**DIstribution Of Multi-view Entertainment using content aware DElivery Systems**

**DIOMEDES**

**Grant Agreement Number: 247996**

# D4.5

# Report on results of integrated MD-SMVD and P2P system

| Document description | |
|---|---|
| Name of document | Report on results of integrated MD-SMVD and P2P system |
| Abstract | Most of the results of the studies on video streaming in P2P networks can be categorized under two major concepts. The first aspect is the application layer framing which corresponds to formation of data flow in a way that each received byte is useful in decoding (video/audio) process. The second aspect is the adaptation of the payload that is delivered from the application layer to the network layer, which we can simply call as adaptive streaming. This document is to describe the steps taken to enable application layer framing and adaptive streaming in the P2P architecture of project DIOMEDES. The document also includes the streaming tests to evaluate the performance of adopted P2P architecture. |
| Document identifier | D4.5 |
| Document class | Deliverable |
| Version | 1.0 |
| Author(s) | C.Göktuğ Gürler, Sedef Savas (KU), Erhan Ekmekcioğlu, Hemantha Kodikara Arachchi (UNIS) |
| QAT team | Haluk Gokmen (ARC), Tamir Adari (OPTEC) |
| Date of creation | 11/08/2011 |
| Date of last modification | 12/31/2011 |
| Status | Final |
| Destination | European Commission |
| WP number | WP4 |

# Table of contents

# List of Figures

# List of Tables

# 1 INTRODUCTION

## 1.1 Purpose of the document

The document is to provide the details of the DIOMEDES P2P overlay architecture that is adopted in project DIOMEDES. First, the details of P2P architecture are introduced along with its differences from the currently available solutions like NextShare (software developed by P2PNext [1]), Tribler [2] and SARACEN project [3] (EU-ICT 7$^{th}$ Framework Project for scalable video distribution). Next, the detailed information about adaptation strategies and how they are implemented are provided. This work includes the rate adaptation (SNR scaling), view adaptation (stream selection) and MDC implementation. Finally, we provide the results of the video streaming tests to evaluate the performance of adaptive streaming capabilities.

## 1.2 Scope of the work and scenarios considered

The full description of P2P architecture is provided that is implemented for the final demonstrator. This includes the cooperative work of Adaptation Decision Engine Module (ADEM) and P2P streaming software. Tests are performed in controlled LAN environment with a P2P swarm that consists of multiple nodes and a single main seed server. In these test, P2P-only scenarios are focused on, in which a peer with no DVB-T reception aims to watch multi-view content.

## 1.3 Objectives

The objectives of the proposed workflow is summarised as:

- Achieving adaptive streaming with the consideration the QoE in multi-view video.
    - Priority ranking carried out by the adaptation decision engine is used by the P2P streaming client to drop/add streams with respect to its particular importance to the overall QoE of the service at times of bandwidth scarcity.
- Achieving a scalable P2P overlay architecture by the increasing number of peers.
    - DIOMEDES has hybrid P2P architecture and there are a number of main seed servers along with the peers that assist the content delivery when P2P fails.
- Reducing the average bandwidth requirement on the main seed servers as low as possible by forcing the peers to take part in active media distribution.
    - Intelligent chunk scheduling algorithms are designed in the P2P streaming clients to decrease the workload on the servers.
- Achieving robust content delivery by using MDC scheme
    - DIOMEDES P2P uses selective and scalable MDC schemes to augment the data delivery while keeping the redundancy as low as possible

## 1.4 Structure of the document

This report is organized as follows: Chapter 2 explains the P2P overlay specifications by providing detailed information about the implementation of adaptive streaming in DIOMEDES. Section 2.1 provides an overview of DIOMEDES P2P approach. Section 2.2 describes the challenges and the adopted methods to overcome the described challenges. Section 2.3 provides the steps that are applied in order to create the multi-view content that is ready for P2P distribution at the server side. Section 2.4 presents a short description of how the system is initialized at the receiver (peer) side using tracker servers. Section 2.5 provides the details on adaptive video streaming methods and the adopted MDC scheme. The Section 2.6 introduces a new policy to keep the bandwidth requirement on the server side as low as possible. Section 2.7 provides more details about the implementation plan in the level of source code. And finally in Section 2.8, we underline the differences of DIOMEDES P2P from the existing architectures. Chapter 3 is dedicated to tests that we perform to evaluate the methodologies described in Chapter 2. In Section 3.2, we present the video coding results to explain the importance of variable bitrate coding (VBR) and variable chunk sizes. The following two section (3.3 and 3.4) are the tests to evaluate the performance of adaptive streaming mehtods in from QoS and QoE aspect respectively. Following that in Section 3.5, we present the results about using "intelligent chunk picking" algorithms to minimize the server load. Section 3.6 is for the evaluation of the adopted MDC scheme. Finally, in Chapter 4, we draw our the conclusions about this document.

# 2 *DIOMEDES* P2P OVERLAY SPECIFICATIONS

## 2.1 Overview of DIOMEDES P2P

The DIOMEDES P2P follows the torrent-based approach in which the content is first split into multiple sub-pieces (chunks) and then distributed among peers that are connected to each other in a mesh-based topology. The peers do not have parent-child relation unlike tree-based solutions and they can send and receive data from/to any other peers in the network (swarm). However, we belive that it is not possible to achieve a fully optimized P2P video streaming scheme when the system is strictly backwards compatible. Therefore, there are also significant differences with the BitTorrent specifications and other P2P distribution projects that are compatible with it. These differences are introduced throughout the rest of this section.

### 2.1.1 DIOMEDES P2P as a Hybrid Solution

The DIOMEDES P2P is a hybrid video distribution system in two aspects. First, it has hybrid network architecture, since DIOMEDES can use both the DVB and the IP channels. In its hybrid mode, the DVB channel delivers stereoscopic 3D video and standard audio, whereas the IP channel augments the user experience by providing additional views and object based audio. This condition is referred to as *dual-network layer*. Naturally, it is also possible to use only the IP channel to receive the content if the DVB signal is not available. In this case, the views that are transmitted over DVB are sent over the IP channel if the available network resources are adequate.

The second hybrid aspect of is DIOMEDES is the P2P overlay architecture. In DIOMEDES, data distribution over IP is performed mainly by P2P overlay networks, but it is also assisted by servers that are called *main seed servers (MSS)*. A MSS uploads video segments (chunks) to sub-list of peers and they assist the data dissemination when P2P overlay fails to deliver the content in a timely fashion. This condition is referred to as *hybrid delivery architecture*.

These two aspects make DIOMEDES architecture unique, when compared to the currently available solutions.

#### 2.1.1.1 Video Streaming using Dual-Network Layer

When streaming is performed in synchronization with the DVB channel, the there has different set of rules compared to simple video streaming over IP. For instance, in some cases the initial video buffering, which is a widely adopted technique and present in almost all streaming solutions may not be available because when the play-out deadlines are determined by the signal from the DVB channel, the streaming solution that operates over IP cannot control the duration of the pre-buffing. Similarly, in most cases streaming solutions can use freeze, re-buffer, and restart playing combination when there is fluctuations in the available network resources and this is also not available in DVB-synchronised streaming solutions, instead DIOMEDES P2P has to resynchronize with the DVB at a later position. For this purpose the P2P engine constantly receives updates from the AV Synchronization Module and tries to determine the chunks to download in order to be in synchronisation with the DVB channel. If it loses synchronicity, it has to jump to a chunk that is further away in time to re-establish synchronous play-out.

#### 2.1.1.2 Balanced distribution load over IP using Main Seed Servers

The second hybrid aspect of the DIOMEDES P2P is the adopted overlay architecture. In DIOMEDES P2P, the data dissemination is mostly performed by the peers. However, for the initial seeding DIOMEDES P2P use main seed servers. Moreover, the main seed servers can assist the video delivery by directly forwarding video chunks to the clients (peers) if the P2P video distribution has failed.

Although can be helpful, the main seed server introduces new challanges for the DIOMEDES P2P. In order to provide a scalable solution, DIOMEDES P2P solution has implemented methods to minimize the data dissemination from the MSSs. These methods are called `intelligent chunk scheduling` and described in Section 2.6.

## 2.2 Definition of the Problem and Adopted Policies

### 2.2.1 Challenges

The challenges that are encountered during the development of DIOMEDES P2P can be listed as follows:

- Working in a hybrid environment is difficult because a remedy for one aspect may not be applicable because of the other. For instance, the buffering mechanism that is widely adopted in video delivery systems over IP may not applicable if the DVB play-out deadline is very close. For such cases, an algorithm that skips a number of video chunks to resynchronize with the DVB channel is developed.
- QoE aware multi-view video delivery is significantly more complex than legacy mono-view video streaming solutions. In legacy video streaming, the prioritization of video streams is straightforward (e.g. base quality layer is more important than the enhancement layers), whereas in the case of multi-view, the prioritization of streams is:
    - more complex, since it is difficult to prioritize streams (e.g., colour versus depth map)
    - dynamic based on the 3D scene and user viewing preferences.

  Therefore, in DIOMEDES-P2P, the prioritization of the streams can change dynamically and ranked by an external module that focuses on this task only (Adaptation Decision Engine Module).
- The servers that assist the video distribution can easily become overwhelmed by the requests from peers. To prevent such problem, a new piece picking policy is adopted that tries to minimize the burden on main seed servers.
- Application layer framing is a well-known concept that is to frame the data transmission boundaries such that upon the loss of a single unit, minimum data loss has occurred. Most of the P2P delivery solutions that are available today has either neglected this issue or implemented sub-optimal solutions using data padding. For DIOMEDES P2P, we have investigated the means of building a transmission mechanism that will honour the application layer-framing concept.

## 2.3 Content Generation at the Server Side

The content preparation is performed in series of steps that is summarized in Figure 1. The steps depicted in Figure 1 are applied to every viewpoint in the multi-view video. This section provides the details of these steps and describes how the content is prepared to enable adaptive multi-view video streaming in a secure way.

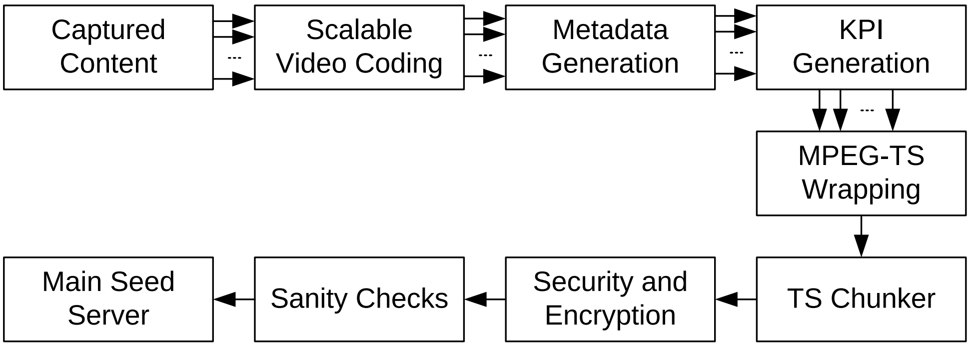


Figure 1: Steps of Content Preparation in DIOMEDES

### 2.3.1 Scalable Multi-view Video Coding

As reported in previous deliverables under WP4, i.e., D4.1 (Specifications of the encoder/ decoder software for the MD-SMVD and the spatial audio codec), D4.3 (Interim report on the developed audio and video codecs) and D4.4 (Final report on the developed audio and video codecs), Scalable Video Coding (SVC) amendment of MPEG-4 Part 10/ H.264 Advanced Video Coding (AVC) video coding standard has been utilised in encoding multi-view videos with depth maps. Readers can refer to these deliverables to learn the basic principles behind using SVC in achieving the desired adaptation capabilities. Here we make a summary of the highlights of scalable multi-view video coding approach.

The scalability dimensions exploited in DIOMEDES are quality (SNR) scalability and inherent viewpoint scalability. Spatial scalability feature is not preferred because of the excessive overhead required. The additional overhead is caused by the periodical insertion of special switching NAL units into the bit-stream, which need to be parsed by decoders to carry out scaling. Furthermore, since only HD video streaming is planned in DIOMEDES and since for the hybrid broadcast-broadband scenario only single resolution video is broadcast over DVB, existence of multiple spatial resolution representations is not sought. On the other hand, adding extra SNR refinement layers on top of the base quality layer does not necessitate extra signalling via new NAL units. The decoder progressively refines the reconstructed video frame with the enhancement layer NAL unit. In addition, the encoder is modified to selectively assign different quantisation parameters to each macroblock in the same layer according to an external visual attention model. This way, it is ensured that most visually salient features are kept in the base layer with higher precision, whereas less salient features' quality is refined in successive enhancement layers. Readers can refer to D4.4 for more information on the visual attention adaptive scalable multi-view video coding structure. Since the elementary streams of different views are generated in different instances, they are completely independent from each other and can simply be selectively streamed, without affecting each other's decoding process. This means 100% view scalability.

Temporal scalability is an inherent result of the employed hierarchical B-frame prediction framework. However, temporal scalability is not functional in the scope of DIOMEDES because of the chunk structure as a result of application layer framing. Nevertheless, temporal scalability is not as effective as SNR scalability in terms of bit-rate adaptation. Each Group of Pictures (GOP) starts with an Intra-coded frame (I-frame) to comply with DVB's broadcasting requirements. The rest of the frames are bi-directional predictive encoded (B-frames). The lowest temporal layer carries the I-frames, which consumes a significant amount of the bit-rate. Hence, dropping the higher temporal layers does not provide remarkable adaptation range. Furthermore, since the side effects are mostly content dependent (e.g. for fast changing scenes with a lot of motion activity, temporal scaling is lesser desired than other adaptation options), temporal scaling cannot be considered as a permanent adaptation option. Therefore, this is sacrificed with the adopted application layer framing.

The multi-view content is encoded using two SNR layers (one base layer and one quality enhancement layer) for all views. All views have similar average bit-rates. This way, a sensible bit-rate adaptation range is reached (i.e., target bit-rate is adjusted by discarding/ truncating one layer stream's packets). Depth maps are encoded using a single layer only. Variable bit-rate (VBR) coding is used for all views/ depth maps included, which has the advantage of producing a better quality-to-space ratio compared to constant bit-rate (CBR) coding, where unnecessary byte/ packet stuffing is done unless low bit-rate coding is used. Stuffing becomes more prominent especially in I-B-B- … I-B-B-… type coding structures, where a lot of stuffing packets are needed before each B-frame to balance the instantaneous rate differences. Variable bit-rate coding is particularly useful for bandwidth constrained and best effort networks, such as IP. Furthermore, chunk based P2P streaming (details are described in Section 2.3.4.2) produces almost equal sized media chunks, all of which contain equal number of Intra and bi-predictive encoded frames. On the other hand, videos that are additionally sent over DVB-T2 are stuffed with NULL transport stream packets to produce constant bit-rate streams as implied by the DVB standard.

### 2.3.2 Key Performance Index (KPI) Transmission Inside Video Stream

Key Performance Index (KPI) values are used to estimate the viewers' QoE for the respective video service in client terminals. Accordingly, they help the adaptation decision making engines to decide whether to continue to stream a particular stream or not, taking into account user's preferences and the usage context. Readers are recommended to refer to D3.4 (Report on the Quality of Experience model and the audio and visual attention models) to review the video QoE modelling in the context of DIOMEDES, as well as the details of KPI metadata models. In summary, there are 2 KPI values, namely Image Quality (IQ) and Depth Perception (DP), which are calculated during the encoding stage per each successive GOP. These two values are then sent along with the encoded video data to the decoder. These two values are combined in a single QoE metric (the average QoE estimate) for the respective decoded GOP in the decoder. In addition to these two values, the Z-directional motion activity in the decoded scene is used to properly weigh the significance of the depth perception in the overall QoE. Because, the significance of the depth perception is proportional to the objects' position with respect to the scene level as well as the motion perpendicular to the screen plane. For decoders' convenience and in order to not to incur additional processing load on them to hinder real-time performance, this value is also calculated in the encoder side per each GOP to be sent along with the other two KPI values.

The bit-stream is formed, such that each GOP is initiated with two special NAL units that refer to Sequence Parameter Set (SPS) and Picture Parameter Set (PPS). They are inserted at the beginning of each GOP to let the decoders recover the broadcast stream at any point aligned with GOP borders. The four KPI values (IQ and DP values for both the base layer and the enhancement layer) and one Z-directional motion value are enclosed in a special Scalable Enhancement Information (SEI) type NAL unit. The payload type of this SEI NAL unit is selected as *user unregistered data*, which is specified in the standard as the means of transmitting user/ service defined parameters that are irrelevant to the actual decoding task. This SEI NAL unit is inserted right after the PPS NAL unit and before the prefix NAL unit of the base layer VCL NAL unit of the I-Frame. The NAL unit is parsed by the decoder and the KPI values are directed to the Adaptation Decision Engine.

### 2.3.3 Encapsulation using MPEG-2 Transport Stream

After the scalable video elementary streams are generated and the computed KPI values are embedded into the elementary stream, all quality layers and depth maps of each camera view are split into different streams to get encapsulated into MPEG-2 Transport Streams (TS). Transport streams consist of fixed length transport packets (188 bytes each). Each Access Unit (AU) that consists of one full base layer VCL NAL unit and one full enhancement layer VCL NAL unit is split to two blocks, one containing the base layer NAL unit and the other containing the enhancement layer NAL unit. Each block is then wrapped into individual Packetised Elementary Streams (PES) packets. Then, all PES packets are further fragmented into smaller chunks to form the transport streams of the base layer, enhancement layer and the depth map. Transport Stream encapsulation is a must for multimedia transmission over DVB. In order to not to introduce unequal transmission formats in the hybrid delivery system, as well as to ease the synchronisation of the streams that are delivered over DVB and P2P using a single clock reference, MPEG-2 TS encapsulation format is deployed for the video and audio transmission over P2P too. MPEG-2 TS encapsulation brings ~4.3% extra overhead in contrast to pure RTP encapsulation that is a more popular encapsulation method for IP streaming and brings ~2% extra overhead. Nevertheless, the difference is not significant that would hinder the transport performance. Figure 2 depicts the encapsulation procedure for one of the camera views.

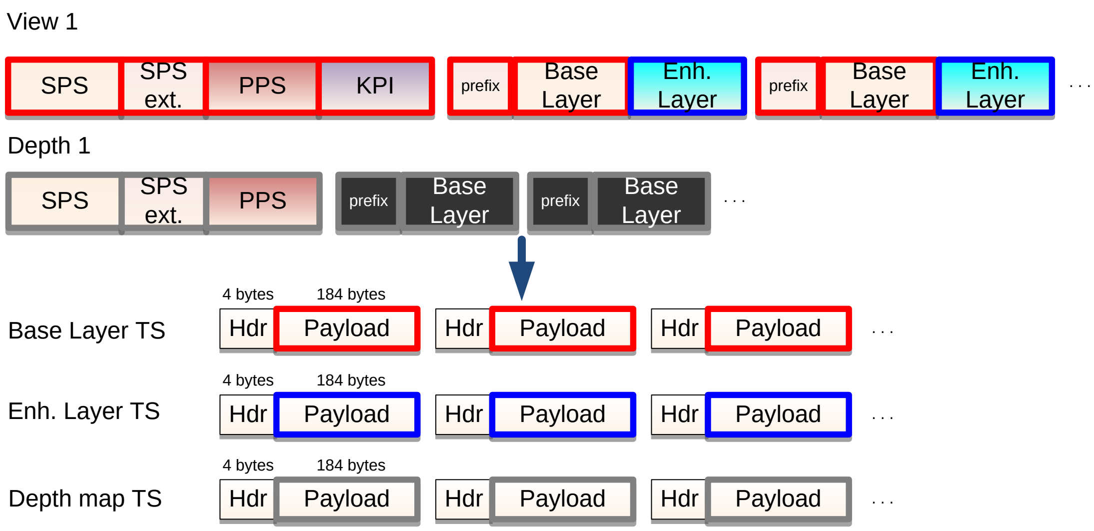


Figure 2: Splitting SVC stream into multiple Transport Streams

Each transport stream has a unique identifier, referred to as PID. These are also the identifiers used by the chunker (see Figure 1) to differentiate different video streams in the multiplexed stream.

### 2.3.4 Content Metadata Format

A content metadata file is generated during audio and video coding and encapsulation process. The content metadata file is used in the subsequent processing stages of the content server and the security server. At the same time, the metadata involved within this file is inserted into a special chunk which is exchanged with downloading media consuming peers to let them be able to properly decode and display the content. The contents are registered in the content server with the title and the globally unique identifier (GUID) and tagged with scene type and sub-type, which are stored in the content metadata file. In order to let the content metadata be widely accessible by the involved processing modules, the format is decided to be in extensible mark-up language (XML). Apart from the descriptive title, unique identifier and tags, multi-view capture specific technical metadata is involved in the content metadata file. This metadata is used in the video and audio player units of the client terminals to correctly render the 3D scene. The number of available camera views, extrinsic and intrinsic parameters of each camera, cameras' IDs (to successfully identify each camera in the corresponding multi-camera set), scene setup (cameras' positions with respect to each other and with respect to the centre of the scene), involved streams' PIDs and types (e.g. base, enhancement, depth, audio) and intended delivery paths (e.g. DVB or P2P) are other elements of the content metadata file.

#### *2.3.4.1* Chunk Mapping

Packetisation of multimedia content considering the underlying network infrastructure has a critical impact over the performance of any streaming solution. For a Torrent based P2P system, packetisation corresponds to formation of video chunks. In BitTorrent protocol, all the chunks have a fixed size, which is based on the total size of the shared content. This approach is not very suitable for video coding applications, because the rate may fluctuate over time. Moreover, chopping the bit-stream at fixed locations generates chunks that are not independently usable, since the actual frame data and the required header information may become separated. Therefore, variable sized chunk generation approach is adopted using Group Of Picture (GOP) boundaries as separation points. It is possible to use multiple GOPs to create a single chunk, if the payload size is too small.

#### 2.3.4.2 Chunk Generation using SVC bit stream

The SVC stream starts with non-VCL NAL units, similar to H.264/AVC. When slice mode is disabled, there is one base layer NAL unit and one enhancement layer NAL unit for each frame. The bit-stream containing only the base layer can generate a video frame at a lower quality, even if the enhancement layer NAL units are not received, indicating that enhancement layer NAL units are discardable. Therefore, each layer is split into separate chunks. This method provides *discardable* chunks (enhancement layer chunks) that can be omitted if the available network resources are not high enough (see Figure 3).

Please note that a *stream* refers to a collection of either base or enhancement layer chunks of a single view, or chunks of a particular audio stream. Each stream is identified by a PID (Program Identifier) number. The list of PIDs for a session is provided in the content metadata, as mentioned in Section 2.3.4.

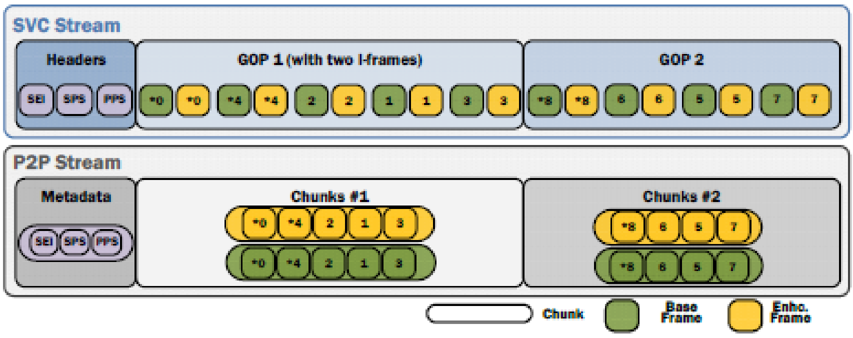


Figure 3: Chunk Generation Using One GOP per Chunk

### 2.3.5 Encryption and Security

The chunks are forwarded to the security server to be encrypted. The server generates a chunk header that is not encrypted and provides information such as *chunkID*, *contentID*, *PID*, *PCR* (clock), *chunkNo*, *viewID* and payload size. When a peer receives a chunk, it checks these fields to validate the chunk. If the peer manipulates the unencrypted header, the security client module can detect such actions and inform the P2P module about the integrity failure. In such cases, the client disconnects from that particular peer and updates the blacklist table, which holds list of blocked peers' IDs.

The security server creates distinct key pairs that are used to encrypt and decrypt the chunks for different content.  Without those keys decryption of the content is not possible. The key is forwarded to the main seed server as a special chunk with the KEY flag set. Similarly, the security server forwards the metadata information that is described in Section 2.3.4 to the main seed server in a special metadata chunk.

### 2.3.6 The Main Seed Server

The main seed server is the final destination of the chunks at the server side. The server initializes a new session upon the reception of the metadata chunk. Correspondingly, it creates the necessary directories under the *wwwRoot* of the web server. The chunks and the KEY information that is used to decrypt the chunks are stored in that directory. When all the content chunks are forwarded by the security server, a termination chunk with zero payload length is used to indicate that the End Of File (EOF) of the streams.

Upon reception of the EOF chunk, the main seed server performs the following sanity checks to confirm that the data is ready for delivery

- For each PID in the metadata, there should be at least one chunk received.
- The PCR value of the first chunks of each stream should be close to each other.
- PCR values should not roll back (i.e. they should be continuously growing).
- The number of streams in the metadata chunk should be equal to the number of total PIDs.
- The GUID string of each content should be unique.

If any of these checks fails, the content is removed from the web server. Otherwise, it is registered to the currently available contents using the given GUID in the metadata.
Once the content is available, peers can request chunks using HTTP protocol. We have used HTTP protocol in order to keep the system compatible with other server-client based adaptive streaming solutions. In HTTP streaming, it is possible to create pull based streaming mechanism that is controlled by the receiver.

The main seed server also operates as the tracker server and helps new peers to find other peers in the swarm (see Section 2.4).

### 2.4 System Initialization and Bootstrapping

The P2P streaming client is in slave mode and waits for directives from the Control Module (CM) that is in charge of regulating the operations of other modules in the user terminal. Consequently, a P2P session is initialized upon receiving the command from the CM that includes the IP address of the main seed server, which also operates as the tracker server.

The DIOMEDES P2P overlay uses a tracker server in order to find new peers in the swarm. The tracker is responsible for tracking peers and content servers and informing peers about the current state of the swarm. In BitTorrent, tracker server randomly forwards a subgroup of peers upon request from a newcomer. Definitely it is possible to follow an intelligent approach for peer selection. Instead of selecting partners randomly, Hei et al. claim that buffer maps can be utilized to monitor network behaviour of a peer and suggested matching peers based on the state of their buffer maps (a data structure that indicates the chunks that are available in a peer) [4]. The authors state that it is important to choose a peer with similar network resources.

Inspired from their approach, the tracking service in DIOMEDES is designed as follows: A tracker server clusters peers according to the requested streams. Since the peers perform rate adaptation, peers that request the same set of streams can be considered to have similar bandwidth capacity. Hence, a similar segmentation is performed among peers using the list of streams that peers are interested.

When a peer connects for the first time, tracker forwards a random subgroup of peers, since the views that are currently requested by this peer is not known in advance. However, in the following iterations, peer forwards the PID number of streams that they are currently interested in. Correspondingly, the tracker server updates its entry for that particular peer and then returns a sub-list of peers that are interested in the same set of streams to the extent that it can.

### 2.5 Streaming Mechanism in DIOMEDES P2P to Deliver the Best QoE over IP

The focus of the P2P overlay system is to create an efficient streaming mechanism, in which the application layer works with the underlying transport and network layers coherently. The two aspects (application layer framing and adaptive streaming) mentioned before are the tools to achieve the best throughput out of a best-effort IP channel. Since the aim is to deliver real-time multimedia content rather than a binary file, the adaptation is performed considering the QoE of the users.

The rest of this section explains how adaptive streaming is performed in the overlay architecture that is developed for project DIOMEDES and how it is different from the currently available adaptive P2P streaming solutions.

### 2.5.1 Sliding Window

Default chunk picking policy of torrent based file sharing systems (i.e., rarest-first approach) does not meet the requirements of delivering time sensitive multimedia content. Therefore, most of the P2P solutions use windowing mechanism that provides timely delivery by restricting scheduling for chunks inside the window and also chunk randomization among peers (to some degree) [5],[5]. The randomization is vital for the success of chunk-based P2P, because if every peer downloads the same chunk, they cannot exchange distinct chunks among themselves.

In our solution, there are separate windows for each stream. These windows can be in perfect synchronization or they can be slightly shifted according to the state of the downloading session. Figure 4 depicts a snapshot of downloading windows in perfect synchronization. Please note that only the chunks inside a window can be scheduled for downloading (chunks in dashed line with yellow background). There can be some chunks that are downloaded but not played back yet (light green), or some chunks that are currently being download (yellow chunks with straight line). When and why the downloading windows get out of synchronisation is explained in the Section 2.7.

The state of the buffer is critical in the adaptation process and it is calculated based on the state of the chunks in window. The buffer length is measured from the end of last played chunk to the start of the downloading window. The duration of the chunks between these borders determines the length of the buffer.

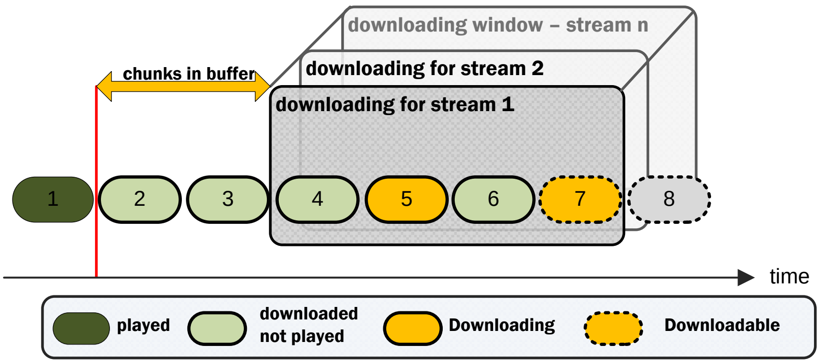


Figure 4: Downloading Windows and Chunks

### 2.5.2 Role of Adaptation Decision Engine

Adaptation decision engine's operation and the overall video adaptation approach were described previously in D3.5/ D3.6 (Report on 3D Audio/ Video rendering and content adaptation). Adaptation decision engine is in charge of deciding on the periodical importance levels of particular streams (video quality layers and depth maps, and audio) and ranking them according to their relative perceptual importance. Actual adaptation with respect to changing network conditions is executed in the P2P streaming client. The input parameters the adaptation decision engine takes are the KPI values for each successive GOP (delivered through the video decoder), user's viewing preferences (i.e., selected free-viewpoint position), list of all available streams associated with the downloaded 3D content (e.g., knowledge on the total number of camera views), camera parameters and the knowledge on the existence of a DVB 3D video reception. Readers are recommended to refer to D3.6 to have a deeper insight on the operation of adaptation decision engine. As long as the user's preferred viewing position stays constant and the KPI parameters do not change significantly over time, the output PID ranking list delivered to the P2P streaming client also stays constant. The ranking order tends to change only if the user slides its viewing position using the user interaction module, if there is significant change in the KPI values in the delivered camera views, or the state of multimedia delivery/ context of consumption changes (e.g., DVB reception becomes active, or user switches to a multi-view screen from stereoscopic screen).Audio streams are given always the highest priority over any other video stream, because of two inherent assumptions. First, the disruptions in the spatial audio playback have more detrimental effects on users' QoE. Therefore they are always given the highest chance to be downloaded even under severe network conditions. Second, since the chunk sizes and the content bit-rate for the audio streams are remarkably smaller than that of the video streams, they are not sacrificed at times of bandwidth adaptation through stream truncation. Because, the adaptation range they would offer would be far less significant than the adaptation range the video streams would offer.

### 2.5.3 Implemented Methods for Adaptive Streaming

The adaptive streaming that is performed in DIOMEDES P2P is performed in two steps. The first one is Stream Selection that is performed each time a download window slides (Section 2.5.3.1). And the second one is adaptive chunk selection that is performed each time a chunk is to be scheduled within a download window (Section 2.5.3.2). In addition to those steps, the robustness of video delivery is augmented using multiple description coding (MDC) by requesting chunks that are available in *unstable* peers from multiple peers.

#### 2.5.3.1 Rate Adaptation by Stream Selection

Once a streaming session is initiated, peers request chunks of all layers to deliver at the highest possible quality. Once the window is downloaded, it advances its window in time to schedule new chunks. Meanwhile, the player starts to consume downloaded multimedia when the buffer reaches to a certain level. Adaptations are based on the streams' prioritization order determined by the ADEM (see Section 2.5.2). The number of streams to be scheduled is updated each time the window slides. If the buffer duration is below a certain threshold, the number of downloaded streams is reduced by discarding the stream that is at the bottom of the prioritization order. (The buffer duration is determined by the duration of received sequential chunks that can be decoded in all layers). In the opposite case, if the duration of the buffer is large and there is an unscheduled stream, then the one with the highest priority are added to the list of active streams. Once the number of streams is determined, chunks that are within the new window can be scheduled for download.

#### 2.5.3.2 Rate Adaptation by Chunk Scheduling

After the list of active streams (active windows) has been determined, a chunk can be scheduled for download. Chunk scheduling is performed in two steps. First, a candidate stream is selected based on the prioritization weights of active streams that are provided by the ADEM. The propability of selecting the $i^{th}$ stream ($p_i$) is the ratio of its weight ($w_i$) to the sum of weights of all streams as shown in the following expression.

$$p_i = \frac{w_i}{\sum_{i=0}^{n} w_i}.$$

The second step is to determine the chunk request order for the selected stream. The criterion here is whether the buffer is short, or long. If it is short, chunks are requested in a weighted random order that gives higher chance to the chunks close to the play-out deadline to be scheduled first. If the buffer is long, chunks that are available in the swarm are scheduled first. If no chunks are available in the swarm, then they are selected randomly within the window. Randomization increases the diversity among the peers and augments the P2P communication, which decreases the overall bit-rate requirements on the server. Figure 5 depicts the chunk selection process.

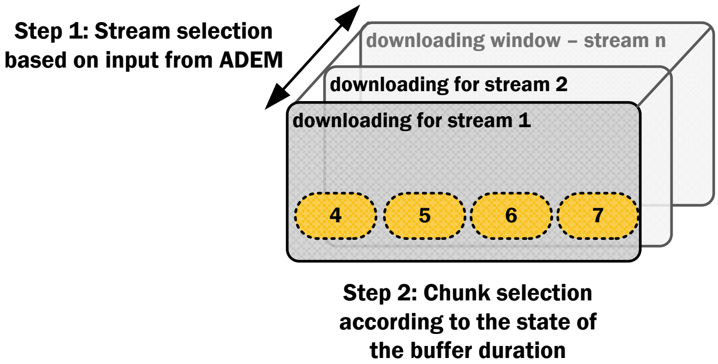


Figure 5: Chunk Selection in Two Steps

### 2.5.4 Robust Delivery Using SMDC

#### 2.5.4.1 Effect of Redundancy and Use Cases

Multiple Description Coding (MDC) increases the robustness of data delivery by transmitting distinct descriptions (self-usable bit-streams) over different paths (path diversity). In theory, if a peer churn (ungraceful exit of a peer) occurs, a receiver can still receive another description and proceed with decoding but at a lower quality output. There are two issues that we want to underline before describing the scalable MDC description implemented in DIOMEDES P2P.

First, one needs to be careful about the implementation of MDC and keep an eye on the redundancy it causes. A very simple example can illustrate the effect of redundancy. Consider two cases. In first case MDC is used and it has 20% overhead. In the second case, MDC is not used so all the available channel capacity is dedicated to source coding (video in our case.) Table 1 presents the state of the peers' video buffer after 10 seconds if no peer exit event has occurred in that duration. Considering that the video rate is 800Kbps, the *Peer1* has 10 seconds buffer where as the second peer has 12.5 seconds of data in buffer. So the second peer has 2.5 more seconds to handle a peer exit event. If it can find a new peer in a shorter interval, then not using MDC is a better option and vice versa. This simple derivation is just to show that if not used properly, MDC can be harmful as well, due to the redundancy causes.

Table 1: Simple Comparison of MDC, State of Peers after 10 seconds

| | Bandwidth | Uses MDC | Redundancy | Actual Receive Rate | Data in buffer after 10 seconds |
|---|---|---|---|---|---|
| ***Peer1*** | 1000Kbps | Yes | 20% | 800Kbps | 8000Kb |
| ***Peer2*** | 1000Kbps | No | 0% | 1000Kbps | 10000Kb |

Secondly, one of the best cases to apply MDC is adopting it in data distribution with multiple-trees in which the content delivered in push manner [10]. Commonly, in push based streams, the sender does not wait for notification from the receiver side and forwards data to its child node as it receives the data from above. Therefore, the sender commonly is unaware of the buffer state of the receiver (especially the data received from over other trees) as depicted in Figure 6. The sender just blindly forwards the content in a way that it maximises the probability of receiving useful data. However, in mesh based topologies in which data requests are receiver driver, the receiver does have the knowledge of the data in its buffer. This has two consequences.

1. The sender does not have to request the redundant part if it already received it from another peer.
2. What the receiver has to estimate is the probability that a request from another peer will be successful.

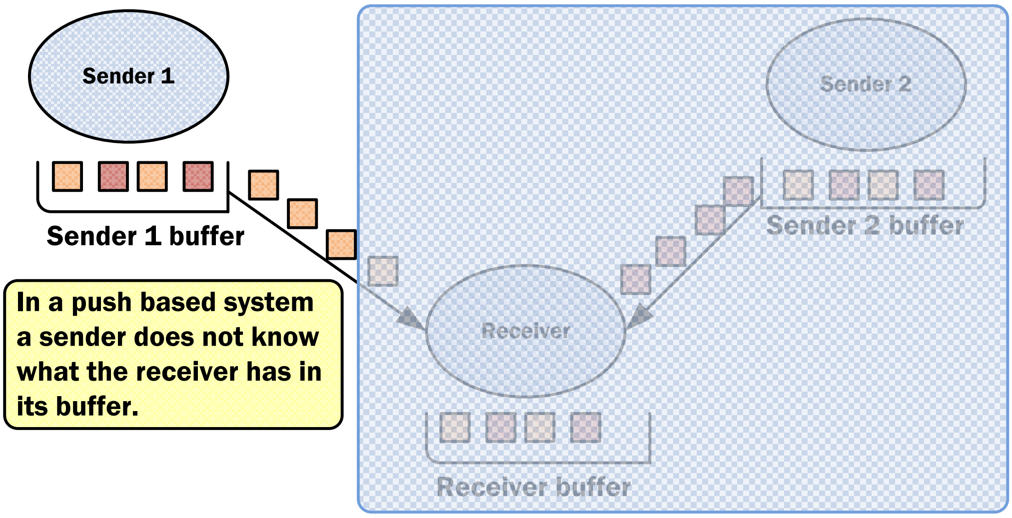


Figure 6: Push-based model (commonly adopted in tree-based topology)

### 2.5.4.2 DIOMEDES SMDC Scheme

DIOMEDES P2P has a mesh based topology and a peer has the advantage of deciding when to request chunk, from whom to request and which layer to request. With these options at hand, we adopt an MDC system in which the multiple descriptions are requested only when necessary in order to keep the bitrate as low as possible. For instance, a peer is to request a chunk that is available in peers which have low upload capacity (at least the current connection quality is poor). If the peer has long buffer duration then it does not have to use MDC because it can retry to download it from another peer if the current transmission fails. On the other hand, if the deadline is close then it is safer to request data from multiple peers and try to receive at least one copy of the chunk.

In order to be able to switch between single description coding and multiple description coding the redundant and the unique data of the descriptions should be individually accessible. If the descriptions are packed in a single chunk for instance, then it is not possible to avoid the redundant part of the descriptions. However, if the descriptions are split in two as redundant and unique data then it is possible to perform MDC streaming selectively.

In DIOMEDES P2P, the descriptions are generated using the base and enhancement layer of the content as depicted in Figure 7. In our case, the redundant data is the base layer and the unique data is the enhancement layer (that is distributed equally among descriptions.) Since each of these data structures can be addressed separately (they are in different chunks) than it is possible to perform selective MDC streaming. Moreover, a description can also be scaled if the enhancement layer is discarded.

The implementation of MDC in DIOMEDES P2P is as follows. A peer always ranks their neighbours according to their upload rate. When a chunk is scheduled for download, candidate peers are sorted based on their upload rate. If peer with high upload capacity is available, then the chunk is requested only from that particular peer. However, if no peers with high upload capacity is available then the state of the buffer determines the next step. If the duration of buffer is long, then we take the risk of downloading a single chunk from a single peer with low upload capacity. However, if the buffer is low and the chunk is a base layer chunk then we make two identical requests from two different peers. The corresponding state diagram is provided in Figure 8.

Description 1

| Frame No: | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| GOP 1 | B | B | B | B | B | B | B | B |
| Frame No: | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| GOP 2 | BE | BE | BE | BE | BE | BE | BE | BE |

Description 2

| Frame No: | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| GOP 1 | BE | BE | BE | BE | BE | BE | BE | BE |
| Frame No: | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| GOP 2 | B | B | B | B | B | B | B | B |

Figure 7: Description Scheme in DIOMEDES for GOP size 8
(B Base layer, E Enhancement layer)

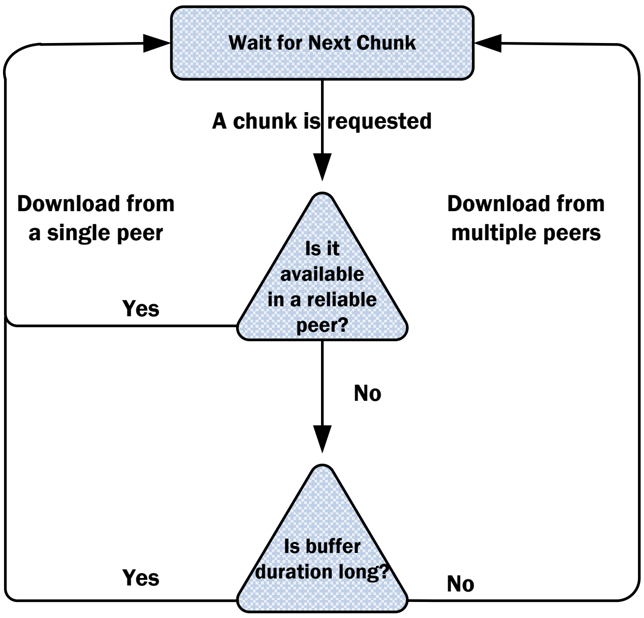


Figure 8: Chunk Scheduling in for MDC

### 2.6 Methods to Decrease the Bandwidth Requirement on MSS

Besides delivering the best QoE by content aware adaptive streaming (Section 2.5), a secondary goal of DIOMEDES-P2P is to keep the bandwidth requirement of the server side as low as possible. Two methods have been implemented to augment the P2P distribution and decrease the burden on the main seed servers. The first one is about adjusting the size of the download window dynamically (Section 2.6.1) and the second one is choosing the chunks in an intelligent way to increase the portion of P2P data distribution (Section 2.6.2). The efficiencies of these approaches are tested in the Section 3.

#### 2.6.1 Dynamic Window Size

In BitTorrent protocol, a peer can choose an arbitrary chunk to download. Most commonly, they prefer a chunk that is least distributed in the swarm. By introducing windowing mechanism for video delivery, the efficiency of this approach is reduced due to the limited number of chunks that the window covers.

In order to decrease the side effect of windowing, different trials with different window sizes are done. Initially, window duration is set to ~2.2 seconds (it is a configurable parameter). Each time a downloading window is complete, the P2P software calculates the duration of the buffer. If the duration of the buffer is larger than the initial value, the window size is advanced correspondingly. Increased window size increases the chances of peers having distinct chunks in their buffer, thus increasing the likelihood of P2P activity rather than direct download from the main seed servers. This feature is active, only if all streams are currently downloaded. If a stream is discarded due to bandwidth scarcity, the window size is not increased over time, because it is more difficult to fill the buffer with a large window size (e.g. a discarded stream can be re-added only if the buffer duration is above a certain threshold).

### 2.6.2 Chunk Selection Policy in Different Modes of Operation

The adaptation and chunk scheduling decisions are always performed with the state of the buffer in consideration. In the case of bandwidth surplus, a peer will have long buffer duration. Thus, it can actually slow its pace of download in order to avoid putting more stress on the main seed server(s). In such cases, a peer enters to a *Safe Mode,* in which the chunk downloads from the server are decreased. Each time a chunk is downloaded from server, it has to wait for a certain amount of time to be able to schedule another chunk from the server again. In Safe Mode, the peer can still exchange chunks with other peers in the swarm.

## 2.7 Details of Implementation

This section provides the implementation details of the developed P2P multimedia distribution protocol.

### 2.7.1 The Source Code

The P2P software is implemented in Java without using any external library for P2P-based communication. By nature, it is platform independent and can run on almost all modern operating systems. It has a multi-threaded architecture to handle tasks that can run concurrently. A *session* object is created each time a start message is received from the Control Module. *Session* is responsible for initializing all the data structures and managers (see 2.7.1.1) and also creating folders to store the chunks that are downloaded.

#### 2.7.1.1 Data Structures

**P2P Metadata**

P2P metadata is an internal data structure that is between the main seed server and the P2P client. It provides some information on the content metadata, such as number of streams, their PIDs and the type of a stream (e.g. base or enhancement). It also includes other information, such as the number of chunks and their average duration in time.

**Bitfield**

It is a bit array to mark the list of available chunks in the peer. This data structure is exchanged among peers upon handshake. It is very similar to the BitTorrent bitfield, but it is multi-dimensional to support multiple streams.

**Chunk States**

This is an array of chunks (a class in the code) that hold the information about the state of a chunk. A chunk can be in one of the following states:

- CHUNK_STATE_UNSCHEDULED = 0;
- CHUNK_STATE_BEING_DOWNLOADED = 1;
- CHUNK_STATE_DOWNLOADED = 2;
- CHUNK_STATE_FORWARDED = 3;
- CHUNK_STATE_ABORTED = 4;

**Peer List**

This is an array of peers (a class in the code) that holds information about the peer (IP address, port number, chunks it has) and also some statistical data such as download rate.

#### 2.7.1.2 Multi-threaded Architecture and Managers

Specific threads are created for certain operations and they are called managers. Each manager shares above mentioned data structure and also an event handling object (event handler). Manager can communicate with each other by pushing new events to the event handler. For example, player manager can request "stream drop event", that is to be delivered to the download manager.

The lifecycle of a manager has three phases.

- **Initialization**: In this phase a manager initializes its own data structures and connections. If initialization fails, the manager terminates.
- **Loop**: A manager performs its regular duties in the loop phase. At the end of each cycle it also checks for the termination case that will break that cycle.
- **Termination**: In this final phase a manager closes its connection and releases any locks it has on any data structure.

**Download Manager**

Download manager handles chunk download operation. It has a data structure (chunk scheduler) that holds the state of the download windows.

- **Initialization**: Constructs the chunk scheduler.
- **Loop**: Request chunks from the scheduler. The scheduler returns a chunk and a peer id if available. Otherwise it returns a null value to indicate that no chunk download can be initialized right now.
- **Termination**: Terminates when all chunks of the stream is downloaded.

**Player Manager**

Player manager is responsible for forwarding downloaded chunks to the player (i.e., to the AV Synchronisation Module via the Decryption Module). It is responsible for receiving PCR (clock information) from the player and marking chunks as aborted, if PCR deadline surpasses. According to the state of the buffer, player manager can request stream add or stream drop.

- **Initialization**: Connects to the player.
- **Loop**: Checks for new chunks to forward to player. Updates buffer states.
- **Termination**: Terminates when all chunks of the stream is forwarded to the player.

**Peer Manager**

Peer manager is responsible for handling requests from the neighbours

- **Initialization**: Creates a port to listen incoming connection requests.
- **Loop**: Listens requests of all peers.
- **Termination**: Never terminates unless the client sends close signal. (Even if download and forwarding finishes, a peer can always upload chunks to other peers.)

**Tracker Manager**

Tracker manager updates its state information to the tracker server. It is also responsible for updating the peer list.

- **Initialization**: none.
- **Loop**: Connect to tracker server and sends the current list of streams it is receiving and also receives the list of new peers (if necessary).
- **Termination**: Never terminates.

#### 2.7.1.3 Graphical Interface

DIOMEDES P2P comes with a built-in graphical interface that provides information about the state of a peer. It is handy for debugging and diagnosis purposes, because the only module that a user can interact is the Control Module. The graphical interface can provide the following information:

- Download rate
- Chunk states
- Player PCR value
- Buffer Duration
- Percentage of chunks that are forwarded to the player
- List of peers

### 2.7.2 Out of Synchronization Windows

As depicted in Figure 4, one would assume that normally windows are in perfect synchronization. However, this is not the case most of the time. First, the duration of the chunks may be different, causing different number of chunks to be in a window. More importantly, when all of the windows are about to be complete, the number of *startable (chunks within a download window)* downloads start to diminish if all windows are slides together. This has a negative effect on channel utilization. Therefore, a window slides if no other chunks can be scheduled because all of the chunks in the windows are either downloaded or currently being downloaded. Naturally, the window of the streams with more importance slides first in such cases.

### 2.7.3 Resynchronization with DVB

Due to the chaotic nature of IP, it is possible that even if the system drops down to a single stream to be downloaded (severe congestion), it may not keep up the synchronization with DVB. In such cases, the windows slide in an exponentially increasing manner. If the last window cannot be completed on time, then it skips one chunk and then slides. If it fails again, it skips two chunks. Next time it is four chunks, and so on. At some point it is highly likely that the IP-stream will resynchronize with the DVB channel, although the contribution from the IP channel will be very low.

## 2.8 List of Differences from Currently Available Solutions

DIOMEDES P2P has different requirements and features from the currently available solutions like Tribler and P2PNext. The differences can be listed as follows:

- **Chunk Generation:** Both Tribler and P2PNext use fixed sized chunks (to be compatible with Torrent). Tribler does not use padding to generate self-decodable video chunks. Consequently, Tribler cannot estimate the duration of playable (decodable) video in its buffer without parsing the content. In the `Tribler Protocol Specifications`, this fact has been stated to cause frequent re-buffering during the play-out if the chunks in the buffer are not useful due to prediction structure of the video [2]. In the P2PNext case, GOP units are split into different chunks using padding. In order to keep the overhead of padding as low as possible video coding is performed in CBR mode [7]. However, in order to avoid quality loss, commonly a high video rate is choosen for the CBR video coding and this is not an efficient solution for video streaming over IP networks [8]. In DIOMEDES, chunks are formed using GOP boundaries. Therefore, it is possible to determine the duration of decodable data in the buffer and perform better adaptation. It is accepted that using application layer framing in a video streaming architecture is a significant contribution for video streaming over P2P overlays. In the literature, there are numerous studies, which suggest that solutions with ALF concept (cross-layer solutions) outperform the content agnostic transmission mechanisms.

- **Security Mechanism and Metadata Format:** DIOMEDES has a different security mechanism to check the integrity of chunks. This has two effects. First, the content authentication mechanisms (comparing hash values) of torrent-based solutions are nullified. Instead, new functionalities are required such as downloading the public key of the session, forwarding it to authentication and decryption module and performing security checks via socket communication. Secondly, using a public-private key pair also nullifies the MD5 hash values that are present in the standard metadata file. Considering that most of the standard metadata is composed of hash values and also considering the amount of new parameters that are required to render multi-view video at the receive side, using a new metadata format is a more convenient solution.
- **Hybrid Model:** The content play-out strategies in DIOMEDES is different from the others. In DIOMEDES, the content is to be consumed in synchronization with DVB channel. Hence, it is possible that a content that is available in Main Seed Server can be played out at a significantly later time than the broadcasting channel. In such cases, it is possible to distribute the video content in an intelligent way to minimize the server load. This also creates an incentive for the content provider to prepare the content at ahead of time to deliver huge amount of data at a low cost if possible. Naturally, the P2P policies that are adopted in DIOMEDES do not rely on the difference between content preparation and play-out deadline. If the time is short, necessary adaptations are performed to deliver very recently uploaded content. However, if there is time then the system uses a very intelligent way of distributing the content seamlessly by consuming significantly less bandwidth and making the distribution cheaper and safer. Therefore, the chunk scheduling policies in project DIOMEDES always keeps an eye on the bandwidth requirement of the MSS.
- **Chunk Picking:** When scheduling a chunk for download, a windowing mechanism is used to enable timely download. In P2PNext uses layered windows in the sense that a chunk from a more important stream is more likely to be selected. Similarly, a chunk that has closer play-out deadline is more likely to be picked. (See Table 2, the entries represent the weight of a chunk that is used to calculate the probabilities)

Table 2: P2PNext Chunk Scheduling in Sliding Window [7]

| | t | t+1 | t+2 | t+3 | t+4 | t+5 | t+6 | t+7 | t+8 |
|---|---|---|---|---|---|---|---|---|---|
| E3 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| E2 | 1.0 | 1.0 | 1.0 | 0.7 | 0.3 | 0.0 | 0.0 | 0.0 | 0.0 |
| E1 | 1.0 | 1.0 | 1.0 | 1.0 | 0.8 | 0.5 | 0.0 | 0.0 | 0.0 |
| Base | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.9 | 0.5 | 0.0 | 0.0 |

DIOMEDES-P2P provides dynamic multiple windows with increasing size over time if the download rate is high enough in order to boost the randomization among chunks and augment the P2P activity. Furthermore, DIOMEDES P2P utilizes an MDC approach by requesting the base layer chunks from two different peers if there is a risk of not receiving a chunk due to poor upload capacity of peers.

# 3 NETWORKING TESTS

## 3.1 Overview

This section provides test that we perform to evaluate the performance of the methods and design approaches discussed in Chapter 2. We provide detailed information about each test, such as the setup, content and link conditions. In most the cases, the tests are performed in controlled LAN environment for the purpose of stability of the results and to be able rerun the tests under the same conditions to evaluate the performance of the solution. For this purpose, we have utilized the NetEM library of Linux OS [13]. Using this tool, it is possible to create any network conditions (bitrate and delay) and adopt different scheduling protocols at the network level to mimic the effect of routing over the Internet.

In Section 3.2, we provide an analysis of VBR coding vs. CBR coding. We believe that the CBR coding that is adopted by many P2P projects has hidden handicaps and moreover, switching from CBR to VBR cannot be done effectively without using variable size chunks. In the other words, this section introduces the benefits of using a true application layer framing concept for a P2P video streaming architecture.

In Section 3.3 and Section 3.4, we investigate the performance of adaptive streaming methods that we adopt in DIOMEDES P2P. The former one mainly provides an evaluation in terms of QoS and mainly provides the number of chunks that are delivered (or lost) under different network conditions. The latter one mostly introduces the effect of adaptation over the perception of the users. In Section 3.5, we perform tests to analyze the performance of intelligent chunk picking policies that are adopted to keep the bitrate of the main seed server as low as possible. Finally in Section 3.6, we evaluate the performance of the adopted MDC policy that we discuss in Chapter 2.

## 3.2 Video Coding Tests for VBR Coding vs. CBR Coding

As the name implies in variable bit rate (VBR) coding, the output data rate can vary according to the characteristics of video context. For instance, when the amount of motion is low it is possible to use the skip mode while encoding a macroblock (MB) and can save significant amount of bits, decreasing the overall bitrate of the encoded stream. On the other hand, when there is high motion or complex pictures, it is possible to increase the video bit rate to maintain the quality of the encoded bitstream. In constant bitrate coding (CBR), data output rate that coding is constant regardless of the content as opposed to VBR. The characteristics of the context such as dynamicity are disregarded and in most of the time the bitrate is fixed at a high level to be able to maintain quality, creating waste of bits for no perceptual improvement. Therefore, CBR would not be the optimal choice for video streaming over IP because of the following handicap:

- If high bit rate in order to encode the complex sections at a good quality, then bits are wasted if content has uniform regions or slow motion.
- If the chosen bitrate is low, then the complex sections would suffer from encoding at a low quality at that bit rate while not wasting bit rate for less complex sections.

The purpose of this test is twofold. First, we would like to demonstrate that using CBR mode can be significantly inefficient for video delivery over IP network. Secondly, we want to show that using fixed sized chunks aggravate the problem due to padding, which is a problem even with VBR coding.

### 3.2.1 Video Coding Tests

We have used the sample movie stream `wildlife.wmv` that is provided freely in Windows 7 and encoded it using SVC extension of H.264/AVC using single layer with no overhead of scalability. The content resolution is *1280x720*. We have performed VBR coding at 5 different quantization parameters to represent 5 different video qualities (highest, high, medium, low, and lowest).

#### 3.2.1.1 Video Coding Results

Figure 9 depicts the change in quality (PSNR) over time and the Figure 10 presents the corresponding bitrate of the sequences for the same time interval. The variation in the PSNR value is due to the characteristics of the content. When the motion is low, it is possible to achieve high quality at the cost of low bitrate. In the other extreme, when there is a complex scene or high motion the bitrate increases but it may not be able to obtain the same quality.

If this scene is to be encoded using CBR mode and if minimum acceptable PSNR value is 32dB then the high quality content (red line) can serve as a reference because in the high quality content, the minimum PSNR value is just over 32dB. Then the CBR coding rate must be about 4000Kbps which is the peak of the red line in Figure 10. For such a case, the amount of overprovision to achieve the min PSNR value requirement is depicted in Figure 11.

Considering that the diminishing gain in RD performance even in the state of the art video codecs makes this situation a serious problem. For instance, if the minimum bitrate is above 34 dB even 5000Kbps is not enough. So, video streaming applications that use CBR coding has this hidded low bitrate utilization problem. This is why DIOMEDES has adopted VBR coding for video delivery over the IP networks using P2P overlay.

### 3.2.2 Effect of Padding in Fixed Sized Chunk Solutions

Switching from CBR to VBR almost mandates the use of variable length chunks. Table 3 presents the overhead of padding if the high quality content is to mapped to fixed sized chunks. Besides the overhead, if the chunk sizes are smaller than multiple chunks are used to contain a single GOP, creating non-self-decodable chunks.

Table 3: Percentage of padding loss for different chunk sizes for high quality content

| 64K* | 128K* | 256K |
|---|---|---|
| 0.230914 | 0.480315 | 0.866485 |

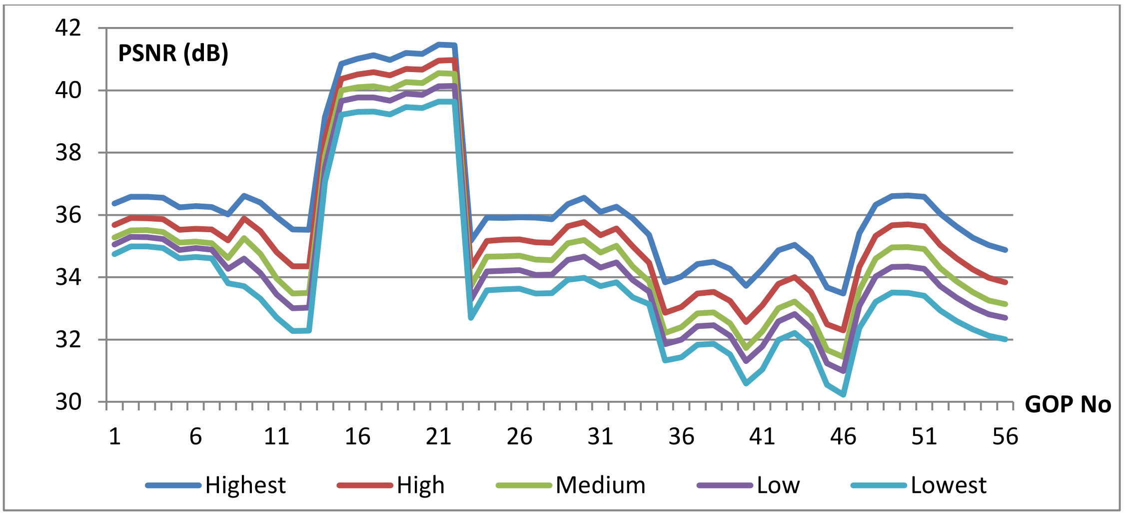


Figure 9: PSNR over Time (1 GOP ~= 0.5 sec) Wildlife

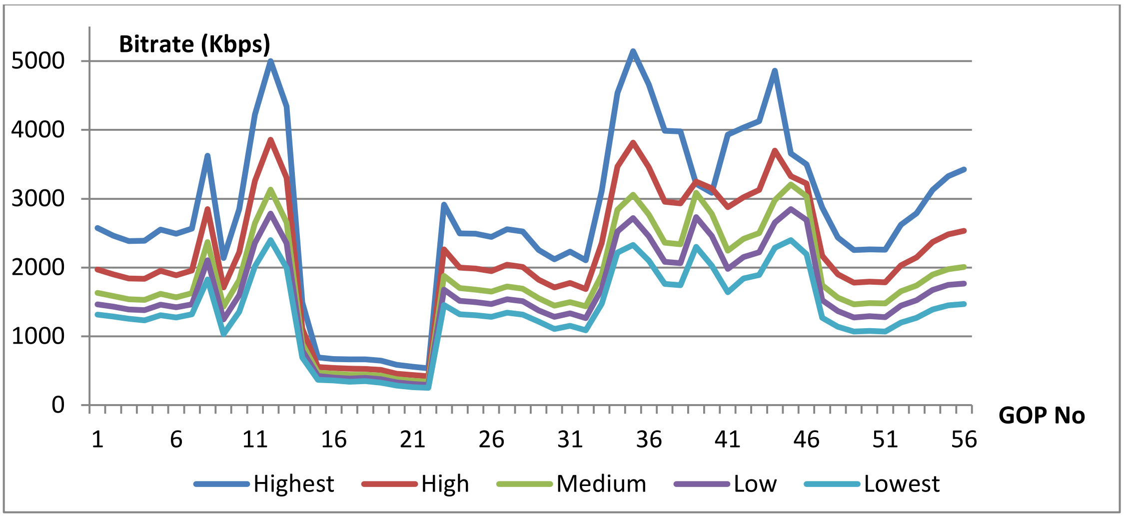


Figure 10: Bitrate over Time (1 GOP ~= 0.5 sec) Wildlife

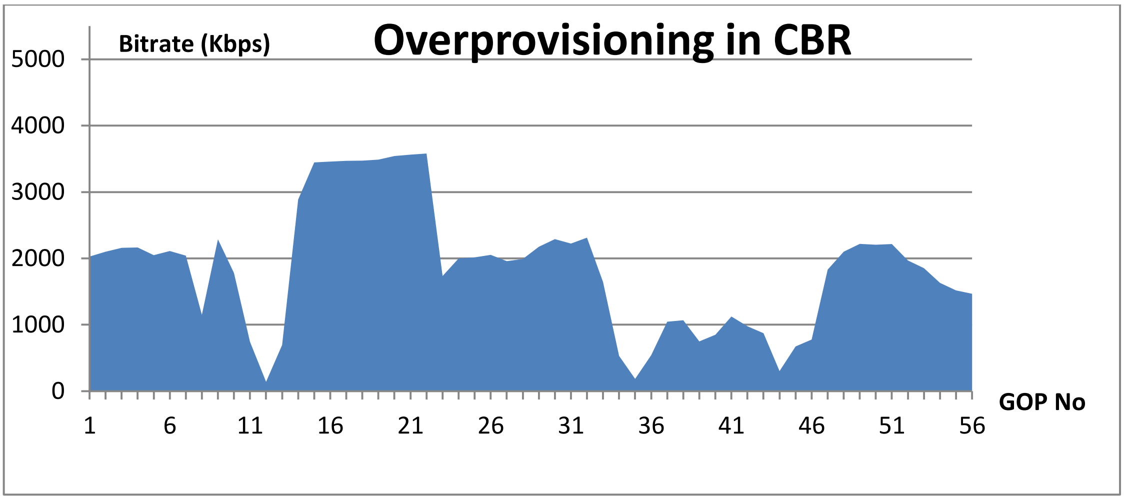


Figure 11: The rate that is above the satisfy the minimum PSNR requirement.

### 3.3 Adaptive Streaming Tests - QoS Perspective

In this section, we evaluate the adaptation methods that are described in Section 2.5. Our goals are to lose as less number of chunks as possible and if a chunk loss is inevitable due to lack of network resources, the system should perform adaptation in a way to lose chunks of the least important stream first.

We have performed two types of tests. In the first four tests, we use fixed bitrate for the total available network resources and evaluated the performance of the system to deliver to best quality under a steady condition. In the last test, the state of the network changes dynamically to evaluate the performance of the system while adapting different network conditions.

#### 3.3.1 Test Setup

In adaptive streaming tests, we record the downloading results of a peer that is connected to 5 neighbouring peers and it is also connected to a main seed server. Each neighbour has 0.5 Mbps upload capacity whereas the main seed server has been

limited to 1Mbps. This is actually a realistic scenario because even in BitTorrent application a peer may have many peers in its list (peer list) but the number of active connections is commonly much less then this number [15].
As for the content, we have used three dummy streams each at 1Mbps. There are two chunks per second per stream. The duration of the streams is 330 seconds long. The gap between the P2P channel and the DVB channel is set to two seconds, which serve as very limited buffering duration.

#### 3.3.1.1 Test Scenarios

We have performed of five different tests, each for a different goal. Table 4 summarizes the test conditions and the list of target results. Figure 12 depicts the total available network resources for the traced peer. In each test scenario, the main seed server is always active but the number of peers that are active can change over time.

Table 4: Test Conditions and Best Case Results for Adaptive Streaming Tests

| Test Conditions | | | Best Case Results | | |
|---|---|---|---|---|---|
| No | Network State | Total Capacity | Stream 1 | Stream 2 | Stream 3 |
| 1 | Fixed | 1.1 MB | Lossless | Total Loss | Total Loss |
| 2 | Fixed | 1.5 MB | Lossless | Partial | Total Loss |
| 3 | Fixed | 2.1 MB | Lossless | Lossless | Total Loss |
| 4 | Fixed | 2.5 MB | Lossless | Lossless | Partial |
| 5 | Dynamic | - | Lossless | Partial | Partial |

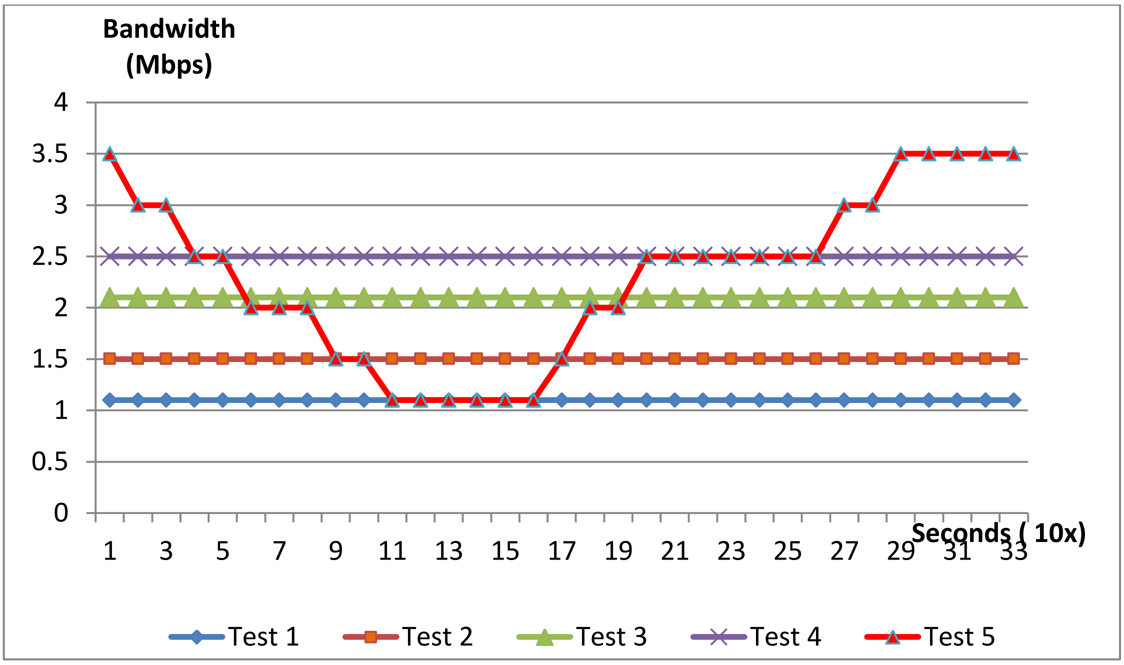


Figure 12: Total bandwidth capacity of the nodes in the network that provide chunks for tests

### 3.3.2 Results

The results of the adaptive streaming tests suggest that the system can adapt to different network conditions within a couple of seconds. Figure 13 to Figure 17 presents the number of lost chunks from each stream for within durations of ten seconds intervals. The losses from the first stream (most important) are coloured as red, whereas the blue and gray colours indicates the chunk losses from stream 2 and stream 3 (least important) respectively. There are 20 chunks within that duration for each stream.

In most of the cases the system achieves the target goals but there are still some points that should be underlined.

- When the system first starts, it tries to receive all the streams. If the bitrate is not high enough within this initial stage, then chunk losses can occur from any stream. Thus the system starts in an optimistic manner but adapts to the network conditions within the first five seconds of the streaming session. (Figure 13, Figure 14, and Figure 15)
- The system can partially receive a stream if the bitrate is not enough to receive the whole stream but it is high enough to receive some bits. If fluctuations due to such cases cause disturbance for the users' QoE, the player can always discard such streams and display only fully received streams.( Figure 14 and Figure 16)
- The system can respond to dynamic changes in the network significantly well if it has a buffer with two-three seconds of duration. The only condition that the adaptation fails is the case when the system has no buffer like in the case of initial start. (Figure 17).

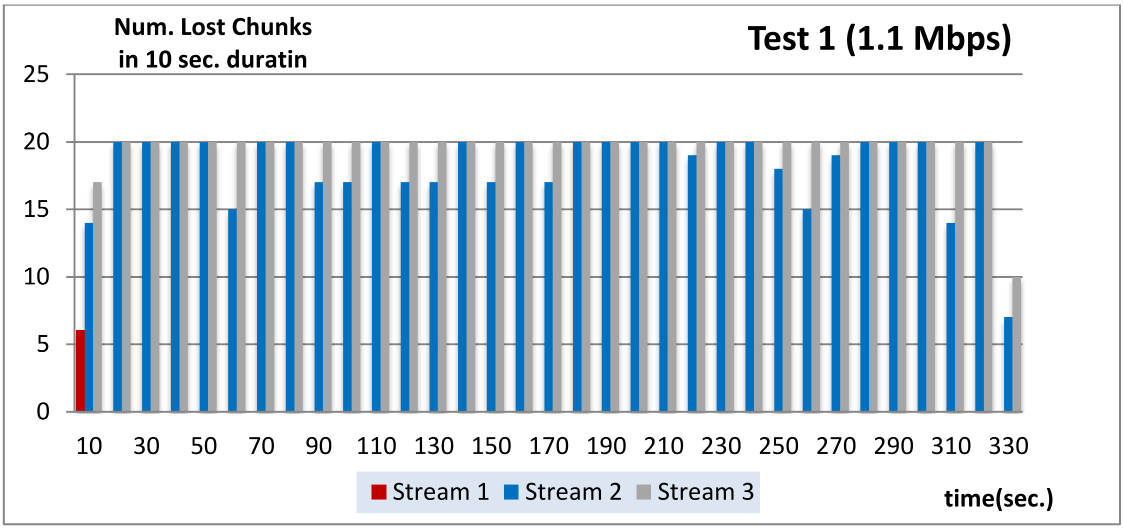


Figure 13: Results for Test 1

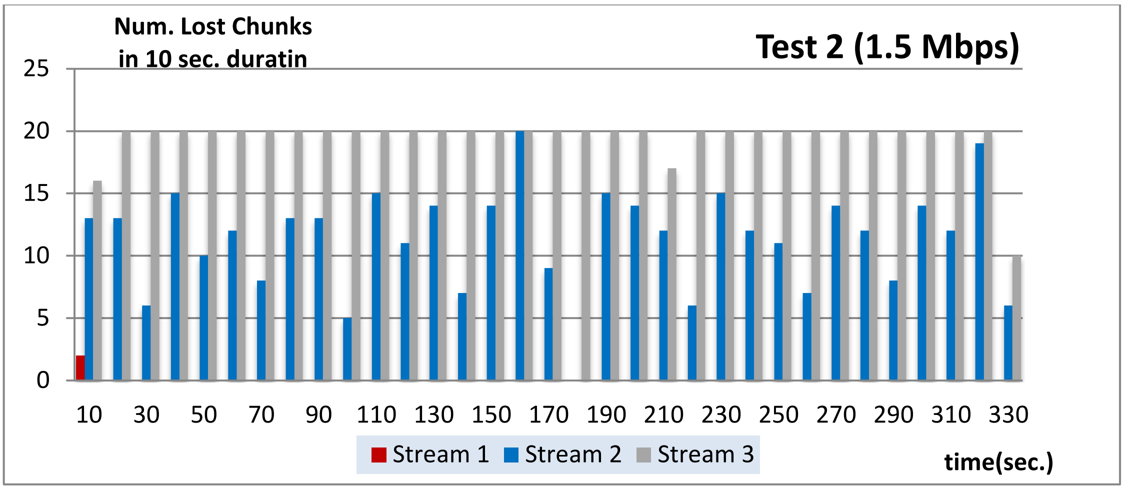


Figure 14: Results for Test 2

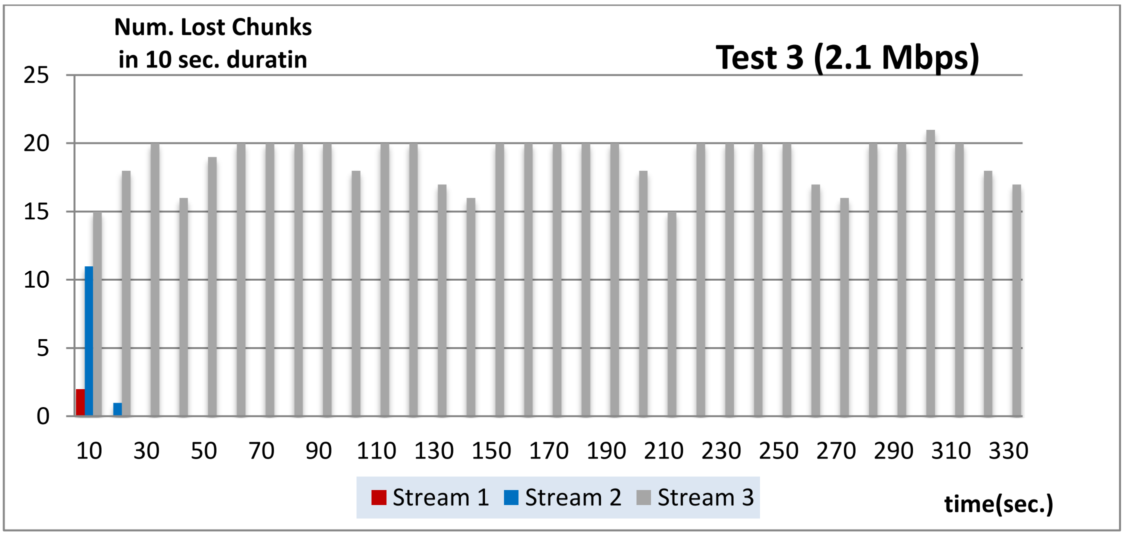


Figure 15: Results for Test 3

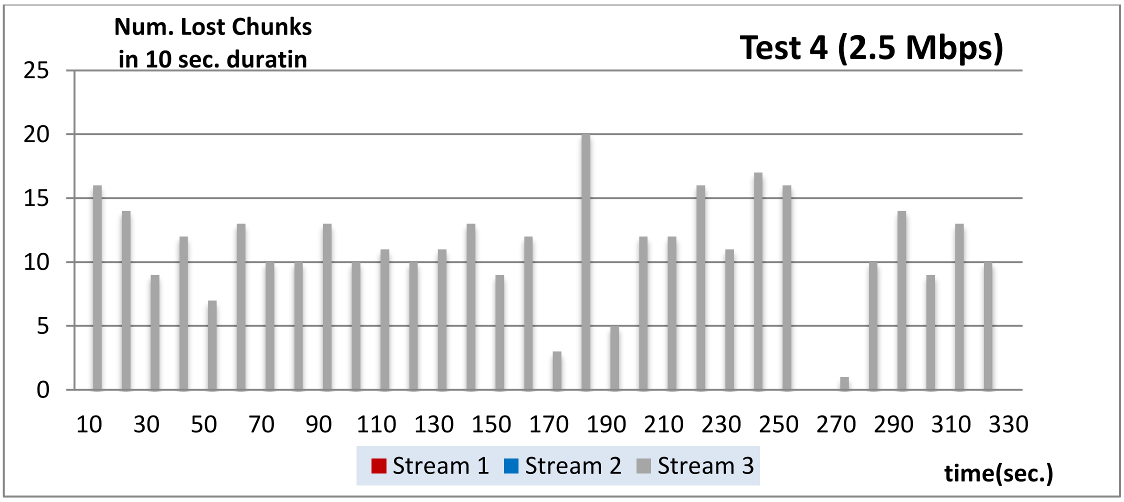


Figure 16: Results for Test 4

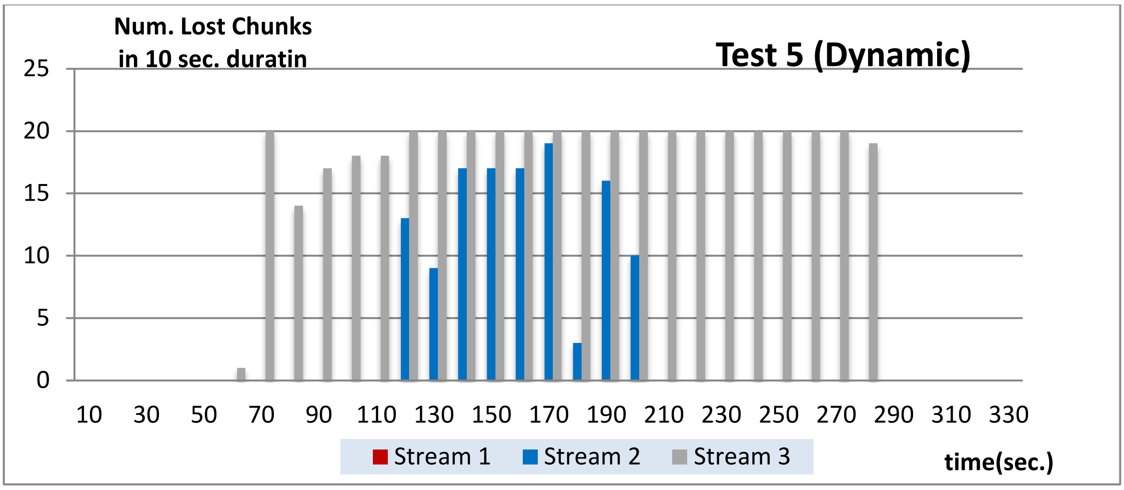


Figure 17: Results for the Test 5 (Dynamic)

### 3.4 Adaptive Streaming Tests - QoE Perspective

The methods deployed in DIOMEDES-P2P to realise efficient adaptive streaming were previously described in Section 2.5.3. Accordingly, the P2P streaming client performs adaptation by means of adding or removing streams (where the streams stands for the stream of chunks with a particular PID). Adaptation decision engine is in charge of assigning the priorities of each stream (identified with their PIDs) depending on the particular contribution of the selected streams in the overall 3D QoE considering user's particular viewing preferences.

The purpose of the series of tests performed in this subsection is to depict the performance of rate adaptation under different server upload rate conditions by measuring the QoE of the user throughout the media playback. The 3D QoE models developed and reported in D3.4 (Report on the Quality of Experience model and the audio and visual attention models) are used. In particular, the QoE models developed and reported for video+depth and stereoscopic video (L-R) are used in outlining the results. It was previously reported that the 3D-QoE models are proved to attain over ~90% correlation with 3D subjective tests scores. Hence, they can be regarded as a reliable measure of users' QoE.

#### 3.4.1 Test Setup

Tests are performed using two video sequences called *Street* (2 views + 2 depth maps 1280x720, 25fps) and *Music* (4 views + 4 depth maps, 1280x720, 25fps). For each of the test video sequence, 8 different streaming tests are performed. First four tests (Tests 1, 2, 3 and 4) contain one seed server and one peer. The QoE measurements are done on that peer. In the second set of tests (Tests 5, 6, 7 and 8), another seeder is added to the existing seed server and the peer. The total duration of the streamed multi-view content is selected as ~35 seconds.

For Music sequence tests, the total upload capacity of the seed server is set as 8.9 Mbps, 7.4 Mbps, 6.0 Mbps and 4.9 Mbps for Test 1, Test 2, Test 3 and Test 4, respectively. In tests 5, 6, 7 and 8, the upload capacities of both seed servers are held identical. They are set as 4.5 Mbps, 4.0 Mbps, 3.0 Mbps and 2.5 Mbps (each), for Test 5, Test 6, Test 7 and Test 8, respectively. The bit-rate of the VBR encoded multi-view video is around 8.2 Mbps in average, where the average base layer and enhancement layer rate are ~900 kbps each and the depth maps are encoded using single layer at ~450 kbps.

For Street sequence tests, the total upload capacity of the seed server is set as 4.5 Mbps, 3.9 Mbps, 3.4 Mbps and 2.0 Mbps for Test 1, Test 2, Test 3 and Test 4, respectively. In tests 5, 6, 7 and 8, the upload capacities of both seed servers are held identical. They are set as 2.5 Mbps, 2.0 Mbps, 1.7 Mbps and 1.0 Mbps (each), for Test 5, Test 6, Test 7 and Test 8, respectively. The bit-rate of the VBR encoded multi-view video is around 5.2 Mbps in average, where the average base layer and enhancement layer rate are ~1.2 Mbps each and the depth maps are encoded using single layer at ~500 kbps.

In both tests, user is interested in (by default) the central stereoscopic video pair. However, based on the adaptation decision engine, the Left video+depth pair is given a higher priority over the Right pair. As per the adaptation decision making algorithm, the enhancement layers of all views are given the least priority. The resulting 3D video quality index is scaled in between 0 and 1, where higher scores (close to 1) mean better subjective performance.

### 3.4.2 Results

Figure 18 and Figure 19 depict user's 3D-QoE results over the duration of the video service (ca. 35 seconds) for the Music test sequence, considering the QoE model for video + depth format. In other words, they show the user's 3D-QoE, if the stereoscopic video was rendered from either of the video+depth couples. Figure 20 depicts the user's 3D-QoE considering the QoE model for stereoscopic (i.e., Left-Right views) format. This format does not comprise the effects of the received depth maps. Similarly, Figure 21, Figure 22 and Figure 23 outline the QoE results for the Street test sequence.

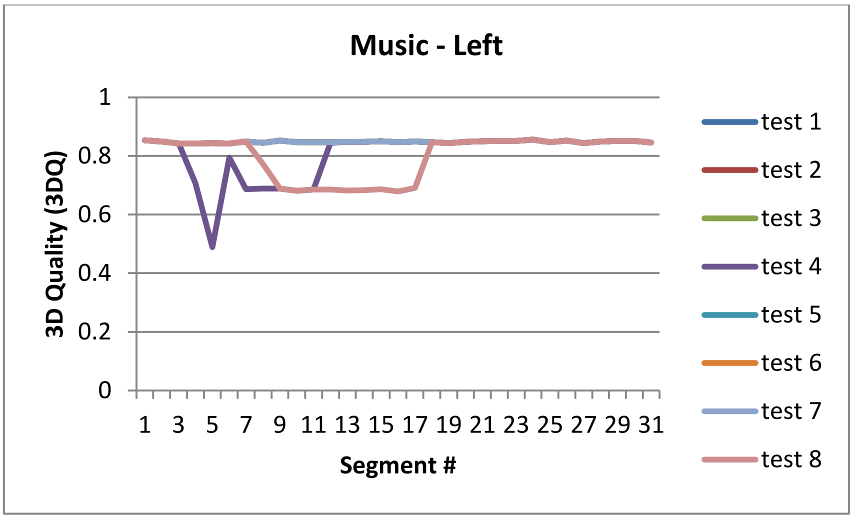


Figure 18: 3D video quality during playback of Music video (considering Left view)

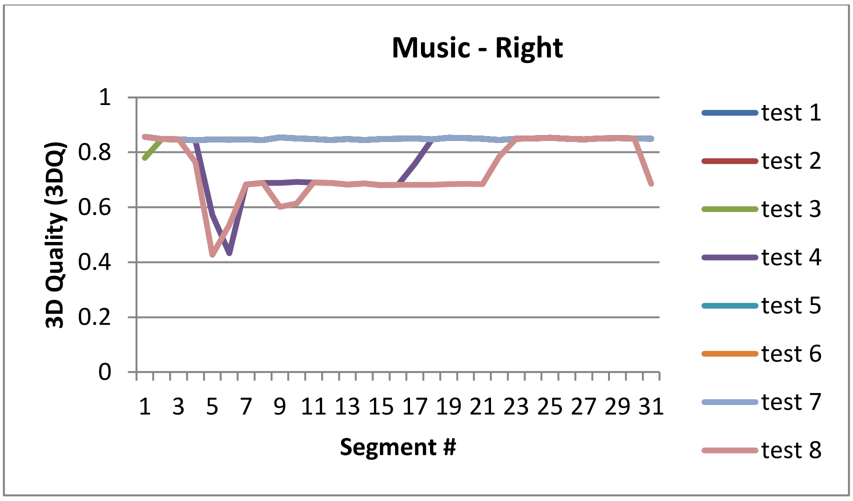


Figure 19: 3D video quality during playback of Music video (considering Right view)

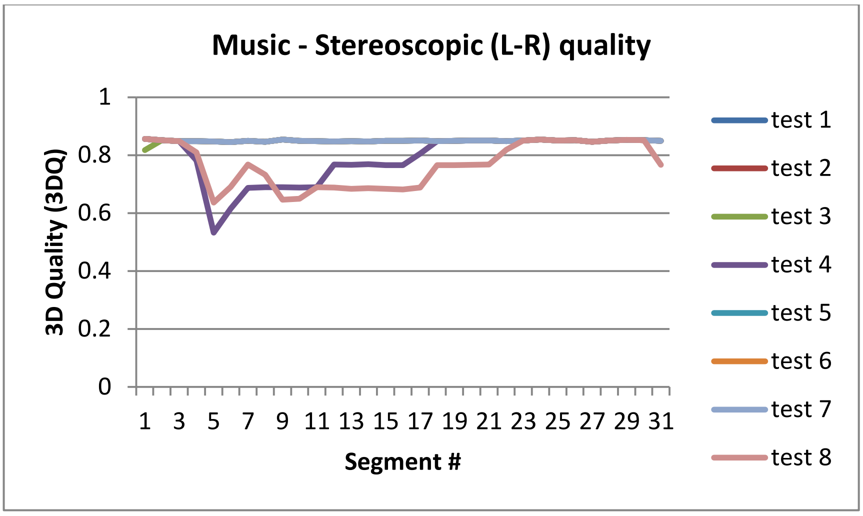


Figure 20: 3D video quality during playback of Music video (considering only colour videos)

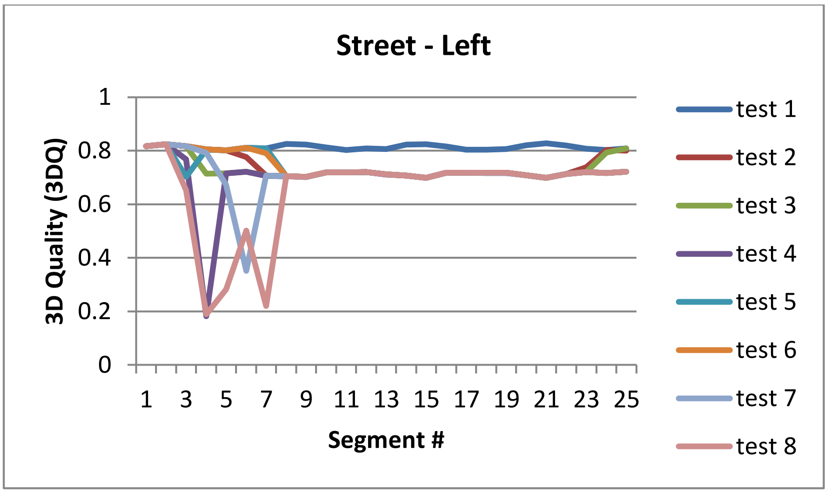


Figure 21: 3D video quality during playback of Street video (considering Left view)

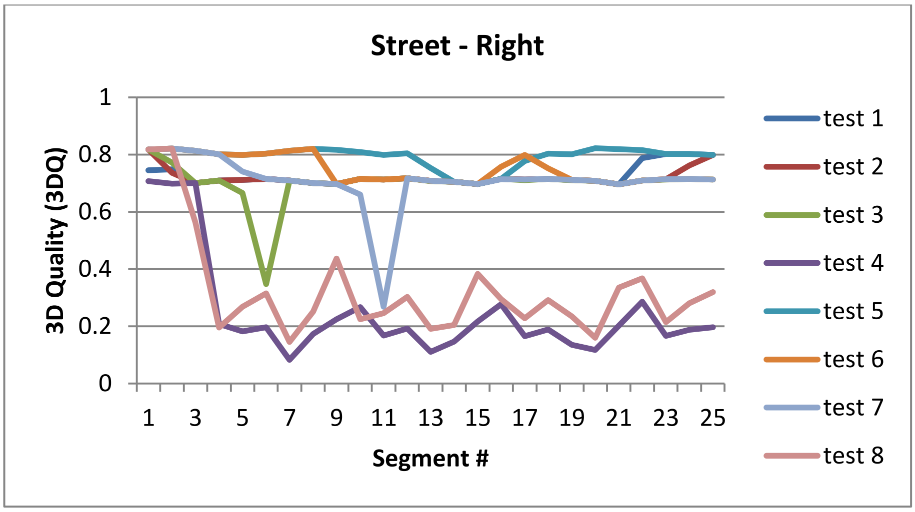


Figure 22: 3D video quality during playback of Street video (considering Right view)

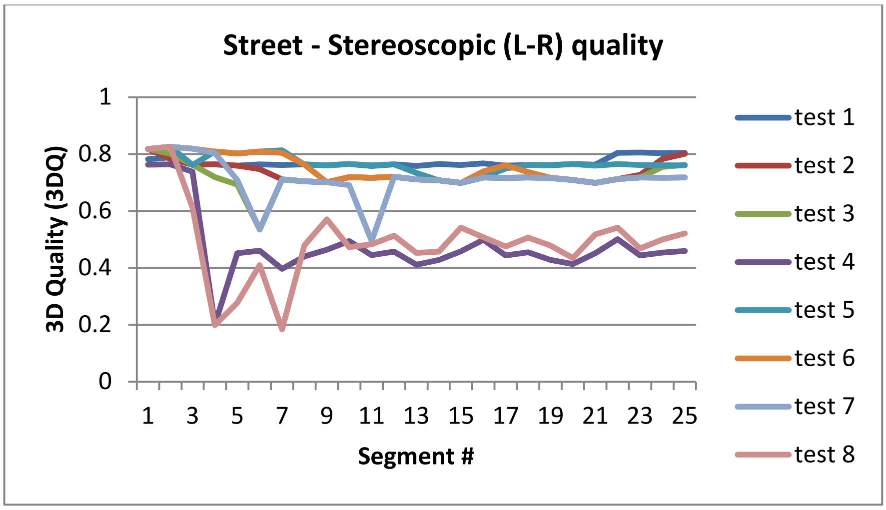


Figure 23: 3D video quality during playback of Street video (considering only colour videos)

Each segment in the horizontal axes of above figures represents a duration of 32 frames (i.e., ~ 1.3 seconds). According to the presented results, it is clearly seen that in both test videos, and taking into consideration all network tests, the 3D video quality perceived using Left video+depth pair is consistently better than the 3D video quality perceived using Right video+depth pair. This shows us the efficiency of the deployed adaptive streaming technique, which favours the more important streams (as per the directives of the adaptation decision engine) throughout the streaming and even under bandwidth scarcity. Remember that Left view had been given a higher priority than the Right view. It is also seen that according to the results of Tests 4 and 8, where the total useful bandwidth drops significantly, as compared to the required source coding rate (less than a half of it), the 3D video quality drops significantly for both pairs, also considering the stereoscopic (L-R) video quality. Nevertheless, the amount of quality degradation in Left pair is remarkably less than the degradation in the Right pair in Tests 4 and 8. Since the Music video originally contains 4 views, the outermost views are given the least importance and they are dropped for most of the time, especially in tests 3, 4, 7 and 8. Nevertheless, since they do not have an effect of the measured QoE in the centre view position, the effects of bandwidth scarcity in Tests 3, 4, 7 and 8 are not as devastating as in Street video, where all involved views are directly affecting the perceived 3D video quality. This is the reason why in average the worst results obtained Music scene are around 0.6, whereas it is around 0.2 in Street, when there are severe chunk losses or stream drops.

### 3.5 Performance of Methods for Bandwidth Reduction at the Server Side

DIOMEDES P2P has adopted a hybrid solution for the delivery of multimedia over IP and for this reason it has to two types of nodes in the network. The first type is the peers that consume the content and assist the video delivery when possible. The second type is the main seed servers that provide initial seeding service and distribute the content among peers. In order to achieve a scalable content delivery solution (against increasing number of peers), DIOMEDES P2P tries to minimize the server load whenever possible (See Section 2.6).

In the pursuit of this requirement, DIOMEDES P2P can take advantage of one of its unique features. In project DIOMEDES, the content from distinct channels (DVB and IP) is played synchronously and it is possible that content becomes available in the IP network before the broadcast starts in the DVB channel. This is an interval which allows the content to be distributed by P2P overlay in an efficient way.

In the following tests, we assume that the content is available in the IP channel one minute before the same content will be broadcasted from the DVB channel. We have initiated a varying number of peers and recorded the percentage of the chunks that are delivered by via

P2P rather than direct download from the server. Using this information, we have also calculated the bandwidth load on the server side.

### 3.5.1 Test Setup

In this test, we have used varying number of peers ranging from one to ten. . Similarly, the bandwidth of the peers is another test parameter. All the peers are initialized with at the same time (can be at most 2-3 seconds time difference) and the DVB signal indicated that the content is to be played out after one minute. Each test scenario has been tested 10 times, and the results in the following section are the average of these tests.

The content is the same content that we have used in Section 3.3.

### 3.5.2 Results

Table 3 presents the outcome of the streaming tests. For the sake completeness, we have also included the case where the number of peers in one. The left most column presents the number of peers. The column in the middle is the tests cases for different link capacities of the peers. It presents the achieved average number of chunks that are received by the P2P overlay. The right most column presents the highest possible ratio. For example, when there is only one peer, no chunks can be distributed via P2P overlay. For the case of two peers, at lease of the peer should download the content from the main seed server. At the best case the other peer will download all the chunks from the first one. In such a case, half of the chunks can be delivered via P2P overlay at best. So in theory, the percentage of chunks that can be delivered by P2P overlay increases as the number of peers increase.

One example result (light blue background) should be interpreted as follows. There are 8 peers. If the content is composed of 100 chunks then there should be 800 chunk transfers. Out of these 800, 68% of them (544) are transferred over P2P overlay, whereas the remaining 256 of them are directly downloaded from a main seed server.

Based on this interpretation we have the following conclusions. Increasing the number of peers augments the P2P delivery, this is very normal since the peers are more likely to find distinct peers within the swarm. The results also suggest that increasing the bandwidth capacity of the peers has a positive effect upon the P2P overlay. This is a promising result and it reveals that the system will become more and more efficient as the users' link capacity increases, which is natural process.

Using the table above, we can also **calculate** some other statistics.

Table 6 presents the average bitrate that each peer requests data from the main seed server. And the next table presents the sum of all peers request from the main seed server. Finally, Table 8 provides the overall bitrate reduction at the main seed server by the P2P system.

Table 5: Percentage of Chunks Received via P2P

| | **Percentage of Chunks Received via P2P** Peers Link Capacity | | | | |
|---|---|---|---|---|---|
| # Peers | 3.5 | 4.5 | 5.5 | 6.5 | Max (%) |
| 1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.00 |
| 2 | 42.5 | 45.5 | 46.8 | 47.9 | 50.0 |
| 4 | 60.4 | 64.4 | 64.8 | 67.7 | 75.0 |
| 6 | 65.5 | 69.9 | 70.8 | 75.0 | 83.3 |
| 8 | 68.0 | 71.7 | 72.7 | 77.3 | 87.5 |
| 10 | 69.7 | 73.4 | 74.3 | 78.8 | 90.0 |

Table 6: Calculated Average Download Rate from MSS (Mbps) per Peer

| | **Calculated Average Download Rate from MSS per Peer (Mbps)** Peers Link Capacity (Mbps) | | | |
|---|---|---|---|---|
| #Clients | 3.5 | 4.5 | 5.5 | 6.5 |
| 1 | 3 | 3 | 3 | 3 |
| 2 | 1.73 | 1.62 | 1.61 | 1.56 |
| 4 | 1.12 | 1.02 | 1.01 | 0.97 |
| 6 | 0.91 | 0.86 | 0.82 | 0.75 |
| 8 | 0.85 | 0.76 | 0.73 | 0.69 |
| 10 | 0.80 | 0.72 | 0.70 | 0.67 |

Table 7: Total Bitrate Requirement from MSS

| | **Calculated Total Bitrate Requirement (Mbps)** | | | |
|---|---|---|---|---|
| | Peers Link Capacity (Mbps) | | | |
| #Clients | 3.5 | 4.5 | 5.5 | 6.5 |
| 1 | 3.00 | 3.00 | 3.00 | 3.00 |
| 2 | 3.45 | 3.25 | 3.21 | 3.13 |
| 4 | 4.46 | 4.07 | 4.03 | 3.87 |
| 6 | 5.44 | 5.16 | 4.93 | 4.52 |
| 8 | 6.81 | 6.12 | 5.82 | 5.48 |
| 10 | 8.01 | 7.19 | 7.01 | 6.73 |

Table 8: Total bitrate Gain Due to P2P Overlay

| | **Calculated Total Bitrate Gain (Mbps)** Peers Link Capacity (Mbps) | | | |
|---|---|---|---|---|
| #Clients | 3.5 | 4.5 | 5.5 | 6.5 |
| 1 | 0 | 0 | 0 | 0 |
| 2 | 2.55 | 2.75 | 2.79 | 2.87 |
| 4 | 7.54 | 7.93 | 7.97 | 8.13 |
| 6 | 12.56 | 12.84 | 13.07 | 13.48 |
| 8 | 17.19 | 17.88 | 18.18 | 18.52 |
| 10 | 21.99 | 22.81 | 22.99 | 23.27 |

### 3.6 Performance Tests on SMDC

In this section, we evaluate the performance of the SMDC scheme that we have described in Section 2.5.4. We have compared three cases:

1. Always MDC: Each base layer chunk is always requested from multiple peers
2. Selective MDC: Base layer chunks are requested from MDC only if the conditions described in Section 2.5.4. ( See Figure 8: Chunk Scheduling in for MDC) are met.
3. No MDC: The case when MDC is disabled.

For each case, we compare the number of received base layer chunks for an interval of 10 seconds.

### 3.6.1 Test Setup

In order to test the performance of MDC, we need a very dynamic peer activity. For this reason, we set the link capacity of the peers (that are uploading chunks to the probe peer) between 0 and 1 Mbps as depicted in Figure 24. We have also provided the total sum of the bitrate that is available to the probe peer.

The same test content in Section 3.3.1 is used. It is 330 seconds long and composed of three layers (each at 1 Mbps).

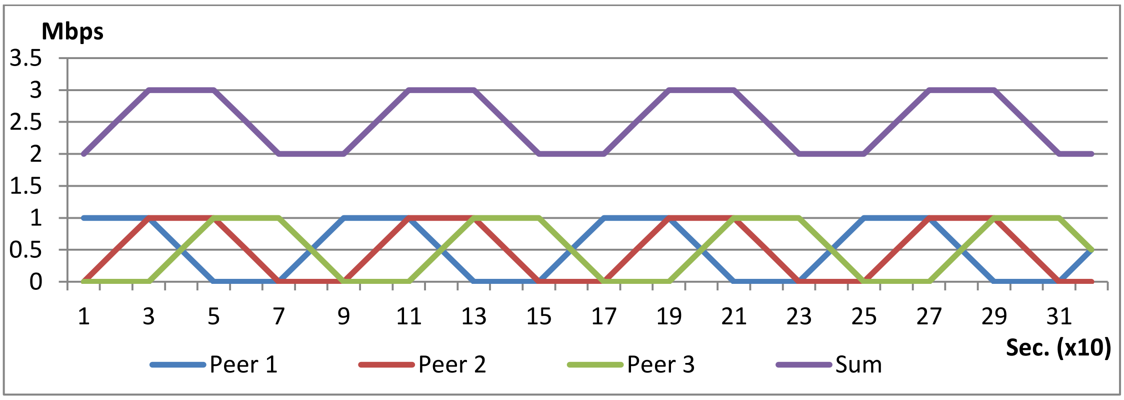


Figure 24: Peers Upload Capacity

### 3.6.2 Results

The number of received base layer chunks for different MDC cases are depicted in Figure 25. The results indicate the using MDC all the time can be significantly worse than even the case where MDC is not utilized. Naturally, the results depend on the redundancy that is caused by the adopted MDC scheme. On the other hand, using selective MDC seems to augment the video delivery and does not cause significant redundant data transfer. However, even with selective MDC scheme, redundancy is not completely eliminated since in some cases it can cause performance drops when compared to no MDC case.

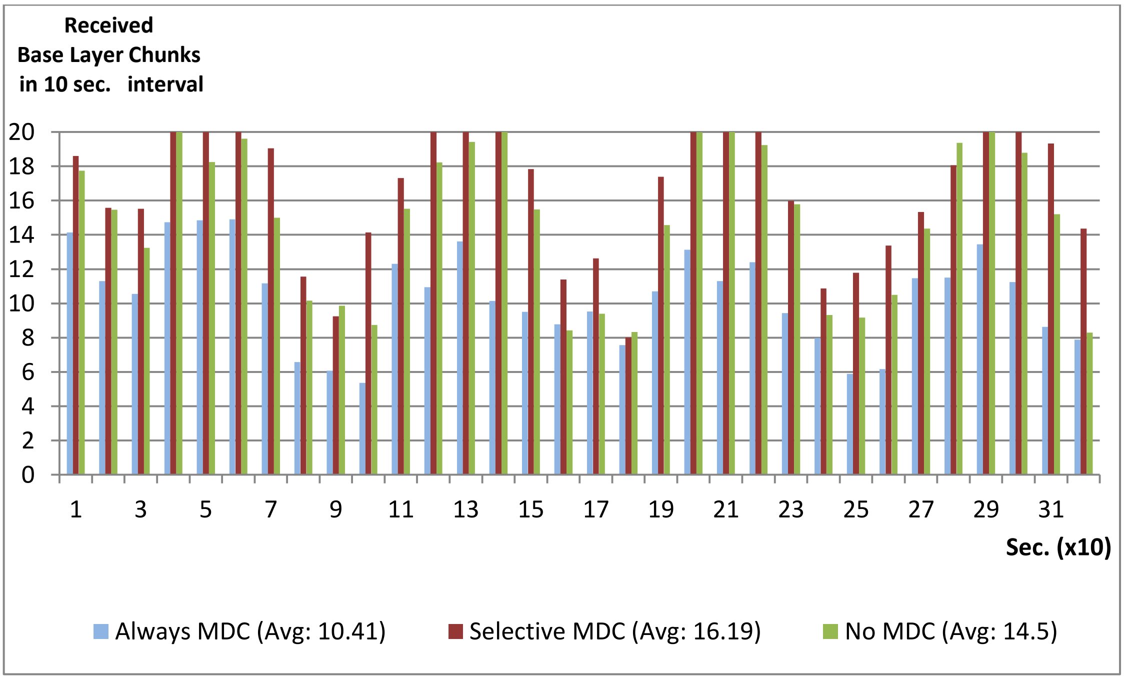


Figure 25: Number of received base layer chunks for different MDC cases

# 4 CONCLUSIONS

Based on the literature review and the tests performed in Chapter 3, the following conclusions about the P2P architecture of the DIOMEDES and the adopted MDC scheme are drown:

- Although being BitTorrent compliant has its own advantages, creating a video streaming architecture by modifying a system originally built to transmit large files with no timeliness concern can be inefficient. The well-known concepts like application layer framing must form the basis of a system that is created to deliver multimedia content over the Internet. Otherwise, issues described in the Tribler Software manual [2] are inevitable.
- As the first item indicates, using constant bitrate coding technique to deliver video over the Internet has hidden problems. In theory, using CBR coding with fixed chunk sizes may seem to work in harmony, however they can decrease the overall system performance due to the inherent redundancies. This is why the RTP protocol was developed, to enable a flexible solution that works in coherence with the characteristics of the IP networks. Therefore, the usefulness of VBR coding is exploited.
- Multiple description coding can augment the data transfer robustness if is used properly. Up to now, there is no known video service that is publicly available that uses MDC over mesh based topologies. In this document, the adopted MDC scheme is described (which is called selective MDC) for DIOMEDES-P2P.
- DIOMEDES-P2P has hybrid architecture for two reasons as explained in Chapter 1. One of the reasons why DIOMEDES-P2P is called hybrid is the adopted delivery mechanism that performs video delivery using P2P overlays that are assisted by main seed servers. For this reason, DIOMEDES P2P has an additional task, to keep the bit-rate requirement over the main seed server as low as possible. It is concluded that it is possible to achieve significant reduction in server bandwidth requirements when "intelligent chunk picking" policies are adopted.

# 5 REFERENCES


[1] N. Capovilla, et. al., “An Architecture for Distributing Scalable Content over Peer-to-Peer Networks”, in Proc. of International Conferences on Advances in Multimedia (MMEDIA’10), Athens, Greece, Jun. 2010.

[2] Tribler Protocol Specification, Jan. 2009. (available online: http://svn.tribler.org/bt2-design/proto-spec-unified/trunk/proto-spec-current.pdf)

[3] Socially aware, collaborative, scalable coding media distribution (online: http://www.saracen-p2p.eu/)

[4] X. Hei, Y. Liu, and K. W. Ross, "Inferring Network-Wide Quality in P2P Live Streaming Systems," JSAC, vol.25, no. 10, Dec. 2007.

[5] Z. Liu, et. al., "Layerp2p: using layered video chunks in p2p live streaming," *IEEE Trans. on Multimedia*, vol. 11, no. 7, pp. 1340-1352, 2009.

[6] N. Magharei, R. Rejaie, and Y. Guo, B, "Mesh or multiple-tree: A comparative study of live P2P streaming approaches", in Proc. IEEE INFOCOM’07, Anchorage, AK, May 2007

[7] M. Eberhard et al., “Knapsack problem-based piecepicking algorithms for layered content in peer-to-peer networks,” *in ACM Workshop on Advanced Video Streaming Techniques for P2P Networks and Social Networking*, 2010, pp. 71–76.

[8] http://en.wikipedia.org/wiki/Constant_bitrate

[9] S. Savas, G. Gurler, M. Tekalp, E. Ekmekcioglu, S. Worrall, A. Kondoz, “Adaptive streaming of multi-view video over P2P networks,” *in Proc. Journal of Signal Processing: Image Communications*, *Special issue on Advances in video streaming for P2P networks,* in press.

[10] P. Baccichet, T. Schierl, T. Wiegand, and B. Girod. Low-delay peer-to-peer streaming using scalable video coding. Packet Video 2007, pages 173{181, 2007

[11] C.G. Gurler, B. Gorkemli, G. Saygili, and M. Tekalp, "Flexible transport of 3D video over networks" in Proceedings of the IEEE, Vol. 99, No. 4, April 2011

[12] R. Penno, S. Raghunath, R. Woundy, V. Gurbani, J. Touch “LEDBAT Practices and Recommendations for Managing Multiple Concurrent TCP Connections”, IETF draft, Feb. 26,2000

[13] online available: http://www.linuxfoundation.org/collaborate/workgroups/networking/netem

[14] V. K. Goyal, “Multiple description coding: Compression meets the network,” *IEEE Signal Processing Magazine*, pages 74–93, Sep. 2001

[15] R. Penno, S. Raghunath, R. Woundy, V. Gurbani, J. Touch “LEDBAT Practices and Recommendations for Managing Multiple Concurrent TCP Connections”, IETF draft, Feb. 26,2000

# Appendix A: Glossary of abbreviations

| | |
|---|---|
| **A** | |
| ADEM | Adaptation Decision Engine Module |
| AU | Access Unit |
| AVC | Advanced Video Coding |
| **C** | |
| CBR | Constant Bit-Rate |
| CM | Control Module |
| **D** | |
| DP | Depth Perception |
| DVB | Digital Video Broadcasting |
| **E** | |
| EOF | End Of File |
| **G** | |
| GOP | Group of Pictures |
| GUID | Globally Unique Identifier |
| **H** | |
| HTTP | Hypertext Transfer Protocol |
| **I** | |
| IQ | Image Quality |
| IP | Internet Protocol |
| **K** | |
| KPI | Key Performance Index |
| **M** | |
| MDC | Multiple Description Coding |
| MSS | Main Seed Server |
| MVV | Multi-View Video |
| **N** | |
| NAL | Network Abstraction Layer |
| **P** | |
| P2P | Peer-to-Peer |
| PCR | Program Clock Rate |
| PES | Packetised Elementary Stream |
| PID | Program ID |
| PPS | Picture Parameter Set |
| **Q** | |
| QoE | Quality of Experience |
| QoS | Quality of Service |
| QP | Quantisation Parameter |
| **R** | |
| RTP | Real Time Protocol |
| **S** | |
| SEI | Supplementary Enhancement Information |

| | |
|---|---|
| SMDC | Scalable Multiple Description Coding |
| SNR | Signal-to-Noise Ratio |
| SPS | Sequence Parameter Set |
| SVC | Scalable Video Coding |
| **T** | |
| TCP | Transmission Control Protocol |
| TS | Transport Stream |
| **X** | |
| XML | Extensible Mark-up Language |
| **V** | |
| VBR | Variable Bit-Rate |
| VCL | Video Coding Layer |