EnvironRC: Integrating mobile communication and collaboration to offshore engineering virtual reality applications

Bernardo Pedras, Alberto Raposo
Departamento de Informática, PUC-Rio
Rio de Janeiro, Brazil
bernspedras@gmail.com, abraposo@tecgraf.puc-rio.br

Ismael Santos
Cenpes, Petrobras
Rio de Janeiro, Brazil
ismaelh@petrobras.com.br

Abstract—**Offshore Engineering visualization applications are, in most cases, very complex and should display a lot of data coming from very computational intensive numerical simulations. To help analyze and better visualize the results, 3D visualization can be used in conjunction with a VR environment. The main idea for this work began as we realized two different demands that engineering applications had when running on VR setups: firstly, a demand for visualization support in the form of better navigation and better data analysis capabilities. Secondly, a demand for collaboration, due to the difficulties of coordinating a team with one member using VR. To meet these demands, we developed a Service Oriented Architecture (SOA) capable of adding external communications to any application. Using the added communications, we built an external collaboration layer. We study the architecture of our solution and how it could be implemented for any application. Furthermore, we study the impact of our solution when running an Offshore Engineering application on VR setups with the support of mobile devices. Such devices can be used to help navigate the virtual world or be used as a second screen, helping visualize and manipulate large sets of data in the form of tables or graphs. As our test application, we used Environ, which is a VR application for visualization of 3D models and simulations**

Keywords--3D Interaction, immersive Environments, Collaboration, mobile communication

I. Introduction

Offshore engineering (OE) visualization applications are, in most cases, very complex and should display a lot of data coming from very computational intensive numerical simulations. To help analyze and better visualize the results, 3D visualization can be used in conjunction with a Virtual Reality (VR) environment. The main idea for this work began as we realized two different demands that engineering applications had when running in VR setups: demand for visualization support in the form of better navigation and better data analysis capabilities and demand for collaboration.

First, since the models being visualized originated from large numerical simulations, we noticed the need for a better user-interface to allow the user to visualize large amounts of data, without disturbing the virtual world representation on the main VR screen. Engineering data are mostly represented by large tables of numbers and graphs, which are not suited for display in VR environments.

Second, we realized that having an immersive VR environment brought real benefits for the review process of large simulations [1]. However, a problem remained; it was hard for the user in the VR setup to communicate and make his/her observations useful. A better way to collaborate and produce useful data from the VR sessions was needed. A further motivation for collaboration comes from the inherited multi-disciplinary aspect of OE. To analyze the results, many different specialists (mostly geographically separated) need to work together on the same model.

Therefore, to solve these problems, we developed a Service Oriented Architecture (SOA) capable of adding external communication and collaboration capabilities to any application. The idea behind our solutions is to enable collaboration-unaware [2] [3] applications to collaborate with as little reworking of the original application as possible. Collaboration-unaware applications are originally developed to be single user applications, but may be used collaboratively by means of an external support system. This external support system may be an application sharing system or a GUI event multiplexing system. In both cases the applications do not explicitly support collaboration; they are implemented as single user applications. This is important since, in our case, the applications developed for OE projects have this characteristic.

In addition, to solve the user-interface problem, we used the added communication mechanism of the application to enable real-time data visualization and manipulation with tablets and smartphones. Such devices can be used to help navigate the virtual world or be used as a second screen, helping visualize and manipulate large sets of data in the form of tables or graphs. Another big advantage that mobile devices bring to the engineering applications is the capability of accessing the data in remote locations, like on oil platforms or refineries, and so allowing the field engineer to check the data or even change it on the fly.

We have developed and tested the proposed architecture with an application called EnvironRC (Environ Remote Control). EnvironRC is an application that adds the benefits of integration with mobile devices, visualization support and collaboration to ENVIRON [4], a VR application for visualization of 3D models and simulations. We opted to demonstrate the benefits of using EnvironRC in VR session with an experiment. Furthermore, we present a real world use of the collaboration aspect of EnvironRC to help offshore engineers review simulation results.

Furthermore, we developed a generic and extensible version of EnvrionRC, called AppRC [5]. AppRC is a framework that can be extended to be used by any other application that wishes to add external communication and collaboration functionalities,

II. Related work

VR visualization technologies enhance the content knowledge within any engineering design activity. Used in conjunction with collaboration, VR visualization provides valuable insights for better Decision Support with risk mitigation. While there are huge benefits of using virtual environments, many problems arise when running complex applications in an engineering virtual environment, especially applications that were not built from the ground up with such environments in mind (as is often the case).

There are many applications that utilize mobile devices to enhance the VR experience. Most of them focus on implementing efficient ways to navigate or to interact with the 3D world [6], [7], [8], [9].

Most implementations of navigation functionalities in VR systems rely on the VRPN (Virtual-Reality Peripheral Network)

[10], which is a device-independent and network-transparent system for accessing virtual reality peripherals in VR applications. VRPN provides a communication standard for developers who want to utilize different sensors as inputs to their application. Although the VRPN framework is widely adopted as the communication channel for navigation input information, it does not provide a good way to handle other kinds of information, like symbolic input, large amount of data or more complex data structures like photos or videos.

There are some proprietary mobile application development systems that integrate a sort of framework to manage the messages from the mobile application to a server application. Flick [11], for example, provides a framework restricted for testing and debugging the server-client communication. The Maximo Integration Framework (MIF) [12] is defined as a framework that provides web services and SOA technologies to support application services and coordination between enterprise systems such as synchronization and integration of data between applications. It is a specific framework that handles data transfers to IBM's Maximo system. Although restricted to the Maximo system, some solutions were developed to integrate the SOA architecture of MIF with mobile devices [13], [14].

We are especially interested in the use of mobile devices to aid the visualization of engineering data and simulations. Many applications and simulators have been developed over the years for traditional desktops and/or clusters. It is still unfeasible to run such simulations or applications in a mobile device as a standalone application because of performance issues, and even if it were possible to run entire simulations on smart devices, it wouldn't be the best way to do it because all the generated data must be stored and centralized in databases or servers, and it wouldn't be practical to transfer such amounts of data for each simulation. So, mobile devices should be used to add their benefits to an existing application or simulator, creating a collaborative environment between the main application (running all the heavy work) and different connected mobile devices. And exactly at this point the main problem we try to address with our proposal comes. How to build a collaborative environment between an existing application and mobile devices with minimum reworking of the existing application?

III. APPRC

As mentioned before, there are two main goals for this project. First, we want to provide a better user interface for users in VR environments. Second, we want them to be able to collaborate with remote users and exchange engineering data in real time.

To achieve these goals, we developed a solution with the following requirements in mind: i) The solution should be generic and extensible to enable its use with other applications with little extra work; ii) The solution should require minimum reworking of the original application; iii) The solution should add very low overhead, i.e., while running the original application with the added benefits of our solution, there should be very little performance impact; and iv) The solution should be portable.

The only requirement we set on the user-application (in our case, Environ), is that the application should have some way to receive commands from an outside source. We left open what exactly this command-passing mechanism is, since it depends heavily on many technical aspects of the application, such as, programing language, platform and internal architecture.

A. *Architecture Overview*

In this section, we present the architecture overview of the generic AppRC solution. EnvironRC, the specific implementation of AppRC for the Environ application follows the same architecture and the specific details of its implementation will be discussed later.

The goal of our architecture is to integrate external communications and enable collaboration in a collaboration-unaware application. Therefore, our architecture is resolved around supporting many instances of the application running on multiple machines and devices.

Our solution is divided in three main components: AppRC Server, AppRC Client, and Mobile Application (Figure 1).

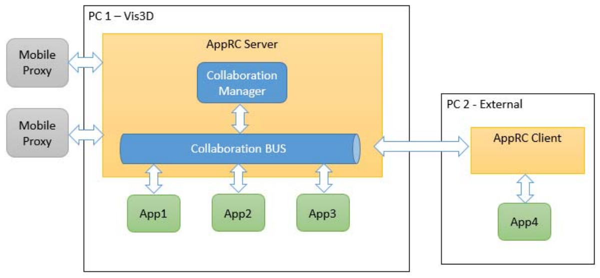


Figure 1: AppRC Architecture Overview.

AppRC Server works as the central HUB of all communications. All users are required to communicate through the Server. It also manages all collaboration aspects including users and permissions. The server receives (via the Collaboration BUS) all messages and routes it to the corresponding receivers.

AppRC Client works as a proxy of the main server running on a different machine. It only passes the messages along to the main server. For the Collaboration BUS, there is no distinction between an instance of the application running on a same machine or on a remote device.

Each application instance is both an Inbound and Outbound Endpoint, capable of receiving and sending messages. The Mobile Proxy works almost the same as an instance of the application, with the important distinction of being exclusively an Inbound Endpoint. The reason for this decision is so that the mobile application can only request information it needs, and does not have to always listen to incoming messages.

B. *Use Scenarios*

We studied two main use-scenarios to demonstrate the benefits of our solution. We chose these scenarios as they represent real world situations and reflect the benefits of adding our solution to engineering applications running in VR setups. In each scenario, we identify the main problem the application has when running in VR and how our solution should solve it.

- Scenario 1 – Single Mobile / Multiple instances
- Scenario 2 – Multiple Mobile / Remote users

1) *Single Mobile / Multiple instances*

We built this scenario around the idea of having a Powerwall setup. Powerwalls are typically very large displays with massive resolution, and so, capable of displaying many contents at the same time. As most applications were not built with such large resolutions in mind, the most effective way to use the benefits of a Powerwall visualization setup is to open many instances of the application at the same time. Each instance using a part of the full Powerwall screen enabling the user to view all instances simultaneous side by side and compare and analyze the results. Another instance of this use-scenario in a VR environment is when the user has multiple VR applications running in background, each of them with a different visualization model. This typically happens in CAVE setups, as each model has to be loaded and properly setup for visualization, a multi-model visualization, where the user is able to change between models by running multiple instances of the application in background.

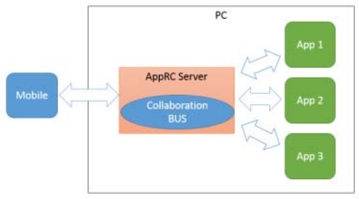


Figure 2: Single Mobile / Multiple instances

Many instances of the application running simultaneous on the same computer bring new challenges. There is now a need to coordinate the different instances of the application on top of managing the collaboration and messages between them. It is also important for the user to be able to broadcast messages to all instances of the application.

In Figure 2 we can see how this scenario is represented using our architecture. Here we need the AppRC Server to be running with the collaboration BUS to manage the communication and route the messages to the corresponding application instances.

2) Multiple Mobile / Remote users

This scenario represents the most complete use of our solution. It encapsulates many different uses of the collaboration infrastructure (Figure 3). It represents for instance the use of the application for a presentation setting, where there are many models being presented in the main screen and each viewer of the presentation has a mobile device and can request on-demand specific details of the main model.

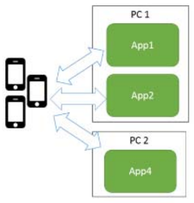


Figure 3: Multiple mobile / remote users

Another use-scenario is during a review process of a model. Sometimes not all the participants are co-located for the presentation. This is especially interesting for offshore engineering applications due to its inherited multi-disciplinary aspect. To analyze the results, many different engineers (mostly geographically separated) need to work together in the same model. The main review process occurs in the main visualization environments while remote users can join in and collaborate with the working session.

Here we need the remote machine to run the AppRC Client and connect it to the running session of the AppRC Server. The connection of AppRC Client (running on a remote machine) and AppRC Server (running on the main visualization machine) is done through a direct socket connection. The remote user running AppRC Client doesn’t affect the collaboration session. For all collaboration participants, all other participants are accessible through the same means (collaboration BUS).

3) Implementation

AppRC was developed in Java as a Netbeans Platform application using Mule ESB [16]. It is designed to have a simple user interface for basic monitoring and controlling different communication channels.

Another important aspect of AppRC is its support for managing multiple instances of the application running at the same time. This functionality is very important when running an application in very large displays for example, as it allows the user to manipulate from a single source, multiple instances of the application, providing an efficient way to compare simulation results or data. To achieve this goal, we adopted the concept of application instances session. Each session can run a single instance of the application and the user can create and manage multiple sessions.

The internal architecture of our solution uses the concept of SOA in the form of Mule Enterprise Service Bus (ESB) implementation. We use the benefits of SOA to build a communication bus inside our framework so that the input services can exchange message with the base application service.

AppRC supports four different communication channels (file, socket, WebService and mobile service). Using Mule ESB implementation, we defined two Mule flows. The first one, called App Flow, handles mobile devices incoming messages. The second one, called Mobile flow, handles all other incoming messages (file, socket and WebService) – Figure 4.

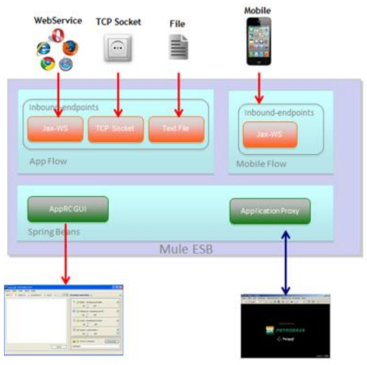


Figure 4: AppRC with Mule ESB

Therefore, with Mule ESB, we were able to implement the four command input channels. The simplest way to exchange messages between two applications is via *text files*. Therefore, we implemented this kind of interface in our solution, even if it is not the most efficient way to exchange message, since both applications have to be running on the same machine or have access to a shared file system. We implemented the *socket input* as a basic TCP socket where the server (AppRC) waits the commands as serialized text bytes on a specific port and passes the message over to the application. We opted for adding the socket communication channel because of the wild spread use of sockets and because of its simplicity. We also use the reference Jersey RESTful (*JAX-RS*) implementation coupled with the Mule ESB framework. The Mule ESB handles the incoming messages through the web service in the form of flows.

The *mobile channel* was implemented the same way as the normal web service with some adjustments to account for some general mobile connectivity functionalities and the use of Mobile Application Service extension point.

In order to keep AppRC generic and extensible, we decided to simplify the work needed to integrate AppRC to a new application and defined only two extension points (Figure 5) that need to be addressed by the application developer.

On top of the four input channels, an adapter had to be implemented to handle the communication between the ESB (AppRC) and the application (Application Proxy). The problem is that each application may export its functionality in a different way. Environ, for example, supports LUA commands via a local socket connection. However, other applications may use direct command line calls, text file input or any kind of custom method of receiving commands. Moreover, since these methods may already be implemented in the application, we decided not to arbitrary force the use of any method as to avoid having to change any code in the application itself. Instead, we leave the implementation of an Application Proxy class as an extension point of AppRC, responsible for sending the message received and processed by AppRC to the application.

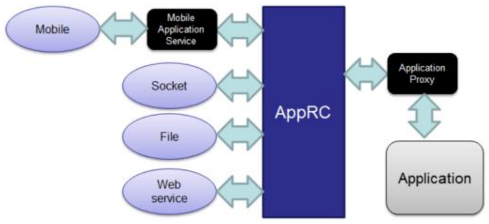
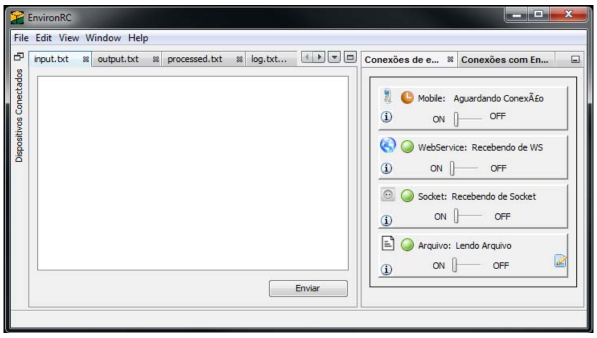


Figure 5: Extension points overview

Another extension point we decided to include is the Mobile Application Service. While developing a mobile application, it is useful to have a middle layer of abstraction between the responses from the application and the mobile application. In this layer, from a single mobile command, the developer could, for instance, request many commands from the application organize the data and then send it back to the mobile app in a way that is easy to process and display. By adding this extension point, we are creating a business logic layer without modifying the original application and keeping the mobile app development simple. If, in the other hand, the application already has a complete business logic or the mobile app already implements a complete set of commands, this extension point can be left out and no additional work is needed to integrate the mobile functionality.



Mobile: Aguardando ConexÃ£o
WebService: Recebendo de WS
Socket: Recebendo de Socket
Arquivo: Lendo Arquivo

Figure 6: Basic AppRC GUI

4) GUI

In Figure 6 we present the basic interface. The left side shows the connected mobile devices, in the middle section we can see the contents of the four basic files that AppRC writes and reads. It is possible for the user to write commands directly to input.txt file and send it to the application. The right side shows the status of each input channel and, in a second tab, the status of the connected application instances. It is also able to run in the background as not to disturb the main visualization.

IV. ENVIRONRC

EnvironRC is our specific use of the AppRC generic solution for the Environ offshore engineering visualization application. Environ [1] is a tool designed to allow visualization of massive CAD models and engineering simulations both in desktop and VR immersive environments. It is a system composed of a 3D environment for real-time visualization and plug-ins to import models from other applications, allowing users to view and interact with different types of 3D data, such as refineries, oil platforms, risers, pipelines and terrain data. It enables the user to view the simulation in real-time as a 3D environment and enables, at the same time, the user to view all the simulation data and engineering information.

Environ was chosen as our test application because it is both a virtual environment application and a complete offshore engineering application, able to manipulate large amounts of simulation data. Environ provides a full script interface using the LUA language. With this script language, it is possible to execute many complex operations inside Environ programmatically. At the same time, it provides a perfect way to send commands from an external application to Environ.

EnvironRC implements both extension points discussed in the previous section.

To help the development process of the mobile App for Environ, we created a custom Mobile Application Service to handle high-level commands coming from the mobile device. The mobile service is also able to do some logic depending on the command. A good example of the benefits of using this extension point is the handling of active selection of model objects in Environ. We display in the mobile device interface a list of selected objects. With many users using the system at once, (many mobile devices at the same time for instance) it can be a problem to maintain the current selected objects list updated in all devices. So, we implemented as part of the Extension Point, a layer of business logic that is in charge of keeping the selection consistent across all connections. When there is a request from a Mobile Device to get the selected elements, the custom mobile service will not pass the command direct to the application, instead it process the command internally and responds accordingly.

The implementation of communication between EnvironRC and Environ was done using a local socket connection. We had to implement the framework Extension Point to send the LUA commands to Environ using the socket structure. The implementation of the socket connection was done in a separate thread that keeps consuming the messages from environ and passing it along to the AppRC collaboration structure. The thread also consumes a message queue end sends the new messages to Environ.

A. EnvironMobile

EnvironMobile is an IOS application developed mainly to help the visualization process of complex offshore engineering models in VR environments. It was developed with the Objective C language as a hybrid application (IPad and IPhone). It works as an extension of the main application, enabling the user to setup parameters and view details of the main model being visualized. EnvionMobile connects to environ using the Mobile connection input channel of EnvironRC.

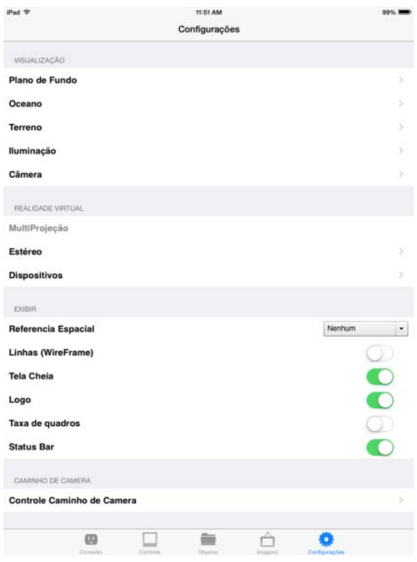
We defined and implemented seven base functionalities for EnvironMobile:

- Configure visualization parameters
- Navigate the model structure
- Help navigate the virtual world
- View and manipulate engineering data
- Create and delete annotations
- Take snapshots of the scene
- Manipulate the collaboration sessions

In the next subsections, we will discuss the implementation of these functionalities in further detail.

1) Visualization parameters

Being a complex application as it is, Environ can have many parameters to setup depending on the kind of visualization the user wants to use. These parameters range from simple visualization tools like showing the ocean or terrain information, to complex values tied in to the simulation, like forces applied to risers or sea current values. As the user will probably need to change the values and inspect different information during the visualization, it was very important that these parameters be made available to the mobile device in his or her hands. Therefore, our first priority was to implement a complete set of commands for visualization parameters and the interface to manage them. (Figure 7 shows a sample of the Mobile interface). We followed the standard settings interface of the IOS platform.


iPad 11:51 AM 99%

Configurações

VISUALIZAÇÃO

Plano de Fundo

Oceano

Terreno

Iluminação

Câmera

REALIDADE VIRTUAL

MultiProjeção

Estéreo

Dispositivos

EXIBIR

Referencia Espacial: Nenhum

Linhas (WireFrame) ○

Tela Cheia ●

Logo ●

Taxa de quadros ○

Status Bar ●

CAMINHO DE CAMERA

Controle Caminho de Camera

Figure 7: Environ mobile visualization settings

The visualization parameters of Environ include settings of the 3D scene, like skybox, ocean and terrain settings. We can also set the camera and light settings of the visualization. Another important set of visualization parameters are the VR settings. These include stereoscopic 3D, multi-projection and devices (like fly sticks) settings.

2) Navigate the model structure

The visualization scene of Environ is structured in a hierarchical way, because for very complex scenes and simulations it becomes harder to select a specific object in the 3D world. So, having a tree like object hierarchical structure can help to select and check more detailed information of a given object. However, bringing up the objects tree in a context menu during a session in an immersive environment like a CAVE can be distracting and disturb the immersion. Thinking about that, we implemented a way for the user to view the complete objects tree with the tablet and to access the details, navigate to or even manipulate the data of any object.

In Figure 8, we can see how the interface works. We present the user with a list of all elements of the current hierarchical level. The user can then tap the arrow on the far right side of the element to navigate to its children or he/she can tap the element to access some commands to be performed on that element, like: navigate to the element, turn the element visualization to invisible, clear the elements selection among other commands.

3) Navigate the virtual world

For the Environ application, only simple navigation methods were implemented. The method mimics the use of navigation with the keyboard (Figure 9). Although it might be a very simplistic implementation of 3D world navigation, it can be of much help to be able to come closer or move away from an object using the mobile device. Environ has two types of 3D world navigation paradigms; “Examine” and “Fly”. Using the Examine mode, the user camera has a set point of view center point, and he/she moves around this center point. It is very useful when analyzing a specific point in the model. The other mode is Fly, in which the controls work as if the user was flying through the scene.

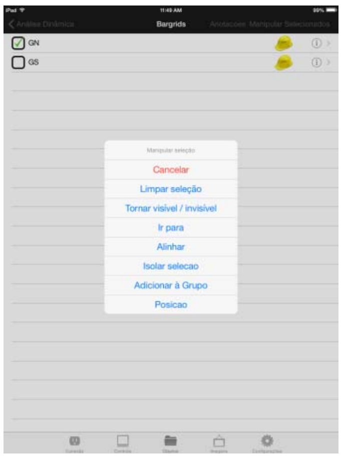


Figure 8: Manipulate the model structure

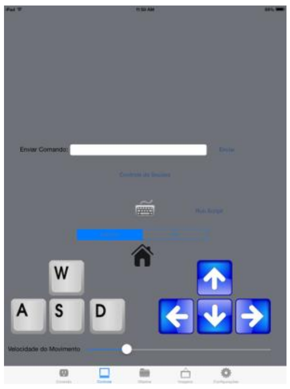


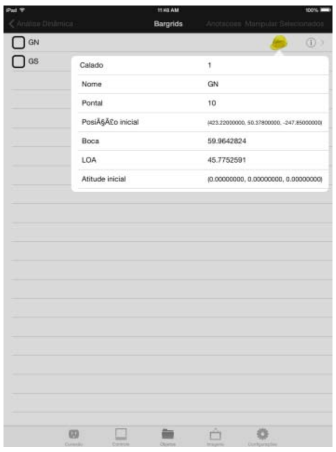
Figure 9: Virtual world navigation

Our communication and collaboration solution supports any kind of navigation, including more complex implementations using the internal sensors of the mobile device like accelerometers and gyroscopes. However, it fell out of the scope of our work and in our experience with EnvironRC in VR Setups, the use of the simple provided controls in conjunction with the ability to select any structure from the model and directly navigate to it, although not optimal, proved very effective in helping the visualization.

4) View and manipulate engineering data

When inside the VR environment, any kind of menu or big text block viewed in the application disturbs the immersion. A mobile device offers a second screen where we could display many kinds of data like: large tables of numbers, graphs, detailed information, etc.

We implemented a way for the user to request engineering data detail of any object of the scene (Figure 10). This feature proved the most useful when using the mobile device in VR setups, as it helps the review process of an offshore simulation when the user can see multiple data tables of the specific simulation timestamp being presented in the main visualization screen.



Calado	1
Nome	GN
Pontal	10
PosiÃ§Ã£o inicial	(423.22000000, 50.37800000, -247.85000000)
Boca	59.9642824
LOA	45.7752591
Atitude inicial	(0.00000000, 0.00000000, 0.00000000)

Figure 10: Manipulate engineering data

5) *Manipulate Annotations*

Virtual 3D annotations are a very useful feature of Environ. They help the user understand details of the virtual model since each annotation is always coupled to a specific part of the model being visualized. All annotations can be seen in the mobile device as a list. This is especially useful when the user needs to setup a list of tasks to be done in a remote location. For example, a user using the desktop application can create a sequence of annotations, each one representing a task that needs to be done in a specific part of the model. A second user in a remote location (an oil platform for example) connected to the same session, can see the annotations and their locations and perform the tasks. He/she can even create new annotations or edit those already present.

6) *Take Snapshots*

A useful feature is the ability to take snapshots of the current simulation. It is a useful way to collaborate with remote users, since they can receive images of the simulation model while on the field. This can help the field engineer using the mobile device identify and compare the simulation with the real world engineering hardware.

A more complete implementation of this feature would include not only the ability to take snapshots, but also the ability to stream as video the content of the visualization directly to the mobile device in real-time. However, to implement this complete feature would require a lot of extra work and was left outside the scope of the current work.

7) *Manipulate the Collaboration Session*

In order to manage the collaboration aspect of EnvironRC, we implemented a special screen to manipulate the collaboration session. Here the user can see the available applications to send commands to, the number of other users in the session, create new instances of the application, start broadcasting a message or remove an application from the session.

V. Results

In this section, we discuss two main results of EnvironRC. First, we describe the experiment that was conducted with users to determine the benefits of having a mobile device while using a VR setup. Second, we present a real-world case of EnvironRC.

A. *Controlled Experiment*

To test the VR setup support, we decided to run an experiment with users to help determine the impact on the immersion experience while using a tablet connected to the main visualization system.

We chose to perform an experiment because of the nature of the problem we were trying to analyze. The user experience when using a CAVE is subject to many different aspects, such as, time needed to complete tasks, usability, user experience, immersion perception, among others. By running a controlled experiment, we will try to analyze qualitative and quantitative aspects of this experience.

The objective of our experiment was both descriptive and explanatory. We wanted to describe how the users interacted with the system and what their thoughts were. At the same time, we wanted to examine if the use of a tablet device in VR setups had positive impact in the VR session, both in terms of time needed to complete a task and user experience.

We recruited participants for the experiment that had familiarity with computers, had experience with complex three dimensional applications such as games and 3D modeling software and had experience using VR setups. The decision to recruit only users with experience in VR setups was made to avoid large distinctions in execution time. VR setups can be very confusing and take a large amount of time for a new user to get used to. So, by having only experienced users, we can avoid problems with training bias.

The experiment consisted of the users performing two tasks in a CAVE VR setup. Both tasks were performed first with Environ (without EnvironRC) and afterwards with EnvironRC. To ensure that all users had the same experience with both EnvironRC and Environ, we conducted a training session before the experiment. In the training session, the users had unlimited time to try both applications and ask questions.

To collect the data, we used different techniques: Pre-test question form; Post-test question form; Time to perform each task; and Semi-structured interview. The pre-test form was used mainly to select the participants' profile. Post-test question form and the time to perform each task were used as a quantitative measurement of each user performance and experience. The semi-structured interview was designed to collect qualitative information about the use of the system and to help describe the experience in further detail.

The goal of our experiment was to analyze and determine the benefits (if any) for the user experience of running Environ in a VR session with and without our solution. For the experiment, the users had to perform a series of tasks during a VR session in two different conditions: first without our solution and then using the mobile device connect to the Environ application through EnvironRC. To complete the tasks without EnvironRC, the participants were asked to use standard mouse and keyboard inputs. This decision was made because although there are many different devices to help navigate the 3D world, none of them can help the user manipulate engineering data or use 2D menus to configure the application. On top of that, most CAVEs setup today use keyboard and mouse to perform these tasks, generally by an user outside the immersive environment.

1) *Experiment Preparation*

We recruited fourteen participants, with ages between 26 and 45. All of them had experience with VR setups. Only 2 of the participants had not used any kind of engineering applications in VR, but had experience with 3D software like modelling tools and games. All participants had at least a graduation degree in the field of computer science and 78% had a postgraduate degree in the field. There were thirteen males and one female participant.

To test our solution, we used a visualization system of type CAVE. A CAVE has the problems we wanted to address with this dissertation. Some of these problems include: Users have to be standing and to use 3D glasses; there are multiple projection screens, making menu navigation hard, and large projection screens, making texts harder to read.

One of the main problems engineering applications face when running in VR setups is the difficulty to represent any kind of 2D menu navigation. This happens specially when trying to read large amounts of data (as numbers or tables) or when trying to navigate 2D menus to change any visualization parameters.

To study the impact of our solution when trying to solve these problems, we defined two different tasks for the participants to perform. One of them to study the impact when reading large amounts of data and the second one was designed to study the impact of changing visualization parameters. Another important distinction between the two tasks is the fact that the first task requires the participant only to read the values he/she sees in the virtual world, while the second task requires the user to input numerical values as parameters.

Task 1 – Reading engineering data

The goal of task 1 is to test the impact of having a mobile device when trying to read engineering data values from a specific object. The participant starts with a full visualization of the offshore engineering model. The participant is asked to use the object navigation interface of Environ to select a specific object in the scene. Once selected, he/she should invoke the "go to" command to go to the selected object and visualize it. After that, the participant is instructed to read out loud the values of three engineering parameters of the selected object.

Task 2 – Changing visualization parameters

Changing visualization parameters is a very common task when using VR setups. Visualization parameters include: graphic quality parameters, immersive system parameters, stereoscopic 3D parameters and others. The goal of this task is to change three visualization parameters:

1. Change the skybox from "color: black" to "sky: twilight".
2. Change the stereoscopic eye distance from 0,0175 to 0,02.
3. Change the distance to parallax zero from 1,0 to 0,9.

2) Experiment Execution

A pilot test of the experiment was conducted to validate the experiment material, documents and process. Based on the results of the pilot test, we decided to make a single adjustment to the experiment. We decided to add a flat surface near the CAVE setup with a mouse and keyboard. Before the pilot test, we had decided that the participants would try to use the mouse and keyboard while standing, but in most VR setups there is often a support table nearby where the user can go to use mouse and keyboard if needed. Therefore, we decided to leave the table near the CAVE and the participant was free to use it to support the mouse and keyboard if he/she so wanted.

The experiment was divided in seven steps:

1. Introduction to the study and pre-test question form
2. Reading of the short Environ and Environ Mobile manuals
3. Training session with Environ and Environ Mobile on Desktop
4. Reading of the Task description documents
5. Test execution
6. Post-test question form
7. Semi-structured interview

Before the experiment, the participant signed a consent form which explains the purpose of the tests and guarantees the reliability of the collected data. After that, the participants were asked to fill in the pre-test question form to determine their profile and their experience with both VR setups and engineering applications.

After the pre-test form, users were given the short manual of Environ and EnvironMobile. This document was designed to teach the user how to use the applications to complete the tasks. After reading the manual, the user was given free access to a desktop computer running Environ and a tablet running EnvironMobile. At this stage, the user was free to experiment with both applications and to ask questions about how they worked. There was no time limit for the training stage.

After the training stage, the user was asked to complete each task twice, once using Environ and once using EnvironMobile inside the CAVE setup. There, the user had to wear stereoscopic 3D glasses and stand up inside the CAVE. He/she was first asked to complete both tasks without EnvironMobile. To do so, the participant could use mouse and keyboard. After that, he/she was asked to complete the same tasks using EnvironMobile. A measurement of time spent in each task with each device was taken.

Finally, after the execution of the tests, the participant was asked to fill in the post-test question form. This form was used as a tool to determine the user experience in the CAVE. Some aspects we wanted to study with the post-test form were: usability, how (or if) both solutions disturbed the immersion, impact of 3D glasses while using a tablet and others. The question form consists of eleven statements. We used a Likert scale for each statement where 1 represents full disagreement and 5 represented complete agreement.

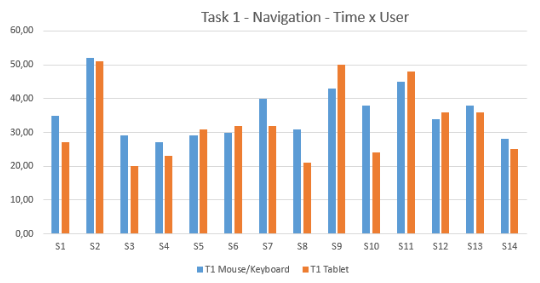
At the end, a semi-structured interview was conducted in order to determine how the participant felt about both ways of completing the tasks. Furthermore, we used the semi-structured interview to clarify the answers given by the participant in the post-test form and ask for improvement suggestions.

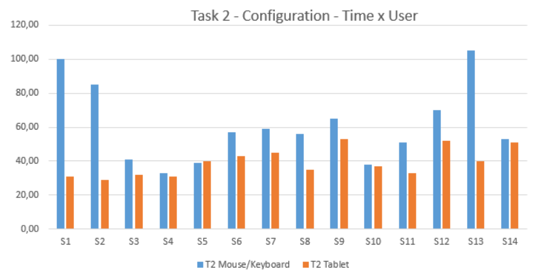
3) Results Analysis

In this section, we present the main observations done while running the experiment, as well as difficulties and suggestions the participants had. First we analyze the time needed by each participant to complete each task. The results can be seen in Figure 11. As most users had experience with VR setups and engineering applications, we could not observe any significant improvement in the time needed to perform task 1. On the other hand, we can observe that most users performed task 2 in significant more time when using mouse and keyboard. The different in results from task 1 and task 2 is most likely explained duo to the fact that task 2 required the user to input numerical values using the keyboard.

To further analyze and understand the results of the experiment, we show the result of the post-test question form in Figure 12. We can see that the users thought the task was easier to complete using EnvironMobile in comparison with standard input devices. (Q1 – Q4). During the semi-structured interview, most users reported that using mouse and keyboard in a CAVE was very unpractical and because of that, using a CAVE setup alone was very hard, often requiring a second person to help navigate the menus. These users thought that having a tablet made it practical to complete the tasks in the CAVE alone.

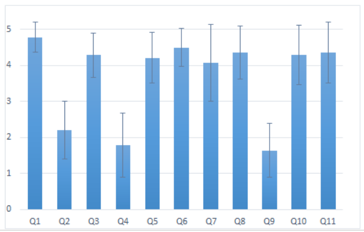
A very common complaint during the interviews was that the size of the text on the projection screen was very hard to read. There are two main reasons for that. First, the projection screen is very large and has a high resolution making the text very small. Second, most stereoscopic 3D glasses work using some kind of color filtering mechanism that makes the image darker, and therefore harder to read. To further analyze the impact of 3D glasses, we can see that most users thought the 3D glasses had a small impact when using the tablet screen (Q9). One user did complain that the colors on the tablet screen were not very clear using the 3D glasses, but he mentioned it was of low impact for his experience.





	Avarage
T1 Mouse/Keyboard	35,64
T1 Tablet	32,57
T2 Mouse/Keyboard	60,86
T2 Tablet	39,43

Figure 11: Time to complete tasks



Q1	It was easy to navigate (T1) to the target object using mouse and keyboard on Desktop
Q2	It was easy to navigate (T1) to the target object using mouse and keyboard on the CAVE setup
Q3	It was easy to navigate (T1) to the target object using tablet device on the CAVE setup
Q4	It was east to alter the configuration values (T2) using mouse and keyboard on the CAVE setup
Q5	It was easy to alter the configuration values (T2) using tablet device on the CAVE setup
Q6	Having a tablet device had a positive effect on my experience in imerssive VR setups
Q7	I learned to use the tablet quickly
Q8	I felt confortable using the tablet
Q9	The 3D glasses made it more dificult to use the tablet device
Q10	I am satisfied with the time it took me to complete task 1 in the CAVE
Q11	I am satisfied with the time it took me to complete task 2 in the CAVE

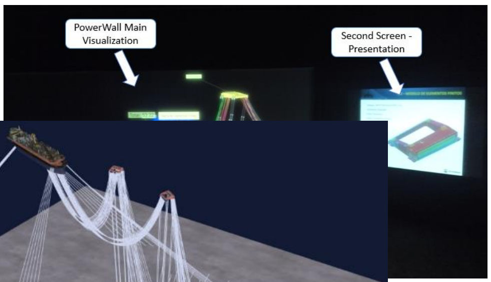
Figure 12: Post-test results

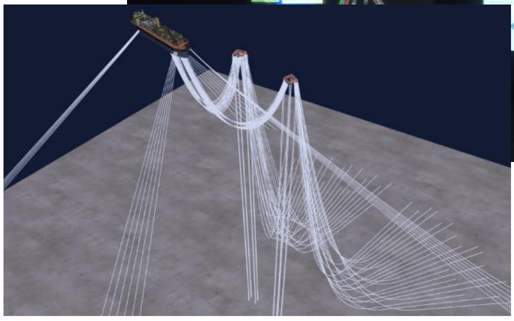
Some users mentioned that the EnvironMobile interface could be improved. One user suggested that the EnvironMobile interface should try to mirror the main Environ application perfectly. Furthermore, when asked about the impact the tablet had on the immersion, almost all users said that having to look away from the main projection screens to the 2D interface of the tablet reduced the immersion, but was of a much lesser impact than having to use mouse and keyboard. The results of Q6 further accentuate this difference, with most users agreeing that the tablet did have a positive impact on the immersion experience.

From the interviews, it became clear that although some users could complete the task in similar times with or without our solution, the use of keyboard and mouse did never offer a good immersion experience.

B. *Case Study*

In this section, we will discuss a special use of EnvironRC during the review process of a very large offshore engineering





project involving a BSR system (Riser sustaining buoy) for deep see oil exploration. The main goal of the buoy is to alleviate the stress on the risers induced by the movement of the platform. Without the BSR all the risers would go from see floor all the way to the platform, and they should be resistant enough to support all the forces coming from the movement of the platform. The platform suffers movement duo to the external forces applied to it (current, wave, winds etc.). The BSR works as a way to separate the movement of the platform from the long risers' lines – Figure 13.

Figure 13: Two BSR overview

The complete BSR Project took many years of study and involved many areas of the offshore engineer field, making it a very good example for our solution. There was a huge amount of simulations done for every step of the project. These simulations where divided in two main phases: installation phase and operational phase. There were more than a thousand simulation cases for each of the phases, involving every kind of failure or condition.

The simulations are a very complex numerical computational problem and have to run offline. Each simulation can be visualized by Environ in VR setups of the company that developed the project. The visualization in VR helped offshore engineers during the review process and these visualizations of big offshore engineering projects were the main motivation of this work. One special case we would like to study in more details is the review process of the BSR project that involved a large amount of simulations and a large amount of engineers.

Because of the number of people present in the day of the review, all the models were to be presented using a PowerWall setup, with a supporting screen for presentation (Figure 14). All offshore engineers were to present their results and show the simulated models that had interesting details and facts that could indicate some problems for the overall project.

Figure 14: VR setup at the company that developed the project: a PowerWall and an auxiliary screen.

The presenter (mostly offshore engineers) had a PowerPoint presentation that would run on the second screen and the simulation results would be presented at the main PowerWall screen. The presenter could send commands and control the simulation via the PowerPoint presentation as well as with an IPad running EnvironMobile. A complete schematic of how our solution was used to achieve this kind of integration and collaboration can be seen in Figure 15.

For the review process of the BSR project, seven simulation results were chosen to be presented. Since each simulation takes a very long time to load using the visualization setup, we chose to have all seven simulations loaded using different instances of Environ and have EnvironRC managing them. In total, there were eight instances of Environ connected to the EnvironRC session, seven for each model and one with a general overview model used only to explain the idea behind BSR.

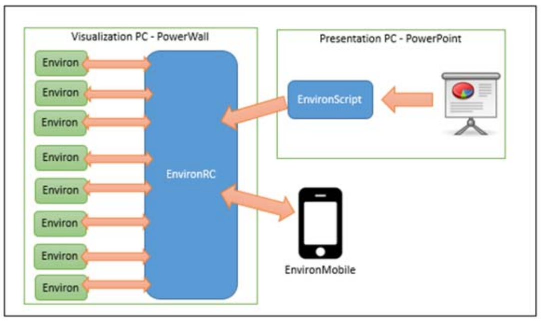


Figure 15: EnvironRC in the BSR Review process

The presentation PC had all the PowerPoints the presenters were going to use. Each PowerPoint was previously setup to be able to invoke EnvironScript to send a command to Environ. On top of the integration with the PowerPoint PC, we had a mobile device assisting the entire process, as seen in Figure 16. With the mobile device, the presenter could, in real time, change the visualization and navigate to a different spot in the scene. He could also check engineering data of the model to better answer the questions of the viewers. Using the already discussed functionalities of EnvironMobile, the presenter was able to control and communicate with all eight instances of the application at the same time.

The PC running on the main visualization screen was a custom build state of the art machine running a 32 core CPU with two dedicated external NVDIA Quadro Plex graphic cards. The presentation PC was a standard laptop since it was used only to run PowerPoint and the mobile device used was an IPad 2. All devices were connected to the same network via WiFi and we achieved real time manipulation and collaboration.

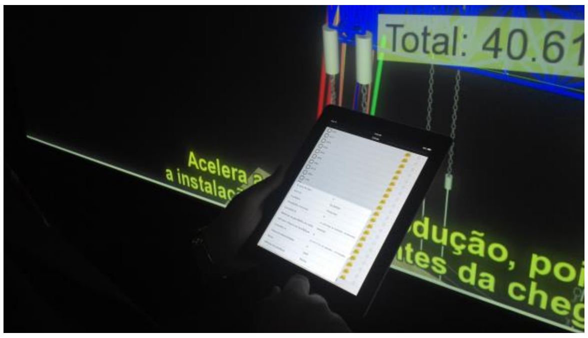


Figure 16: Mobile in BSR review

The review process of BSR was a real world, heavy load test case for our solution and we were able to handle all the communication in a real time scenario and still add all the benefits of having a collaborating system in an offshore engineering project review in a VR environment.

The average response time of a request from the mobile device to the main visualization application was of 0.67 seconds. Nielsen [16] defined three main time limits for applications responses. Our result stay within the second limit category where the users has the impression that the computer is working on the request but can still freely navigate the interface.

We performed tests of Environ running with and without EnvironRC connected, but could not find any significant performance difference. This probably happens because of the nature of Environ (and engineering applications), the load of the visualization and simulation part of Environ is much greater than the load of the communications between applications, making the performance difference negligible.

VI. CONCLUSION

Our goal with this work was to study the use of EnvironRC to help offshore engineering applications run in VR environments. On top of that we wanted to enable collaboration unaware application to collaborate and exchange messages. As the result of this work, we could demonstrate that EnvironRC brought real benefits to offshore visualization applications running in VR setups. The main benefits were that having a connected tablet device when inside a CAVE setup enabled the users to manipulate engineering data and configure the visualization parameters. Furthermore, we showed that using a tablet had a lesser negative impact to the immersion experience then traditional input devices.

EnvironRC is in use as a way to enhance the experience of Environ in many different kinds of virtual reality setups like a CAVE and an L-shape display. EnvironRC is also in use with an Ultra-HD display (20x HD) that, because of the very high resolution, enables the use of multiple instances of Environ on the same display with many users at the same time. Furthermore, it enables the user of Environ to navigate the 3D world, review engineering data, create 3D annotations or even take snapshot pictures of the simulations from a mobile device.

Furthermore, AppRC was designed to be lightweight, extensible, as well as flexible enough to accommodate the different kinds of needs of any kind of application. AppRC requires little development work in order to achieve the integration. By implementing only two basic classes, and with minimum change to the base application, it is possible for any application to export its functionalities, not only to a mobile device, but also to any other application using one of the four available input channels.

We had success running EnvironRC in a collaborative setup, but further studies are still needed. When adding collaboration capabilities to an application, we add a large amount of new ways to interact with the application and other users. A possible future work is to conduct a more complete study with different collaboration sessions and how much each session type benefits the users.

Further studies not included this work involve the use of our solution to integrate applications outside the field of offshore engineering. The benefits of having collaborating applications in VR Setups could be studied other kinds of applications outside the scope of engineering. One field that could possibly be of interest is medical applications. There are already many 3D visualization applications in the field of medicine that could use our solution to run different visualizations to compare data

References

[1] I. H. F. Santos; A. B. Raposo and M. Gattass. A software architecture for an engineering collaborative problem solving environment. In *32nd Annual IEEE Software Engineering Workshop*. IEEE, 2008. pages 43-51

[2] W. Reinhard et al. CSCW Tools: Concepts and Architectures. *IEEE Computer*, 27(5): 28-36, 1994.

[3] I. H. F. Santos; A. B. Raposo and M. Gattass. Finding Solutions for Effective Collaboration in a Heterogeneous Industrial Scenario. In *7th International Conference on Computer Supported Cooperative Work in Design - CSCWD 2002*, pages74-79.

[4] A. B. Raposo et al. Environ: Integrating VR and CAD in Engineering Projects. *IEEE Computer Graphics & Applications*, 29(6):91-95, 2009.

[5] B. F. V. Pedras, A. B. Raposo; I. H. F. Santos. AppRC: A framework for integrating mobile communication to virtual reality applications. In *Proceedings of the 12th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry*. ACM, 2013, pages 305-308.

[6] D. Medeiros et al. An interaction tool for immersive environments using mobile devices. In *XV Symposium on Virtual and Augmented Reality (SVR),* IEEE, 2013, pages 90-96.

[7] H. Noronha et al. Designing a mobile collaborative system for navigating and reviewing oil industry cad models. In *Proceedings of the 7th Nordic Conference on Human-Computer Interaction: Making Sense through Design-Industrial Experiences*, NordChi, 2012.

[8] N. Katzakis; M. Hori; K Kiyokawa and H. Takemura. Smartphone game controller. In *Proceedings of the 74th HIS SigVR Workshop*. 2011.

[9] M. P. Guimarães, B. Gnecco and M. K. Zuffo. Graphical interaction devices for distributed virtual reality systems. In *Proceedings of the 2004 ACM SIGGRAPH international conference on Virtual Reality continuum and its applications in industry*, ACM, pages 363–367.

[10] R. M. Taylor et al. VRPN: a device-independent, network-transparent VR peripheral system. In *Proceedings of the ACM symposium on Virtual reality software and technology*, ACM, 2001 pages 55-61.

[11] Flick Software, 2015. http://flicksoftware.com/flick-technologies/flick-development-libraries/.

[12] Maximo, 2015. http://www-03.ibm.com/software/products/us/em/maximoassetmanagement/

[13] Ezmaxmobile, 2015. http://ezmaxmobile.interprosoft.com/.

[14] Imaxeam, 2015. http://imaxeam.com/maxinterface-integration-tool.

[15] MULE ESB Framework, 2015. http://www.mulesoft.org/documentation/display/current/Home.

[16] Jakob Nielsen. Usability engineering. Elsevier, 1994.