Collaborative 3D Environments over Windows Azure

Jiri Danihelka, Lukas Kencl
R&D Centre for Mobile Applications (RDC)
Czech Technical University in Prague (CTU)
Prague, Czech Republic
{jiri.danihelka, lukas.kencl}@rdc.cz

***Abstract*—Microsoft Windows Azure and other cloud platforms are potentially well suited for building and deploying rapidly scalable 3D environments, usable for multi-player online games or interactive 3D education and e-commerce. Such 3D environments need to allow cooperation and interaction of multiple users in a single world. This may be difficult to synchronize due to network and cloud-operation delays. We present the project on building 3D services in the cloud, an example implementation of such an application, a shared 3D Teapot in Windows Azure, and a measurements-based discussion of performance issues when the teapot is shared and actively manipulated by many simultaneous users.**

***Index Terms*—3D environments; mobile cloud services; scalability; latency.**

I. Introduction

Cloud platforms, such as Microsoft® Windows® Azure™ [1], [2], are potentially well-suited for building and deploying rapidly scalable 3D environments, usable for example for multi-player online games or interactive 3D education and e-commerce. Cloud platforms might be particularly useful for their ability to rapidly scale with fluctuating workloads (due to e.g. fluctuating number of users) and might provide any level of high-performance computing power needed to operate the back-end of 3D applications and environments. Such 3D environments need to allow cooperation and interaction of multiple users in a single world, interacting with each other in real-time and sharing and effecting changes in the same environment. However, this could be difficult to synchronize due to network and cloud-operation delays. Delays of 10s or 100s of milliseconds may severely hamper the ability of users to share the 3D environments effectively.

Further questions arise when implementing such 3D shared service in the cloud: how to most effectively utilize the resources of the cloud platform when architecting such service, and how to distribute well the efforts of 3D rendering and action monitoring and synchronization between the cloud and the many individual clients with potentially different capabilities.

Proper understanding of distributed 3D worlds in the cloud should allow adding new kind of cloud services. An open platform for interactive 3D graphics in the cloud will result in strong social and commercial impact worldwide – new assistive services may be deployed or users may enjoy the experience of 3D internet shopping or other interactive services [3]. Such distributed 3D environments may improve productivity – e.g. by reducing travel – and have positive impact on the environment by the same token as well as by optimizing its own energy consumption.

In this work we present a prototype approach on building interactive multi-user 3D services in the Windows Azure cloud environment. The ultimate goal is to build a platform that would allow creating 3D virtual environments stored in the cloud and accessible by a variety of devices with very different performance and capabilities. This platform ought to allow easy creation of 3D virtual shops and games. Shopkeeper-users will be able to upload models of 3D-goods and easily customize the 3D talking-head and environment appearance. Some of the devices (e.g. smartphones) might not be able to store the entire scene geometry in their memory and will have to use only partial content downloading. The created 3D worlds need to allow cooperation of multiple users in a single world – and we investigate methods for providing scalable 3D worlds management in the cloud. The cloud-side application will automatically and instantaneously adjust and communicate the evolving world content to reflect the actions and changes effected by the many users interacting with objects in the 3D environment.

We present a prototype of such an application, a shared 3D Teapot in Windows Azure, and a measurements-based discussion of performance issues when the teapot is shared and actively manipulated by many simultaneous users.

Cloud computing is a highly suitable platform for introducing more sophisticated services beyond the current WWW client-server paradigm. This research focuses on the topic of dynamic sharing of 3D virtual-world content. Managing network and cloud latency and concurrency are vital aspects for usability of such platforms in highly interactive scenarios. Our practical, measurements-based analysis show the limits of the current cloud-based environment.

II. Related work

Maggiorini and Ripamonti [6] showed that game providers can benefit from scalable allocation of resources in the cloud thanks to the reduced cost of ownership and shorter deployment time. Cloud computing allows to avoid over- or under-provisioning of the game-server infrastructure. They also proposed a three-tier architecture for massively multiplayer online role-playing games (MMORPG) in the cloud.

Iosup at al. [7] presented an architecture for analytics of cloud massively multiplayer games and used it for analyzing

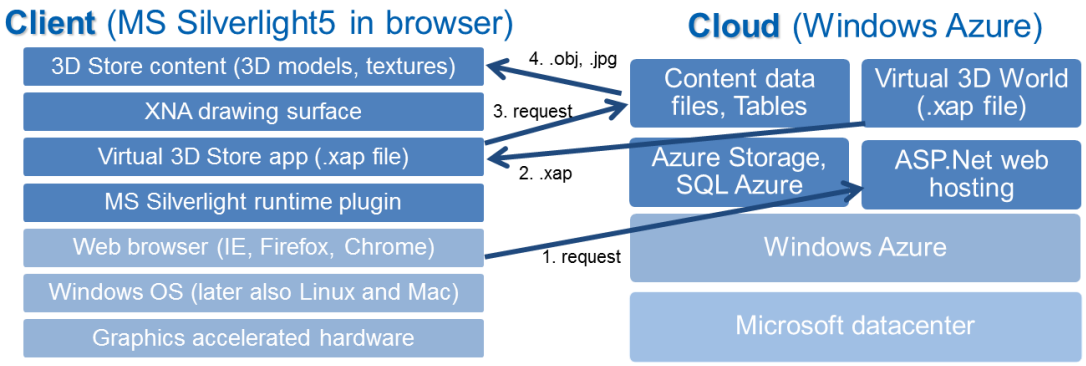

Fig. 1. Client and Cloud Functionality distribution

the popular game RuneScape. Their system is capable to record progress of 500,000 players for over a week.

Najaran and Krasic [8] presented an architecture for hosting first-person-shooting games in the cloud, reducing the aggregate bandwidth requirements by a scalable publish-subscribe subsystem. Players can set preferences for updates they receive, allowing to scale the game up to hundreds of players.

Windows Azure lacks support for unreliable services — it does not support communication over the User Datagram Protocol (UDP), often used in real-time games for its low latency, yet with the penalty of a decreased reliability. Many free-to-play games utilize the less reliable services of other providers (e.g. Amazon) that cost less.

Currently only a few games are based on Windows Azure. They are mostly social (multiplayer), turn-based games, i.e. games where players take turns to play. Tankster [9] is a prototype of turn-based shooting game by Microsoft. It is based on HTML 5 and also has a client running on the Windows Phone 7 smartphone operating system. This game serves as an example how to use Windows Azure Toolkit for

Fig. 2. Screenshot of a Client-server 3D talking head based on Windows Mobile 6.5 and OpenGL ES technology (without cloud support), previously developed at RDC [4], [5]

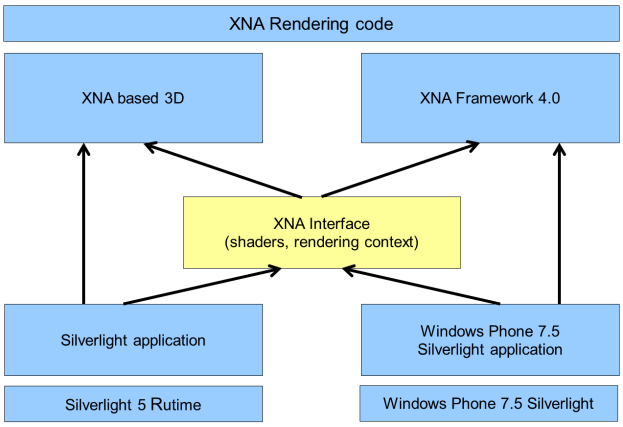

Fig. 3. Code reuse, browser and phone versions

Social Games [10]. Vampire Legacy [11] is a Facebook turn-based RPG game written in Adobe Flash that uses servers in Windows Azure. RobotZ [12] is a Facebook social robot-fighting game. According to the case study [13], Windows Azure provided it with a reliable and scalable infrastructure with simplified maintenance. Zombie Pandemic [14] is a turn-based MMORPG based on Windows Azure that is being developed by Pixel Pandemix startup.

The 3D Mobile Internet project at RDC [3] has, over the past few years, created multiple prototypes of platform-independent client-server 3D applications for mobile phones (see Fig. 2) and desktop computers. Recently we have oriented ourselves towards investigating cloud support for such platforms.

III. Architecture and Implementation

Our goal is to create 3D environments and objects that are stored in the cloud and can be accessed by a variety of devices with very different performance and capabilities. We seek to provide such virtual environments both within web browsers and on smartphones.

The application architecture consists of multiple layers, as described in Fig. 1. We use Silverlight 5 technology, as it introduces the ability to use hardware-accelerated 3D graphics using XNA in Silverlight applications. This opens up a whole new set of possible scenarios for 3D world visualization.

For the mobile client we use smartphones equipped with with Microsoft Windows Phone 7 operating system. Graph-

ical rendering is based on a combination of Silverlight and XNA [15] technologies. The cloud service is implemented using the Windows Azure platform and is accessible via a desktop Windows PC or any using a browser with a Silverlight 5 plug-in.

To reuse the rendering code between the web browser with Silverlight 5 plugin and Windows Phone 7, we have created a library that encapsulates the differences between those platforms (see Fig. 3). The most challenging issue of the library was with the graphics pixel shaders (programs that runs on a graphics card or chip). Windows Phone 7 forbids using a custom vertex and pixel shader, because it runs on a device with a graphics chip that does not support shader development and thus only a set of pre-defined shaders can be used. On the other hand, Silverlight 5 3D does not have some high-level rendering functions, to save on download size and to achieve higher portability. Those functions have to be implemented using low-level vertex and pixel shaders by developers. Thus Silverlight 5 forces its developers to use shaders. To overcome the shader problem we had to transfer the pre-defined Windows Phone 7 pixel shaders to Siverlight 5 and implement Windows Phone 7 pre-defined high-level rendering functions in Silverlight 5 using low-level pixel-shader code. In our applications, we have to use only those functions available on both platforms, thus we cannot use other shaders than those that are pre-defined on Windows Phone 7.

Our first prototype of a 3D environment is the 3D Teapot application (Fig. 4), a 3D object that can be both manipulated (i.e. have orientation or color changed by user-effected action) and observed in real-time by multiple users simultaneously, using different terminals. At every instant, the single currently valid teapot representation is being stored within Windows Azure. A Windows Azure Web Role is used as access frontend, while the actual 3D Teapot state is being permanently stored in Azure SQL and updated with every instance of manipulation of the teapot (see Figs. 5 and 6).

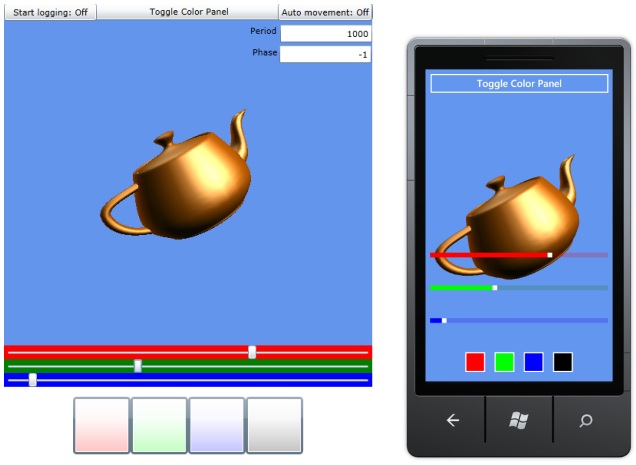

Fig. 4. Screenshots of two client versions of the synchronized Windows-Azure-based 3D Teapot, accessible simultaneously by multiple clients: mobile (XNA-based) and desktop (Silverlight 5 in a browser).

IV. Performance evaluation

A. Measurement setup

In the first experiment, we let 24 automated client instances attempt to manipulate the 3D teapot in Windows Azure simultaneously (see Figs. 7-8), to test scalability, reliability and response latency under a shared interaction scenario. The machines are connected at our university laboratory using a high-speed, 1 Mbps academic network. The simulation runs for about 1 hour, starting with only one instance of the teapot and incrementally adding other instances, always 2 per PC. We also keep decreasing the period of issuing a turning command to the 3D Teapot, starting with 1000 ms on the first machine and ending with 50 ms frequency on the last.

Another experiment we have carried out is to run the teapot shared between two instances on a single machine, who both tried to manipulate periodically on the order of 500ms, over laptop WiFi in a busy *Starbucks Cafe*, located in Prague city centre.

B. Results

1) University lab: It is clearly visible that already with about 6 instances running, the individual 3D Teapots are not

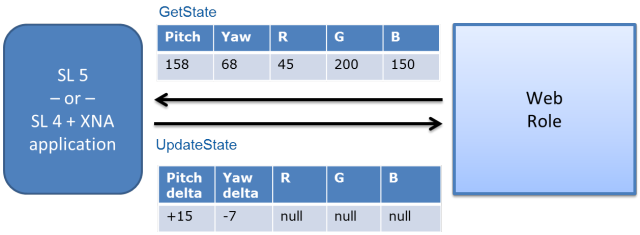

Fig. 5. Teapot state synchronization protocol. `GetState` returns all fields of the current Teapot state, `UpdateState` sends relative change on client to server, only the changed fields are bing sent. The client thus must keep track of changes (maintain a session), while the server does not.

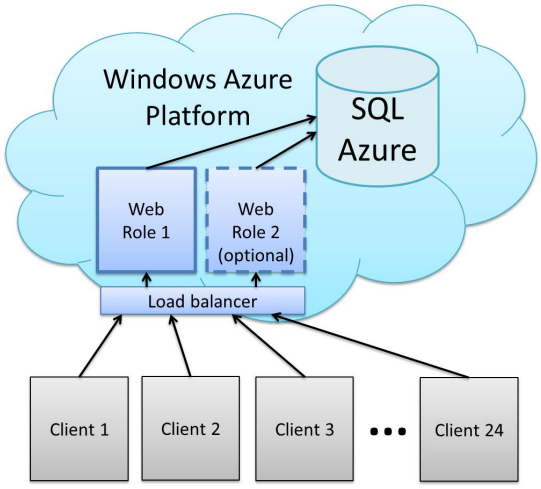

Fig. 7. Architecture of the experiment: Up to 24 client instances were simultaneously trying to connect to the web service to synchronize state of a single object in the database every 100 ms.

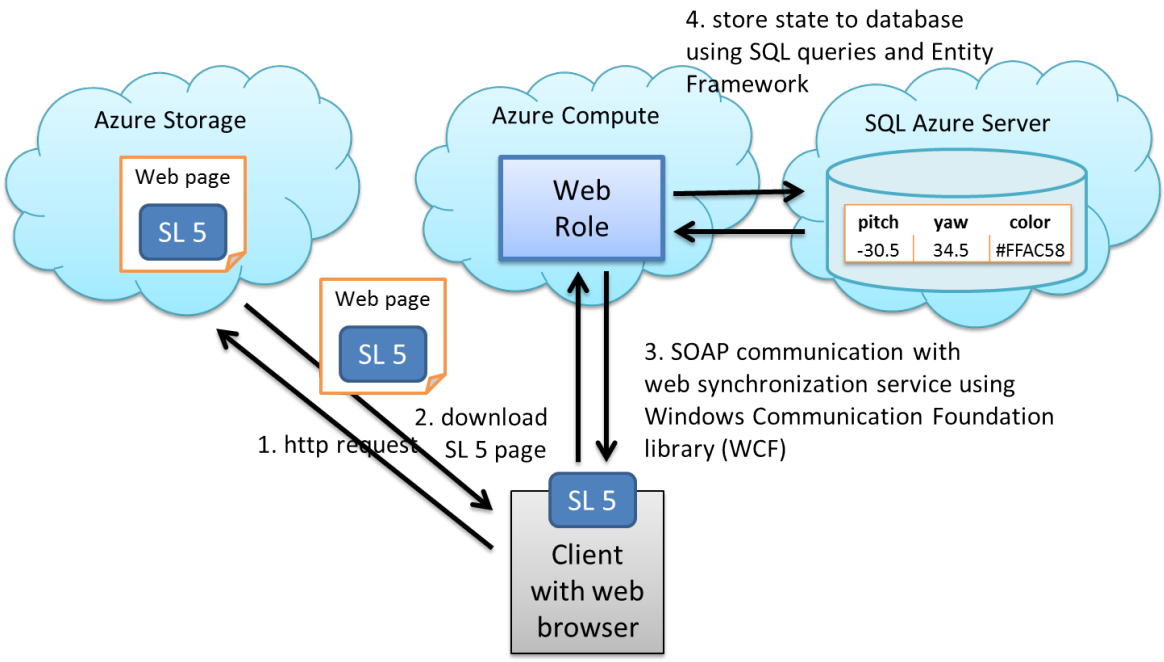

Fig. 6. Internal architecture of the 3D Teapot. The Azure SQL record stores the entire state of the Teapot.

properly synchronized, while with 24, there is no trace of synchronization at all.

Measurements of latency of responses from Windows Azure are presented in Figs. 9-11. In Fig. 9, we analyze the evolution of the experiment at one single host machine. At this host, per the entire experiment the mean latency is 91.40 ms with standard deviation 165.86, minimum and maximum latency of 46 ms and 8078 ms, and *median* latency of 63 ms. Clearly, there is a high degree of unpredictability, with most responses taking around 70 ms, some around 1200 ms (packet retransmits?) and some outliers above 5000 ms. Most likely this is the influence of TCP, clearly indicating packet losses and retransmits. With the growing number of teapot instances, latency becomes less predictable and manageable. This is confirmed by Fig. 10, analyzing the experiment over time as

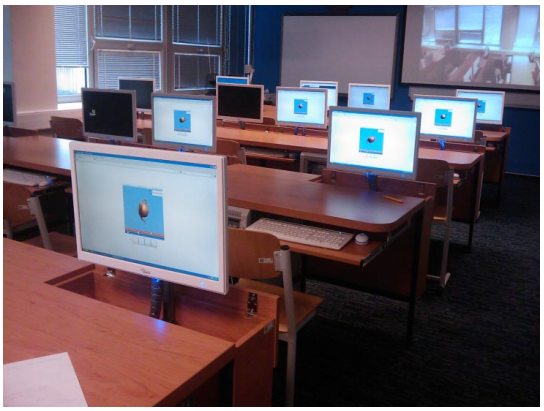

Fig. 8. Tests of multiple parallel clients simultaneously attempting to manipulate an identical 3D Teapot stored in Windows Azure cloud.

12 machines gradually join in the simultaneous manipulation of the teapot (all the 24 clients eventually join in, however, we have not been able to analyze all the 24 traces together due to the sheer volume of data). With further clients joining in, even the median-filtered response latency frequently exceeds 300 ms (see Fig. 10). Note the concentration around certain fixed time values - this is caused by the granularity of the measuring clock at the host machines. Figure 11 shows the growing trend in mean response time over all clients having joined, as well as the joint effects of latency and jitter. Finally, Fig. 11 shows that updating the Teapot state is more time-consuming in general than just reading it, indicating slightly longer time in processing the SQL write operation.

For the purpose of this simulation, we have only used a single Web Role frontend instance on the side of the cloud – it is likely that scaling there would yield better results.

2) City cafe: The latency measurements of simultaneous manipulations of the 3D Teapot in a Prague city cafe are summarized in Fig. 11. Here, the network effects are clearly much stronger than on the university high-speed connection network above. The mean delay value is 204.38 ms, with standard deviation of 100.953, minimum and maximum latency of 78 ms and 2101 ms, and *median* latency of 195 ms. We observe concentration around the median value with some responses being faster - this is most likely due to route caching in the provider network.

V. Conclusion and Future Work

We have investigated and prototyped novel methods and techniques for providing scalable, state-of-the-art multi-user interactive 3D services in the mobile cloud using Windows Azure as back-end. The early measured results show some

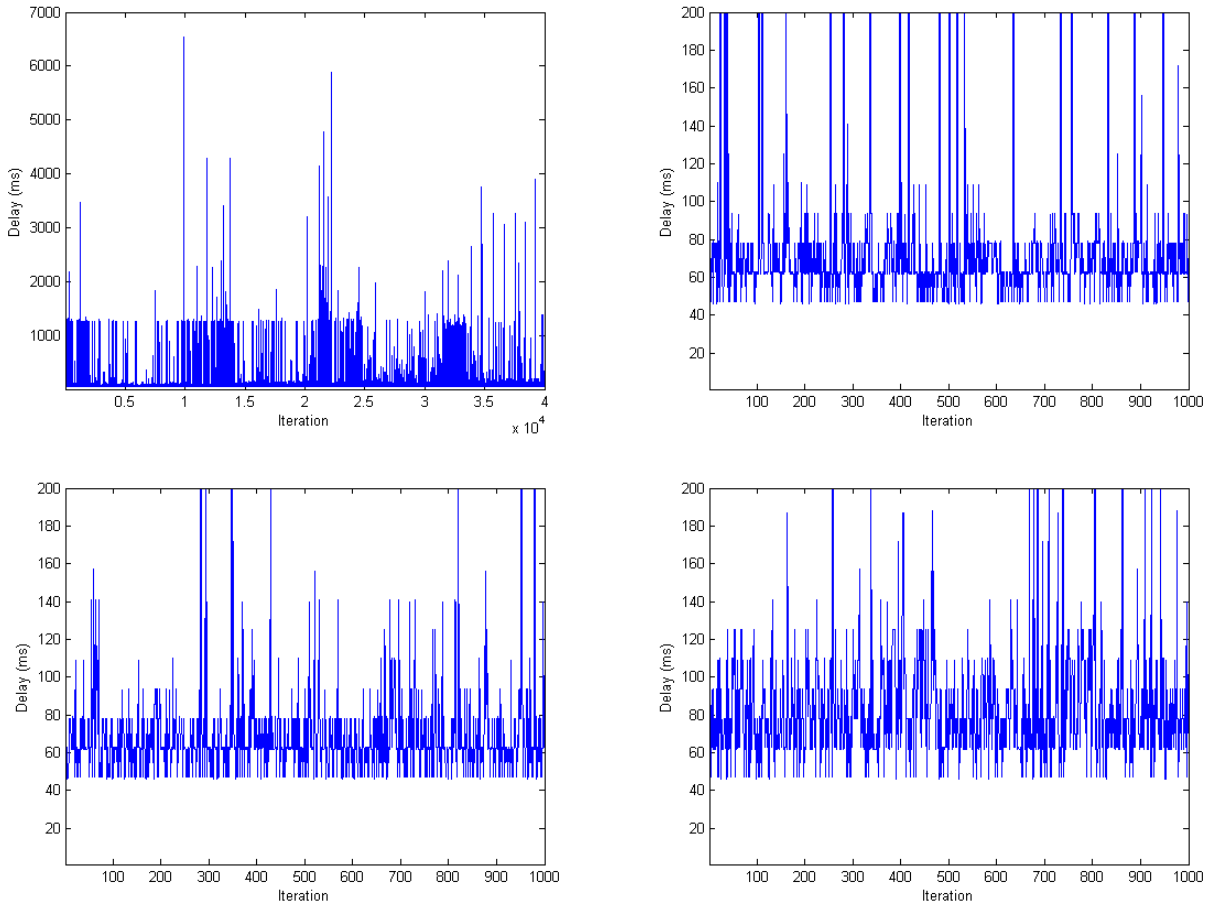

Fig. 9. 3D Teapot in windows Azure, response latency in milliseconds, the entire simulation (top left), requests 1000-2000 (top right) 20000-21000 (bottom left) and 40000-41000 (bottom right).

question marks as to whether Windows Azure can provide enough reliability and short-enough latency to support true multi-user interaction in 3D environments in the cloud. In particular the relatively high variability in the latency (standard deviation) may be a factor; the reliance on TCP is hurtful in this respect. More measurements and perhaps alternative approaches to SQL Azure for storing the immediate 3D environment state need to be tried to determine the best approach. However, an early conclusion can be reached that for truly real-time environments, intermediate nodes (cloudlets [16]), deployed geographically closer to the users, are probably necessary to reduce the overall latency.

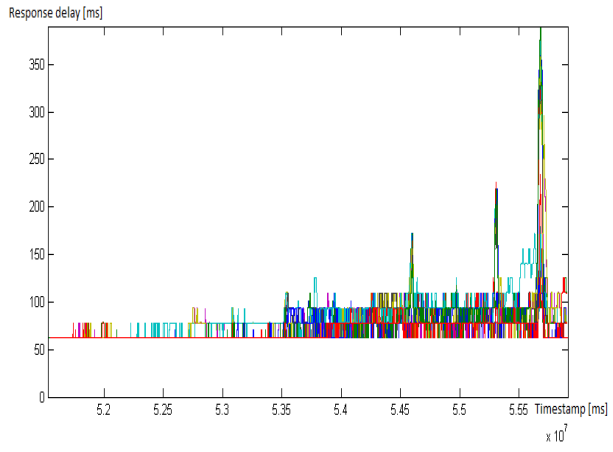

Fig. 10. Measurement scenario when 12 clients gradually connect to a single Azure server. Outliers are filtered out by using a median over a window of 100 values. Notice that response time increases with multiple clients connecting.

This work makes an important step by demonstrating the feasibility of easily sharing a cloud-based 3D object, using mobile as well as fixed terminals. Our ultimate goal is to provide an open platform for other researchers to build on and a tool for easy deployment of 3D services, having potential impact in many interactive domains (e-commerce, assistive technologies, real-time interaction, gaming, etc.). It will foster rapid growth of visually attractive 3D applications appearing in the mobile cloud marketplace.

A further open research question is how to allow dynamic component-code migration between client/phone and server/cloud at runtime. This concept is beginning to attract attention among computer science researchers – first prototype approaches have been described by Cuervo et al [17] or Chun and Maniatis [18]. In the context of 3D environments in the cloud, clarity is still lacking as to the proper distribution of work between the mobile client, the datacenter and a possible intermediate entity such as a cloudlet [16].

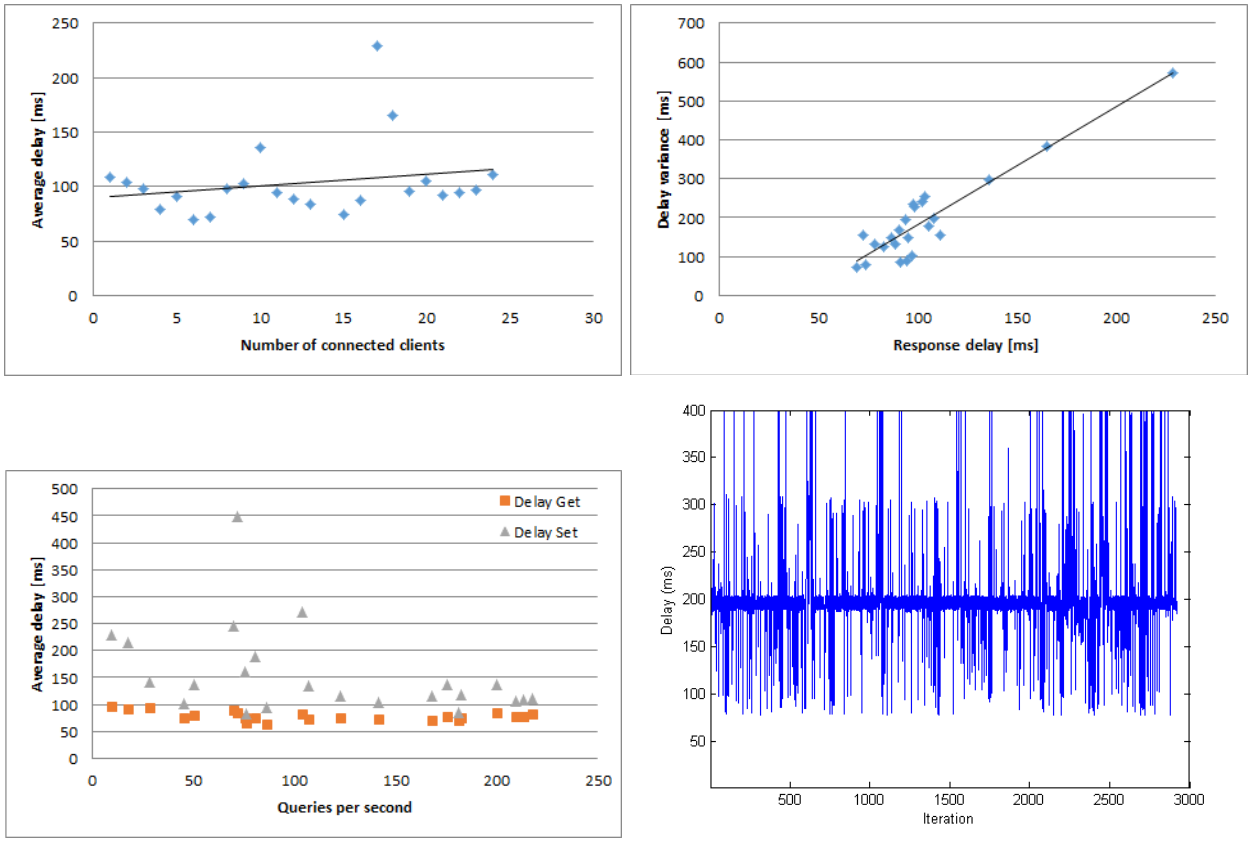

Fig. 11. Top left: Average response time based on the number of connected clients. Response time increase with additional clients. Top right: Response variance based on response delay. Intervals with higher response delay have also higher delay variance. Bottom left: Response times per different type of queries. Queries that change state in the database take more time than merely reading. Bottom right: 3D Teapot in Windows Azure, response latency in milliseconds, when used for sharing and active manipulation by two users in a City Cafe.

ACKNOWLEDGMENT

This project was partially sponsored by Microsoft Czech Republic and the CTU FEE Interoperability Lab. Many thanks also belong to researchers in the Microsoft Research Redmond Connections and Networking Research Group, especially Dr. Judith Bishop, Dr. Victor Bahl and Dr. Stefan Saroiu, for initial project consultations. This work was supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS10/291/OHK3/3T/13. The research was supported by the grant No. DF12P01OVV028 of the Ministry of Culture of the Czech Republic.

REFERENCES

[1] R. Jennings, *Cloud Computing with the Windows Azure platform*. Indianapolis, IN, USA: Wiley Publishing Inc., 2009.

[2] Microsoft, "Windows azure cloud platform," http://www.windowsazure.com/.

[3] RDC, "3D Mobile Internet project," http://www.rdc.cz/en/projects/3DMobileInternet/.

[4] J. Danihelka, R. Hak, L. Kencl, and J. Zara, "3d talking-head interface to voice-interactive services on mobile phones," *International Journal of Mobile Human Computer Interaction IJMHCI*, vol. 3, no. 2, 2010.

[5] J. Danihelka, L. Kencl, and J. Zara, "Reduction of animated models for embedded devices," in *International Conference on Computer Graphics, Visualization and Computer Vision*, 2010.

[6] D. Maggiorini and L. Ripamonti, "Cloud computing to support the evolution of massive multiplayer online games," *ENTERprise Information Systems*, pp. 101–110, 2011.

[7] A. Iosup, A. Lăscăteu, and N. Ţăpuş, "Cameo: enabling social networks for massively multiplayer online games through continuous analytics and cloud computing," in *Proceedings of the 9th Annual Workshop on Network and Systems Support for Games*. IEEE Press, 2010, p. 7.

[8] M. Najaran and C. Krasic, "Scaling online games with adaptive interest management in the cloud," in *Proceedings of the 9th Annual Workshop on Network and Systems Support for Games*. IEEE Press, 2010, p. 9.

[9] Microsoft, "Tankster beta," http://http://www.tankster.net/.

[10] Nathan Totten, Microsoft Research Redmond, "Windows azure toolkit for social games," https://github.com/WindowsAzure-Toolkits/wa-toolkit-games.

[11] Sneaky Games, "Vampire legacy," http://apps.facebook.com/vampirelegacy/.

[12] Kobojo, "RobotZ Arena," http://apps.facebook.com/robotzarena/.

[13] Kobojo, "Game developer meets demand and prepares for success with scalable cloud solution," http://www.microsoft.com/casestudies/Case_Study_Detail.aspx?CaseStudyID=4000008216.

[14] P. Pandemic, "Zombie pandemic," http://pixelpandemic.net/zombie-pandemic/.

[15] MSDN Library, "Combine Silverlight and the XNA Framework in a Windows Phone Application," http://msdn.microsoft.com/en-us/library/hh202938(v=vs.92).aspx.

[16] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14–23, 2009.

[17] E. Cuervo, A. Balasubramanian, D. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "Maui: making smartphones last longer with code offload," in *Proceedings of the 8th international conference on Mobile systems, applications, and services*. ACM, 2010, pp. 49–62.

[18] B. Chun and P. Maniatis, "Augmented smartphone applications through clone cloud execution," in *Proceedings of the 12th conference on Hot topics in operating systems*. USENIX Association, 2009, pp. 8–8.