# Level Aware Model for Peer to Peer Live Video Streaming

Zoran Kotevski and Pece Mitrevski
*University of "St. Kliment Ohridski", Boulevard 1th of May bb,7000 Bitola, R. Macedonia*
*E-mail(s): zoran.kotevski@uklo.edu.mk, pece.mitrevski@uklo.edu.mk*

**Abstract**. *Peer to Peer (P2P) live streaming paradigm is realized using several approaches based on combinations of network overlay formation (tree, mesh or hybrid) and data scheduling algorithms. Source driven (push based), Data driven (pull based) and Receiver driven approaches are widey utilized for modeling and designing P2P video streaming application for live video broadcast. In this paper we propose new P2P network overlay construction for live P2P video streaming that we intend to evaluate via well established formalism of Fluid Stochastic Petri Nets (FSPNs) and/or Discrete Event Simulations. Our model accounts for many issues of these systems such as: locality awareness, low control traffic, low playback delay, fast network recovery, incentives etc.*

**Keywords.** Modeling, performance analysis, discrete event simulation, P2P live streaming, fluid stochastic petri nets.

## 1. Introduction

Today, the use of Internet video streaming services is spreading rapidly. Web locations for live video broadcast attract more visitors every day. In the classical client/server system architecture the increase in number of clients requires more resources in manner of high bandwidth transmission channels with large upload rates. Since they are extremely expensive it results in a limited number of unicast connections that a server can support at a given time. In the early '90s it was expected that IP Multicast will be the natural technology to satisfy the requirements of large number of users with lower cost. However, lack of support for functionality of higher level, scalability issues and requirements for hardware Internet technology changes have prevented its wider deployment. In the last decade, the limited deployment of IP Multicast has motivated the science community to work in the field of new approach for Internet video streaming by use of Peer to Peer networking technologies. In this paradigm every user (peer, node) maintains connections with other peers and forms an application level logical network on top of the physical network. Video stream originates at a source and every peer acts as a client as well as a server forwarding the received video packets to the next peer. P2P logical network is used to deliver video without the need of broadband server connections. This class of "One to Many" video streaming is easy to deploy because P2P technology does not require network infrastructure support and offers scalability of resources having peers act as clients or servers. This leads to small bandwidth server being able to transmit video to hundreds of thousands of users. P2P networks have huge economical benefit in deploying and managing IP video streaming, but bring a lot of open issues and research challenges that need to be tackled. Besides the existing numerous applications, P2P video streaming systems are still in the early stages.

Two types of data circulate in these P2P networks: *control data* and *video data*. One algorithm uses the *control data* to construct and maintain an application level logical network (*control scheme*) that is usually in the form of: single spanning multicast tree [8], [10], [18], multiple multicast trees [1], [3], [5], [6], [9], [11], or mesh (unstructured) [2], [4], [7], [12], [13], [14], [16], [17], [21] where peers are organized in swarming or gossiping like environment. In hybrid systems combination of tree and mesh constructions is used [15], [19], [20]. The second algorithm takes care of *video data* dissemination and tends to deliver video packets on time and without losses (*data scheme*).

Combinations of *control scheme* and *data scheme* form several different approaches for video data dissemination. *Source driven* (Push) approach means that data scheme is built as a tree on top of the control scheme, where data is pushed down the tree from the root (source) to the leaves (peers). Typically source driven approach forms tree data scheme on top of tree control scheme [11], but some P2P protocols build tree data scheme on top of mesh control scheme such as Narada [2]. *Data driven* (Pull) approach is data oriented and doesn't form data distribution trees [12], [21]. Instead peers periodically exchange information about pieces of video data they possess, and every piece of data is explicitly requested. Data driven approach is usually related to mesh based systems. There are many efforts that combine these two approaches (Push/Pull) and present substantial results [15], [20]. *Receiver driven* approach, means that the data scheme is a tree rooted at the receiver [1], [7], [16]. Beside the formation of logical network the receivers also organize resources. This approach is usually related to scalable video coding techniques where different descriptions or layers are distributed through different branches. All these types of network formation and approaches for data dissemination have their pros and cons. Basically, tree based systems are more vulnerable to peer churn and mesh systems introduce more control traffic and playback delay. Since we analyzed all the technical characteristics of such systems in our previous work [26] this particular research is concerned with analysis of nework types and approaches and proposes novel concept for organization of peers in P2P logical network for live video streaming. Our new model uses different approach compared to existing ones and we beleve that it can address several nown issues of these systems while adding to its simplicity.

This paper is organized as follows. Section 2 gives detailed description of our model. Section 3 shortly speaks about the tools that we are going to use to evaluate the performance of the system and section 4 gives summary of our contribution.

## 2. Level – aware concept for P2P live video streaming

The many requirements of P2P live streaming systems contribute to complexity of the system making its design a challenging task. Basing our idea on Einstein's principle that "Things should be made as simple as possible, but not any simpler" we try to implement as many issues as possible to prototype simple, robust and resilient P2P live video streaming system.

The first step toward modeling a new system, is to follow an intuition. So, we propose a novel concept of Level Aware P2P Live Video Streaming solution presented in Fig. 1. The model has one control server, one streaming server and peers are organized in segments and levels. Hence, the basic units of consumption and distribution of video stream aren't the individual peers, but rather the levels, since we assume that they become self-aware. It means that the number of nodes that one peer can stream to, is dynamically determined. This isn't a classical tree nor mesh based system. The model tries to combine the better characteristics of the two previously mentioned in a hybrid model that tends to address many of the known issues of these systems.

The network formation algorithm tends to organize peers in separate levels. It's a logical decision since all connected peers cannot receive

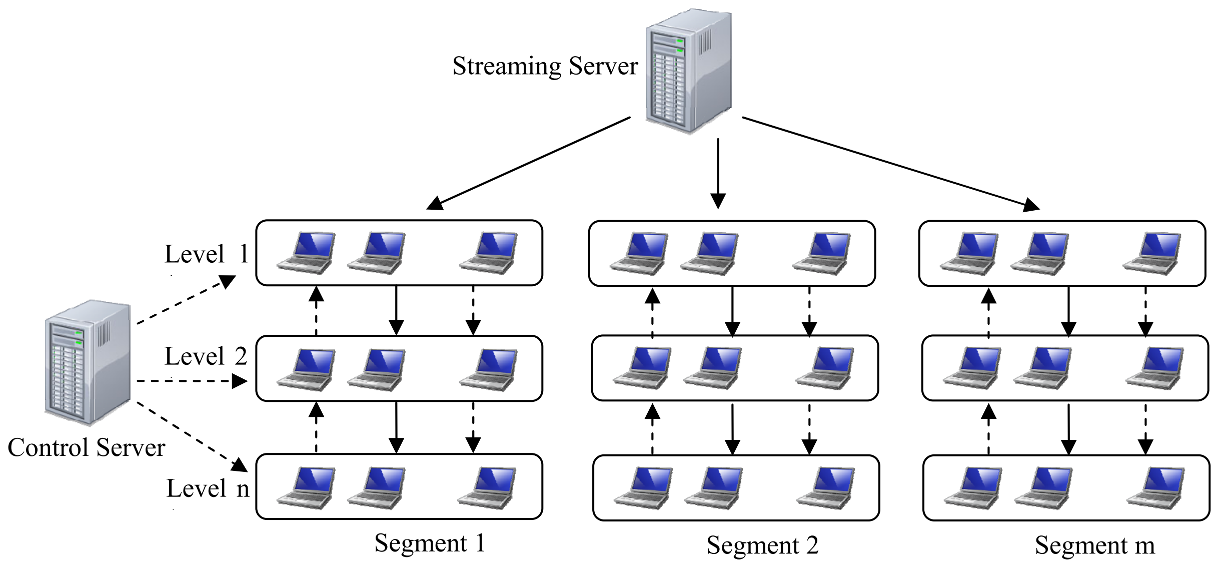


**Figure 1. Level Aware concept for p2p live video streaming**

video directly from the server and they must be organized in some kind of groups. This kind of organization looks also promising in one other aspect: it organizes peers in levels with same or similar video stream delay which can be useful in the second step of creating the Chunk Scheduling Algorithm. We define that the system has one streaming server which streams the video data only to the first level of peers. The first level forwards the stream to the second level and so on. Video data traffic is never exchanged between peers from the same level. Oppositely, they exchange only control messages.

Now we turn to addressing some issues beginning with Locality Awareness. Locality awareness is used for organizing peers according to their geographic location in order to preserve availability of International transcontinental links (ITLs) which are also expensive. Beside reduction of traffic load of these ITLs there is another benefit of organizing peers based on locality and that is lower video playback and stream delay. On the other hand, the number of clients in these logical networks can rise up to several hundreds of thousands and it would be very impractical to operate whole network as one gigantic net. Hence using the principle of the Roman Empire "Divide et Impera" we can resolve this first step with dividing this logical network in smaller segments. This network division is made in vertical rather than horizontal manner, and is presented in Fig. 2.

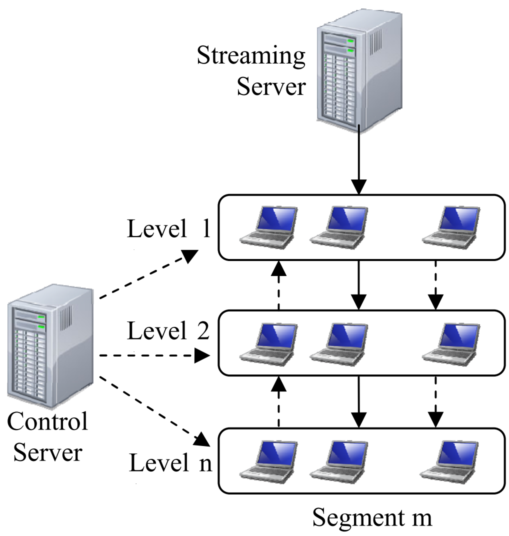


**Figure 2. One segment of the level aware P2P live streaming network**

In this network, peers from different segments never exchange any traffic between them and can be treated as separate small P2P live streaming networks. All segments belong to separate geographical areas, so locality concept is preserved. Depending on the Server's upload bandwidth capacity and number of separate segments, we can define the width of the Segment (how many peers in the first level of a segment can the server support).

Next we can concentrate on modeling one segment of the network and analyze its performance. This analysis will also stand for any segment of the network since all segments are identical. As we mentioned, one segment of the network looks like in Fig. 2. Even though the system's structure looks a bit like a logical tree structure, we must emphasize that more precise description would be logical level structure.

## 2.1 Joining of peers

Although these P2P networks are based on the basic communist principle where each member requires from the system according to its needs and contributes according to its capabilities, some incentives must be present. Peers with higher upload bandwidth capacity are more valuable to the system and therefore they must be positioned to better serve other peers with the benefit of receiving the video signal with lower stream and playback delay.

When a new peer intends to join the system it sends joining request to the control server which is responsible for network construction and maintenance. The control server (CS) first decides to which segment the new peer should be assigned. For this first decision it uses the IP address to determine client's geographic locality. Afterwards a new decision is made for the most suitable level. The choice for the level should be beneficial to the overall system, as well as to the new client. First, the peer's upload bandwidth capacity (UBC) needs to be determined. Whether the new peers report their UBC, or CS determines by test, it is used to classify the peers in classes of UBC. These classes have different priorities/restrictions for entering a level and this represents the first principle of built in incentive for better contributing peers. Peers with larger UBC can contribute more to the system and therefore these peers should be placed in the upper levels. Therefore, CS checks the levels for a vacancy from the top to the bottom. Oppositely, peers with less contribution to the system, in manner of UBC, are placed in the lower levels, so CS checks for a level vacancy from the bottom to the top.

Since the peers from one level exhibit similar playback delay and video stream delay it is only logical that these peers don't exchange video packets between themselves. They acquire video packets from the upper level (or the server) and forward the video packets downward (to the lower level). It is very important to mention that peers can move only between neighboring levels meaning that at a time one peer can move up or down only one level. This is necessary because moving up/down many levels at once will result in disruption of the video playback due to larger increment of video playback lag with every next level. It depends on the buffer capacity and the video chunks delay between levels.

## 2.2 Level switching

Peers churn. It means they join and leave at free will. Sometimes they notify other peers for their intention to leave and sometimes they don't (infrastructure/power failure). In any case when a peer leaves its level may create lack of level UBC needed to support the lower level. This peer can be replaced by a new one if there is one waiting, and if there isn't one, the level acquires one from the lower level. So, when the level acquires a peer from the lower level it is a must! Now the lower level needs to decide which peer it should send to the upper level. The decision is always for the peer with the highest UBC which represents the second principle of incentives where peers with higher UBC travel up the segment level by level. In another scenario, when replacement peer has lower UBC then the replaced one, and the upper level available bandwidth is consumed, the level orders the lower level to drop one peer down one level. Oppositely to previous case, peer with the lowest UBC is the one that must go down one level. This incentive makes lower contributing peer to travel down the segment level by level. As mentioned before, this way the network constantly reconstructs itself, in a network where upload bandwidth capabilities of peers are greater in the upper levels and degrade with moving down the levels. If free riders occur, they are placed or pushed at the last level of the segment, rather than refused, so no special action is needed (built in incentive mechanism).

Because we have implemented level awareness we can address the level as unique part of the network. The level always starts sending packets upon request. It isn't pull based system, but rather pull-push approach where the streaming session is initialized with request and after that the packets are sent continuously without re-requesting them. In special occasions when some packet is lost the receiving peer sends request with the number or numbers of missed packets (if there is time to receive them). The decision about swapping levels can be made at the level, or sometimes can be made by the control server.

## 2.3 Model Summary

Peer churn initiates constant reorganization of the network and tends to generate video stream disruptions. To cope with these disruptions many models have implemented someway of splitting the video stream in many sub streams such as: layered video coding (LVC) or multiple description coding (MDC). In all these systems the video signal is split in more description which are dependently (LVC) or independently (MDC) decodable. Even though these efforts present promising results we will stay by our idea of simple system and work on a model with one video description.

Theoretically, compared to tree based systems, our model is to be more resilient to peer churn and variations in peer's UBC, since the number of nodes that one peer streams to, is dynamically determined. Compared to Mesh based systems, it's designed to exhibit lower control traffic, lower stream and playback delay and require smaller buffer. Also, chunk scheduling should be easier to implement in comparison to both, tree and mesh systems.

From a practical point of view, the strength of this model would be its simplicity filled with a lot of built in mechanisms, such as:

1. Better network management, since the huge network is segmented into smaller logical networks
2. Locality awareness principle is easily implemented because in one segment the members are peers from neighboring geographic areas.
3. Preservation of expensive ITLs.
4. Video stream delay is kept on a minimal level since video data isn't sent back and forth via previously mentioned ITLs.
5. Video playback delay is kept on a minimal level
6. Built in incentives for peers that contribute more resources since they go up, and low contributing peers go down the levels.
7. Built in admission control

## 3. Tools for modeling and performance analysis

The idea of a modeling is simple. Why building a system, spending a lot of time and sometimes large amount of money, and then test its performance when its performance can be calculated using known modeling formalism that will bring good insight in the system's behavior. Mathematical models are used widely in a lot of areas and as we researched P2P streaming we found only several efforts that speak about modeling and performance analysis of such systems. Oppositely to modeling P2P live video streaming systems, there is large number of created applications that are tested afterwards usually on PlanetLab. Here lays our idea of contribution to the area of modeling and performance analysis of P2P live video streaming systems.

### 3.1 Fluid Stochastic Petri Nets

The modeling formalism of FSPNs was firstly introduced in 1993 by Trivedi et al. [22] as a tool for modeling and analysis of systems that are characterized as being dynamic, concurrent, parallel, distributed and stochastic [23], [24], [25]. Even though FSPNs are being used for almost two decades now, until this date there is no prior work on using FSPNs as a tool for modeling and analysis of P2P live video streaming systems. Since P2P live streaming systems and their behavior can be described as processes with alternately changing states, the natural conclusion is that such systems can be described with FSPNs.

The decision to use fluid stochastic petri nets is on the parallel characteristics with P2P live streaming networks. If we represent peers with discrete tokens moving in and out of discrete places and video bits as fluid that flows through fluid arcs, then we have numerous possibilities to calculate and analyze system's performance.

The solution of the FSPN models can be performed analytically, numerically or by simulation.

### 3.2 Python and SimPy

The second approach that we intend to use is discrete-event simulation (DES) using SimPy. SimPy is DES package based on standard Python programming language.

*Python* [27] is a general-purpose high-level, interpreted and most important powerful programming language. It supports multiple programming paradigms as object oriented, imperative and functional programming. It features a fully dynamic type system and automatic memory management, similar to that of Scheme, Ruby and Perl. Like other dynamic languages, Python is often used as a scripting language, but is also used in a wide range of non-scripting contexts. Python is an interpreted language and has a multiplatform support.

*SimPy* [28] stands for *Simulation in Python.* It's an object-oriented, process-based discrete-event simulation language based on standard Python, developed at Victoria University of Wellington. It provides the modeller with components of a simulation model including processes for active components like customers, messages and vehicles, and resources for passive components that form limited capacity congestion points like servers, checkout counters and tunnels. It also provides monitor variables to aid in gathering statistics. Random variables are provided by the standard Python random module.

Both packages are platform independent, open source free software released under compatible GNU GPL licenses which makes them highly suitable and available for the research community.

## 4. Summary of contribution

In this paper we have presented several characteristics of P2P live video streaming systems with an accent on the network formation and different approaches for data dissemination. We also presented a new concept of level aware system for P2P live video streaming. At the end of the paper we presented our future work and we gave short description of the tools that we are going to use for performance analysis of our level aware model as well as its comparison to other well established concepts.

## 5. References


[1] McCanne S, Jacobson V. Receiver Driven Layered Multicast. In: ACM SIGCOMM, Stanford, California; 1996

[2] Chu Y, Rao SG, Seshan S, Zhang H. A Case for End System Multicast. In: ACM SIGMETRICS, Santa Clara; 2000

[3] Jannotti J, Gilford DK, Johnson KL, Kaashoek MF, O'Toole Jr.JW. Overcast:

Reliable Multicasting with an Overlay Network. In: Proceedings of OSDI, San Diego; 2000;
[4] Rowstron A, Druschel P. Pastry: Scalable, Decentralized Object Location and Routing for Large Scale Peer to Peer Systems. In: 18th IFIP/ACM Int. Conf. on Distributed Systems Platforms, Heidelberg; 2001
[5] Castro M, Druschel P, Kermarrec AM, Nandi A, Rowstron A, Singh A. SplitStream: High Bandwidth Content Distribution In Cooperative Environments. In: 19th ACM Symposium on Operating Systems Principles, New York; 2003
[6] Tran D A, Hua KA, Do TT. ZIGZAG: An Efficient Peer to Peer Scheme for Media Steaming. In: IEEE INFOCOM 2003, San Francisco; 2003
[7] Jiang X, Dong Y, Xu D, Bhargava B. GnuStream: A P2P Media Streaming System Prototype. In: IEEE Int. Conference on Multimedia and Expo, Baltimore; 2003
[8] Kostic D, Rodriguez A, Albrecht J, Vahdat A. Bullet: High Bandwidth Data Dissemination Using an Overlay Mesh. In: ACM Symposium on Operating Systems Principles, New York; 2003
[9] Hefeeda M, Habib A, Botev B, Xu D, Bhargava B. PROMISE: Peer-to-Peer Media Streaming Using CollectCast. In: ACM Multimedia, Berkeley, 2003
[10] Dobuzhskaya M, Liu R, Roewe J, Sharma N. Zebra: P2P Multicast for Live Streaming Video. Technical report, Massachusetts Institute of Technology; 2004
[11] Setton E, Noh J, Girod B. Rate-Distortion Optimized Video Peer-to-Peer Multicast Streaming. In: ACM P2PMMS, Singapore; 2005
[12] Pai V, Kumar K, Tamilmani K, Sambamurthy V, Mohr AE. Chainsaw: Eliminating Trees from Overlay Multicast. In: 4th Int. Workshop on P2P Systems, New York; 2005
[13] Zhang X, Liu J, Li B, Yum T-SP. CoolStreaming/DONet: A Data Driven Overlay Network for Efficient Live Media Streaming. In: IEEE INFOCOM, Miami; 2005
[14] Pianese F, Keller J, Biersack EW. PULSE, a Flexible P2P Live Streaming System. In 25th IEEE Int. Conference on Computer Communications, Barcelona; 2006
[15] Wang F, Xiong Y, Liu J. mTreebone: A Hybrid Tree/Mesh Overlay for Application Layer Live Video Multicast. In: International Conference on Distributed Computer Systems, Toronto; 2007
[16] Magharei N, Rejaie R. PRIME: Peer to Peer Receiver Driven Mesh Based Streaming. In: IEEE INFOCOM, Anchorage; 2007
[17] Covino F, Mecella M. Design and Evaluation of a System for Mesh-based P2P Live Video Streaming. In: ACM MOMM, Linz; 2008
[18] Tu X, Jin H, Liao X. Nearcast: A Locality-Aware P2P Live Streaming Approach for Distance Education. In: ACM Transactions on Internet Technology; 2008
[19] Lu Z, Li Y, Wu J, Zhang SY, Zhong YP. MultiPeerCast: A Tree-mesh-hybrid P2P Live Streaming Scheme Design and Implementation based on PeerCast. In10th IEEE Int. Conf. on High Performance Computing and Communications, Dalian; 2008
[20] Li Z, Yu Y, Hei X, Tsang DHK. Towards Low-Redundancy Push-Pull P2P Live Streaming. In: ICST QShine, Hong Kong; 2008
[21] Chakaereski J, Frossard P. Dellay-Based Overlay Construction in P2P video Broadcast. In: IEEE ICASSP, Taipei; 2009
[22] Trivedi KS, Kulkarni VG. FSPNs: Fluid Stochastic Petri Nets. In: Proceedings of the 14th Int. Conf. on Application and Theory of Petri Nets, Springer – Verlag, London, 1993
[23] Murata T. Petri Nets: Properties, Analysis and Aplications. In: Proceedings of the IEEE, 1989
[24] Horton G, Kulkarni VG, Nicol DM, Trivedi KS. Fluid Stochastic Petri Nets: Theory, Applications and Solutions Techniques. In: European Journal of Operational Research, Elsevier Science, 1998
[25] Wolter K, Horton G, German R. Non-Markovian Fluid Stochastic Petri Nets. Technical Report, Berlin, 1996
[26] Kotevski Z, Mitrevski P. A Modeling Framework for Performance Analysis of P2P Live Video Streaming Systems. In: Proc. of the International Conference ICT-Innovations-10, Macedonia, Springer; 2010
[27] Python Programming Language http://www.python.org/
[28] SimPy – Simulation in Python http://simpy.sourceforge.net/