. Furthermore the peer to peer software models are in a initial status (Fig. 3). At this point of time we mostly deal with the *P2P data placement simulation environment.*

The outline of this proposal is as follows: Within the next Section we describe general P2P techniques in the Internet followed by the P2P middleware *Project JXTA* by Sun Mic. Section 3 describes the general architecture of the P2P video storage network and finally gives a short overview of the data placement simulation environment.

2 Peer to Peer Networks

As the web continues to expand its scope to wireless devices and sensors, its growth is expected to explode to billions of new devices[2]. The popularity of the web has also demonstrated its limitations. Denial of service attacks have shown its fragility and lack of resilience to simple attacks. Services like DNS have created centralized dependencies and constrained the Internet's growth. The computing model on the web is primarily based on a client/server model where information and services are published at well-known and fixed locations (URLs). Such addressing, along with centralized web sites, have created single points of failure and bandwidth bottlenecks to popular sites: Hot spots become hotter while cold locations remain cold. A more decentralized and self-adapting computing model has been proposed by systems such as *Freenet*, *Gnutella*, and *FreeHaven* for addressing many of these limitations and have taken advantage of the increasing bandwidth, processing, and storage available on devices connected at the edge of the Internet.

The main concept of peer-to-peer computing is that each peer is client and server at the same time. Each peer may release and allocate

- **processing power**: like at *Seti at home* a bunch of peers performs a distributed computation.
- **data storage**: data is not owned by a particular member or server, but is passed around, flowing freely towards the end subscribers. When member demands for some data increase, more copies will be propagated and replicated within the community. Otherwise, fewer copies will be available as existing copies slowly disappear and are replaced by more popular data.
- **control**: each peer can offer the possibility of being controlled or illustrate monitored data. This is especially dedicated to sensors or small embedded devices.

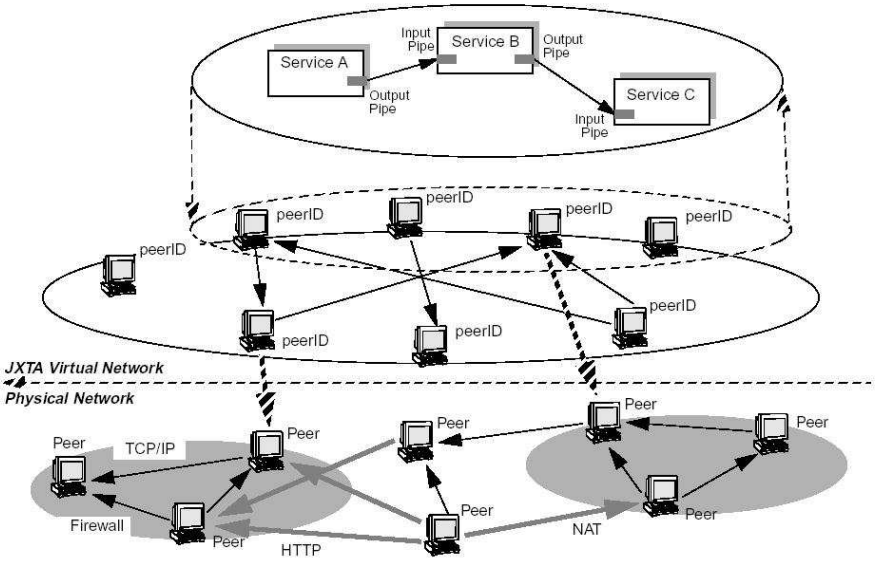

Figure 1: Jxta Virtual Network [10]

Peers have direct connection to other peers avoiding communication via mediating servers. The peer (or peer group) community as a whole is supposed to ensure the protection and persistency of data through its unique ability to adapt, resist, and protect data by scattering multiple copies within the community boundary. Community members tend to interact more heavily with their neighbors for searching and accessing information, reducing overall network traffic and data latency, and leading to a better utilization of available bandwidth. Chaining and delegation capabilities enable members to forward requests to their neighbors for searching data beyond their own view. Within a community, each member can access the entire community's knowledge. There is no single, centralized search engine; every member contributes to a search.

The described P2P architectures don't only make sense within the scope of the internet. They can easily be adapted on LANs/Intranets: also here it is reasonable to distribute content to avoid the need of centralized servers creating hot-spots.

Project JXTA

Project JXTA is an open-source project originally conceived by Sun Microsystems [8] and designed with the participation of a small but growing number of experts from academic institutions and industry. JXTA was initiated to standardize a common set of protocols for building P2P applications. Prior to this time, existing peer-to-peer systems were built in isolation, delivering a single type of service, and employing protocols incompatible with other services. For example, *Gnutella* defines a generic file sharing protocol and *Jabber* defines an instant messaging protocol, but none of these protocols are interoperable. Each system creates its own P2P community, duplicating efforts in creating software and system primitives required by P2P systems, such as managing the underlying physical network (e.g. dealing with firewalls, peer discovery, and message routing). Project JXTA attempts to define a generic network layer usable by a wide variety of P2P applications. It is designed to be independent of programming languages (e.g. C, C++ or Java), system platforms (such as the MS Windows, UNIX, Linux etc), service definitions (such as RMI and WSDL), and network protocols (such as TCP/IP or Bluetooth). The JXTA protocols have been designed to be implementable on any device with a network heartbeat, including sensors, consumer electronics, PDAs, appliances, network routers, desktop computers, data-center servers, and storage systems.

The Project JXTA protocols create a virtual network on top the existing physical network infrastructure of which services and applications are built (cf. Fig. 1). The developers designed this virtual network layer to be thin and simple, but with interesting and powerful primitives for use by services and applications. The main purpose of JXTA virtual network is to hide all of the complexity of the underlying physical network topology, and provide a uniform addressable network for all peers in the network. The JXTA virtual network allows a peer to exchange messages with any other peers independently of its network location (firewalls, NATs or non-IP networks).

It standardizes the manner in which peers discover each other, self-organize into peergroups, advertise and dis-

cover network resources, communicate with each other and monitor each other. The network transport layer is built of a uniform peer addressing scheme based on peer ID, relay peers that relay messages between peers, and a binary message format to transport binary and XML payloads.

3 Video-P2P-Network

Consider distributed video peers for storing and streaming videos where single video processing machines are connected to a network of distributed peers connected via switches and routers(cf. Fig. 2). This peer interconnection provides a virtual video server to the player application.

The management for the virtual server can be implemented by a JXTA-based middleware which ensures on the one hand *Quality of Service* for video streaming and manages on the other hand the storage and distribution of recorded videos. Each peer offers video content and may play videos from other peers by streaming simultaneously.

Details of streaming, rerouting and reallocation of storage may be completely managed and thus hidden by the middleware. The user only deals with one virtual video server and does not have to consider any network details.

In Fig. 2 you can see the raw architecture of our testbed. For our experiment we have connected peers to 3 LANs which is managed by a fast ethernet switch each. The LANs are connected to routers. Due to the experimental status the routers in the network are Linux machines. For our experiments, each peer locally stores a small number (up to now 4-7 videos) and the disk quota of each peer is limited to just a few GByte. Movies are additionally saved redundantly in order to have most possible fast access. How often movies are saved redundantly is presently evaluated by the data placement simulation.

Each peer holds a JXTA instance but also on the Linux-Router C a JXTA peer is running. Hoever here is also a so called *Rendezvous Service* working. Due to the fact that there are several subnets it is not possible for the peers to get aware of each other. Because of this the Rendezvous Server stores references of the individual peers and the peer content. However the data exchange just takes place between the individual peers. Though we need this kind of server this model does not stand in contrast to P2P models. Due to [2] these models can and do coexist.

Due to the fact that the number of peers in the network is not too large we decided to select IntServ supported Quality of Service [1]. The QoS support is achieved by the use of *traffic control (tc)* [7] [6]: The network interface is assigned to a *Queuing Discipline.* The traffic going out does not line up in a standard-FIFO rather in a *Weighted Fair Queue (WFQ).*

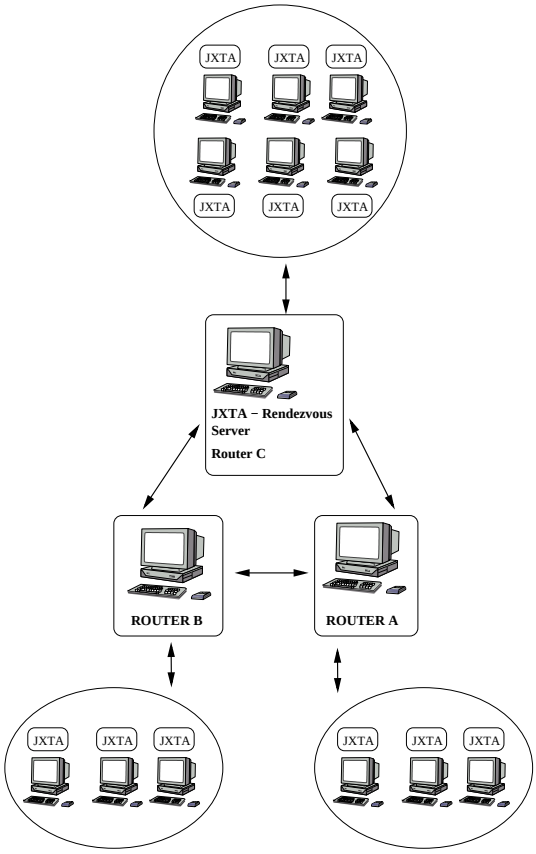

Figure 2: Architecture of the P2P-VoD testbed

3.1 Short Protocol Overview

In Fig. 3 you can see the protocol stack which shall be implemented on every peer. The *Player Application* directly interacts with the P2P instance in order to control the video stream. The P2P instance consists of several protocols which mostly are parts of the JXTA framework. Their detailed tasks are explained in [9] and [5]. Just the *Video Allocation Protocol* and the *Peer Discovery Protocol* shall be mentioned here.

- The Discovery Protocol (as part of JXTA framework) is responsible for the peer to get aware of the distributed content of the other peers. This presumes that each peer lets the rendezvous server know about its local content. This is realized by publishing *Advertisements.*

- When a user intents to view a movie the *Video Allocation Protocol* locates the most efficient source with the information delivered by the *Peer Discovery Protocol.* The RSVP protocol then reserves bandwidth along the route. We are presently implementing the VAP.

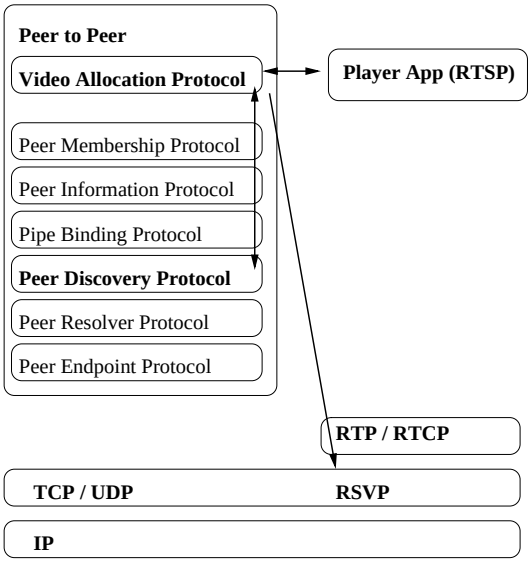

Figure 3: Protocol Overview

3.2 Simulation Environment for P2P Data Placement Strategies

In Fig. 4 you can see our Simulation Environment to evaluate different video placement strategies.
This optimization problem depends on demand distribution of the individual users, bandwidth, probability of peer failure and wether often requested movies are in the local subnetwork. The simulation applies the OSGI layer model. Besides different IP routing protocols (OSPF, RIP, DPS [3]) it is possible to reserve bandwidth resources simulating RSVP.
Later on it could make sense (when causing lower resource usage)if a serving video peer is changed at runtime to another one which is holding the redundant data.

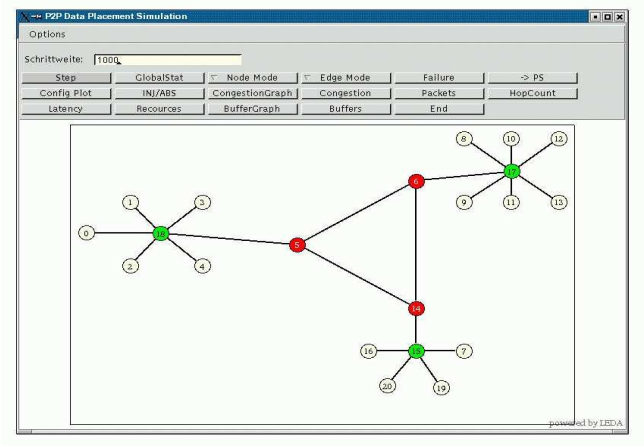

Figure 4: P2P Data Placement Simulation Environment

4 Conclusions & Future Work

Within this proposal we presented an architecture of an P2P-VoD-Network. Due to the fact that our work is still in progress we expect further results concerning the simulations within the next several month. The so gained strategies are then integrated in the Video Allocation Protocol. Moreover it might be useful to partition the video content.

Acknowledgements

Parts of the work described herein is funded by German Ministry of Technology and Research [4].

References

[1] G. Armitage. *Quality of Service in IP Networks*. Macmillan Technical Publishing, 2000.
[2] D. Barkai. *Peer-to-Peer Computing*. Intel Corporation, 2001.
[3] P. Berenbrink, A. Brinkmann, and C. Scheideler. Distributed path selection for storage networks. In *Proceedings of the 2000 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'2000), Las Vegas, USA*, pages 1097–1105, June 2000.
[4] Information Technology for European Advancement (ITEA). *Middleware for Virtual Home Environments*. `http://www.vhe-middleware.org`.
[5] S. Li. *Jxta. Peer-to-Peer Computing with Java*. 2002.
[6] G. Maxwell, R. van Mook, M. van Oosterhout, P. Schroeder, and J. Spaans. *Linux Advanced Routing and Traffic Control HOWTO*, 2001. `http://www.tldp.org/HOWTO/Adv-Routing-HOWTO.html`.
[7] S. Radhakrishnan. *Linux - Advanced Networking Overview*. University of Kansas, 2000. `http://qos.ittc.ku.edu/howto/`.
[8] Sun Microsystems. *Project JXTA*. `http://www.jxta.org`.
[9] Sun Microsystems. *JXTA Protocol Specification*, 2002. `http://spec.jxta.org/v1.0/docbook/JXTAProtocols.html`.
[10] B. Traversat, M. Abdelaziz, M. Duigou, J.-C. Hugley, E. Pouyoul, and B. Yeager. *Project JXTA Virtual Network*. Sun Microsystems, Februrary 2002.