Virtual and Augmented Reality: Improved User Experience through a Service Oriented Infrastructure

Michael Braitmaier
University of Stuttgart
Stuttgart, Germany
Braitmaier@hlrs.de

Dimosthenis Kyriazis
National Technical University of Athens
Athens, Greece
dimos@mail.ntua.gr

Abstract—Service Oriented Infrastructures allow for on-demand provision of Information and Communication Technologies assets. What virtual environments require from such infrastructures refers to the offered Quality of Service (QoS) levels, which have direct impact on the end users experience. In this paper, we describe a new approach to improve the user experience, with regard to QoS, of collaborative Virtual Reality (VR) and Augmented Reality (AR) working sessions. One of the main issues that have to be addressed in such a use case is the requirement for keeping QoS with regards to real-time of the shared multimedia content of the AR. Therefore we adapted our application to a new and innovative real-time enabled framework for service-based infrastructures, implemented within the framework of the IRMOS EU project. This framework provides the necessary guarantees for successful collaborative VR and AR sessions as derived from the desired user experience. We also demonstrate the operation of the implemented framework and evaluate its effectiveness to the collaborative Virtual and Augmented Reality scenario.

Virtual Reality, Augmented Reality, Quality of Service, Realtime, Collaborative Environments, Service Oriented Infrastructure

I. INTRODUCTION

Augmented Reality (AR) and especially Virtual Reality (VR) [1] are around for quite some years. While Virtual Reality has evolved quite well and also made the step forward into industry, Augmented Reality still stays behind. Even though it is spread among research facilities worldwide and in some industry areas, especially in the training sectors for industrial productions, the final step into the production phases of industry environments is yet to come. But with increased interest of industry in this technology, the question arises how it actually can be applied, not only locally, but also in the context of collaborative working environments, as these become more and more important. Collaborative working environments push the requirements for AR even further in terms of network and computing performance, especially with an increasing quality of the AR video streams as it is desired by the industry. Considering that AR also requires computing tasks to be performed in parallel to actually create the visualization, it causes a serious computing load, for overlaying the visualizations on the AR video images. In the end this results in requirements for very high-end computing systems. However, for the non-technical user in terms of Information and Communication Technologies (ICT), this mainly boils down to user experience. User experience is the key factor for intuitive use of scientific visualizations in collaborative environments. Given the aforementioned, user experience in such environments heavily depends on the infrastructure and the guarantees in terms of QoS it can offer. To this direction, emerging service-based platforms are being developed targeting to meet these requirements.

Service Oriented Architectures (SOAs) [2] aim at improving the user experience with on-demand provision of computing, storage and network resources. The vision of this improvement, which actually refers to the guaranteed level of QoS, has driven the emergence of a new generation of Service Oriented Cloud Infrastructure [3] implemented within the framework of the IRMOS project [4]. The proposed cloud solutions address these issues by allowing the specification of QoS parameters in the description of an application prior to execution, and the provision of resources that meet the QoS requirements during application execution.

Over the last 10-15 years Augmented Reality has been developed from an initially researched focused to an industrial affiliated technology. Interest in industry increases more and more. This has been prepared by applications such as [5], but also [6]. However in 1997 the statement [7] is that AR is still far away providing usability that is acceptable, a statement that hasn’t changed until nowadays as the authors of [8] highlight. One of the aspects that contribute to usability is the user experience, which is directly influenced by the workload on network and computing resources for such tasks as AR video transmission and also video transcoding, mainly caused by delays in the workflow and network drops [9]. However considered in local usage by industry, AR in the meanwhile is seen as a means to improve productivity, e.g. in training of employees and design processes. Therefore the need of stable remote AR with guaranteed QoS to achieve a certain user experience is required.

In this paper we present an approach for improving user experience in a collaborative Virtual and Augmented Reality session. The approach focuses on the use of a Service Oriented Infrastructure (SOI) as well as the adaptation of the application in order to enable its execution on the service based environment. We start by discussing how the user experience in the Augmented and Virtual Reality application is reflected to specific parameters (in application terminology) and based on
these parameters what the workload of this application on a SOI is expected to be. During this process we present the mechanisms behind the tools for making the process as intuitive as possible.

The remainder of the paper is structured as follows: Section II provides an overview of the VR and AR scenario, discussing also how QoS is related to the user experience and what are the current limitations and drawbacks. Furthermore, it describes what these QoS parameters finally mean in terms of hardware resources. Once identified the QoS parameters the chapter about “Service Oriented Infrastructure Offerings” is then actually describing the framework of a cloud infrastructure that is able to provide the required QoS guarantees. In Section IV we present both how VR & AR application has been adapted and how it is executed on the SOI, while Section V includes a brief evaluation of the proposed approach. The paper concludes with results from the application executions and a prospect to future work.

II. VIRTUAL AND AUGMENTED REALITY SCENARIO
OVERVIEW

In this section we present an overview of the Virtual and Augmented Reality scenario as well as the requirements set by the end users of the application, related to their perceived experience. Moreover, we discuss on the current limitations of the application and the need for service platforms that can provide QoS guarantees and thus improve the users’ experience.

The collaborative Augmented Reality application has been setup in the context of the Collaborative Visualization and Simulation Environment (COVISE [10]) to allow a multidisciplinary team from an automotive company to collaboratively work on an early car product model. The intention for the user group is to eliminate early problems in one of the disciplines that could impact on the other one. An individual or a part of that team however might not be physically located at the work location, e.g. due to other obligations in another country or due to the fact that their office is located in a foreign country. In our case the local AR working environment is a big wind tunnel to evaluate simulation results of wind drag for the car. Therefore the AR session is collaboratively shared among the local group in the wind tunnel and the remote groups at their individual location. The following figure (Figure 1.) depicts the high-level setup of the application in the conventional way with the local working environment represented AR workstation and the remote individual or group, being represented by the CAVE workstation. The figure clearly depicts how the transcoding nodes for AR are located at some location in the internet, resulting in links crossing many subnets until the stream reaches its final destination. Each subnet thereby has its own priority in routing and performance feature. Thus a clear statement on the overall performance is hard to make.

Figure 1. Conceptual schema of the Remote Augmented Reality application

In the application setup we have the above mentioned two user endpoints, where the AR workstation is transmitting an AR stream and the CAVE workstation is receiving the stream. In between these two endpoints, there is the AR video transmission link. For purpose of adaptation to the network performance, we use two transcoders that can apply different video codecs to adapt the workload on the network path through the internet as desired and decode it at the end into a format the COVISE application can process. This allows delivering different quality and performance levels to different remote users. The reason for that could be either the network performance towards a remote user or group or the locally available performance of the remote user’s visualization hardware in contrast to the AR sending visualization hardware.

The bottleneck is definitely the video transmission of the AR video stream over the internet, especially if we are working with stereoscopic AR, which doubles the workload. In addition the computing resource required for transcoding needs also to be taken into consideration. For a uncompressed AR video with a resolution of 640 x 480 at 24 bits per pixel at 25 frames per second (FPS) in stereoscopic mode a bandwidth of approximately 44 MByte/s. Approaching the problem from the user experience, the user group itself expects the whole visualization to run smoothly and not having any lags in the AR video stream, while experiencing a good quality to evaluate their simulation results. These requirements apply especially to the remote group which is depending on the remote AR video stream. In general, a frame rate of 25 to 30 frames per second (FPS) results in a smooth running video at a sufficient video quality. Therefore the parameter the user group intuitively specifies as QoS parameter is the smooth running AR video stream. The latter can be expressed in the frame rate, which allows staying in application terminology for the specification purposes and not describing the needed resources in terms of CPU, memory, bandwidth, etc. Another set of criteria referring to the image resolution, video codec and the color depth, have also to be specified to draw conclusions on the resulting frame rate in the end. This bundle of parameters altogether describes the actual workload that has to be considered for the transcoders and the network links connecting them. Using the Internet for transmission the performance provided on the network link however varies dependent on the load on particular network segments. Under this condition a reliable QoS can not be guaranteed for the AR stream from the local wind tunnel facility to the remote locations. In the worst case,
high variances in the network performance have to be accepted, which directly leads to a loss in image quality through increased video compression or lagging video playback of the AR stream at the remote location.

III. SERVICE-ORIENTED INFRASTRUCTURE OFFERINGS

Service Oriented Infrastructures are based on SOA principles [2], mainly emphasizing implementation of components as modular services that can be discovered and used by the end users. Such architectural paradigms are incorporated into Clouds, which provide the ability to efficiently adapt resource provisioning to the dynamic demands of Internet users. A classification [11] that is generally accepted separates the cloud offerings into: a) Infrastructure as a Service (IaaS), which refers to the hardware provision, b) Platform as a Service (PaaS), which refers to the provision of a development platform and environment providing cloud-based services, c) Software as a Service (SaaS), which refers to the provision of an application as a service.

An implementation of the aforementioned offerings has been provided by the IRMOS project, which as depicted in Figure 2., is composed by the Service Engineering Tools [12] that enable the adaptation process of applications to a SOI (covering mainly the SaaS layer) which will be described in more detail in chapter IV.A, the Framework Services [13] (covering the PaaS service model) and ISONI [14] (an Intelligent Service Oriented Network Infrastructure covering the IaaS service model). The final outcome is an infrastructure that aims at guaranteeing the requested QoS for an application.

The specific environment considers the full service lifecycle of service-based systems deployed on cloud resources including service engineering, Service Level Agreement design, provisioning and monitoring. QoS parameters at application, platform and infrastructure levels are given specific attention as the basis for provisioning policies in the context of temporal constraints. For specification of the QoS a Service Level Agreement (SLA) has to be negotiated, which is named in this context as Application-SLA (A-SLA) since it includes application related terms (e.g. frames per second) and not resource attributes (e.g. memory). In order to include the high level QoS parameters in the A-SLA, the IRMOS framework offers tools to model the application components based on a UML language notation. The main offerings of the specific SOI span across the Cloud classes that have been briefly described previously and are summarized in the following sub-sections.

A. Real-time QoS Specification

It mainly refers to a complete specification of the application and its components in order to enable the definition of the QoS parameters that affect the user experience in such a virtual environment. The specification is achieved through models and parameters with regard to the workload that affect the performance requirements (e.g. number of connected users). The main outcome is a complete set of the application service components, their descriptions as well as the corresponding application description consisting of the whole application setup in terms of topology. Thus the end-user’s experience is reflected to specific QoS terms that are included in an Application-SLA and according to which the provision of the resources will take place.

B. Dynamic SLA Negotiation & Resource Provisioning

Another offering of the specific SOI, refers to services supporting the dynamic negotiation of Application-SLAs, which include the end-user’s QoS requirements, and Technical-SLAs, which include the corresponding resource attributes in order to meet the user’s requirements. These attributes are being produced according to the QoS specifications (as discussed in the previous section). The negotiation services enable re-negotiation that may be triggered in case of performance degradation in order not to deteriorate the user experience. Furthermore, platform services allow for virtual service networks to be provided as a means to access the IaaS layer.

C. QoS Event Monitoring

Measuring QoS at both application and infrastructure levels that trigger events for runtime adaptability of resource provisioning estimation and decision making is being provided by the IRMOS platform. The response times of the monitoring framework enable the improvement of user experience in virtual environments given that in cases of SLA violations or performance degradations, these are being monitored and corresponding events are triggered in order to guarantee the agreed QoS levels. To enable the latter, an instance of the monitoring service is deployed in the virtualized environment to monitor the application service components and provide corresponding reports to the monitoring service.

D. Event Management

Event management refers to a framework that triggers and handles events being produced both from the application and the infrastructure in order to guarantee the offered level of quality. The events produced target the workflow management services, which are responsible for configuring, starting and

Figure 2. Architectural overview in the IRMOS framework
stopping the applications. Similar with the monitoring framework, an instance of the workflow enactor service is deployed in the virtualized environment to invoke the application service components according to the events being consumed.

IV. VR AND AR IN SOI

Applications in SOI are provided through application providers, e.g. an application provider offers the collaborative AR application to an interdisciplinary team of engineers and designers of the automotive industry. For specification of the QoS, a Service Level Agreement (SLA) has to be negotiated, which is named in this context as Application-SLA (A-SLA) since it includes application related terms (e.g. frames per second) and not resource attributes (e.g. memory). In order to include the high level QoS parameters in the A-SLA, the IRMOS framework offers tools to model the application components based on UML language notation. On a higher level, the whole application has to be modeled in terms of topology and link specifications which together form the basis of the A-SLA that is afterwards negotiated with the PaaS provider. That means that the A-SLA contains the outcome of what has previously been described in chapter III.A. Therefore a web interface is provided that allows the customer to specify the QoS parameters with which the application is going to be deployed.

A. Adaptation of the application

Prior to the execution of the VR and AR application on a SOI that facilitates real-time and provide QoS guarantees improving the user experience, there are a number of steps that have to be completed. In this section we describe these steps along with the tools used in each step.

The application has to be adapted in different aspects and dependent on their native architecture. In principal distinction is made between two application classes: Monolithic application and componentized applications. The effort to adapt a monolithic application compared to an already componentized application is larger, as it first has to be “broken up” into several components that can be deployed as services in a SOI. Once this task is performed it ends up on the same procedure that has to be applied to already componentized applications. To bring these applications on the cloud infrastructure, the first steps refers to modeling each component, packaging it with the model description and providing it to the application provider. Furthermore an application description is required that describes the topology and the composition of components that form the application that will be deployed. The UML modeling of the application takes place in the Open-Source UML tool Papyrus [15]. When the components of an application are adapted, an application developer has to model the components in terms of properties and valid property value ranges in order to allow afterwards the PaaS provider to provide the necessary resources that meet the QoS requirements of the application. For simplification of this modeling process the UML environment has been enriched with special UML templates, which provide the necessary framework to map the application requirements into the UML language. Each component that has been developed for an application has to be described using the aforementioned modified UML template. A component is thereby given as a class in UML notation, where the properties describing the component in terms of performance constraints are modeled using class members. Components offering a service to the overall application in the context of the IRMOS framework are called Application Service Components (ASC) whereas their UML description is called Application Service Component Description (ASCD). An example of a video streaming server ASC’s description used in the Virtual and Augmented Reality Application is given in Figure 3. The ASCD describes the most important parameters, the FPS and the image resolution in conjunction with some additional parameters, which are partially application dependent. One of such parameters is the “ServerReady” parameter in Figure 3. that signals the Framework Services (FS) of the PaaS layer when the AR video transcoding server is ready to accept incoming client connections. Through monitoring, this is feed into a workflow management component that based on this signal executes the further AR video streaming clients first when the according server is ready to accept their connection requests. The goal of the whole modeling is to have a complete description of an application given in UML, consisting of the individually modeled components as well as a model of the application composition and its topology, including network links. Beside the ASCD mentioned above, furthermore an application description (AD) is needed. For the actual value specification an instantiation of a previously modeled ASC description is required. The instantiation of an ASCD differs from an ASCD with regards to the properties of the component. In the ASCD instantiation each parameter is defined with a concrete value or a value range. Where concrete values have been specified no further changes are possible in the A-SLA negotiation later on when an application should be deployed in a service oriented infrastructure. For parameters that have only a value range specified in the ASCD instantiation these can be later on refined when negotiating the SLA. A specially developed conversion algorithm for Papyrus converts the UML-based instantiation models into XML-style service level agreement templates being used by the customer as the basis of an A-SLA negotiation in the web portal. In this web portal the template can then be concretized to a final A-SLA by exactly defining any open parameter values.

Figure 3. Example of an Application Service Component Description modelled in the extended UML notation of IRMOS
Besides describing the components and the application, using UML as described above, a binding between the ASCs and the PaaS services has to be provided by the application component developer in order to allow monitoring. IRMOS offers the opportunity to do this binding in a very simple and effective way. Each ASC is controlled by the PaaS services for configuration (based on the parameters as negotiated in the A-SLA), controlling the start and stop processes of a component and monitoring of performance critical application parameters (as previously specified in the individual A-SLA). This approach poses three requirements on every application component developer:

- Implementing monitoring of application parameters that are performance critical
- Implementing binding-scripts that transform the PaaS services invocations into application specific actions
- Implement binding-scripts for configuration of the application dependent execution environment, as well as basic execution control, e.g. start and stop commands towards the ASC

Our Virtual and Augmented Reality application components implement these binding scripts using perl scripts. The FS use a fixed command line format for passing the parameters to the scripts which resolves to the format as shown in (1).

\[
<\text{parameter\_name}> = <\text{value}>
\] (1)

The processed parameters are passed on to the application components or stored in files that can be processed by the application. The binding-scripts can be implemented in anyway and language the developer desires as long as their command line processing adheres to the format as described in (1). In Figure 4, the integration of the monitoring mechanism and its binding to the Framework Services is depicted. The measured value is expressed as FPS. The measurement is performed at two different points in the processing chain. First the FPS at the “AR Video stream Receiver Server” (see Figure 4.) side coming from the application is measured to evaluate the input data staying to the QoS constraints. Additionally a second measurement is made on the “AR Video stream Transcode Server” (see Figure 4.) side when the transcoded data is sent onto the QoS guaranteed network within the IRMOS domain. This helps detecting deviations from the requested QoS in the A-SLA. It is however clear that this monitoring functionality is application dependent and it is recommended for an adapted application to at least monitor once per component the parameter that has been identified as a QoS constraining parameter. To avoid serious impact of the FPS monitoring on the frame-rate the ASCs provided through their application dependent mechanism and writes them to a XML-style file. The information in this file can then be parsed by the PaaS monitoring services’ on a regular basis [16], using a polling schema, in order to trigger corrective actions towards the IaaS provider if necessary (e.g. performance degradation during the execution of the application). The monitored data, which in the case of the AR video streaming is actually the frame-rate of the AR video stream is additionally going to be presented in the web interface, where the consumer can cross-check it against the initial requirements (as included in the A-SLA).

In case the requested Quality of Service cannot be kept, which is referred to as an SLA breach, dependent on the business model of the involved parties either the user will be informed on the breach or the SOI will try to compensate for the underperforming QoS through various mechanisms, e.g. live migration or improving resource allocation of components. In the case of a user triggered change of the QoS, e.g. the user desires better quality than he originally requested, a renegotiation of the A-SLA can be triggered to cause the infrastructure to adjust computation and network resources to fulfill the new QoS constraints. In the collaborative AR application this is the case when a remote participant decides for this QoS change during the running session, e.g. improving resolution of the AR video stream.

Figure 4. Example architecture of application-based monitoring and its binding to the IRMOS Framework Services
Figure 5. Image comparison for the two tested scenarios: Left side lossy compression with traditional approach; Right side lossless compression in HQ with QoS guaranteed approach. The image to the left clearly shows the artifacts around the splashboard that accompany the lossy best-effort setup.

V. RESULTS

To evaluate our approach we tested the performance of our collaborative VR and AR application COVISE in two different setups. The first setup was under the traditional setup as it has been described in chapter II, while the second setup utilizes the SOI for Quality of Service of the collaborative session. In both cases mono AR video was streamed. In the first scenario the video was transmitted through an application specific collaboration provider, while in the second scenario a video streaming server in conjunction with a video streaming client was used in the IRMOS infrastructure. Both times the video was recorded at a resolution of 640x480 with 20 fps. The image page shows the differences in video quality between the two scenarios. The previously outlined application achieved the same performance at a better image quality, while the AR application running on best effort network had to be run with the lossy MPEG-2 video compression, to continuously keep the same performance. Performance aspects that are contributing to the different FPS are on the one side network performance directly impacting the need for video compression and on the other side computation power required for higher degrees of compression. In the case of guaranteed QoS the IRMOS framework provisioned a continuous level of network bandwidth and computing power to ensure the processing and transmitting of the AR video stream at the required speed to keep the user requested FPS. In contrast to this the non-QoS case was under the influence of interfering network traffic which required applying the lossy-compression to keep the overall data volume used for AR video transmission low. Variances in processing load on the compute resources for the case of lossy-compression has not been considered in this evaluation and is expected to introduce partial FPS drops over the application lifetime, but is subject to further measurements.

Even though there might be several subnets below the SOI this is transparent to the application due to the virtualization of the network. Thus the application interacts with the deployed services as if they all were in the same subnet space of the network. It can clearly be seen in TABLE I. that the approach without the IRMOS SOI has nearly the same performance. However, the image panel in Figure 5. shows obvious differences in image quality. To better see the deviation in quality, a part of the image has been scaled up and inserted in the image panel, clearly showing the MPEG-2 artifacts that are especially occurring when there is a lot of movement in the video. Nevertheless, movement in our application is caused repeatedly because of the need to move around the real-world object with recording camera for the AR.

<table>
<thead>
<tr>
<th>Application</th>
<th>FPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR/VR application on best-effort internet lossy compression</td>
<td>18</td>
</tr>
<tr>
<td>AR/VR application on IRMOS SOI with guaranteed QoS (captured 20 FPS)</td>
<td>15†</td>
</tr>
</tbody>
</table>

† The difference in the FPS results from network IO overhead with large amount of data as it is experienced when using low compression of frames.

VI. CONCLUSIONS AND FUTURE WORK

In this paper we presented a new approach to guarantee an user experience for collaborative Augmented Reality applications, based on an initial negotiation with a Service Level Agreement by a customer. For this purpose the user experience has been expressed as parameters on the application level (e.g. frames per second) describing the workload of the application in terms of Augmented Reality. Expressing the parameters on application level doesn’t mean that low-level (i.e. resource parameters) are not taken into account. The proposed approach allows mapping of application parameters to low-level ones as explained in Section III of our paper, enabling the users to deal only with parameters / terms that are directly related with the experience of the Virtual and Augmented Reality sessions (i.e. frames per second). The parameters have been modeled on a per component-basis in advance by an application developer using new service
engineering tools, that have been developed in the context of the IRMOS EU project. The components form the foundation for application service level agreements. Once negotiated the A-SLA provides the basis for the Quality of Service level of the application. This level is actually the desired user experience for the execution phase of the application. During the execution, the framework services continuously monitor the application parameters while the infrastructure provider guarantees the required hardware performance. To achieve the latter, virtualization of the computational and network resources as well as isolation of the application in this virtualized environment towards other simultaneously running applications have been implemented in the IaaS layer as a means to guarantee QoS. With this the need for expensive hardware has been eliminated.

The improvement in user experience has been significant in terms of quality while keeping up the same performance as could be shown with performance results and images of the running application. However so far we only could evaluate the results in a research environment with industrial spin-offs. As a future task a case study with industrial partners would be beneficial to get clearer impression on the performance improvement in real world scenarios. Additionally the evaluation as mentioned in chapter V should also be performed taking into account varying load on the computing nodes in both test cases to show the benefit of application isolation on the IRMOS framework and thus stress the significance of the framework for performance improvements of real-time applications in Virtual and Augmented Reality. As a future improvement, the integration and adaptation of an application to SOIs could be improved to be more intuitive considering the steps to be undertaken by an application developer in terms of preparing his components with regards to performance estimation and benchmarking. The configuration data that is provided by the PaaS to enable a user client GUI to connect to the components running within the virtualized IRMOS framework could be automatically consumed by an application through a similar approach as it has also been introduced for the ASCs in the virtual machine units as described in chapter IV.A, using a wrapping component in between. Eventually the possibility to have in-application transitions of components from best-effort resources to resources with guaranteed QoS with the consequence of application topology renegotiation would help application consumers to overcome problems in actual running sessions. These activities all together would help establishing collaborative VR and AR in wide consumer-base as they clearly diminish the effort to access such VR and AR applications at very well user experience level.

VII. ACKNOWLEDGEMENT

This work has been supported by the IRMOS project and has been partly funded by the European Commission’s IST activity of the 7th Framework Programme under contract number 214777.

VIII. REFERENCES


