Community Tools for Massively Multiplayer Online Games

Shakeel Ahmad*, Christos Bouras†, Raouf Hamzaoui*, Jiayi Liu§, Andreas Papazois†, Erez Perelman‡, Alex Shani ‡, Gwendal Simon §, George Tsichritzis †

*Department of Engineering
De Montfort University, Leicester, UK
sahmad@dmu.ac.uk, rhamzaoui@dmu.ac.uk

†Computer Technology Institute & Press "Diophantus"
N. Kazantzaki, GR26500 Patras, Greece
bouras@cti.gr, papazois@ceid.upatras.gr, tsixritzis@cti.gr

§Institut Telecom
Telecom Bretagne, France
jiayi.liu@telecom-bretagne.eu, gwendal.simon@telecom-bretagne.eu

‡Exent Technologies
Bazel 25, 49125 Petach-Tikva, Israel
eperelman@exent.com, ashani@exent.com

***Abstract*—One of the most attractive features of Massively Multiplayer Online Games (MMOGs) is the possibility for users to interact with a large number of other users in a variety of collaborative and competitive situations. Gamers within an MMOG typically become members of active communities with mutual interests, shared adventures, and common objectives. We present the EU funded Community Network Game (CNG) project. The CNG project provides tools to enhance collaborative activities between online gamers and offers new tools for the generation, distribution and insertion of user-generated content in MMOGs. CNG allows the addition of new engaging community services without changing the game code and without adding new processing or network loads to the MMOG central servers. The user-generated content considered by the CNG project includes 3D objects and graphics, as well as screen-captured live video of the game, which is shared using peer-to-peer technology. We survey the state of the art in all areas related to the project and present its concept, objectives, and innovations.**

***Keywords*-Massively Multiplayer Online Games; user generated content; P2P video streaming; graphics insertion.**

I. Introduction

Massively Multiplayer Online Games (MMOGs) allow a large number of online users (in some cases millions) to inhabit the same virtual world and interact with each other in a variety of collaborative and competing scenarios. MMOG gamers can build and become members of active communities with mutual interests, shared adventures, and common objectives. Players can play against other players (player versus player) or build groups (guilds) to compete against other groups (realm versus realm) or against computer-controlled enemies. MMOGs are rapidly gaining in popularity. According to IDATE [1], the number of MMOG players worldwide is expected to grow from 82 million in 2008 to more than 140 million by 2012.

This paper presents the Community Network Game (CNG) project [2], an EU funded project within the Seventh Framework Programme. The project, which started in February 2010 and has a duration of 30 months, aims at enhancing community activities for MMOG gamers. This is achieved by providing Web collaboration tools and developing new tools for the generation, distribution and insertion of User-Generated Content (UGC) into existing MMOGs. This UGC may include textures and 3D objects to be added to the game, live video captured from the game screen and streamed to other players, as well as videos showing walkthroughs, game tutorials, or changes in the virtual world to be watched on demand.

The main technologies proposed by the CNG project are the In-game Graphical Insertion Technology (IGIT) and a Peer-to-Peer (P2P) system for the distribution of live video. IGIT is an innovative technology of replacing or inserting content into a game in real time without the need to change the game code in the client or server. For example, billboards can be inserted, tattoos can be added to in-game characters, an area on the screen can be assigned to display user information, and any type of window (browser, chat, etc.) can be inserted floating on or outside the game area. The technology can be implemented on multiple games, making it possible to create a community that is not limited to a specific game or publisher.

Enabling thousands of users to communicate UGC represents a significant challenge to networks already occupied with the MMOG client-server data. The CNG project develops new techniques for UGC distribution that are "friendly" (supportive and not disruptive) to the MMOG client-server traffic. The key innovation is a P2P system that allows MMOG gamers to stream live video of the game without interrupting the MMOG data flow and without the need to

upload the video data to a central server.

The remainder of the paper is organized as follows. Section II gives an overview of the state-of-the art in the areas of UGC, Web collaboration tools, P2P live video streaming, and game adaptation technologies. Section III presents the project's concept, objectives and technologies. Section IV concludes the paper by discussing the project's benefits and expected impact. The paper is an extension of the conference paper [3].

II. Related Work

In this section, we review the state-of-the art in the areas of UGC, Web collaboration tools, P2P live video streaming, and game adaptation technologies.

A. UGC

UGC includes various kinds of media content produced by end-users. In a game context, for example, this may be screen-captured video. Another example of UGC is the various mods created by the users. Sharing and remixing UGC is a widespread online activity that crosses borders of age and gender. Avid players go to great lengths in their efforts to create shared content in which they reveal their mastery. Additional data layers are always included: narration, animation and primarily soundtrack. Most MMOG-based UGC content is confined to dedicated game company sites as in World of Warcraft [4]. Many MMOG games also have their own community pages in social networking sites such as Facebook. In April 2010, Facebook released significant updates to its API by allowing external websites to uniformly represent objects in the graph (e.g., people, photos, events, and community pages) and the connections between them (e.g., friend relationships, shared content, and photo tags). As a result, the Facebook API [5] can provide an unprecedented bridge between gamespaces and the social web.

Current UGC tools can be classified into three categories: tools for capturing, for editing, and for uploading/broadcasting.

Capturing: Capturing videos can be done within the gamespace as in Spore [6]. This is not a common feature in MMOGs. More commonly, capturing is done with external video capture software such as Camtasia [7] or Fraps [8]. Fraps is the preferred software for users who want to capture high quality video. However, its free version has a 30 s recording limitation.

Editing: UGC sharing and remixing within game platforms is currently not supported. To edit the video and add effects, narration, soundtrack and text overlays, users tend to use readily available software such as Windows Movie Maker for Windows or iMovie for Mac that allow for the inclusion of additional content: audio, images and other videos. Annotations can only be added after the capture is done and cannot include other participants' comments.

Uploading/broadcasting: Once a user has captured and edited the video, a final step is needed to upload it for viewing. Many MMOG players use sites such as YouTube to share their game-based UGC. In 2008, Maxis incorporated YouTube APIs within their game, Spore, enabling players to upload videos of their creations to their YouTube account with only two clicks [9]. The collaboration between YouTube and a game creator (Electronic Arts), including revenue share from advertisements, is unique to date. Players of other games need to upload their video creation from their computer and cannot do it from within the game itself [10].

A user can capture the video of the game and broadcast it live to other users via a video server. This feature is offered by Xfire [11], which allows anyone to watch a live feed of a user's game screen. When a user begins a stream, a chat room is opened that anyone watching the live feed can join.

B. Web collaboration tools

Web 2.0 based collaboration applications may include instant messaging, audio and video chat, file sharing and online voting and polling. For audio/video capturing and playback the Flash software platform [12] is commonly deployed. Other solutions are the Java Applet technology or standalone applications which run on a Web browser and offer interoperability over different platforms. For instant messaging, online polling/voting and file sharing, Asynchronous JavaScript and XML (AJAX) [13] are commonly used. AJAX allows Web applications to retrieve data from the server asynchronously in the background without interfering with the display and behavior of the existing page. The use of AJAX techniques has led to an increase in interactive interfaces on webpages. Finally, for WWW client-server communication, most of the Web 2.0 applications are based on Simple Object Access Protocol (SOAP) [14]. SOAP relies on XML as its message format, and usually relies on other Application Layer protocols, most notably the Remote Procedure Call (RPC) and HTTP.

In the following, we give examples of Web 2.0 collaboration software.

1) Instant messaging: Instant messaging software is mainly based on AJAX technology. A typical AJAX chat application uses a database (MySQL) and AJAX to store and retrieve the users' messages and pass them between the client and the server. Examples of instant messaging software include AJAX Chat [15], Google Talk [16], ChatZilla [17], Mibbit [18], and Java/JavaScript Chat [19].

2) File sharing: Examples of popular Web2.0 file sharing systems include Meebo [20], iGoogle [21], Orkut [22], and FlashComs Community chat [23].

3) Audio and video chat: Audio and video chat applications are based on the Flash Platform. Some typical examples of Web-based audio and video chat tools are AVChat 3 [24], Red5Chat [25], MeBeam [26], Web Voice Chat [27], and 123 Live Help Chat Server Software [28].

Table I: POPULAR CHAT TOOLS.

Tools	Instant Messaging	Audio and video chat	File sharing	Protocols
CGI:IRC	Perl/CGI	Not supported	Not supported	IRC
PJIRC	Java Applet	Not supported	Not supported	IRC
qwebirc	Ajax Applet	Not supported	Not supported	IRC
Parachat	Java Applet	Not supported	Not supported	Jabber/XMPP
Pichat	Ajax	Not supported	Not supported	Unknown
Facebookchat	Ajax	Not supported	Not supported	Jabber/XMPP
eBuddy	Ajax	Not supported	Not supported	Jabber/XMPP
Omegle	Ajax	Flash	Not supported	Jabber/XMPP
webcamnow	Ajax	Flash	Not supported	Jabber/XMPP
JatChat	Java Applet	Java Applet	Not supported	Jabber/XMPP
campfire	Ajax	Not supported	Ajax	Unknown
Single Operator Ajax chat	Ajax	Not supported	Ajax	Unknown

Table I lists some widely used chat tools together with their underlying technology.

Table II: POPULAR VOTING AND POLLING APPLICATIONS.

Tools	Technology
Poll4Web	Flash
Flash Web Poll	Flash
ABPollMaster Polling	Java Applet
Fly06 Poll	Ajax

Table III: POPULAR BLOGWARES.

Tool	Technology
Kontain	Flash
Blogsmith	Ajax
TypePad	Ajax
Gawker bespoke software	Ajax

4) Online voting and polling: Examples of Web-based collaborative voting and polling tools are VotingPoll [29], DPolls [30], and XML Flash Voting Poll [31]. Table II lists other examples and the technology used for their implementation.

5) Blogging: Important tools used for the building of online blogging applications are WordPress [32], and Movable Type [33]. Table III lists popular blogwares.

C. P2P live video systems

Traditional client-server video streaming systems have critical issues of high cost and low scalability on the server. P2P networking has been shown to be cost effective and easy to deploy. The main idea of P2P is to encourage users (peers) to act as both clients and servers. A peer in a P2P system not only downloads data, but also uploads it to serve other peers. The upload bandwidth, computing power and storage space on the end user are exploited to reduce the burden on the servers.

Viewers of a live event wish to watch the video as soon as possible. That is, the time lag between the video source and end users is expected to be small. In a live streaming system, the live video content is diffused to all users in real time and video playback for all users is synchronized. Users that are watching the same live video can help each other to alleviate the load on the server. P2P live streaming systems allow viewers to delete the historic data after the playback, and hence have no requirement for any data storage and backup.

Based on the overlay network structure, the current approaches for P2P live streaming systems can be broadly classified into two categories: tree-based and mesh-based. In tree-based systems, peers form an overlay tree, and video data are pushed from the parent node to its children. However, a mesh-based system has no static streaming topology. Peers pull video data from each other for content delivery. Over the years, many tree-based systems have been proposed and evaluated, however, never took of commercially. Mesh-based P2P streaming systems achieve a large-scale deployment successfully, such as PPLive [34], PPStream [35], etc.

1) Tree-based systems: Many early P2P streaming systems use a tree-based approach that is typically based on application-level multicast architectures. Tree-based systems, such as ESM [36] and P2Cast [37], organize peers into a tree structure for delivering data. The data are diffused following this well-defined structure, typically pushed from a peer to its children. Tree-based solutions are perhaps the most natural and efficient approach, but they face several challenges. One major drawback of tree-based systems is the system fragility due to peer churn. A peer departure will disrupt data delivery to all its descendants, particularly for the peers in the higher level of the tree. The high dynamicity of peers in a P2P network potentially deteriorates transient performance. Another drawback is the under-utilized upload bandwidth of the peers. The leaf nodes in the tree cannot contribute any upload bandwidth resource to the system. Since a majority of nodes are leaves in the tree structure, this significantly reduces the overall efficiency. To address the issues of leaf nodes, multi-tree structures were introduced [38], [39]. In a multi-tree system, the source encodes the stream into several sub-streams and diffuses each sub-stream along one sub-tree. Each peer participates in many or all

Table IV: TRANSPORT PROTOCOLS IN P2P LIVE VIDEO STREAMING SYSTEMS.

System	Protocol
CoolStreaming [41]	TCP
PPStream [65]	TCP
PPLive [66]	Combination of TCP and UDP
TVAnts [65]	Combination of TCP and UDP
Joost [67]	UDP and TCP with UDP being the dominant traffic
SopCast [66]	Combination of TCP and UDP
[49]	UDP with FEC
[53]	UDP with ARQ
[61], [62], [68]	UDP
GridMedia [69]	UDP
iGridMedia [70]	UDP
PULSE [71]	UDP for control messages and TCP for data exchange
R2 [72]	UDP or TCP when UDP cannot be used due to firewall blocking

sub-trees to retrieve sub-streams. Hence, a peer might be deployed on an intermediate position in one sub-tree or a leaf position in another sub-tree. Compared with the single-tree approach, the multiple-tree solution has two advantages. First, the system's robustness is improved, as the failure of a high-level node would not completely disrupt all its descendants. Second, the upload bandwidth of all nodes could be well utilized, since each node stands a good chance to be both a leaf and an intermediate node. However, since the multiple-tree approach is still a tree-based solution, the drawbacks of tree-based systems remain basically unsolved. First, the construction and maintenance of the multiple-tree structure are costly because of frequent peer churn behaviours. Second, the upload bandwidth contribution of a node, which depends on the position in each sub-tree, is deficient. Furthermore, the design involves overhead.

Viewers in P2P live streaming systems only focus on the live video data that currently are output from the source. Hence, the video playbacks for all users are synchronized. In tree-based P2P live systems [36], all users participating in a video streaming session can form a tree at the application layer with the root at the video source. Each peer receives the live video data from its parent and immediately forwards the data to its children. Usually peers at lower levels receive the live data after peers at upper levels. The major consideration is to balance the depth of the tree and the out-degree of the intermediate nodes. Multi-Tree based approaches for P2P live systems are described in [38].

2) Mesh-based systems: In a mesh-based P2P streaming system, peer relationships are built and terminated according to data availability and bandwidth availability on peers. A video is typically divided into many chunks. Moreover, a tracker server maintains the relationship between peers and video data. Then, a peer can dynamically connect to a peer list that is chosen randomly from the tracker server according to which chunk the peer requests. After that, the peer maintains multiple neighbours and exchanges chunks with these neighbours simultaneously. A gossip protocol [40] is typically used for the topology management. Peers also periodically exchange information of the chunk availability using a buffer map. Usually, a chunk is pulled by a peer from its neighbours who have the requested chunk. The pull policy can avoid redundant chunk transmission.

If a neighbour leaves, the peer can continue retrieving chunks from other neighbours. The peer also explores some new neighbours to keep a certain number of neighbours. Due to the maintenance of multiple neighbours, mesh-based systems are highly robust against peer churns and fully utilize users upload bandwidth. However, transmission delay presents a challenge to mesh-based systems (for example, long start-up delay and channel switching problems for live streaming systems).

Many successful P2P live streaming systems [34], [35], [41] use the mesh-based streaming approach. The design of mesh-based P2P live streaming systems is relatively simple. All users are interested in the same live data. All chunks downloaded at a peer are always useful to other peers that have not retrieved these chunks. Some studies investigate the quality of peering connections. Several strategies are proposed to construct the peer relationship. The first consideration is the workload and resource availability on both peers, such as the current number of connections, upload and download bandwidth, and system resources. Other considerations are the network condition, which includes end-to-end delay and loss rate, and the network proximity, including geographical position, bandwidth, delay and network distance.

3) Server-assisted P2P systems: Most peer-to-peer systems rely on a server, either a bootstrapping server or a tracker server. A bootstrapping server is only used when a new peer joins an overlay. The bootstrapping server is expected to give to this newcomer a list of peers that are currently in the system. In this way, the new peer can quickly open connections. It has been shown, however, that a popular P2P streaming system like PPLive fails to provide accurate information to newcomers, resulting in a too long start-up delay [42]. Actually, a large proportion of peers that are given by the bootstrapping server do not answer the initial request of the newcomer, either because they are

no longer in the system, or because they do not need any new connections. A tracker server extends the bootstrapping function. Every peer periodically sends a message to the tracker, which gives in return a list of peers (peerlist). That is, the participants to a peer-to-peer system can discover new peers on a periodic basis. The bit-torrent system has popularized this hybrid architecture, which guarantees, among other suitable properties, that new peers can quickly find matching peers.

In general, implementations of tracker-based peer-to-peer systems are simple. The tracker sends to a requesting peer a list of randomly chosen peers among the set of peers that are expectedly active in the system. Interestingly, the resulting topology is a random regular graph: every peer is connected to a given number of randomly chosen other peers. This random-like underlying topology is interesting on several aspects, especially random regular graphs are connected with high probability (so, any information is accessible from any peer), and the diameter of a random regular graph is small (therefore any information is close to any peer, if it is able to find it).

This topology links "acquaintances". A peer can contact any subset of peers in the peerlist, but it is free to choose, among them, some privileged peers with which it will exchange data. This presents some problems. First, the peerlist contains peers exhibiting a broad scope of capacities, although the overlay tends to connect peers having similar characteristics [43]. The overlay would converge faster if the peerlist could contain preferentially the peers having the closest characteristics to the requester. However, it would require to authorize the tracker to determine as accurately as possible the capacity of peers, which appears to be impossible or costly in many cases. Second, the peerlist topology does not take into account the location of peers. Therefore the overlay wastes network resources.

4) Hybrid CDN-P2P systems: Peer-assisted (PAS) Content Delivery Networks (CDNs) have attracted a lot of attention in recent years. In this section, we present the architecture of a real-world CDN-P2P live video streaming system called LiveSky [44], which has been deployed in China. The system is designed to solve a set of problems in current CDN and P2P live video streaming systems such as scaling, fast startup and upload fairness.

Server side organization: The CDN overlay is constructed according to a tree-based structure, where leaves are edge servers, whose role is to serve end users. All other intermediate nodes are core servers, which are responsible for delivering content to edge servers. Because of their work load, edge servers are not allowed to transfer content between each other. To realize a P2P organization at the client side, an edge server has several roles: 1) a regular server for legacy clients; 2) a tracker for the P2P operation; 3) a seed in the P2P system.

Client side organization: There are two types of clients: legacy clients and P2P clients. Legacy clients are served in the traditional CDN manner and receive low quality streams. P2P clients are organized in a hybrid scheme proposed in [45], [46] that combines the multi-tree and mesh topologies. As usual a video is divided into several substreams. Each substream contains nonconsecutive frames. The peers that host the same substream compose a tree-based overlay. This ensures upload fairness of each peer. On the other hand, peers also use a mesh-style pull mechanism to retrieve missing frames for continuous playback. This enhances the robustness of the network. Moreover, P2P clients are allowed to access high quality videos.

Adaptive scaling and improvements: In the system, each edge server decides whether a new arrival client should be treated as a legacy client or a P2P client independently. A threshold is pre-configured in every edge server. When the number of clients is below the threshold, all clients retrieve the content directly from the edge server. If the number of clients exceeds the threshold, new arrival clients will be redirected to other clients to form a P2P organization. Both the threshold and the capacity of an edge server are calculated by some parameters, including the level of the P2P tree overlay, peer arrival rate, peer leaving rate and peer contribution rate. When an edge server reaches its capacity limitation, new clients will be redirected to a less loaded edge server.

Fast startup: Startup time is optimized in LiveSky in two ways. First, the buffer size is reduced to 15 seconds. Second, the first request of a client is always replied directly by an edge server, thus it is very quick to retrieve startup streams.

5) Video transmission protocols: In private, well-managed IP networks, the quality of service (QoS) is maintained by calibrating the end-to-end infrastructure. This is not possible in P2P overlays since they are built on open IP networks, which are best-effort in nature. Real-time video communication over P2P overlays on the public Internet mainly relies on the transmission control protocol (TCP). TCP guarantees reliable transmission of the data by automatic retransmission of lost packets. However, as TCP requires in order delivery of the data and keeps on retransmitting a packet until an acknowledgement is received, significant delays may be introduced. Further delays are caused by the congestion control algorithm used by TCP, which reacts to packet loss by reducing the transmission rate, leading occasionally to service interruption. This presents a serious drawback for real-time video communication where the data must be available to the receiver at its playback time. Lost and delayed packets that miss their playback deadline not only are useless, they also consume the available bandwidth unnecessarily.

An alternative to TCP is to use UDP as the transport protocol and apply application-layer error control. For example, the Darwin Streaming Server, which is the open-source version of Apple's QuickTime Streaming Server, uses a simple

timeout-based ARQ scheme [47]. The Helix DNA streaming system, which is the open-source version of RealNetworks Helix streaming suite, also uses timeout based ARQ [47]. Windows Media uses a selective retransmission scheme. If the client detects gaps in the packet sequence numbers, it sends a retransmission request to the server, which retransmits the missing packets. Packet retransmissions are limited to a certain percentage of the available bandwidth and packets to be retransmitted are prioritized according to their content. Audio packets are given the highest priority. Video packets close to their playout deadlines are given the lowest priority, on the presumption that retransmissions are most likely to miss the playout deadline [48]. VideoLAN, a popular open-source streaming system, uses either TCP or UDP without packet loss mechanisms [47]. These streaming techniques are suitable for well-managed networks. However, they face considerable problems in open IP networks where the packet loss rate may be significant, and the available bandwidth may be variable. In P2P overlays, in particular, packet loss is not only due to congestion at routers but also to the heterogeneity in node stay-time duration.

Most P2P streaming systems use TCP (CoolStreaming, PPStream), a combination of UDP without error control and TCP (PPLive, TVAnts), UDP with Forward Error Correction (FEC) [38], [49], [50], [51], [52], and ARQ [53]. Another approach is Multiple Description Coding (MDC) [54], [55]. However, MDC schemes are rarely used in practice because they rely on non-standard video coders.

Thomos and Frossard [56] use network coding with rateless codes [57] for P2P video streaming. The technique exploits path diversity and lessens the burden of re-encoding on an intermediate forwarding peer.

Wu and Li [58] also use network coding based on rateless codes for P2P live video streaming. A peer can recover the original video source block by receiving enough encoded symbols from multiple receivers. As soon as a receiving peer successfully decodes the source block, it becomes a source and applies rateless coding on the decoded source block to generate encoded symbols for other peers. To avoid receiving redundant symbols, each peer uses a different seed for rateless encoding. The authors propose a distributed algorithm for best peer selection and optimize rate allocation to guarantee minimum delay.

Grangetto, Gaeta, and Sereno [59] propose an improvement to the method of Wu and Li [58]. In their method, called 'Relay and Encode' (RE), a receiving peer relays the received encoded symbols immediately. Once it decodes the source block, a rateless code is applied on the source block and newly produced encoded symbols are sent to its children. The authors show that RE has a lower delay than the method of [58]. However, the paper does not consider the effects of varying channel conditions and does not exploit feedback to minimize bandwidth usage. Moreover, the protocol is not robust against failures. For example, if one peer cannot decode the source block, all its descendent peers are affected. A similar system is proposed by the same authors in [60]. Here, receiver feedback is used to ask the sender to stop sending more symbols when the source block is decoded.

In [52], rateless coding is used to make a P2P VoD system resilient to peer churn. The source partitions the video into source blocks and applies rateless coding on these blocks. During a push phase, for each source block, distinct groups of encoded symbols are distributed among a number of volunteering peers, which may not be interested to watch the video. This pushing can be done during low network utilization time. In a pull phase, a peer who wants to watch the video needs to collect a minimum number of distinct encoded symbols from any subset of volunteering peers.

Setton, Noh, and Girod [61] propose a system for live video streaming over P2P networks aimed at low latencies and congestion avoidance. Video packets are sent using UDP/IP, and a scheduling algorithm is used to maximize the received video quality, while minimizing network congestion.

In [62], a P2P live video streaming system aimed at low delays is presented. The system uses the Stanford Peer to Peer Multicast Protocol to build multiple complementary multicast trees, all originating at the source. The source exploits path diversity and sends different packets over different multicast trees. The video is compressed with the Scalable Video Coding (SVC) and packets are sent over UDP/IP. Depending on the available bandwidth and packet loss rates, intermediate nodes decide how many layers should be sent to their children. A simple ARQ mechanism is used to deal with packet loss.

One limitation of the UDP protocol is the lack of a congestion control mechanism. Congestion control with UDP can be realized with the Datagram Congestion Control Protocol (DCCP) [63]. DCCP uses an Explicit Congestion Notification (ECN) bit, which is set on by a congested router. When a receiver receives a packet with an ECN bit set on, it asks the sender to react to the congestion accordingly. However, most routers disable ECN [64].

Table IV gives an overview of the transport protocols used in P2P systems.

D. Game adaptation technologies

In-game technologies have been used in the gaming market for several years. The gaming industry has adopted these technologies to increase its revenue by finding more financial sources and by attracting more users. In-game overlay allows to view and interact with windows outside the game, but without "Alt-Tabbing". It does so by rendering the window inside the game. Texture replacement enables to replace an original game texture with a different texture. In this way, the newly placed textures are seen as part of the original game content. This method is commonly used for dynamic

Table V: IN-GAME TOOLS. AN X INDICATES THAT THE TOOL IS OFFERED BY THE PRODUCT.

Product	XFIRE	PLAYXPERT	Massive	FreeRideGames	Double Fusion	Steam	Overwolf	Raptr
Texture replacement			x		x			
Game resize				x				
In-game overlay	x	x		x		x	x	x
Video capture	x	x					x	
Video edit								
Video upload	x						x	
Live video	x							
Instant messaging	x	x				x	x	x
Audio chat	x					x		x
File sharing								
Online blogging								
Need for SDK	No	No	Yes	No	Yes	Yes	No	No

in-game advertisement. Game size modification technology adapts the original game by decreasing its original size and surrounding it with an external content.

Some of these technologies are distributed as an external utility that can overlay a pack of games while others are part of MMOG service features which are provided to the users.

The main available in-game adaptation products on the market are Xfire [11], PLAYXPERT [73], FreeRidesGames [74], Massive [75], Double Fusion [76], Steam [77], Overwolf [78], and Raptr [79]. Some of the products require for the game developer to integrate the product's Software Development Kit (SDK) (for each game to be developed the game developer must use the products SDK). The use of those in-game adaptation products is not available for the existing games catalogue. Table V lists the in-game tools surveyed in this paper, showing those offered by existing products.

III. TECHNOLOGIES AND INNOVATIONS

Fig. 1 shows the CNG architecture. While the MMOG architecture is not modified (the game content and the game data are still transferred through the MMOG servers), the following components are added: (i) Sandbox on the client side that is responsible for modifying the game environment; (ii) CNG Server for monitoring the P2P UGC communication. The CNG server acts as a tracker for the system in the sense that it is in charge of introducing peers to other peers. It has persistent communication with the clients and manages the organization of the P2P exchanges.

A. IGIT

IGIT enables the user to resize the game and surround it with external content, overlay the game, and replace an existing game texture with an external content. This is done in a way that does not harm the game experience and without the need for SDK integration. Fig. 2 and Fig. 3 illustrate some of these features. Fig. 2 is a screenshot from the MMOG game "Roma Victor" [80] by RedBedlam. Fig. 3 shows the same game scene with a mock-up of CNG features. The modifications, which are numbered in Fig. 3, are as follows:

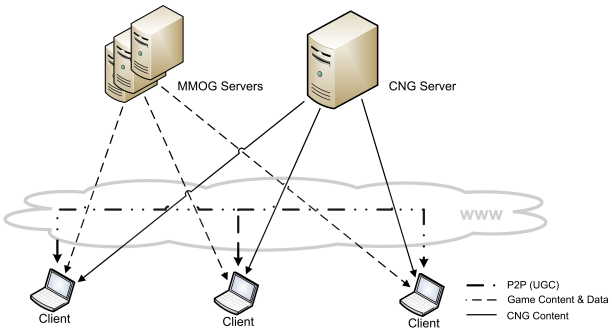

Figure 1. CNG architecture.

- (1) The original resolution of the game was modified to enable an additional frame around the game to hold the in-frame objects. IGIT uses the GPU of the user's machine for changing the resolution of the game to avoid reduction in the image quality;
- (2) Instant messaging window as an example of active Web 2.0 application;
- (3) Web browser that presents online passive information (in this example, a leader board);
- (4) Another Web browser window that presents an updated advertisement;
- (5) MMOG specific chat to enable the users in a specific scene to cooperate;
- (6) In-game 3D UGC. In this example, a user added a note on a tree to publish an eBay auction;
- (7) Two windows of a video chat with casual friends or cooperative players.

The choice of application and the application's screen location are under the control of the user (player). The Web 2.0 applications are browser-based applications that are downloaded from the CNG Server and run in the web-browser instances within the CNG Client. The purpose of Web 2.0 Applications is to offer online collaboration services to the user. They are browser-based and downloaded from the CNG Server and run in the web-browser instances within the CNG Client.

Since they are web-based the CNG client retrieves all the necessary information from the CNG Server

In addition, the CNG toolbox includes video recording and editing tools that allow users to capture the video of the game and

1) trim the captured game video
2) split, duplicate and sequence the recorded video cuts
3) remix trimmed cuts of the recorded video
4) upload the edited video to YouTube

Figure 2. Original MMOG screenshot.

Figure 3. IGIT-modified MMOG screenshot.

B. P2P live video system

In existing MMOGs, a player can capture the video of the game and send it to a central server which broadcasts it live to other users [11]. However, this solution, which heavily relies on central servers, has many drawbacks such as high costs for bandwidth, storage, and maintenance. Moreover, this solution is not easily scalable to increasing

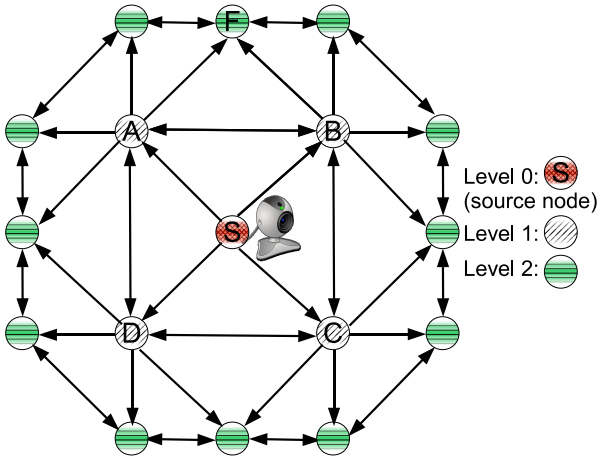

Figure 4. P2P topology.

number of users. The CNG project proposes a P2P live video streaming system to address the limitations of server-based solutions. The CNG P2P live video system allows every peer to become a source of a user-generated video stream for a potentially large set of receivers. While many P2P live video systems have been proposed, none of them has been specifically designed for the unique environment of MMOGs. In particular, none of the existing P2P live video systems addresses the following challenges:

- Any MMOG player should be able to multicast live video. The video can potentially be received by any other player in the P2P network.
- Live video streaming should not consume the upload and download bandwidth that is necessary for the smooth operation of the MMOG (MMOG client-server traffic).
- Live video should be delivered at about the same time to all peers at the same "level". For example, a level can be a priority class in a multi-tiered premium service. Peers in a higher priority class should be able to watch the video before those in a lower priority class. Alternatively, a level can be defined as the set of MMOG players that are in the same region of the virtual world.

The CNG P2P live video system is designed as follows. Peers are organized in levels with the source peer placed at level 0 (Fig. 4).

The video is captured in real time from the source screen, compressed, and partitioned into source blocks. Each source block corresponds to one GOP (Group of Pictures) and is an independent unit of fixed playback duration (e.g., 1 s). The source peer applies rateless coding on each source block and sends the resulting encoded symbols in successive UDP packets to level-1 peers. Packets are sent according to a scheduling strategy. The strategy specifies the maximum

number n of encoded packets that can be sent by the source for this block, the time t_i at which packet i is sent, and a hierarchical forwarding scheme $F_i,\ i = 1, 2, \ldots, n$. An example of a scheduling strategy for $n = 4$ and the four level-1 peers of Fig. 4 is as follows.

- $1 : t_1 : A \rightarrow B + D(\rightarrow C)$,
- $2 : t_2 : B \rightarrow A + C(\rightarrow D)$,
- 3: $t_3 : C \rightarrow B(\rightarrow A) + D$,
- 4: $t_4 : D \rightarrow C(\rightarrow B) + A$

The strategy says that packet 1 should be transmitted at time t_1 to A. A forwards the packet to B and D. D forwards it to C. Packet 2 should be transmitted at time t_2 to B. B forwards it to A and C. C forwards it to D. Packet 3 should be transmitted at time t_3 to C. C forwards it to B and D. B forwards it to A. Packet 4 should be transmitted at time t_4 to D. D forwards it to C and A. C forwards it to B.

A level-1 peer uses its own scheduling strategy to immediately forward any packet received from the source to level-2 peers. Moreover, as soon as it successfully decodes a source block, it sends an acknowledgment to the source, so that it stops sending it packets. Then it applies rateless coding on the source block and creates new packets. These new packets are sent to level-2 peers according to the scheduling strategy. Since a level-2 peer may receive packets from different level-1 peers, level-1 peers use randomly chosen rateless code seeds to minimize the probability that a level-2 peer receives duplicate packets. The value of the seed is sent as part of the header, so that the receiving peer can generate the same graph as the encoding peer. The procedure described above for two levels is repeated for the next levels.

Note that with the exception of peers situated at the last level, a peer will usually have two phases: a forwarding one (before the decoding is successful) and an encoding one (after decoding the block).

One of the main challenges consists of optimizing the scheduling strategy. The details of this optimization will be presented in a future paper.

Our system extends previous ideas proposed in [58], [59]. However there are many important differences between these works and our scheme. For example, the systems of [58], [59] do not have the notion of scheduling strategy. Also in [58], [59], there is no notion of levels.

As UDP does not have a built-in congestion control mechanism, a pure UDP-based application may overwhelm the network, leading to packet loss and degraded video quality. Therefore, the CNG project proposes an application-layer congestion control mechanism for the P2P system. The source adapts the video bit rate according to feedback received periodically from all peers. This feedback consists of the outage rate, i.e., the percentage of source blocks that were not decoded in time.

IV. Conclusion

We presented the EU funded CNG project. CNG supports and enhances community activities between MMOG gamers by enabling them to create, share, and insert UGC. The UGC considered by the CNG project includes 3D objects, graphics, and video. CNG develops in-game community activities using an in-game graphical insertion technology that replaces or inserts content in real time without the need to change the game's code in the client or server. CNG uses an architecture that efficiently combines the client-server infrastructure for the MMOG activities with a P2P overlay for the delivery of live video. The video traffic represents a real challenge to the network already occupied by the MMOG client-server data. The project proposes new techniques for P2P live video streaming that are "friendly to the MMOG client-server traffic. Since video can be resource heavy, the network indirectly benefits from the increased locality of communication. CNG also provides Web 2.0 tools for audio and video chat, instant messaging, in-game voting, reviewing, and polling. This will reduce the need for visiting forums outside the game and diluting the MMOG experience.

CNG has the potential to provide huge benefits to MMOG developers and operators. New community building tools will be offered cost-effectively and efficiently, without the need to redesign or recode the existing game offerings. The user experience will be enriched, and the needs of the end-users will be better addressed. The community will be brought into the content, and the game communities will become more engaged, reducing churn to other MMOGs. New income streams will be delivered with the help of in-game and around game advertising. Yet, MMOG developers and operators will be able to maintain control over how various commercial and UGC content is displayed, thus keeping editorial control of the look and feel of their MMOG.

Acknowledgment

The research leading to these results has received funding from the European Commission's Seventh Framework Programme (FP7, 2007-2013) under the grant agreement no. ICT-248175 (CNG project).

References

[1] [Online]. Available at: http://www.slideshare.net/ICOPartners/free-to-play-games-in-europe-2009. Last accessed: 7/2/2012.

[2] [Online]. Available at: http://www.cng-project.eu/. Last accessed: 7/2/2012.

[3] S. Ahmad, C. Bouras, R. Hamzaoui, A. Papazois, E. Perelman, A. Shani, G. Simon, and G. Tsichritzis, "The Community Network Game project: Enriching online gamers experience with user generated content," in Proc. 2nd Int. Conf. Creative Content Technologies (CONTENT 2010), Lisbon, Nov. 2010.

[4] [Online]. Available at:: http://eu.battle.net/wow/en/. Last accessed: 7/2/2012.

[5] Facebook API. [Online]. Available at: http://developers.facebook.com/docs/. Last accessed: 7/2/2012.

[6] [Online]. Available at: http://eu.spore.com/home.cfm?lang=en. Last accessed: 7/2/2012.

[7] [Online]. Available at: http://www.techsmith.com/camtasia/. Last accessed: 7/2/2012.

[8] [Online]. Available at: http://www.fraps.com. Last accessed: 7/2/2012.

[9] Youtube API. [Online]. Available at: http://code.google.com/apis/youtube/casestudies/ea.html. Last accessed: 7/2/2012.

[10] D. Takahashi, "YouTube game videos become a big channel for game marketers," Dec. 2008. [Online]. Available at: http://games.venturebeat.com/2008/12/18/youtube-game-videos-become-a-big-channel-for-game-marketers/. Last accessed: 7/2/2012.

[11] [Online]. Available at: http://xfire.com. Last accessed 7/2/2012.

[12] Adobe Flash. [Online]. Available at: http://www.adobe.com/support/flash/downloads.html. Last accessed: 7/2/2012.

[13] XMLHttpRequest, W3C Working Draft, 2009. [Online]. Available at: http://www.w3.org/TR/2009/WD-XMLHttpRequest-20091119/. Last accessed: 7/2/2012.

[14] SOAP Version 1.2 Part 1: Messaging Framework (Second Edition), W3C Recommendation, 2007. [Online]. Available at: http://www.w3.org/TR/soap12-part1. Last accessed: 7/2/2012.

[15] [Online] Available at: http://blueimp.net/ajax. Last accessed: 7/2/2012.

[16] [Online] Available at: http://www.gmail.com. Last accessed: 7/2/2012.

[17] [Online] Available at: https://addons.mozilla.org/en-US/firefox/addon/chatzilla/. Last accessed: 7/2/2012.

[18] [Online] Available at: http://www.mibbit.com. Last accessed: 7/2/2012.

[19] [Online] Available at: http://www.php-development.ru/software/chat-java.php. Last accessed: 7/2/2012.

[20] [Online] Available at: http://www.meebo.com/. Last accessed: 7/2/2012.

[21] [Online] Available at: http://www.google.com/ig. Last accessed: 7/2/2012.

[22] [Online] Available at: http://www.orkut.com/. Last accessed: 7/2/2012.

[23] [Online] Available at: http://www.flashcoms.com/products/community_video_chat/. Last accessed: 7/2/2012.

[24] [Online] Available at: http://avchat.net/. Last accessed: 7/2/2012.

[25] [Online] Available at: http://www.red5chat.com/. Last accessed: 7/2/2012.

[26] [Online] Available at: http://www.mebeam.com/. Last accessed: 7/2/2012.

[27] [Online] Available at: http://www.fileguru.com/Web-Voice-Chat/info. Last accessed: 7/2/2012.

[28] [Online] Available at: http://www.123flashchat.com. Last accessed: 7/2/2012.

[29] [Online] Available at: http://acidjs.wemakesites.net/voting-poll.html. Last accessed: 7/2/2012.

[30] [Online] Available at: http://ajaxian.com/archives/dpolls-an-ajax-pollster. Last accessed: 7/2/2012.

[31] [Online] Available at: http://www.flabell.com/flash/XML-Flash-Voting-Poll-39. Last accessed: 7/2/2012.

[32] [Online] Available at: http://wordpress.org/. Last accessed: 7/2/2012.

[33] [Online] Available at: http://www.movabletype.org/. Last accessed: 7/2/2012.

[34] [Online]. Available at: http://www.pptv.com/. Last accessed: 7/2/2012.

[35] [Online]. Available at: http://www.PPstream.com. Last accessed: 7/2/2012.

[36] Y.-H. C. Sanjay, S. G. Rao, S. Seshan, and H. Zhang, "A case for end system multicast," in Proc. ACM Sigmetrics, pp. 1–12, 2002.

[37] G. Yang, S. Kyoungwon, K. Jim, and T. Don, "P2cast: peer-to-peer patching scheme for VoD service," in Proc. 12th Int. Conf. World Wide Web, pp. 301–309, 2003.

[38] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: High-bandwidth multicast in a cooperative environment," in Proc. ACM SOSP, 2003.

[39] V. N. Padmanabhan, H. J. Wang, P. A. Chou, and K. Sripanidkulchai, "Distributing streaming media content using cooperative networking," in Proc. NOSSDAV '02, pp. 177–186, 2002.

[40] M. Jelasity, R. Guerraoui, A.-M. Kermarrec, and M. V. Steen, "The peer sampling service: experimental evaluation of unstructured gossip-based implementations," in Proc. ACM/IFIP/USENIX Int. Conf on Middleware, pp. 79–98, 2004.

[41] B. Li, G. Y. Keung, C. Lin, J. Liu, and X. Zhang, "Inside the new coolstreaming: Principles, measurements and performance implications," in Proc. INFOCOM'08, 2008.

[42] A.-M. Kermarrec, E. Le Merrer, Y. Liu, and G. Simon, "Surfing peer-to-peer IPTV system: distributed channel switching," in Proc. EuroPar, 2009.

[43] A.-T. Gai, F. Mathieu, F. de Montgolfier, and J. Reynier, "Stratification in P2P networks: application to BitTorrent," in Proc. 27th IEEE International Conference on Distributed Computing Systems (ICDCS), 2007.

[44] H. Yin, X. Liu, T. Zhan, V. Sekar, F. Qiu, C. Lin, H. Zhang, and B. Li, "Design and deployment of a hybrid CDN-P2P system for live video streaming: Experiences with livesky," in Proc. ACM Multimedia, 2009.

[45] H. Xie, Y. R. Yang, A. Krishnamurthy, Y. G. Liu, and A. Silberschatz, "P4p: Provider portal for applications," ACM SIGCOMM Computer Communication Review, Vol. 38, No. 4, Oct. 2008.

[46] M. Zhang, J.-G. Luo, L. Zhao, and S.-Q. Yang, "A peer-to-peer network for live media streaming using a push-pull approach," in Proc. 13th ACM International Conference on Multimedia, 2005.

[47] M. Röder, *Efficient Rate-Distortion Optimized Media Streaming*, PhD Thesis, University of Konstanz, 2007.

[48] P.A. Chou, "Streaming media on demand," in M. van der Schaar, P.A. Chou (editors), Multimedia over IP and Wireless Networks: Compression, Networking, and Systems, Academic Press, 2007.

[49] V. N. Padmanabhan, H. J. Wang, and P. A. Chou, "Resilient peer-to-peer streaming," in Proc. IEEE ICNP, pp. 16–27, Atlanta, GA, Nov. 2003.

[50] M. Hefeeda, A. Habib, B. Botev, D. Xu, and B. Bhargava, "PROMISE: peer-to-peer media streaming using CollectCast," in Proc. ACM Multimedia (MM'03), Berkeley, CA, Nov. 2003.

[51] X. Xu, Y. Wang, S.§. Panwar, and K. W. Ross, "A peer-to-peer video-on-demand system using multiple description coding and server diversity," in Proc. IEEE Int. Conf. Image Processing, Singapore, Oct. 2004.

[52] K. Suh, C. Diot, J. Kurosey, L. Massoulie, C. Neumann, D. Towsley, and M. Varvello, "Push-to-Peer video-on-demand system: design and evaluation," IEEE Journal on Selected Areas in Communications, vol. 25, no. 9, pp. 1706–1716, Dec. 2007.

[53] E. Setton, J. Noh, and B. Girod, "Rate-distortion optimized video peer-to-peer multicast streaming," in Proc. P2PMMS'05, pp. 39–48, Singapore, Nov. 2005.

[54] E. Akyol, A. M. Tekalp, and M. R. Civanlar, "A flexible multiple description coding framework for adaptive peer-to-peer video streaming," IEEE Journal of Selected Topics in Signal Processing, vol. 1, no. 2, pp. 231–245, Aug. 2007.

[55] M.-T. Lu, J.-C. Wu, K.-J. Peng, P. Huang, J. J. Yao, and H. H. Chen, "Design and evaluation of a P2P IPTV system for heterogeneous networks," IEEE Transactions on Multimedia, vol. 9, pp. 1568–1579, Dec. 2007.

[56] N. Thomos and P. Frossard, "Raptor network video coding," in Proc. 1st ACM International Workshop on Mobile video (in conjunction with ACM Multimedia 2007), Augsburg, Germany, Sep. 2007.

[57] A. Shokrollahi, "Raptor codes," IEEE Trans. Inf. Theory, vol. 52, pp. 2551–2567, June 2006.

[58] C. Wu and B. Li, "rStream: resilient and optimal peer-to-peer streaming with rateless codes," IEEE Transactions on Parallel and Distributed Systems, vol. 19, pp. 77–92, Jan. 2008.

[59] M. Grangetto, R. Gaeta, and M. Sereno, "Rateless codes network coding for simple and efficient P2P video streaming," in Proc. IEEE ICME 2009, Cancun, Mexico, 2009.

[60] M. Grangetto, R. Gaeta, and M. Sereno, "Reducing content distribution time in P2P based multicast using rateless codes," in Proc. Italian Networking Workshop, pp. 1–12, Cortina, 2009.

[61] E. Setton, J. Noh, and B. Girod, "Congestion-distortion optimized peer-to-peer video streaming," in Proc. IEEE ICIP-2006, Atlanta, GA, Oct. 2006.

[62] P. Baccichet, T. Schierl, T. Wiegand, and B. Girod, "Low-delay peer-to-peer streaming using scalable video coding," in Proc. International Packet Video Workshop, PV2007, Lausanne, Switzerland, Nov. 2007.

[63] [Online]. Available at: http://tools.ietf.org/html/rfc4340. Last accessed 7/2/2012.

[64] R. Diana and E. Lochin, "ECN verbose mode: A statistical method for network path congestion estimation," in Proc. INFOCOM, San Diego, CA, 2010.

[65] T. Silverston and O. Fourmaux, "Measuring P2P IPTV systems," in Proc. ACM NOSSDAV'07, June 2007.

[66] S. Ali, A. Mathur, and H. Zhang, "Measurement of commercial peer-to-peer live video streaming," in Proc. ICST Workshop Recent Adv. Peer-To-Peer Streaming, Waterloo, Aug. 2006.

[67] M. Alhaisoni and A. Liotta, "Characterization of signalling and traffic in Joost," Journal of Peer-to-Peer Networking and Applications, Vol. 2, issue 1, pp. 75–83, 2009.

[68] P. Baccichet, J. Noh, E. Setton, and B. Girod, "Content-aware P2P video streaming with low latency," in Proc. IEEE International Conference on Multimedia and Expo, ICME 2007, Beijing, China, Jul. 2007.

[69] M. Zhang, Q. Zhang, L. Sun, and S. Yang, "Understanding the power of pull-based streaming protocol: Can we do better?," IEEE Journal on Selected Areas in Communications, Vol. 25, pp. 1678–1694, Dec. 2007.

[70] M. Zhang, L. Sun, and S. Yang, "iGridMedia: Providing delay-guaranteed peer-to-peer streaming service on Internet," in Proc. IEEE GLOBECOM 2008, New Orleans, LO, Dec. 2008.

[71] F. Pianese, D. Perino, J. Keller, and E.W. Biersack, "PULSE: An adaptive, incentive-based, unstructured P2P Live streaming system," IEEE Transactions on Multimedia, Vol. 9, Number 8, pp. 1645–1660, Dec. 2007.

[72] M. Wang and B. Li, "R2: Random push with random network coding in live peer-to-peer streaming," IEEE Journal on Selected Areas in Communications, Vol. 25, Number 9, pp. 1655–1666, Dec. 2007.

[73] [Online]. Available at: http://www.playxpert.com/. Last accessed 7/2/2012.

[74] [Online]. Available at: http://www.freeridegames.com/. Last accessed 7/2/2012.

[75] [Online]. Available at: http://en.wikipedia.org/wiki/Massive_Incorporated. Last accessed 7/2/2012.

[76] [Online]. Available at: http://www.doublefusion.com/. Last accessed 7/2/2012.

[77] [Online]. Available at: http://store.steampowered.com/. Last accessed 7/2/2012.

[78] [Online]. Available at: http://www.overwolf.com/. Last accessed 7/2/2012.

[79] [Online]. Available at: http://raptr.com/. Last accessed 7/2/2012.

[80] [Online]. Available at: http://www.roma-victor.com. Last accessed 7/2/2012.