Int J Interact Des Manuf (2017) 11:569–580
DOI 10.1007/s12008-016-0311-x

ORIGINAL PAPER

# Towards a methodology to build virtual reality manufacturing systems based on free open software technologies

Isidro Calvo[1] · Fernando López[1] · Ekaitz Zulueta[1] · Pablo González-Nalda[2]

Received: 23 March 2016 / Accepted: 24 March 2016 / Published online: 23 April 2016

**Abstract** Virtual reality and augmented reality are being increasingly used in the manufacturing field for different purposes. As a result, these technologies seem to have a very promising future in the manufacturing industry. However, most of the current applications are based on the use of expensive proprietary software packages (such as specific CAD modules). This work presents a methodology that helps programmers to build virtual and augmented reality systems, which is valid for a broad number of industrial plants. Differently to other approaches, ours is based on open technologies, most of them free of cost. The technologies proposed to build the virtual worlds are already mature at the entertainment industry. The presented approach opens new possibilities for creating virtual reality models aimed at two major directions: (1) *pure simulation* of real processes and products for different purposes (e.g. staff training; prototype designing; manufacturing optimization; marketing) and (2) *teleoperation* of real processes (e.g. in dangerous environments or in micro and macro scale manufacturing). The application of the methodology is illustrated by means of the creation of a non-immersive virtual world which represents a part-classifying station, allowing both the simulation of the real plant as well as its remote operation.

**Keywords** Virtual reality for engineering · Virtual engineering · Industrial automation · Simulators for design and manufacturing · Teleoperation · Serious games

✉ Isidro Calvo
isidro.calvo@ehu.eus

Ekaitz Zulueta
ekaitz.zulueta@ehu.eus

Pablo González-Nalda
pablo.gonzalez@ehu.eus

1 Systems Engineering and Automatic Control Dept, University of the Basque Country (UPV/EHU), U. College of Engineering, C/Nieves Cano, 12, Vitoria-Gasteiz, Spain

2 Computer Languages and Systems Dept, University of the Basque Country (UPV/EHU), U. College of Engineering, C/Nieves Cano, 12, Vitoria-Gasteiz, Spain

## 1 Introduction

Virtual reality (VR) and augmented reality (AR) generate or combine virtual representations of the physical world by means of real-time computing systems that allow human beings to interact with them. The increasing availability of computing power opens new opportunities by the use of new techniques and tools which were firstly explored in the entertainment industry, but which are now being successfully extended to other domains such as medicine or education [35,38]. Even though the manufacturing sector tends to be more conservative, these tools are being increasingly used in industrial automation with promising results [16,18,29]. Actually, this has become a rapidly growing sector, with an increasing number of companies involved in the application of these tools into the production processes. This new vision is integrated in new paradigms such as Industrie 4.0 and Internet of Things (IoT) which allow the integration of real-time data from different endpoints. It is possible to find in the literature examples about how virtual and augmented reality were used for different purposes such as: (1) analyzing the products at early design stages [14,37]; (2) studying the interaction of the customers with the final products [27,42]; (3) designing and improving the manufacturing processes [2,41] and (4) monitoring and supervising remotely the processes [23,43].

The use of virtual reality offers new possibilities in industrial processes which may be grouped in two major categories [16,29]:

1. **Pure simulation of real processes:** Simulation may be a very valuable tool for: (a) staff training (e.g. flight simulators), (b) building virtual prototypes before its construction in order to analyze their operation (e.g. building a new machine or a whole automation cell), (c) conceptualizing new products or systems, and (d) marketing purposes.
2. **Operation of real processes:** Also, when combined with physical devices, virtual worlds provide enriched human-machine interfaces (HMIs) for industrial processes capable of providing: (a) remote operation of plants, (b) operation in dangerous environments, (c) process operation at micro or macro scales (e.g. at micro assembly devices [5]).

Augmented reality is also a very interesting tool for helping operators to benefit from human-robot collaboration. For example, humans could be guided by virtual reality training systems that assist them in the tasks they must carry out [20,30]. From the previous discussion, we may conclude that even though these technologies still are in the early stages they seem to have a very promising future in the manufacturing field. Unfortunately, most of the current applications are based on approaches that use expensive software packages (e.g. CAD modules).

This work presents a methodology that helps programmers to build virtual environments, which is valid for a broad number of industrial plants. Our approach is mostly based on free of cost, open technologies, which were already mature in the entertainment industry. In our view, the use of free and open software provides several advantages regardless of the costs. For example, if we analyze the evolution of the robotics software, we can observe that it has been chronically facing problems in industry and academy due to the lack of standardization, interoperability and reuse of software libraries [4]. For a long time, the most relevant problems that prevented the robotics community from producing a healthy software ecosystem were: (1) lack of code reuse; (2) higher needs of integration of components and; (3) finding the appropriate trade-off between efficiency and robustness. As a solution in the robotics domain, free software and open source software (FOS) initiatives such as the robot operating system (ROS) initiative has been promoted [4]. In our opinion this evolution of the robotics software is illustrative of the value of the FOS in critical domains and justifies its promotion to other domains.

The presented approach is illustrated by means of the creation of a non-immersive virtual world which represents a part-classifying station, allowing both the simulation of the real plant as well as its remote operation. However, the tools used in this work could be used in order to create an immersive virtual world if necessary.

In this work, the authors combine both pure simulation and teleoperation in a two step approach. Firstly, they create a virtual representation of an existing automation station (a part-classifying station). This virtual model allows analyzing and visualizing the behaviour of the station and may be used for different purposes such as staff training, process optimizing or analyzing. Secondly, they build the software infrastructure that combines virtual and physical worlds allowing remote users to teleoperate the physical plant by means of TCP/IP technologies and visualize in real-time the effect of their actions. It is important to note that from the point of view of bandwidth performance it is more efficient to connect virtual worlds with real plants than sending real camera images.

This approach is based on the connection of the virtual world with industrial controllers, like Programmable Logic Controllers (PLC), Industrial PCs (IPC) or Robot controllers, which are responsible for controlling locally the industrial processes. Consequently, our approach is open to be used to build virtual representations of a broad number of industrial processes extending the benefits of its application. This approach may be considered as an extension of the approach presented in [6] but with the application of the state-of-the-art technology.

Last but not least, the presented approach opens new possibilities to build remote laboratories in education, especially in scientific and engineering disciplines. The combination of virtual and remote laboratories in education has been analyzed by several authors [11,21,27,35]. Actually, as explained in [7], from the technical viewpoint, the problem of teleoperating remote laboratories and industrial plants is essentially the same.

The proposed layout of the article is as follows: Sect. 2 describes briefly some related works in the field; Sect. 3 describes the proposed approach, including the proposed architecture and methodology; Sect. 4 provides its application to a case study consisting of a part-classifying station in order to build a virtual representation. It also describes how the virtual model and the real plant are connected in order to be operated remotely; finally Sect. 5 draws some conclusions.

## 2 Related work

Virtual reality (VR) is defined as an alternative world that is similar to the original world, but generated through computer graphic images [8,40]. There are basically two types of VR systems: (1) *desktop systems*, in which the virtual environments are displayed on a computer screen and (2) *immersive systems*, in which users are immersed in an environment created by projectors and screens and, occasionally, head-mounting devices (HMD) or 3-D vision googles [9].

Also, augmented reality (AR) is a live direct or indirect view of a physical, real-world environment whose elements are augmented (or supplemented) by computer-generated sensory input such as sound, video, graphics or GPS data.

Even though originally these technologies were aimed at the entertainment industry, clearly, these technologies are opening new opportunities in other applications. These are frequently known as serious games. For example, the use of 3D immersive telepresence technologies as a way to communicate groups of people located in two or more geographically separate rooms, such as offices or lounges, by means of virtual joining of the spaces is analyzed in [15]. There are some works that describe how virtual reality may be used to analyzing the products at the earlier design stages by reproducing the highest number of our senses [14,42]. A similar approach is followed in [27] to analyze the ergonomic of the cars. In the manufacturing field virtual reality may be a valuable tool for building models of machines [39] or for improving the manufacturing processes previously to its implantation or deployment [8,41,44].

There are several cases in the literature that show how virtual reality is applied for staff training. An example of a training simulator for mechanical maintenance of F-16 engines is presented in [34]. In this case, the authors use based on OpenSimulator which is one of the tools used in the current work. Augmented reality may be also a very valuable tool for staff training and assistance. Some examples that show its application may be found in [20,32,45].

These technologies also allow the remote collaboration of several users in virtual environments in order to carry out common tasks. Some examples may be found in [10,16,18].

In the field of teleoperation these technologies are very valuable for operating remotely different kinds of plants such as robots [3,23,30] or for implementing advanced manufacturing techniques such as micro-device assembly [5,26]

Moreover, these technologies offer lots of new opportunities in education [11,17], especially in engineering disciplines. In [12] it is presented a taxonomy that shows the opportunities of the usage of virtual worlds in education. Some authors propose the creation of virtual laboratories where the students could reproduce their experiments. For example, Rico et al. [38] presents an architecture for building virtual laboratories in engineering education. Other authors propose the combination of both virtual laboratories with remote laboratories [7,35]. More examples can be found at [21,28,31].

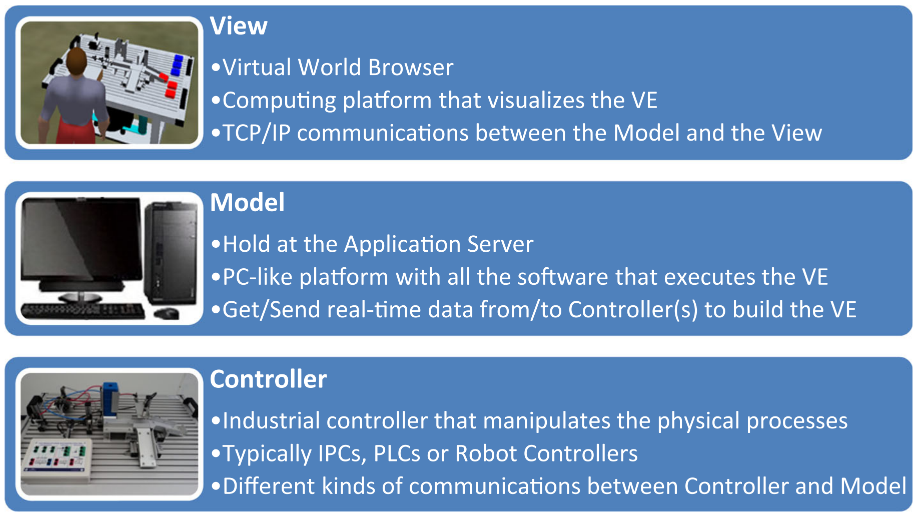
## 3 Proposed approach

The development of virtual worlds in industrial automation is typically carried out with proprietary software packages that tend to be expensive and complex to use. In addition, most frequently these virtual worlds are not expected to be connected to real equipment, but they provide a pure simulation of real processes. In this work, the authors present an approach based on free and open software packages that allows both (1) simulate the behaviour of machines and real processes and (2) operate remotely real processes. However, since the followed approach requires the combination of different tools that must be connected in real-time and share information, some guidelines are also provided in order to support the designers in the creation of the virtual environments.

### 3.1 Short overview of the architecture

The proposed approach uses the distributed architecture outlined in Fig. 1. It implements the well-known three-tier software design pattern known as Model-View-Controller (MVC) [19] which separates the business logic in three different software tiers. According to the MVC pattern *Models* are those components of the system application that actually do the work (simulation of the application domain). They are kept quite distinct from *views*, which display aspects of the models. *Controllers* are used to send messages to the model, and provide the interface between the model with its associated views and the interactive user interface devices (e.g. keyboard, mouse). Each view may be thought of as being closely associated with a controller, each having exactly one model, but a model may have many view/controller pairs [24].

In our approach: (1) The *Model* of the virtual world is held in a server running on a PC-like platform responsible of executing the Virtual Environment; (2) the *View* of the model (i.e. the virtual world) is shown by means of any compatible Virtual World Browser that may be executed in any remote computer and; (3) the *Controller*, which is responsible for controlling locally the physical process. Most frequently industrial processes are managed by means of PLCs, IPCs or Industrial Robot Controllers, so the proposed approach should be able to integrate those kinds of devices since they will be the end devices controlling the processes. From the point of view of communications, the application is divided into a two-segmented (public and private) network structures to address the requirements of efficient and secure communications between the plant and the remote operators. On one side, the communication between the View and the Model will be typically based on HTTP over TCP/IP communications, whereas the communication between the Model and the Controller may use any blend of factory communication



**View**

- Virtual World Browser
- Computing platform that visualizes the VE
- TCP/IP communications between the Model and the View

**Model**

- Hold at the Application Server
- PC-like platform with all the software that executes the VE
- Get/Send real-time data from/to Controller(s) to build the VE

**Controller**

- Industrial controller that manipulates the physical processes
- Typically IPCs, PLCs or Robot Controllers
- Different kinds of communications between Controller and Model

**Fig. 1** Application of the MVC paradigm in the methodology

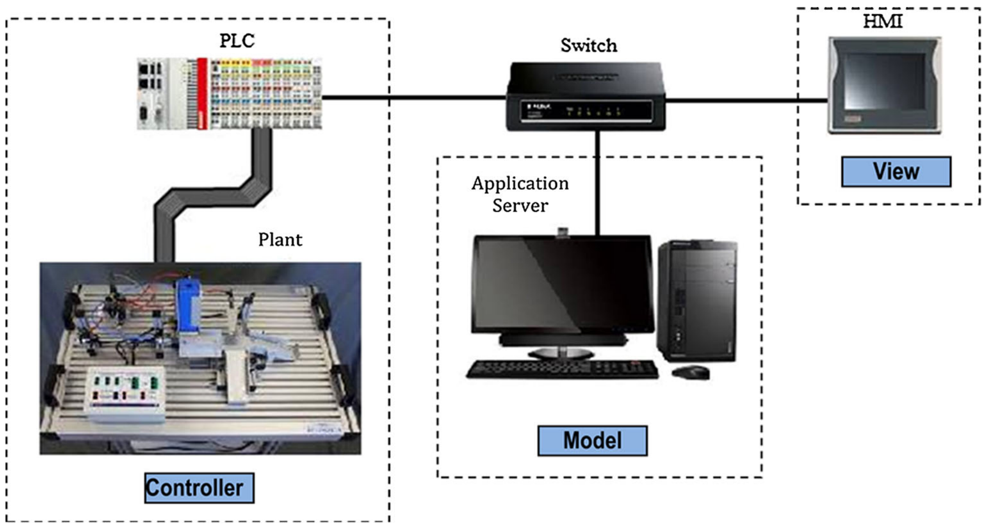


**Fig. 2** Example of implementation of the three-tier architecture

protocols, including standards like TCP/IP, OPC, Profibus or even proprietary protocols, such as ADS [1].

## 3.2 Hardware architecture

The previous three-tier architecture defined above is implemented by three kinds of devices: (1) Industrial Controllers, such as PLCs, IPCs or Robot Controllers that interact with the real plants for implementing the *Controller* tier; (2) PC-like platforms to implement the *Model* tier that holds the Virtual Environment and; (3) Visualization devices, such as HMIs or Virtual World Browsers that allow the visualization of the VE. As a matter of example, Fig. 2 shows how this architecture could be implemented by means of diverse kinds of hardware devices. Even though in this figure an Ethernet switch is used to separate Controller-Model and Model-View communications, other communication means could be equally used.

It is important to note that this three-tier architecture is open to the collaboration among several devices in each tier. Thus, the Model could be connected to several devices from which to acquire information in order to build an enriched Virtual Environment. Also, several remote viewers could represent the state of the real process at the same time. However, in such case access policies should be defined in order to avoid access conflicts from remote viewers. Similarly, complex Virtual Worlds could be divided into several application servers for performance reasons.

## 3.3 Software architecture

From the software development perspective, developing simulations of complex processes (e.g. mechanical machines) by means of simulators may be costly in terms of time and resources. This is has been somewhat lessened by employing game engines or development platforms rather than developing from scratch, rendering the early-stage selection of the engine or platform for development critical [33]. We have tried to lessen the resource requirements of both simulation development and updating by employing a readily available virtual world platform rather than a game engine. Several platforms were considered, most notably SecondLife, OpenSimulator, BSColaborate, OpenWoderland and Unity 3D. For the selection of this platform the following criteria were considered:

1. *Type of license*: Preferably free and open source software
2. *Native Operating System*: Software available to be executed at several heterogeneous environments (Windows, Linux, etc.)
3. *Scripting language*: Ease of programming. Existence of APIs.
4. *Creation of 3D objects*: Use of basic figures
5. *Importing 3D objects*: Possibility of uploading previously created objects.

After a careful evaluation *OpenSimulator* [36] was selected.

In order to allow the behaviours and operations to be accessed remotely a virtual world viewer is required. This is the software responsible for visualizing remotely the virtual representation of the virtual reality, similarly to a Web browser at the client side. Since *OpenSimulator* is an open source implementation of SecondLife, all available viewers will be compatible. The following viewers were evaluated: Firestorm Viewer, Imprudence Viewer, Kokua Viewer and Singularity Viewer. Again after a careful evaluation in which the following criteria were considered, the *Firestorm Viewer* [13] was selected.

1. *Grid Manager*: Possibility of adding or modifying connections between several remote users in the virtual world.
2. *Mesh*: Support for imported 3D models
3. *Chat*: For cooperative work between the remote users.

In addition, it was necessary to use XAMPP, which is a well-known free and open source cross-platform web server solution stack package consisting of a HTTP server, a MySQL database and several interpreters for scripts written in PHP and Perl. This tool was needed to ease the communication between the virtual and real world, as it will be described below. Actually, this step was necessary to achieve remote operation of the real automation plant. Next it is detailed the different relationship between the different software packages and the three tiers of the architecture.

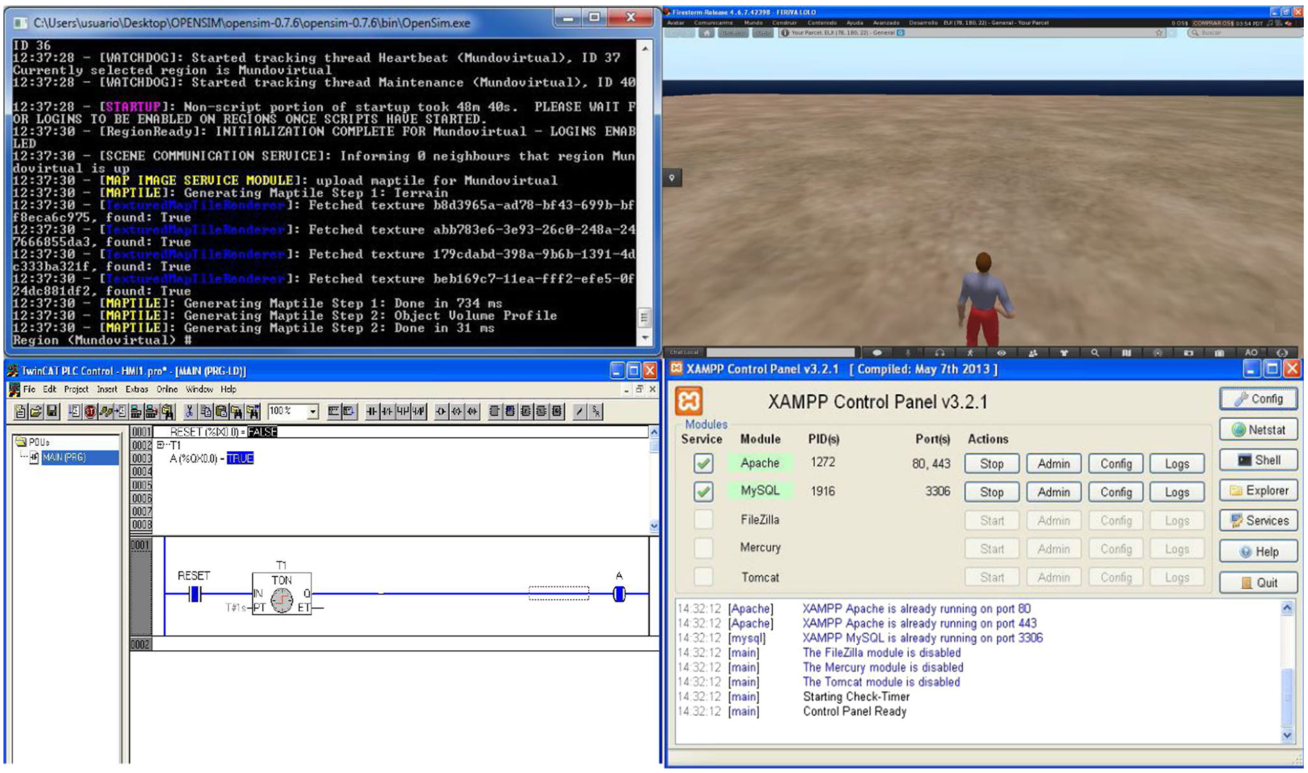
- The *Model* of the Virtual World, is implemented by means of the *OpenSimulator* platform, which is based on the source code used for the Second Life environment. More specifically, OpenSimulator is an open source 3D server that allows the creation of virtual worlds that may be accessed by means of a variety of viewers (browsers). In order to create a virtual world it is necessary: (1) to build all the different objects with which the virtual representation of the human beings in the virtual world (avatars) would be able to interact, as well as (2) to model all their possible behaviours (actions, movements, etc.). The animations of the objects in the virtual world (e.g. the movements of the components of a machine) were defined by means of the Linden Scripting Language (LSL), [25], which is a C-like language. This language allows the communication of the model with the external world by means of the Internet.
- The *View* of the Virtual World will be executed at the remote computers. These will just have to install a compatible browser with the OpenSimulator technology and manipulate the Virtual World and hence with the physical plant. The authors used the *Firestorm Viewer*.
- Finally, the *Controller* is typically locally implemented by means of industrial controllers like PLCs or IPCs which are programmed in different ways. For example, in the case of PLCs they are typically programmed according to the IEC 61131-3 standard [22].

Figure 3 shows the different software packages proposed in our approach. They are: (1) the Model (OpenSimulator), (2) the View (FirestormViewer), (3) the Controller (e.g. the program executed at the PLC) and (4) the XAMPP server in order to provide a database as well as a remote HTTP server.

In order to connect both the virtual and the physical worlds it is not enough with just installing the previous packages, but it is necessary to share information among them in real-time, especially when remote operation between the virtual plant executed at the Virtual World and the physical plant is required. This operation requires the help of additional software in order to share the information, such as Visual Basic .NET. Figure 4 shows the deployment of the software packages used in our approach as well as the software packages used to share information between the different tiers of the proposed MVC architecture. The communications issues will be analyzed in the following section.

### 3.4 Communication protocols

One of the key issues in the current approach is the management of the communication issues, since different kinds of information must be shared between the different devices. Figure 5 shows the protocols involved.


C:\Users\usuario\Desktop\OPENSIM\opensim-0.7.6\opensim-0.7.6\bin\OpenSim.exe
TwinCAT PLC Control - HMI1.pro* - [MAIN (PRG-LD)]
File Edit Project Insert Extras Online Window Help
POUs
MAIN (PRG)
RESET (%IX0.0) = FALSE
T1
A (%QX0.0) = TRUE
RESET
T1
TON
IN
PT
Q
ET
T#1s
A
XAMPP Control Panel v3.2.1 [ Compiled: May 7th 2013 ]
XAMPP Control Panel v3.2.1
Config
Netstat
Shell
Explorer
Services
Help
Quit

```
ID 36
12:37:28 - [WATCHDOG]: Started tracking thread Heartbeat (Mundovirtual), ID 37
Currently selected region is Mundovirtual
12:37:28 - [WATCHDOG]: Started tracking thread Maintenance (Mundovirtual), ID 40

12:37:28 - [STARTUP]: Non-script portion of startup took 48m 40s.  PLEASE WAIT F
OR LOGINS TO BE ENABLED ON REGIONS ONCE SCRIPTS HAVE STARTED.
12:37:30 - [RegionReady]: INITIALIZATION COMPLETE FOR Mundovirtual - LOGINS ENAB
LED
12:37:30 - [SCENE COMMUNICATION SERVICE]: Informing 0 neighbours that region Mun
dovirtual is up
12:37:30 - [MAP IMAGE SERVICE MODULE]: upload maptile for Mundovirtual
12:37:30 - [MAPTILE]: Generating Maptile Step 1: Terrain
12:37:30 - [TexturedMapTileRenderer]: Fetched texture b8d3965a-ad78-bf43-699b-bf
f8eca6c975, found: True
12:37:30 - [TexturedMapTileRenderer]: Fetched texture abb783e6-3e93-26c0-248a-24
7666855da3, found: True
12:37:30 - [TexturedMapTileRenderer]: Fetched texture 179cdabd-398a-9b6b-1391-4d
c333ba321f, found: True
12:37:30 - [TexturedMapTileRenderer]: Fetched texture beb169c7-11ea-fff2-efe5-0f
24dc881df2, found: True
12:37:30 - [MAPTILE]: Generating Maptile Step 1: Done in 734 ms
12:37:30 - [MAPTILE]: Generating Maptile Step 2: Object Volume Profile
12:37:30 - [MAPTILE]: Generating Maptile Step 2: Done in 31 ms
Region (Mundovirtual) #
```

| Service | Module | PID(s) | Port(s) | Actions | | | |
|---|---|---|---|---|---|---|---|
| ✓ | Apache | 1272 | 80, 443 | Stop | Admin | Config | Logs |
| ✓ | MySQL | 1916 | 3306 | Stop | Admin | Config | Logs |
| | FileZilla | | | Start | Admin | Config | Logs |
| | Mercury | | | Start | Admin | Config | Logs |
| | Tomcat | | | Start | Admin | Config | Logs |

```
14:32:12  [Apache]   XAMPP Apache is already running on port 80
14:32:12  [Apache]   XAMPP Apache is already running on port 443
14:32:12  [mysql]    XAMPP MySQL is already running on port 3306
14:32:12  [main]     The FileZilla module is disabled
14:32:12  [main]     The Mercury module is disabled
14:32:12  [main]     The Tomcat module is disabled
14:32:12  [main]     Starting Check-Timer
14:32:12  [main]     Control Panel Ready
```

**Fig. 3** Software packages used. Clockwise: *1* Opensimulator; *2* Firestorm Viewer; *3* XAMPP and *4* PLC programming tool

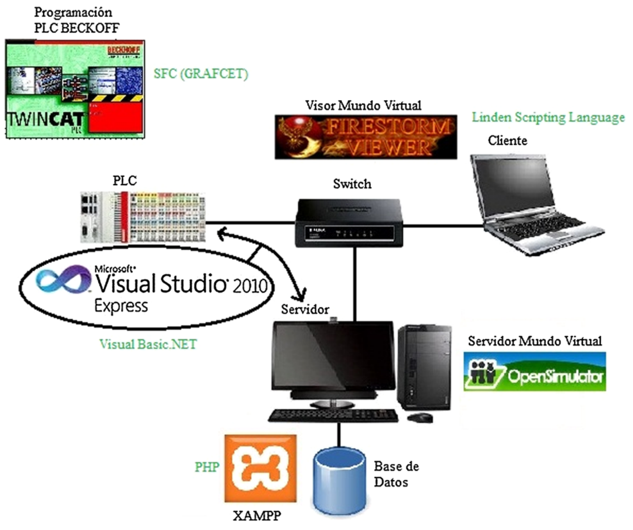


**Fig. 4** Software tools and programming languages used in the proposed approach

1. *Communication model-controller* Even though most current industrial controllers allow the use of TCP/IP protocols, sometimes they may not be the best alternative, either for performance, programming convenience or legacy reasons. Sometimes, it is a better option to carry out the communication between the industrial controller and the Model server by means of proprietary protocols. Actually, in the case study presented in this current work (see Sect. 4) it was used the ADS protocol [1], which is a proprietary protocol by Beckhoff Automation, to carry out the communication between the industrial controller (a Beckhoff IPC) and the Model server. This data must be handled at the Application Server side in order to be *inserted* into the Virtual World platform. The interface between OpenSimulator and the real process will be the MySQL database contained in the XAMPP. More specifically, OpenSimulator reads real-time process data from the database, which was previously inserted by means of any programming language (e.g. Visual Basic .NET and PHP) used to establish the connections with the Controllers (i.e. PLC or IPC).
2. *Communication model-view* The communication between the OpenSimulator server and the different viewers, such as Firestorm Viewer, is based on the Hyper Text Transfer Protocol (HTTP) and handled internally by the software tools. It is available on top of any TCP/IP connection, including the Internet.

It is important to remark that from the point of view of communications performance, the use of virtual 3D models is typically much more efficient than the use of camera images. As a matter of example, Wang [43] presents a robot monitoring and control platform for a six joint angles

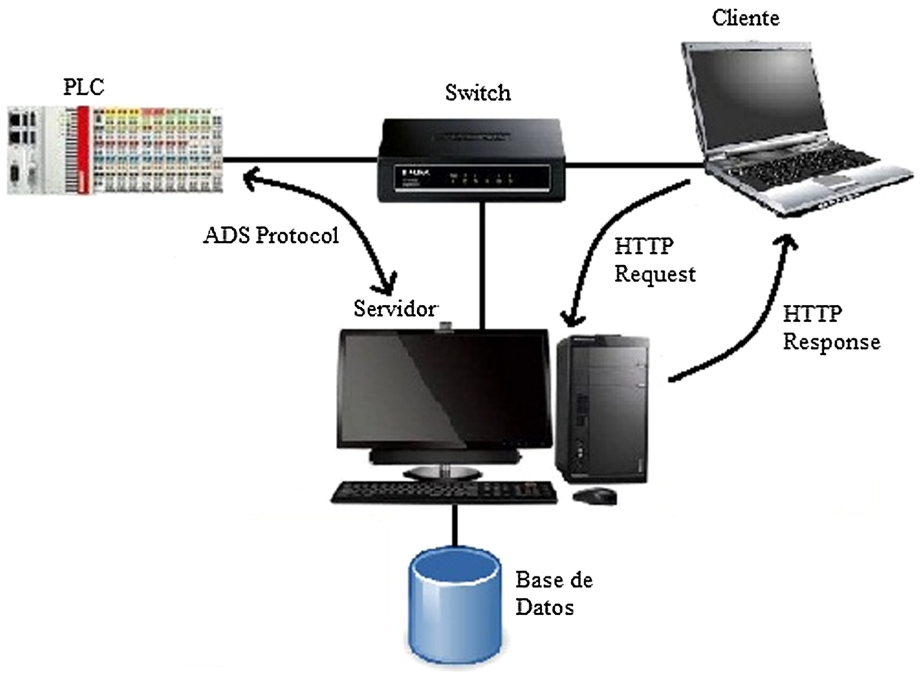


**Fig. 5** Communication protocols employed

robot. In this work is compared the bandwidth consumed by one 8-bit-depth VGA camera image, which would be $640 \times 48 = 307,200$ bytes, with the data packet size of the robot model used in their approach, consisting of six joint angles is only 52 bytes. It is clear that by using this kind of technologies it is possible to obtain a big reduction on the network bandwidth consumption.

## 3.5 Methodology

This section provides some guidelines in order to help designer to build any new virtual reality systems according to the presented approach. This methodology is divided in two parts: The first five steps are aimed at creating the Virtual World, whereas the steps from 6 to 10 are aimed at connecting both the Virtual World with the Physical Process.

*Creation of the Virtual World*

1. **Analysis of the plant or industrial process**: The first step involves knowing the details of the plant or industrial process that is going to be modeled in the virtual world. Mobile parts or elements that take part in the process must be identified. It may be necessary to build mathematical models that describe the behavior of some parts or elements.
2. **Design of the 3D model**: Once the plant or industrial process has been analyzed, a 3D model is built with all the mobile elements with the *SketchUp Make* Software. This 3D model must be exported in COLLADA format (COLLAborative Design Activity) in order to be imported from OpenSimulator. This is an interchange file format (.dae extension) for interactive 3D applications adopted as ISO/PAS 17506 specification.
3. **Creation of a virtual region in OpenSimulator**: The next step involves creating a virtual region for our plant in OpenSimulator. In case no regions were previously created, some parameters must be configured to define both the region and the *avatar.* Otherwise, in case a region already exists, it is possible to add new regions close to the already existing regions, as well as new avatars.
4. **Upload of the 3D model to the Virtual World**: Once a region is available, it is possible to import a 3D model into OpenSimulator. It may be necessary to repeat this step for every element previously created with SketchUp Make that represents the different components of the real plant.
5. **Programming OpenSimulator scripts**: In order to provide the behavior (e.g. movement of mechanical parts) to the virtual world elements and define their actions, it is necessary to program the scripts for every virtual object. The programming language used in OpenSimulator is the Linden Scripting Language (LSL), [25], which is a C-like language.

*Connection between the Virtual World and the Physical Plant*

6. **Programming the industrial controller:** The physical plant will be controlled by one or several industrial controllers, typically PLCs, IPCs or industrial robot controllers. It may be necessary to use the proprietary tools for programming the controllers. Sometimes, the pro-

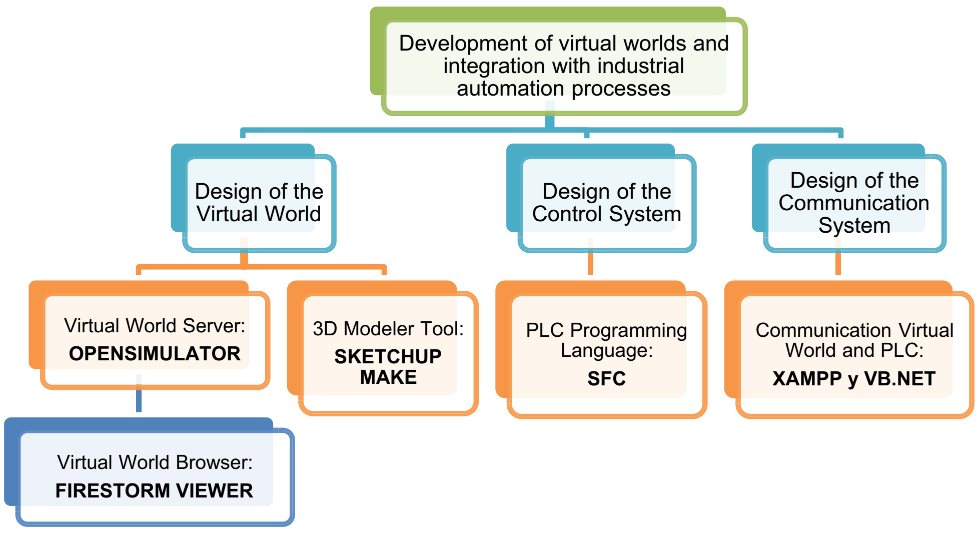

Development of virtual worlds and integration with industrial automation processes
Design of the Virtual World
Design of the Control System
Design of the Communication System
Virtual World Server: OPENSIMULATOR
3D Modeler Tool: SKETCHUP MAKE
PLC Programming Language: SFC
Communication Virtual World and PLC: XAMPP y VB.NET
Virtual World Browser: FIRESTORM VIEWER

**Fig. 6** Methodology overview and software packages used at every stage

gram will have to be created from scratch whereas in other situations it will be possible to connect legacy control programs.

7. **Connection of the physical plant and virtual world**: Once the controller program is created, some variables and scripts must be defined in order to connect the physical world controller (e.g. the PLC) with the virtual world (OpenSimulator). These variables are common to both worlds and will be shared by means of a MySQL database.
8. **Update of OpenSimulator scripts**: The scripts created in step 5 must be updated in order to access the MySQL database. In this step it will be necessary to do some PHP programming to include some instructions that allow requesting some variables from the virtual world to the database.
9. **Update of the Controller**: The program of the controller (PLC, IPC, etc.) created in step 6 must be also updated according to the variables defined in the database. It will be necessary to create a program (e.g. in Visual Basic .NET) that establishes the connection with the controller and puts the data into the database.
10. **Validate and test**: Finally, tests should be carried out in order to check the functionality of the connection between the virtual world and the physical plant.

These steps can be summarized in several stages according to Fig. 6. This figure includes the software packages used at every stage.

## 4 Case study

This section describes briefly how the previous approach has been applied to a real plant: a part-classifying station controlled with a Beckhoff CX5020 IPC which is connected with the Application Server by means of an Ethernet-ADS connection (See Fig. 5). Figure 7 provides an overview of the part-classifying station used as case study.

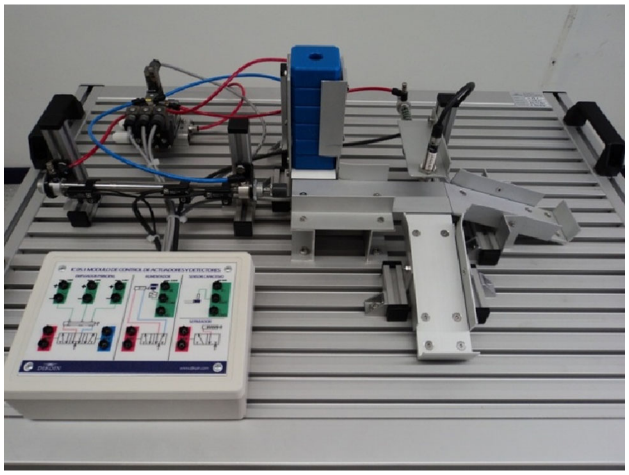


**Fig. 7** View of the real part-classifying station

### 4.1 Design of the virtual world

This section follows briefly the steps 1– 5 from the methodology explained in Sect. 3 in order to achieve a Virtual World that represents the physical plant shown in Fig. 7.

Firstly, the behaviour and components of the plant was analyzed (*Step 1*). Several sensors were observed to detect the position of the parts and its nature (i.e. whether they have a hole in the middle or not) in order to separate them. Also, the mobile elements were identified. In this case the station includes several pneumatic cylinders that move the parts and

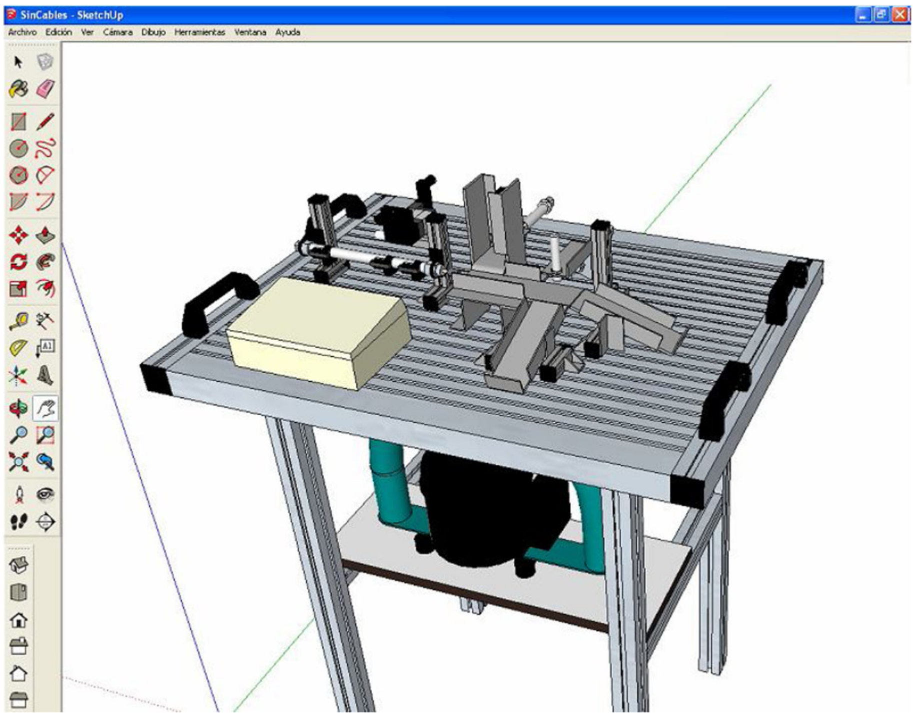

SinCables - SketchUp
Archivo Edición Ver Cámara Dibujo Herramientas Ventana Ayuda

**Fig. 8** 3D SketchUp model of the plant

carry out the classification process. These are the only mobile elements of the plant to be represented in the virtual world. A SketchUp representation of the plant was created (see Fig. 8) in order to be imported from OpenSimulator (*Step 2*). Later, a new region in OpenSimulator was created to hold the Virtual World as well as an *avatar* (*Step 3*). To connect a client to the OpenSimulator server it is necessary to execute the Firestorm Viewer (See Fig. 3). Next step involves uploading the 3D model (See Fig. 8) into the virtual world (*Step 4*). This model should be exported in COLLADA format (.dae extension), which is the one used by OpenSimulator for 3D objects. In the importing process it is possible to choose from several textures and levels of detail. Finally, it will be necessary to animate the mobile objects by means of scripts written in the LSL (*Step 5*). LSL is a state-event driven scripting language, in the sense of a finite state machine. A script consists of variables, functions and one or more states. Each state contains a description of how to react to events that occur while the program is within that state.

Once all these steps have been carried out, the virtual world that holds the representation of the physical plant is executed at the OpenSimulator server. The OpenSimulator server allows downloading both the description of the objects and the animation code that describes the logic of their movements by means of the HTTP protocol for their execution at any remote viewer like the Firestorm Viewer. Figure 9 shows a virtual representation of the real plant in a remote viewer which may be used for designing or improving the processes as discussed above.

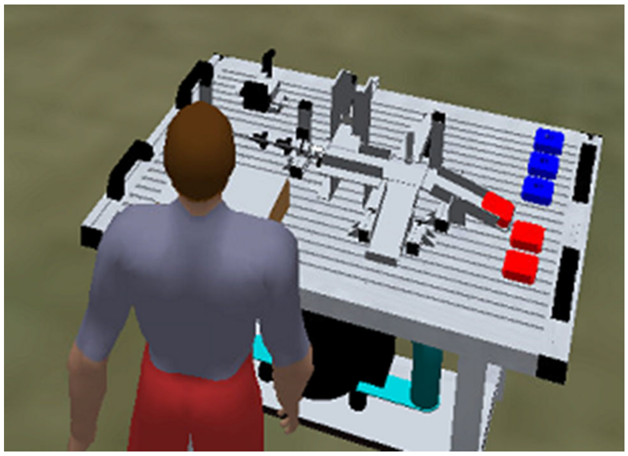
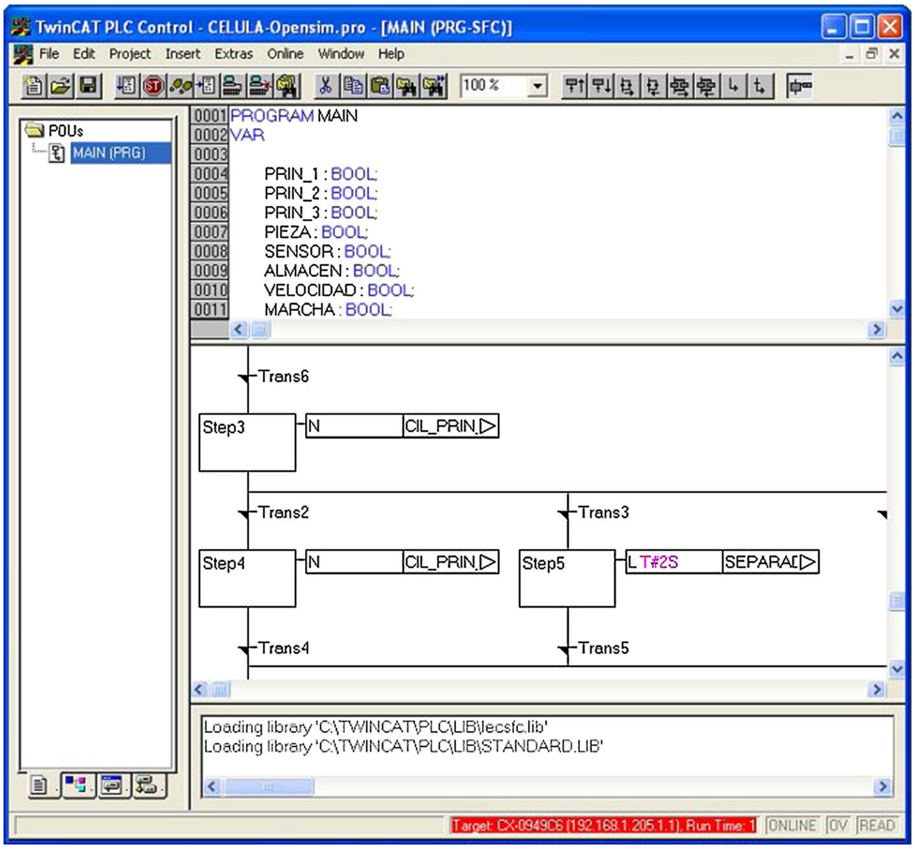


**Fig. 9** Virtual model of the part-classifying station

## 4.2 Design of the Control System

In this case study it was also necessary to develop the program at the Beckhoff CX5020 IPC that controls locally the plant shown in Fig. 7 (*Step 6*). In this case the 61131-3 SFC (Sequential Function Chart) programming language, also known as GRAFCET, was used in order to define the local behaviour of the plant. TwinCAT PLC Control, which is a proprietary tool by Beckhoff, was used for this task. Figure 10 shows an overview of the programming environment.

This step is independent from the construction of the virtual representation of the plant, so it does not need to be


TwinCAT PLC Control - CELULA-Opensim.pro - [MAIN (PRG-SFC)]
File Edit Project Insert Extras Online Window Help
100 %
POUs
MAIN (PRG)
Trans6
Step3
N
CIL_PRIN
Trans2
Trans3
Step4
N
CIL_PRIN
Step5
L T#2S
SEPARAD
Trans4
Trans5
ONLINE
OV
READ

```
PROGRAM MAIN
VAR

    PRIN_1 : BOOL;
    PRIN_2 : BOOL;
    PRIN_3 : BOOL;
    PIEZA : BOOL;
    SENSOR : BOOL;
    ALMACEN : BOOL;
    VELOCIDAD : BOOL;
    MARCHA : BOOL;
```

```
Loading library 'C:\TWINCAT\PLC\LIB\Iecsfc.lib'
Loading library 'C:\TWINCAT\PLC\LIB\STANDARD.LIB'
```

```
Target: CX-0949C6 (192.168.1.205.1.1), Run Time: 1
```

**Fig. 10** Programming environment for the Controller

necessarily carried out at this stage. Actually, it is possible to wrap the behaviour of existing plants after carrying out some modifications in the software as defined in next section.

### 4.3 Design of the Communication System

Once the behaviour of both the virtual plant and the real plant were defined respectively in Steps 5 and 6, the next step involved connecting both worlds (*Step 7*). This step requires declaring common variables in the MySQL database available at both worlds. The addition of these new variables requires updating both the OpenSimulator scripts (*Step 8*) as well as the program executed at the Controller side (*Step 9*). The interaction between OpenSimulator and the database is executed by means of PHP scripts, whereas the communication between the database and the program executed at the PLC controller is carried out by means of different protocols, including proprietary protocols. In the presented case study ADS and Visual Basic .NET were used for this task. ADS is a proprietary protocol by Beckhoff Automation and specific libraries were required to allow its use from Visual Basic .NET. The communication between the Application Server (*Model*) and the Firestorm Viewer (*View*) was carried out directly by OpenSimulator on top of HTTP. This approach allows the remote communication of the application over any TCP/IP network, including Internet. Figure 11 shows the interaction between the real and virtual world. The authors also developed a local HMI to interact with the physical plant (shown at screen of the left image).

The following link provides a visualization of the interaction between both the physical plant and the virtual world used at the case study: https://cps4pss.wordpress.com/2015/12/23/integracion-de-mundos-virtuales-y-procesos-reales/

## 5 Conclusions and future work

This work discusses about the integration of virtual worlds with industrial automation processes by means of free open software tools which are already mature at the entertainment industry. More specifically, it presents an architecture and methodology that assists programmers in the creation of virtual reality systems and its integration with real manufacturing processes. These processes will be typically controlled by industrial devices such as PLCs, IPCs or industrial robots. The presented approach, which is valid for a broad number of industrial plants, opens new possibilities for creating virtual reality models aimed at two major directions: (1) *Pure simulation* of real processes and products for different purposes (e.g. staff training; prototype designing; manufacturing optimization; marketing) and (2) *Teleoperation* of real processes (e.g. in dangerous environments or in micro and

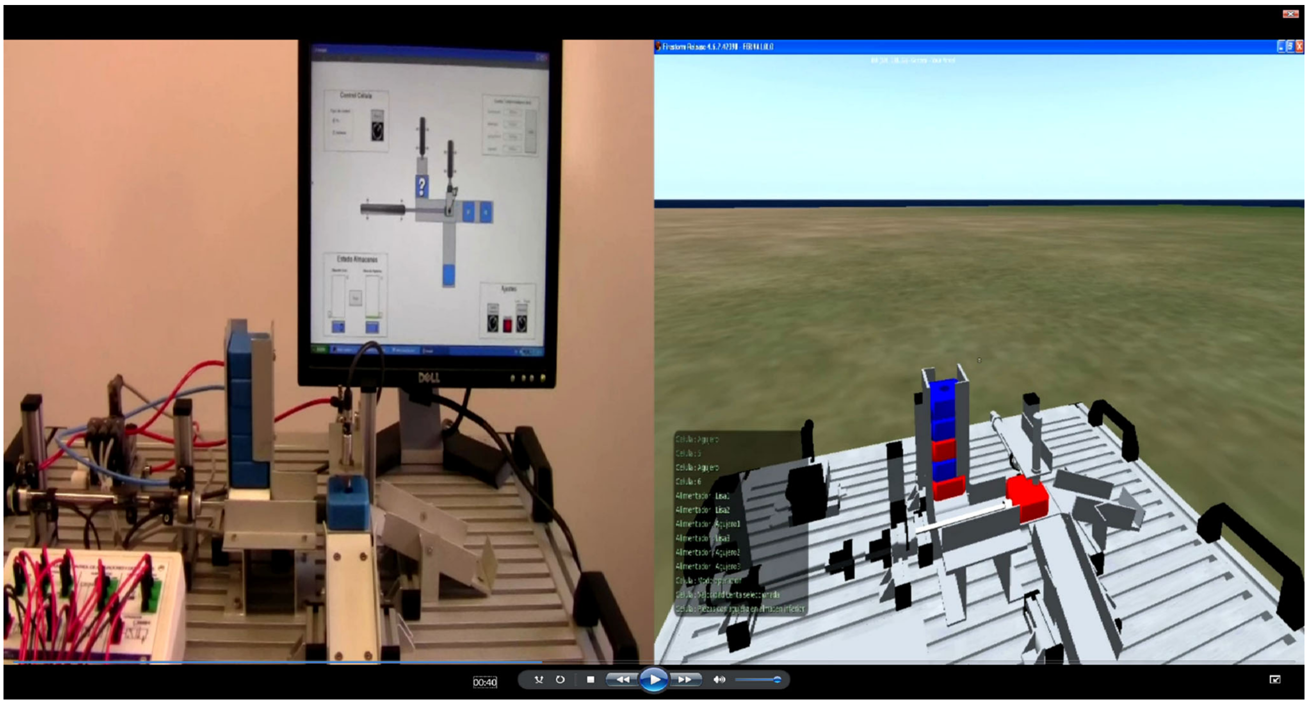


Fig. 11 Interaction between the physical and virtual world

macro scale manufacturing). It is also interesting to remark that this approach also opens new possibilities in education, especially in engineering disciplines, since it increments the possibilities of collaborative learning through the Internet, by exploring the possible interactions in the virtual world.

The presented approach proposed the use of OpenSimulator as the platform to execute the virtual worlds. This software, based on the technology developed for the Second Life environment benefits from all the tools developed for this environment. Thus, it is compatible with all its available browsers and one of them, Firestorm Viewer, was selected.

By following the proposed methodology, the designers will have some guidelines to follow in order to create and connect the physical plants to the virtual worlds. Also, they will be able to have collaborative environments in which avatars remotely controlled will be able to cooperate in the execution of one task.

The application of the methodology was illustrated by means of the creation of a non-inmersive virtual world which represents a part-classifying station, allowing both the simulation of the real plant as well as its remote operation. Also, the authors illustrated how a virtual model may be connected to a physical plant by means of an industrial controller (a PLC or an IPC) allowing the remote operation of almost any plant. This approach provides an enriched Human-Machine Interface (HMI) that introduces new possibilities in the fields of teleoperation or the operation in dangerous environments or different scale operation. In addition, from the point of view of bandwidth consumption, it is more efficient to use this kind of technologies than sending full camera images.

In the future, the authors will work with the possibilities that this technology offers to build immersive systems by means of Head-Mounted-Displays (HMDs).

**Acknowledgments** This work was supported in part by the University of the Basque Country (UPV/EHU) and the Basque Government (GV/EJ) by projects EHU13/42 and CPS4PSS ETORTEK14/10 respectively.

## References

1. Beckhoff ADS Specification (2016). https://infosys.beckhoff.com/english.php?content=../content/1033/tcadsamsspec/html/tcadsamsspec_intro.htm&id=
2. Andrisano, A.O., Leali, F., Pellicciari, M., Pini, F., Vergnano, A.: Hybrid reconfigurable system design and optimization through virtual prototyping and digital manufacturing tools. Int. J. Interact. Des. Manuf. **6**(1), 17–27 (2012)
3. Ang, Q.-Z., Horan, B., Nahavandi, S.: Multipoint haptic mediator interface for robotic teleoperation. IEEE Syst. J. **9**(1), 86–97 (2015)
4. Bruyninckx, H.: Robotics software: the future should be open. IEEE Robot. Autom. Mag. **15**(1), 9–11 (2008)
5. Cecil, J., Bharathi Raj Kumar, M.B., Lu, Y., Basallali, V.: A review of micro-devices assembly techniques and technology. Int. J. Adv. Manuf. Technol (2015). doi:10.1007/s00170-015-7698-6 **(Article in Press)**
6. Calvo, I., Marcos, M., Orive, D., Sarachaga, I.: A methodology based on distributed object-oriented technologies for providing remote access to industrial plants. Control Engineering Practice **14**(8), 975–990 (2006)

7. Calvo, I., Marcos, M., Orive, D., Sarachaga, I.: Building complex remote learning laboratories. Comput. Appl. Eng. Educ. **18**(1), 53–66 (2010)
8. Comm, C.L.: Mathaisel Dennis FX: A paradigm for benchmarking lean initiatives for quality improvement. Benchmark Int J **7**(2), 118–127 (2000)
9. Dewar R.G, Carpenter I.D, Ritchie J.M, Simmons J.E.L.: Assembly planning in a virtual environment. In: Proc. Portland Int'l Conf. on Management of Engineering and Technology. IEEE Press, Piscataway, pp 664–667 (1997)
10. Dow, L., Campbell, A., Miller, A., McCaffery, J., Oliver, I., Davies, C.J., Kennedy, S., Allison, C.: An immersive platform for collaborative projects: In: Proc. Frontiers in Education Conf., FIE, 2015-February, (2015)
11. De Jong, T., Linn, M.C., Zacharia, Z.C.: Physical and virtual laboratories in science and engineering education. Science **340**(6130), 305–308 (2013)
12. Duncan, I., Miller, A., Jiang, S.: A taxonomy of virtual worlds usage in education. Brit. J. Educ. Technol. **43**(6), 949–964 (2012)
13. Firestorm Viewer (2016). http://www.firestormviewer.org/
14. Ferrise, F., Graziosi, S., Bordegoni, M.: Prototyping strategies for multisensory product experience engineering, J. Intell. Manuf. 1–13 (2015). doi:10.1007/s10845-015-1163-0 **(Article in Press)**
15. Fuchs, H., State, A., Bazin, J.-C.: Immersive 3D telepresence. Computer **47**(7), 46–52 (2014)
16. Galambos, P., Baranyi, P., Rudas, I. J.: Merged physical and virtual reality in collaborative virtual workspaces: The VirCA approach. In: Proc. of the 40th IEEE Annual Conf. of the IEEE (IECON), pp. 2585-2590, (2014)
17. Gomes, L., Bogosyan, S.: Current trends in remote laboratories. IEEE Trans. Ind. Electron. **56**(12), 4744–4756 (2009)
18. Galambos, P., Csapó, A., Zentay, P., Fülöp, I.M., Haidegger, T., Baranyi, P., Rudas, I.J.: Design, programming and orchestration of heterogeneous manufacturing systems through VR-powered remote collaboration. Robotics and Computer-Integrated Manufacturing **33**, 68–77 (2015)
19. Gamma, E., Helm, R., Johnson, R., Vlissides, J.: Design Patterns Elem. Reusable Object-Orient. Softw. Addison-Wesley, Reading (1995)
20. Hořejší, P.: Augmented reality system for virtual training of parts assembly. Procedia Eng. **100**, 699–706 (2015)
21. Hu, W., Liu, G.-P., Zhou, H.: Web-based 3-D control laboratory for remote real-time experimentation. IEEE Trans. Ind. Electron. **60**(10), 4673–4682 (2013)
22. IEC 61131-3 Standard (2013) Programmable controllers—Part 3: programming languages. https://webstore.iec.ch/publication/4552
23. Kato, Y.: A remote navigation system for a simple tele-presence robot with virtual reality. In: IEEE Int. Conf. on Intelligent Robots and Systems, pp. 4524-4529 (2015)
24. Krasner, G.E., Pope, S.T.: A cookbook for using the model-view controller user interface paradigm in Smalltalk-80. J. Object-Orient. Programm. **1**(3), 26–49 (1988)
25. Linden Scripting Language (LSL) (2016). http://wiki.secondlife.com/wiki/LSL_Portal/es
26. Liang, J.S., Chao, K.-M., Ivey, P.: VR-based wheeled mobile robot in application of remote real-time assembly. Int. J. Adv. Manuf. Technol. **64**(9–12), 1765–1779 (2013)
27. Lawson, G., Herriotts, P., Malcolm, L., Gabrecht, K., Hermawati, S.: The use of virtual reality and physical tools in the development and validation of ease of entry and exit in passenger vehicles. Applied Ergonomics **48**, 240–251 (2015)
28. Lei, Z., Hu, W., Zhou, H., Zhong, L., Gao, X.: A DC motor position control system in a 3D real-time virtual laboratory environment based on NCSLab 3D. Int. J. Online Eng. **11**(3), 49–55 (2015)
29. Lee, C., Park, S.: Survey on the virtual commissioning of manufacturing systems. Journal of Computational Design and Engineering **3**(1), 213–222 (2014)
30. Mostefa, M., Boudadi, L.K.E., Loukil, A., Mohamed, K., Amine, D.: Design of mobile robot teleoperation system based on virtual reality. In: Proc. 3rd Int. Conf. on Control, Engineering and Information Technology, CEIT (2015)
31. Müller, D., Chilliischi, A., Langer, S.: Integrating immersive 3D worlds and real lab equipment for teaching mechatronics. In: Proceedings of the 9th International Conference on Remote Engineering and Virtual Instrumentation, REV (2012)
32. Matsas, E., Vosniakos, G.C.: Design of a virtual reality training system for human–robot collaboration in manufacturing tasks. Int. J. Interact. Des. Manuf. (2015). doi:10.1007/s12008-015-0259-2 **(Article in Press)**
33. Petridis, P., Dunwell, I., Panzoli, D., Arnab, S., Protopsaltis, A., Hendrix, M., Freitas, S.: Game engines selection framework for high-fidelity serious applications. Int. J. Interact. World 1–19 (2012)
34. Pinheiro, A., Fernandes, P., Maia, A., Cruz, G., Pedrosa, D., Fonseca, B., Paredes, H., Martins, P., Morgado, L., Rafael, J.: Development of a mechanical maintenance training simulator in OpenSimulator for F-16 aircraft engines. Entertainment Computing **5**(4), 347–355 (2014)
35. Pereira, C.E., Paladini, S., Schaf, F.M.: Control and automation engineering education: combining physical, remote and virtual labs. In: Proceedings of the 9th International Multi-Conference on Systems, Signals and Devices (2012)
36. OpenSimulator (2016). http://opensimulator.org/wiki/Main_Page
37. Raffaeli, R., Germani, M.: Advanced computer aided technologies for design automation in footwear industry. Int. J. Interact. Des. Manuf. **5**(3), 137–149 (2011)
38. Rico, M., Ramirez, J., Riofrío, D., Berrocal, M., de Antonio, A.: An architecture for virtual labs in engineering education. In: Proc. of the 2012 IEEE Global Engineering Education Conf. EDUCON, pp. 210–214 (2012)
39. Sghaier, A., Soriano, T.: Using high-level models for modeling industrial machines in a virtual environment". International Journal on Interactive Design and Manufacturing **2**(2), 99–106 (2008)
40. Tyagi, S.K., Ghorpade, A., Karunakaran, K.P., Tiwari, M.K.: Optimal part orientation in layered manufacturing using evolutionary stickers-based DNA algorithm. Virtual Phys. Prototype **2**(1), 3–19 (2007)
41. Tyagi, S., Vadrevu, S.: Immersive virtual reality to vindicate the application of value stream mapping in an US-based SME. International Journal of Advanced Manufacturing Technology **81**(5–8), 1259–1272 (2015)
42. Tideman, M., van der Voort, M.C., van Houten, F.J.A.M.: A new product design method based on virtual reality, gaming and scenarios". Int. J. Interact. Des. Manuf. **2**(4), 1–11 (2008)
43. Wang, L.: Collaborative robot monitoring and control for enhanced sustainability. International Journal of Advanced Manufacturing Technology **81**(9–12), 1433–1445 (2015)
44. Wang, C.G., Mitrouchev, P., Li, G.Q., Lu, L.X.: Disassembly operations' efficiency evaluation in a virtual environment. Int. J. Comput. Integrat. Manuf. **29**(3), 309–322 (2015)
45. Yew, A.W.W., Ong, S.K., Nee, A.Y.C.: Towards a griddable distributed manufacturing system with augmented reality interfaces. Robotics and Computer-Integrated Manufacturing **39**, 43–55 (2016)