RADE: Resource-aware Distributed Browser-to-browser 3D Graphics Delivery in the Web

Timo Koskela, Arto Heikkinen, Erkki Harjula, Mikko Levanto, Mika Ylianttila
Center for Internet Excellence
University of Oulu
Oulu, Finland
firstname.lastname@cie.fi

***Abstract*—With the introduction of novel standardized web technologies such as WebGL, 3D virtual environments (VEs) are making their way into the web. Concurrently, resource rich mobile devices, such as smartphones and tablets, have become the primary access medium for the web. This paper introduces a method called RADE that enables resource-aware P2P-assisted 3D asset delivery in a web browser using WebRTC. Use of P2P technology in 3D asset delivery can (1) decrease the load on 3D asset servers; (2) decrease the application response times; and (3) reduce the operational costs of VE service providers. The performance and resource fairness of RADE was evaluated in real-life wireless networks using a prototype implementation. Based on the results, RADE can significantly reduce the server load and the 3D asset delivery times especially when 3D asset servers are under heavy load. For instance, with a scarce server bandwidth of 2Mbps, use of RADE resulted in 55% shorter 3D asset delivery times on average. Although RADE has been developed for 3D asset delivery, it is applicable for many types of web applications, including video delivery.**

Keywords—P2P; peer-to-peer; webrtc; mobile; virtual environment

I. INTRODUCTION

Fuelled by the introduction of novel web technologies including WebGL [1], XML3D [2] and X3DOM [3], 3D graphics and 3D virtual environments (hence referred as VEs) are quickly making their way into the web. At the same time, mobile devices have substituted PCs as the primary access medium for the web[1]. With the rapidly growing hardware capacities, modern mobile devices are already capable of reproducing complex and high-quality 3D graphics [4]. In the future, the proportion of mobile users participating in VEs will predictably grow. This assumption is supported at least by the two following facts. First, already today, many Internet applications are primarily used with mobile devices. Second, in 2014 alone, the combined number of smartphones and tablets sold was five times bigger than the number of PCs sold[2].

For accessing a VE, users are typically required to install standalone client software that the service provider may need to version for multiple operating system and platform combinations including mobile environments [1]. When using the web browser as a platform, only a single version of client software needs to be implemented and no software installations or updates are required on the terminal side. Despite the benefits introduced by the web, the delivery of 3D graphics (i.e. 3D assets) used in VEs relies traditionally on the client-server architecture, which may lead to scalability issues due to the limitations in processing capacity and available bandwidth as the 3D assets are typically large in file size [5]. Efficient delivery of 3D assets is especially challenging in extensible VEs such as SecondLife[3] and realXtend[4] in which the users are able to modify the 3D assets and to create entirely new ones [6]. As the 3D assets in an extensible VE are constantly changing, heavy burden is inflicted on the asset servers responsible for the storage and the distribution of the 3D assets.

To alleviate the load on the asset servers, a proportion of 3D assets could also be transferred directly between web browsers using the Web Real-Time Communications (WebRTC) technology [7]. This way, the vast combined capacity of end-user devices residing at the edges of the Internet could be harnessed. Although WebRTC is still in standardization, working implementations are already available in some of the major web browsers including Chrome and Firefox for Android. When implementing the delivery of 3D assets directly between end-user devices, one must take into consideration that in the future more and more end-user devices will be mobile. Mobile devices are rather heterogeneous in their capacities, which must be taken into account when selecting both the appropriate level of detail (LOD) of the 3D assets and the end-user devices participating in the 3D asset delivery [4].

This paper builds on a scalable VE communication architecture introduced in [8][9] and proposes a method (hence called RADE) that enables resource-aware P2P-assisted 3D assets delivery in a web browser using WebRTC. Based on the current status and the estimated resources of end-user devices, RADE is capable of proposing suitable candidates for conducting the direct browser-to-browser 3D asset delivery. Thus, RADE can (1) decrease the load on asset servers; (2) decrease the application response times; and (3) reduce the operational costs of the VE provider. A prototype of RADE was implemented and its operation was evaluated in real-life

[1] http://marketingland.com/outside-us-60-percent-internet-access-mostly-mobile-74498
[2] http://www.extremetech.com/computing/185937-in-2015-tablet-sales-will-finally-surpass-pcs-fulfilling-steve-jobs-post-pc-prophecy
[3] www.secondlife.com
[4] http://www.meshmoon.com

978-1-4673-7701-0/15/$31.00 ©2015 IEEE

wireless networks using end-user devices with varying capacities. Although RADE has been implemented particularly for the delivery of 3D assets, it is applicable for many types of web applications including video delivery.

The rest of the paper is organized as follows. Section II examines the related work, Section III describes the communication architecture for adaptive and scalable 3D asset delivery, Section IV introduces RADE, Section V presents the prototype implementation of RADE and the experimental setup, Section VI presents the results of the experiments, and finally, Section VII concludes the paper.

II. Related Work

Introduction of WebGL [1], XML3D [2] and X3DOM [3] technologies have brought 3D graphics into the web enabling the implementation of web-based VEs. Use of the web browser as the development platform has also facilitated the implementation of VEs for mobile devices. In this respect, the challenges of deploying 3D graphics on mobile devices are examined thoroughly in [4]. This work introduces different techniques that can be used for optimizing performance and/or battery life of mobile devices in 3D graphics deployment.

Use of peer-to-peer (P2P) technologies for implementing scalable VEs has recently received lots of attention in the research community (e.g. [5][9][10][11]). Despite the vast efforts, completely P2P-based VEs still face issues regarding the resource-aware load distribution for heterogeneous nodes, the high communication overhead caused by constant joining and leaving of nodes (i.e. churn) as well as the reliability of VE state consistency and persistency maintenance [11]. Therefore, hybrid solutions have also been proposed that implement a client-server backbone of which scalability and performance is enhanced with P2P technologies (e.g. [9][12]).

When comparing our work with the previous research on VEs, none of the existing solutions are provided for the cross-platform supported web browser. Furthermore, participation of mobile devices has been mainly neglected apart from few remarks on their special characteristics that should be taken into account in the VE architecture design (e.g. [5]). Therefore, this paper presents a pioneering attempt to implement resource-aware P2P-assisted 3D assets delivery in the web. It should be noted that mobile devices are naturally the centerpiece in VEs that are based on mobile ad hoc networks (MANETs) (e.g. [13][14]). However, MANET-based VEs are primarily meant for collaborative use in rescue or military operations when no network infrastructure is available [14]. Therefore, MANET-based VEs are spatially very restricted and temporary in nature compared to the (traditional) Internet-based VEs that are in the focus of this paper.

Use of BitTorrent based protocols for the delivery of 3D assets has also been examined in the previous research (e.g. [10][15]). However, the mechanisms of BitTorrent favor nodes that are actively contributing to the swarm (i.e. group of nodes downloading the same file), which may result in poor performance for mobile nodes. This is due to two facts. First, due to the asymmetry in mobile networks, the upload bandwidth is significantly lower than the download bandwidth. Second, NATs/FWs may sometimes prevent the establishment of incoming connections [16]. For making BitTorrent based protocols more suitable for mobile nodes, modified versions have also been proposed (e.g. [16][17]). Finally, it should be noted that BitTorrent has not been designed for real-time data delivery in high churn environments [18], where the availability of nodes may be poor. On the contrary, BitTorrent is very efficient for popular, static and large files, but in extensible VEs, the 3D assets have real-time requirements, are of various size and may be invariably changing.

The performance of WebRTC with mobile devices has only been studied in few recent works. In [19], session establishment delay, session maintenance overhead and resource consumption of multiple simultaneous file transfers were examined. Based on the results, establishing a connection between two devices connected to a mobile network may take several seconds. However, the results also indicate that today's mobile devices are clearly capable of running multiple simultaneous WebRTC data transmissions. In [20], the applicability of content delivery networks, cloud computing, P2P networking, and WebRTC is comparatively analyzed for on-demand data transmission between mobile clients. Although WebRTC has already been utilized in few existing P2P data delivery solutions such as PeerCDN (proprietary), Peer5 (proprietary) and WebTorrent (open-source), according to our best knowledge, these solutions do not take the special characteristics of mobile devices into account. For this purpose, studies such as [21][22] have provided models for approximating energy consumption of mobile devices with different traffic profiles to enable resource-aware load balancing in distributed systems.

III. Communication Architecture for Adaptive and Scalable 3D Asset Delivery

The communication architecture for adaptive and scalable 3D asset delivery is illustrated in Figure 1. The communication architecture was originally introduced in [8], but is now briefly revisited as its design has been further developed.

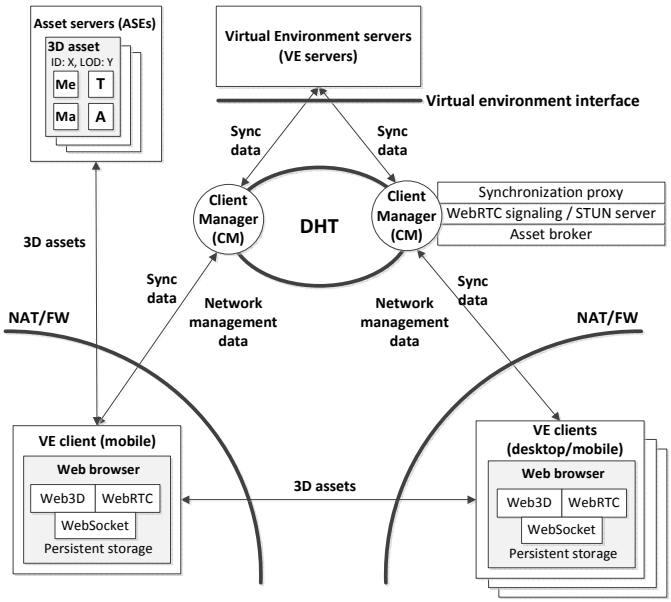

Fig. 1. The overall communication architecture for adaptive and scalable 3D asset delivery.

The entities in the communication architecture are VE clients, Client Managers (CMs), Asset Servers (ASEs) and VE servers. VE clients implement a 3D rendering framework, WebSocket protocol, WebRTC protocol and persistent storage. CMs, in turn, have a triple role. Each CM acts as (1) a synchronization proxy responsible for keeping the state of the VE consistent between VE clients; (2) a WebRTC signaling/STUN server responsible for initiating the direct browser-to-browser 3D asset delivery; (3) an asset broker responsible for determining the suitable VE client candidates for conducting the direct browser-to-browser 3D asset delivery. Each CM is connected to a specific group of VE clients, to all other CMs and to the VE servers. Communication between VE clients and CMs is implemented using the WebSocket protocol. Communication between CMs, in turn, is implemented using a distributed hash table (DHT) -based P2P network that enables discovery of any CM with a single hop [23]. In the P2P network, each CM is provided a unique numerical identifier in the DHT address space. This is done using a hash function such as SHA-1, for instance, on the IP address of a CM. VE clients are also provided unique identifiers in a similar way.

In the P2P network, each CM maintains two tables: (1) a VE client table that is locally stored; and (2) 3D asset location table that is distributed among all CMs. In the VE client table, the following items are stored (if available) per VE client:

- a list of 3D assets (and their lower quality variants if any) stored in the persistent memory,
- a list of open WebRTC connections and their status (including information about active data transmissions between other peers),
- web browser type,
- connection type,
- connection speed,
- battery status,
- logical network location,
- JavaScript processing speed,
- 3D rendering speed.

For implementing the distributed 3D asset table, each 3D asset is given a unique numerical identifier in the DHT address space. The location of this identifier in the DHT address space determines the CM that is responsible for maintaining the list VE clients possessing this particular 3D asset. For each VE client, also the corresponding CM is stored.

ASEs are responsible for hosting all 3D assets in a VE. With a 3D asset, we do not mean any single piece of 3D graphics, but rather a collection of 3D graphics (including geometry, textures, surface materials and animations) forming some logical entity in a VE. For instance, in a 3D virtual city, each building could be treated as an individual 3D asset. If every piece of 3D graphics would be treated as an individual 3D asset, the number 3D asset requests and the number of items stored in the distributed 3D asset location table would explode inflicting heavy maintenance burden on CMs. In addition to the original version, ASEs may also host one or more low quality variants (i.e. LODs) of each 3D asset. Depending on the overall number of 3D assets in a VE, different LODs can either be included inside a single item or introduced as individual items in the distributed 3D asset location table. In any case, CMs maintain a list of 3D assets and the associated LODs possessed by their VE clients. To compose a 3D asset, 3D graphics are grouped together using a container format such as BLAST [24]. The advantage of BLAST is that it enables requesting a single piece of 3D graphics within a 3D asset without downloading the whole 3D asset. This is particularly useful in extensible VEs, where 3D assets or their parts are constantly changing.

In our communication architecture, signaling for WebRTC connection establishment is implemented by CMs in a distributed way. As each VE client is only connected to a single CM, signaling between two VE clients travels through their corresponding CMs. If the VE clients are connected to the same CM, only this particular CM is involved in the signaling. This way, no dedicated signaling servers are required.

Establishing a WebRTC connection between two browsers may induce some delays particularly in mobile networks [19]. This mainly results from the delay caused by the NAT/FW traversal and the high latencies typical to mobile networks. By contrast, the maintenance burden of an idle WebRTC connection is rather small especially with Firefox that implements a long keepalive messaging interval [19]. Therefore, it is important that a VE client establishes an initial set of WebRTC connections to the other VE clients already when starting to use the VE. As a basic principle, the connections should be established with VE clients that possess 3D assets needed immediately or in the near future. In time, the set of WebRTC connections will need to be adapted based on user's own activity and the joining and leaving of the other VE clients. However, the policies governing the WebRTC connection management are outside the scope of this paper.

IV. Method for Resource-aware P2P-Assisted 3D Asset Delivery (RADE)

In this Section, the general operational principles of RADE are presented. Although RADE is introduced in the context of 3D asset delivery, it is applicable for many types of web applications which require delivering large amount of content data.

Figure 2 presents a sequence diagram that depicts how different system entities communicate with each other for enabling resource-aware P2P-assisted 3D asset delivery with WebRTC. The VE client that acts as the requestor in the 3D asset delivery is referred to as R. Depending on the implementation of the VE, (1) the 3D asset discovery can be initiated either by R itself or by the VE servers. After this, (2) the suitable LOD is determined by R's CM based on the estimated capabilities of R. This process is a separate research challenge and left outside the scope of this paper. Next, (3)(4) R's CM uses the hash function on the requested 3D asset to discover and contact the CM responsible for maintaining the list of VE clients (and their CMs) possessing this 3D asset. After this, (5) ASEs' status information is retrieved. Next, (6) the status information of the VE clients possessing the requested 3D asset is retrieved by contacting their CMs. Based

on the acquired status information (see Section III), (7) the most suitable VE clients for the 3D asset delivery are determined. Next, (8)(9) the 3D asset delivery is initiated and the WebRTC connections are established if needed. This is followed by (10) parallel downloading of the requested 3D asset from both the ASEs and multiple VE clients if suitable VEs client were found. Otherwise, the 3D asset is downloaded from the ASEs. The 3D asset is transferred in fixed sized chunks P_S. After this, (11) R informs its CM about the completion of the download process. Finally, (12) R's CM updates the distributed 3D asset location table as R now possesses a new 3D asset.

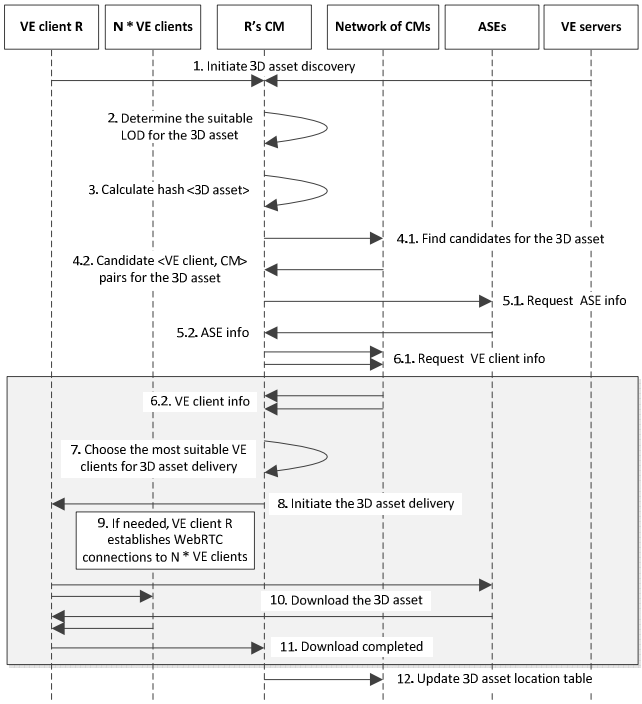

Fig. 2. A high-level diagram describing the sequence of messages in resource-aware P2P-assisted 3D asset delivery.

The section in Figure 2 marked with a grey rectangle is also presented in more detail in Figure 3 and Figure 4 using a state diagram. In Figure 3, the decision making procedure is presented for choosing suitable VE clients for the P2P-assisted 3D asset delivery. This procedure is conducted by R's CM. For each VE client possessing the requested 3D asset, a suitability score is calculated. The calculations are continued until the VE client timer T_C has expired or all candidates have been scored. Next, D_N suitable VE clients are chosen for the P2P-assisted 3D asset delivery. If no suitable ones are found, the 3D asset is only downloaded from the ASEs.

The VE clients are ranked by calculating a suitability score based on the status information (see Section III) of each VE client. This information is retrieved from each VE client's CM. The score is calculated as a weighted sum of different suitability factors as shown in (1):

$$S_c^{score} = \sum_{i=1}^{n} \omega^i S_c^i \text{, where } \sum_{i=1}^{n} \omega^i = 1 \text{ and } \forall\, i\colon S_c^i \in [0,1] \quad (1)$$

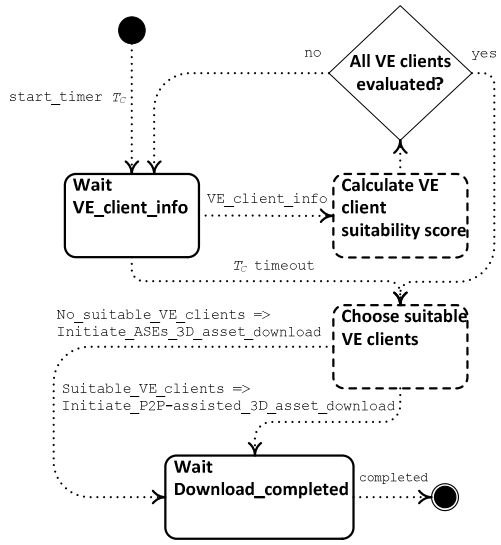

Fig. 3. A state diagram describing the decision making process for choosing suitable VE clients for the P2P-assisted 3D asset delivery.

In (1), S_c^{score} denotes the overall suitability score of a VE client *c;* S_c^i is a suitability factor *i* calculated for the VE client *c*. ω^i is a weight for the suitability factor S_c^i. The generic suitability factors used in RADE are presented in (2). This generic equation (2) does not include the exact formulas for calculating the values of the suitability factors. As Web browser technology is constantly evolving and providing more and more status information to the applications, the formulas can also be constantly improved and optimized. However, the preliminary mathematical formulas defined specifically for our experiments are presented in section V.A.

$$S_c^{score} = \omega^{conn\_est} S_c^{conn\_est} + \omega^{conn\_speed} S_c^{conn\_speed} + \omega^{proximity} S_c^{proximity} + \omega^{battery} S_c^{battery} + \omega^{cpu} S_c^{cpu} \quad (2)$$

Suitability factors in (2) are shortly described below including the status information that can be used in the calculations (R=requestor of the 3D asset; C=candidate for the 3D asset delivery):

- $S_c^{conn\_est}$ is calculated based on the estimated delay for establishing the WebRTC connection between R and C: connection type (C), browser type (C), connection already established via WebRTC (R, C),
- $S_c^{conn\_speed}$ is calculated based on the estimated upload speed of C: connection upload speed (C), the number of active WebRTC uploads (C), the number of active WebRTC downloads (R, C),
- $S_c^{proximity}$ is calculated based on the estimated logical distance in the Internet between R and C: logical network location (R, C),
- $S_c^{battery}$ is calculated based on the estimated battery level of the end-user device: battery level (C), battery recharging (C),
- S_c^{cpu} is calculated based on the estimated processing s of the end-user device: processing speed (C).

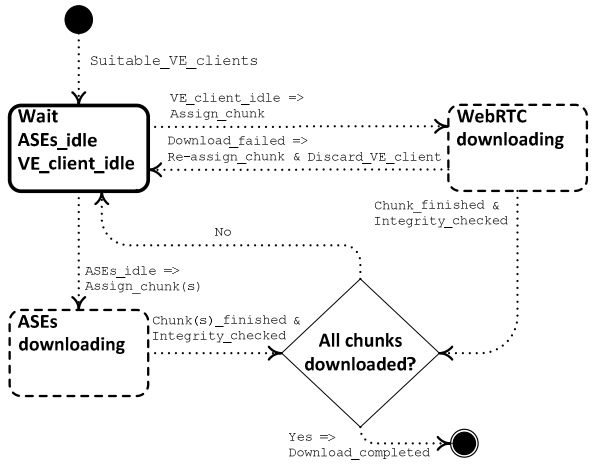

Fig. 4. A state diagram describing a procedure for managing the P2P-assisted 3D asset delivery.

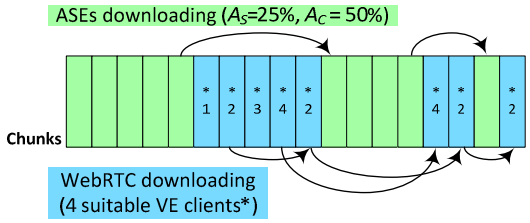

Fig. 5. An example visualization of chunk download scheduling in P2P-assisted 3D asset delivery with a 3D asset divided into 18 chunks.

It should be noted that not all these pieces of status information are available for every VE client. Thus, the implementation of RADE must allow for the scarcity of VE client status information. As part of the suitability score calculations, rejection rules can be defined for RADE that may lead to automatic dismissal of some VE clients. These rules can include, for instance, a minimum threshold for the suitability score S_T, a minimum threshold for the battery level of a mobile device E_T and the maximum allowed number of active WebRTC uploads U_M per each VE client.

In Figure 4, the procedure for P2P-assisted 3D asset delivery is presented from the viewpoint of R. The 3D asset is delivered in fixed sized chunks. After the suitable VE clients have been determined, R assigns the first chunks to be downloaded. ASEs are assigned A_S percentage of the chunks (rounded up) starting from the beginning of the file after which each VE client is assigned a single consecutive chunk (see Figure 5). The chosen delivery order also enables the use of 3D asset streaming. Notice that the unconnected VE clients are assigned the first chunks once they get connected. After R has downloaded the first set of chunks from the ASEs, R assigns A_C percentage of yet unassigned chunks (rounded up) to the ASEs. After R has downloaded a chunk from a VE client, R assigns the next yet unassigned chunk to that VE client (see Figure 5). If downloading of a chunk from a VE client fails, R re-assigns the chunk and discards the VE client in question. Delivery of a chunk may fail due to a connection failure or low bitrate determined by the bitrate threshold B_T. After each successful delivery of a chunk, its integrity is checked. This is done using the hash function on each chunk. Depending on the download source, the hash value is compared either with the ASEs or a VE client. After each successful 3D asset delivery, the VE clients report their upload performance to their CMs, which store the information in the client table.

V. Evaluation of RADE

In this section, a prototype implementation of RADE is presented and its performance and resource fairness are evaluated. Resource fairness means the use of hardware resources of a device in proportion to its available resources.

A. Experimental Setup

Performance and resource fairness of RADE was evaluated using a prototype implementation consisting of a VE client realizing the WebRTC client functionality and a CM realizing both WebRTC signaling server and Asset broker functionality (https://github.com/Chiru/RADE) as shown in Figure 6. The CM implementation uses an SQL database for storing the VE client status information. For NAT/FW traversal, Google's public STUN server was used. RADE's suitability score calculation logic was implemented within the asset broker that is responsible for determining the suitable VE client candidates for conducting the direct browser-to-browser 3D asset delivery.

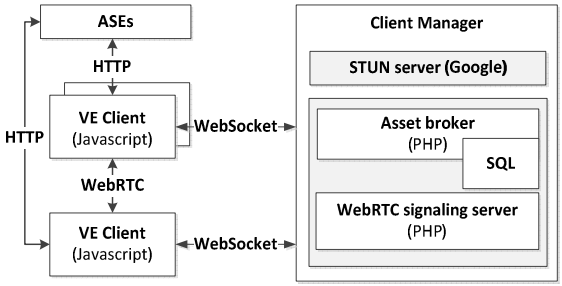

Fig. 6. An overview of the prototype implementation.

The experimental setup consisted of eleven VE clients of which four were smartphones, two were tablets, four were laptops and one was a desktop computer (see Table I). Smartphones and one tablet were connected to cellular networks and the rest of the devices to different WLANs.

TABLE I. Devices used in the experiments.

Device	Connection type	Connection upload speed	SunSpider v1.0.2
Smartphone (R)	Cellular (HSPA+, 21Mbps)	--	1084ms
Smartphone	Cellular (3G, 256kbps)	0.120Mbps	1157ms
Smartphone	Cellular (HSPA+, 21Mbps)	1.060Mbps	1107ms
Smartphone	Cellular (LTE, 80Mbps)	2.120Mbps	942ms
Tablet	Cellular (HSPA+, 21Mbps)	1.177Mbps	1718ms
Tablet	WLAN (802.11n)	2.120Mbps	677ms
Laptop	WLAN (802.11n)	0.883Mbps	224ms
Laptop	WLAN (802.11n)	1.766Mbps	227ms
Laptop	WLAN (802.11n)	1.060Mbps	230ms
Laptop	WLAN (802.11n)	2.650Mbps	196ms
Desktop	WLAN (802.11n)	2.650Mbps	165ms

TABLE II. VALUES OF PARAMETERS USED IN RADE.

Target number of VE client downloads	D_N	5
Maximum number of VE client uploads	U_N	1
Suitability score threshold	S_T	20%
Battery-level threshold	E_T	20%
Chunk size	P_S	256kB
ASEs start percentage	A_S	25%
ASEs continuation percentage	A_C	25%

Table II shows values for the various parameters of RADE used in the prototype implementation. D_N defines the number of VE clients that R can simultaneously download data from. The value of D_N was intentionally set low (5) to allow RADE to choose only the most suitable ones from the small number of VE clients (10) in our test network. U_N defines the maximum number of concurrent uploads per VE client. It was set to 1 as our experiments did not include test cases with concurrent uploads. S_T defines the minimum suitability score below which candidates are rejected. This was set to 20% to reject clearly unsuitable candidates that did not meet any of the other rejection rules. E_T defines the minimum acceptable battery level for candidates. It was set to 20% as many mobile devices enter a power saving mode around this battery level. P_S defines the size of the chunks in which the data is transmitted. The chunk size was set to 256kB as it is the most commonly used value in BitTorrent [25]. A_S defines how large portion of data is requested from ASEs at the beginning of the download process. A_C defines how large portion of the remaining data is requested from ASEs in subsequent requests. Both A_S and A_C were set to 25% in order to allow notable part of data to be downloaded using WebRTC, especially when multiple fast VE clients are available.

```
S_c^conn_est:
    if R_connected_to_C == true:
        score = 1;
    else:
        if browser_type_C == chrome:
            score += 0,3;
        if connection_type_C == wifi:
            score += 0,3;
        if connection_type_C == ethernet:
            score += 0,3;
S_c^conn_speed:
    if active_download_from_C_to_R == true:
        score = -1; (reject candidate)
    else if num_total_active_uploads_C >= U_N:
        score = -1; (reject candidate)
    else:
        score = min(1, (1-exp(-upload_speed_kbps_estimate/300)));
S_c^proximity:
    score = max(0, 1-sqrt((r1-c1)^2+(r2-c2)^2+(r3-c3)^2+(r4-c4)^2)/1000);
S_c^battery:
    if battery_level < E_T:
        score = -1; (reject candidate)
    else if battery_recharging == true:
        score = 1;
    else:
        score = battery_level ^ 1.3;
S_c^cpu:
    score = min(1, exp(-(sunspider_result-200)/1000));
```

Fig. 7. Suitability factor score calculations for RADE.

Figure 7 shows how the scores for the suitability factors S_c^i of RADE were calculated in the prototype implementation. Rejection rules are defined for some of the suitability factors and if a rejection condition is met the suitability factor score gets a value of -1. A VE client is rejected if any of the suitability factors indicates rejection, i.e. has a score of -1.

$S_c^{conn\_est}$: The score is 1 if the WebRTC connection is already established. Otherwise, the use of Chrome as a web browser and either WLAN or Ethernet as an Internet connection adds 0.3 to the score. Chrome is substantially faster than Firefox in the connection establishment as it uses Trickle ICE [19]. With cellular networks, the connection establishment is typically slower than with WLAN or Ethernet [19]. The Internet connection type is retrieved using Network Information API that is currently only available in Firefox.

$S_c^{conn\_speed}$: The upload speed of a VE client was initially estimated using speedtest.net. After this, upload performance can be continuously recorded and calculated as a time weighted sum, where the most recent values have a higher weight. However, this continuous monitoring was not implemented in the prototype. The score is calculated based on the formula shown in Figure 7 and Figure 8. The formula gives roughly half of the maximum score for 200kb/s, which is in the lower end of previously reported average upload speed for BitTorrent clients [25]. If a candidate has U_N or more active uploads or if R is already downloading from the candidate, the candidate is rejected. From R's standpoint, downloading multiple 3D assets from the same candidate does not increase the upload performance.

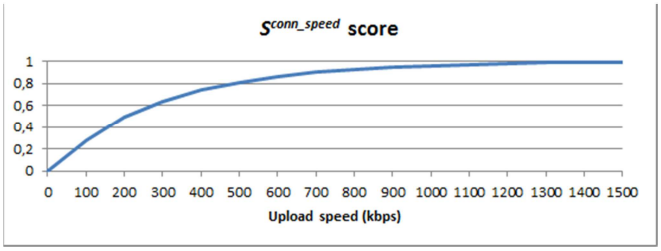

Fig. 8. The $S_c^{conn\_speed}$ score based on the candidate upload speed.

$S_c^{proximity}$: A HTTP-based polling mechanism was implemented to calculate round-trip times (RTT) to four known landmark servers. The Euclidian distance between the four-dimensional RTT vectors of R and the candidate is calculated. For calculating the score, the Euclidian distance is scaled between 0 and 1. Distance value greater than 1000 gives score of 0.

$S_c^{battery}$: If the battery is completely full, the device is connected to a recharger or the device is a desktop computer, the score is 1. If the battery level is below E_T, the candidate is rejected. Otherwise, the score is calculated based on the formula shown in Figure 7. The score attenuates quickly with low battery level values. The battery level is retrieved using Battery API currently available in both Chrome and Firefox.

S_c^{cpu}: For evaluating the JavaScript performance, a JavaScript benchmark called SunSpider 1.0.2 was used for each VE client. The score is calculated based on the formula shown in Figure 7. Execution time of 200ms is a typical result for a basic desktop computer, and was set as a threshold to provide the full score.

B. Experiments

In the experiments, RADE is evaluated from performance and resource fairness standpoints. In the performance evaluation, the suitability factor weights of RADE were adjusted for maximizing the 3D asset downloading speed (see ω^1 in Table III), whereas in the resource fairness evaluation, the suitability factor weights of RADE were adjusted for minimizing the load on low-capacity battery-operated devices in the 3D asset downloading (see ω^2 in Table III).

TABLE III. VALUES OF SUITABILITY FACTOR WEIGHTS FOR RADE.

	ω^1	ω^2
ω^{conn_est}	0.3	0
ω^{conn_speed}	0.6	0
$\omega^{proximity}$	0.1	0
$\omega^{battery}$	0	0.8
ω^{cpu}	0	0.2

In the performance evaluation, server bandwidth usage and transfer completion time were examined. For this, 3D assets were downloaded from (1) ASEs only; (2) ASEs and VE clients in parallel; and (3) VE clients only. In (1) and (2), ASEs' upload bandwidth was first limited to 10 Mbps, and second, to 2 Mbps. In (2) and (3), RADE was also compared against random VE client selection. Altogether, the performance evaluation consisted of 8 different experiments.

In the resource fairness evaluation, the average battery status and processing speed of the chosen VE clients were examined. For this, 3D assets were downloaded from (1) ASEs and VE clients in parallel; and (2) VE clients only. In both (1) and (2), RADE was compared against random VE client selection. In (1), ASEs' upload bandwidth was first limited to 10 Mbps, and second, to 2 Mbps. Altogether, the resource fairness evaluation consisted of 6 different experiments.

In all experiments, one smartphone acted as the requestor (i.e. R) of the 3D asset. All other VE clients possessed the requested 3D asset of which size was set to 10 MBs in all experiments. All 14 experiments were repeated 5 times. In the beginning of each experiment round (consisting of all 14 experiments), some features of every VE client were randomized as shown in Table IV. The approximations for logical network location were measured in the beginning of each experiment round. The approximations for connection upload speed to R and processing speed were measured once before the experimentation (see Table I).

TABLE IV. RANDOMIZED FEATURES OF THE PARTICIPATING VE CLIENTS.

Browser type	Connected to R via WebRTC	Battery level	Battery recharging
50% Firefox 50% Chrome	50% Yes 50% No	1-100%	10% Yes 90% No

In the experiments, the following steps were taken. (1) All VE clients connected to the CM and provided their status information. (2) WebRTC connections between R and randomly selected VE clients were established. (3) R initiated a request for a single 3D asset to its CM. (4) R's CM calculated the suitability score for each VE client and returned the ranked list of VE clients to R. (5) R selected D_N best ranked VE clients or D_N random VE clients for 3D asset downloading. (6) R initiated the downloading of the 3D asset from ASEs only, ASEs and VE clients in parallel or VE clients only. In ASEs only downloading, only steps (1), (3) and (6) were taken.

The experiment data was logged and time stamped by both R and its CM. For limiting the complexity of the experiments, it was ensured that all WebRTC connection attempts were successful and no VE client failures appeared. This was done by repeating the failed experiments. The integrity testing of 3D assets and the bitrate monitoring are not yet implemented in the prototype and were excluded from the experiments.

VI. RESULTS

The results of the experiments are presented in Figures 9-11, where ASE10 and ASE2 denote the ASEs' upload bandwidth limitations of 10Mbps and 2Mbps. RADE1 and RADE2 denote RADE's different weight factors settings ω^1 and ω^2, and RAND denotes the random VE client selection.

In Figure 9, the transfer completion times for the 3D asset delivery are presented. It is clearly visible that use of WebRTC in parallel with ASEs can significantly reduce the transfer completion times particularly when the server bandwidth is scarce (ASE2). Use of RADE1 with ASE2 results in 55% shorter transfer completion time on average compared to ASE2 only. However, with ASE2, RADE1 outperforms RAND only marginally. One reason for this is that RADE1 rejects all VE clients even with fast upload speed if these VE clients are running out of battery (E_T < 20). RAND, in turn, does not concern with the resources of VE clients. Another reason is that the VE clients' upload speed varies in time, and in the experiments, this variation was not continuously recorded having some impact on the performance of RADE1. The variation in VE clients' connection speed also resulted in high standard deviations when WebRTC was more heavily utilized in the 3D asset delivery.

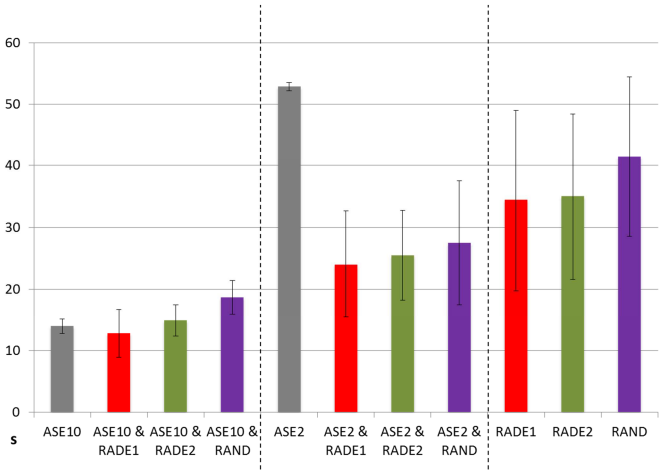

Fig. 9. Transfer completion times for the 3D asset delivery.

With higher server bandwidth (ASE10), RADE1 decreases the transfer completion time by 8% on average. However, use of RAND with ASE10 even slightly increases the transfer completion time compared to ASE10 only. Whereas RADE1

favors VE clients that are already connected to R and have fast connection speed, RAND has higher probability to select VE clients that are not already connected with R and have slow connection speed. It should be noted that establishing a WebRTC connection may induce some delays [19] and if a VE client with a very slow upload speed is assigned a chunk in the end of the transmission, this VE client may delay the whole delivery of the 3D asset. However, this kind of issues can be at least partially solved with bitrate monitoring.

As seen in Figure 9, RADE2 also performs relatively well with both ASE2 and ASE10. This is due to the fact that RADE2 favors VE clients that are run on plug-in devices (battery=100%) and have good processing speed. Most of these VE clients are either laptop or desktop computers that are also connected to fast WLANs or Ethernet.

When using only WebRTC for 3D asset delivery, RADE1 performs 17% faster than RAND on average. RADE1 outperforms also ASE2 by 35%, whereas ASE10 is already clearly faster than RADE1. In the experiments, the maximum number of VE clients used in the 3D asset delivery was only five (D_N=5) due to the small size of the test network. If the download bandwidth of R is abundant (e.g. LTE), increasing D_N would also presumably decrease the transfer completion times with WebRTC. Therefore, more experiments should be conducted in larger test networks with different values of D_N in the future.

Figure 10 illustrates the server bandwidth usage in the 3D asset delivery. When using WebRTC with ASE10, approximately one-third of the data was transmitted directly from VE clients, and with ASE2, two-thirds. With RADE, the use of WebRTC is increased automatically based on the level of load on the ASEs. Therefore, with bitrate monitoring in place, the delivery of a 3D asset would not cease even if the ASEs were temporarily unavailable.

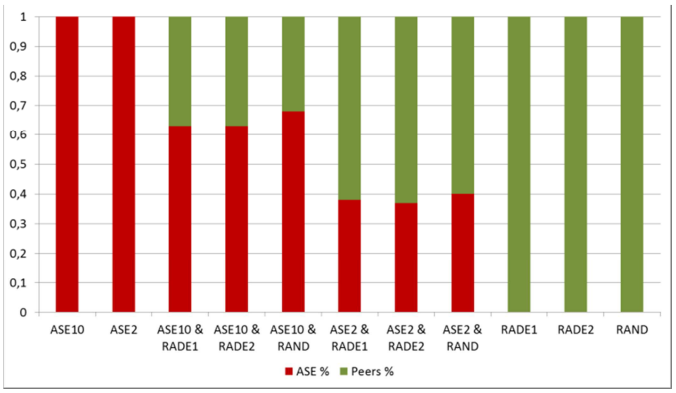

Fig. 10. Server bandwidth usage in the 3D asset delivery.

In Figure 11, the average battery status and processing speed of the chosen VE clients are presented for RADE2 and RAND. RADE2 clearly prefers VE clients with higher battery level (~87%) compared to RAND (~60%). Regarding processing speed score, RADE2 selected devices with about 11% better score compared to RAND. The difference in the processing speed score is not as significant, as 80% (see ω^2 in table III) of RADE2's total score was based on the battery status.

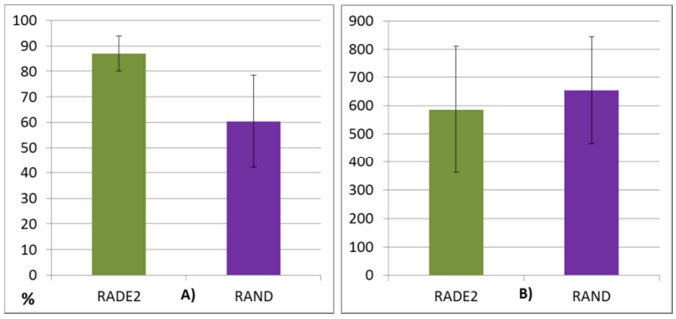

Fig. 11. A) Average battery status (higher is better) and B) average processing speed (lower is better) of the chosen VE clients.

VII. Discussion and Conclusions

This paper presents RADE, a method that enables resource-aware P2P-assisted 3D assets delivery in a web browser using WebRTC. Performance and resource-awareness of RADE was evaluated in real-life wireless networks using a prototype implementation. The results indicate that RADE can dramatically alleviate the load on ASEs, and thus reduce operation costs of VE service providers. It was also shown to improve application performance by reducing 3D asset delivery times, especially when the ASEs are under heavy load. Compared to purely random VE client selection, RADE was shown to improve performance, even though the utilized suitability score calculation formulas and system parametrization were in fairly early stage of development.

Although use of RADE can significantly reduce the load on the ASEs, the current implementation does not yet fully support circumstances when a VE client publishes a new or a modified 3D asset in a part of VE, which is populated by a large number of VE clients. In this case, several simultaneous download requests are initiated for the new or modified 3D asset, which can at first only be served by the publisher of the 3D asset and the ASEs. In case of a small 3D asset, the peak load on the ASEs is rather marginal, but with larger 3D assets (a few tens of MBs), super-seeding algorithm could be applied. In addition to storing a list of VE clients possessing a 3D asset, CMs could also act as trackers and maintain a list of VE clients currently downloading this 3D asset. In this way, new VE clients could be introduced as potential candidates during the download process if the target number of suitable VE clients were not otherwise found. However, using CMs also as trackers would increase their load and this feature should only be used in the circumstances mentioned above.

In the previous research [19], it was discovered that using multiple WebRTC data channels in parallel can increase the total transmission rate between two peers. This finding should be investigated further as it could improve data transmission performance, especially when only a few suitable VE clients are found.

With the current browser implementations, it is burdensome to gain estimates on VE clients' performance. It would be highly beneficial particularly for WebRTC data delivery if the estimates for VE client's Javascript, 3D graphics and data transmission performance could be directly provided by the browser. The estimates could be retrieved using either

in-build benchmarks and/or by continuously recording statistics of the browser performance in the background. The latter method would not cause practically any overhead and would not affect user experience in any way.

In the future, we aim to develop RADE further and to conduct more evaluations in terms of performance and energy-efficiency. Suitability score calculation formulas and system parametrization of RADE can be further optimized for different types of web applications with large-scale simulations and experiments in live networks. In addition, some system parameters could be adjusted during runtime, for example the ASEs allocation percentages (A_S and A_C) could vary in relation to the ASEs current load. Furthermore, as WebRTC is specifically developed for real-time data delivery, it can be assumed that RADE will be effective in the context of video delivery as well. Thus, future work will include evaluating RADE also for video delivery. Finally, we will also implement identification of users' that are mobile either through an accelerator or a GPS sensor to improve the resource-awareness of RADE. This is important especially from the standpoint of connection stability and speed as the mobile VE clients are constantly roaming between base stations and potentially between different wireless interfaces.

ACKNOWLEDGMENT

This work is supported by the European Celtic-Plus project CONVINcE and was partially funded by Finland, France, Romania, Sweden and Turkey. This work is also supported by the CADIST3D project funded by the Academy of Finland.

REFERENCES

[1] T. Dahl, T. Koskela, S. Hickey, and J. Vatjus-anttila, "A virtual world web client utilizing an entity-component model," in 7th Intl. Conference on Next Generation Mobile Apps, Services and Technologies, 2013, pp. 7–12.

[2] K. Sons, F. Klein, D. Rubinstein, S. Byelozyorov, and P. Slusallek, "XML3D: Interactive 3D Graphics for the Web," in 15th International Conference on Web 3D Technology, 2010, vol. 1, no. 212, pp. 175–184.

[3] J. Jankowski, S. Ressler, K. Sons, Y. Jung, J. Behr, and P. Slusallek, "Declarative integration of interactive 3D graphics into the world-wide web: principles, current approaches, and research agenda," in 8th International Conference on 3D Web Technology, 2013, pp. 39–45.

[4] T. Koskela and J. Vatjus-Anttila, "Optimization Techniques for 3D Graphics Deployment on Mobile Devices," 3D Reseasrch, vol. 6, no. 8, 2015.

[5] W. Wang, G. Yang, and N. Xiong, "A General P2P Scheme for Constructing Large-Scale Virtual Environments," in IEEE 28th International Parallel & Distributed Processing Symposium Workshops (IPDPSW), 2014, pp. 1648–1655.

[6] T. Alatalo, "An Entity-Component Model for Extensible Virtual Worlds," IEEE Internet Comput., vol. 15, no. 5, pp. 30–37, 2011.

[7] C. Jennings, T. Hardie, and M. Westerlund, "Real-time communications for the web," IEEE Commun. Mag., vol. 51, no. 4, pp. 20–26, 2013.

[8] T. Koskela, J. Vatjus-anttila, and T. Dahl, "Communication Architecture for a P2P-enhanced Virtual Environment Client in a Web Browser," in 6th International Conference on New Technologies, Mobility and Security, 2014, pp. 1–5.

[9] H. Liu, M. Bowman, R. Adams, J. Hurliman, and D. Lake, "Scaling virtual worlds: Simulation requirements and challenges," in Proceedings of the Winter Simulation Conference, 2010, pp. 778–790.

[10] J. Botev, A. Hohfeld, H. Schloss, I. Scholtes, P. Sturm, and M. Esch, "The HyperVerse: concepts for a federated and Torrent-based '3D Web,'" International Journal of Advanced Media and Communication, vol. 2, no. 4. p. 331, 2008.

[11] B. Shen, J. Guo, and P. Chen, "A survey of P2P virtual world infrastructure," in Proceedings of the 9th IEEE International Conference on E-Business Engineering, ICEBE 2012, 2012, pp. 296–303.

[12] C. Carter, A. Rhalibi, and M. Merabti, "Analysis of a novel hybrid P2P architecture," in Proceedings of the 11th IEEE Consumer Communications and Networking Conference (CCNC), 2014, pp. 508–513.

[13] A. Yu and S. T. Vuong, "A DHT-based hierarchical overlay for Peer-to-Peer MMOGs over MANETs," in IWCMC 2011 - 7th International Wireless Communications and Mobile Computing Conference, 2011, pp. 1475–1480.

[14] A. Zarrad, "A Survey of Routing Techniques for Mobile Collaborative Virtual Environment Applications," Int. J. Comput. Sci. Issues, vol. 9, no. 6, pp. 379–388, 2012.

[15] C. H. Chien, S. Y. Hu, and J. R. Jiang, "Bandwidth-aware peer-to-peer 3D streaming," in 2009 8th Annual Workshop on Network and Systems Support for Games, NetGames 2009, 2009.

[16] I. Kelényi and J. K. Nurminen, "Bursty content sharing mechanism for energy-limited mobile devices," in Proceedings of the 4th ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks - PM2HW2N '09, 2009, pp. 216–223.

[17] E. Esquivel, M. Rivero-Angeles, and A. Vázquez, "Priority Scheme for Mobile Nodes in P2P Bit-Torrent Based Networks," in 28th International Conference on Advanced Information Networking and Applications Workshops, 2014, pp. 469–473.

[18] Y. R. Choe, D. L. Schuff, J. M. Dyaberi, and V. S. Pai, "Improving VoD server efficiency with bittorrent," in Proceedings of the 15th international conference on Multimedia - MULTIMEDIA '07, 2007, pp. 117–126.

[19] A. Heikkinen, T. Koskela, and M. Ylianttila, "Performance Evaluation of Distributed Data Delivery on Mobile Devices using WebRTC," in Proceedings of the IWCMC Conference, 2015, pp. 1–7.

[20] A. Stephenson and D. Namiot, "On data transfer between mobile web clients," Int. J. Open Inf. Technol., vol. 3, no. 1, pp. 30–40, 2015.

[21] E. Harjula, O. Kassinen, and M. Ylianttila, "Energy consumption model for mobile devices in 3G and wlan networks," in Consumer Communications and Networking Conference, 2012, pp. 532–537.

[22] E. J. Vergara, S. Nadjm-Tehrani, and M. Prihodko, "EnergyBox: Disclosing the wireless transmission energy cost for mobile devices," Sustain. Comput. Informatics Syst., vol. 4, no. 2, pp. 118–135, 2014.

[23] A. Gupta, B. Liskov, and R. Rodrigues, "One Hop Lookups for Peer-to-Peer Overlays," in 9th Workshop on Hot Topics in Operating Systems, 2003, pp. 1–6.

[24] J. Sutter, K. Sons, and P. Slusallek, "Blast: A Binary Large Structured Transmission Format for the Web," in 19th International ACM Conference on 3D Web Technologies, 2014, pp. 45–52.

[25] M. Varvello, M. Steiner, and K. Laevens, "Understanding BitTorrent: A reality check from the ISP's perspective," Comput. Networks, vol. 56, no. 3, pp. 1054–1065, 2012.