PASSENGER INFORMATION SYSTEM SIMULATION

Louis Botha

Master's thesis
December 2015
Master of Engineering
Information Technology

TAMPEREEN AMMATTIKORKEAKOULU
Tampere University of Applied Sciences

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Master of Engineering
Information Technology

Louis Botha:
Passenger Information System Simulation

Master's thesis 32 pages
December 2015

This thesis goes through the development process of a Passenger Information System simulator for a customer to test their route information databases on before distributing these to the rolling stock.

The paper starts by describing the basic components and devices of a Mitron passenger information system and the purpose of the passenger information system simulator ordered by the customer.

The paper continues by describing the initial design that was offered to the customer. The 11 main functionalities/requirements of the simulator and their acceptance criteria that was agreed upon with the customer are being explained which are followed by the implementation details of the solution.

Basically the solution uses a VirtualBox virtual machine as a base for the simulator which run the Mitron PIS Software. The solution allows for installing and updating databases, driving of routes and following the route visually and hearing the audio announcements.

The solution also uses a VirtualBox appliance as the way to pack and distribute the solution to the customer.

CONTENTS

ABBREVIATIONS AND TERMS

ALSA	Advanced Linux Sound Architecture
DCP	Driver Control Panel (Mitron hardware component)
DCU	Display Control Unit (Mitron hardware component)
Infotainment	Extra information like POI or advertisements
FPGA	Field-programmable gate array
IDE	Integrated development environment
INOI	Company providing remote monitoring and management software
Mitron	Mitron Oy
On-route	Mission loaded, PIS system in on-route mode
Off-route	Mission not loaded, PIS system in off-route mode
PICS	Passenger Information Creation System software
PIS	Passenger Information System
POI	Points of interest
SAE	Society of Automotive Engineers vehicle bus communication
scp	Secure copy
sftp	Secure File Transfer Protocol
ssh	Secure Shell
SVG	Scalable Vector Graphics
TCU	Train Central Unit, refers to software on the DCU
TFT	TFT display (Mitron hardware component)
VirtualBox	Computer virtualization software
VirtualBox Appliance	Pre-configured virtual machine image, ready to run in VirtualBox

1 INTRODUCTION

Mitron is an internationally acknowledged specialist in Internet Protocol based on-board systems for trains, trams and metros. Mitron products extend from displays and individual products to comprehensive passenger information systems for public address, intercommunication, CCTV and entertainment.

PICTURE 1: Mitron On-Board Solutions

1.1 Basic system components

A typical passenger information system installed into rolling stock consists of a train central unit, one or more driver control panels, loudspeakers and amplifiers and some internal and/or external displays. The display control unit is at the heart of the system.

PICTURE 2: Mitron display control unit hardware

The display control unit (DCU) hardware is also referred to as the train central unit (TCU). The DCU hardware is a computer that is rack mounted into a Mitron designed rack and can run at 24V or 36V. The computer has Linux installed, with a common base set of software and libraries that are needed for communication between the Mitron devices. Project specific software and configurations are installed on top of the base system. At the center of the base system is a database and the application interfaces to access the data. The database contains all the route and departure information as well as the media that needs to be played over the audio system. The databases are mostly created with Mitron database creation tools but the system also allows for customers to use the Mitron XML schema to generate the database from other sources. The train central unit contains also the different controller software for the displays and audio.

The DCU has a GPS/3G module built in, to which antennas mounted on the outside of the rolling stock, are connected. On most systems GPS are used as the primary source for positioning. On some rolling stock, the system also receives an odometer reading as backup for positioning. Positioning is an essential part of the system. All event triggering and all automated announcements are based on trigger points and distances defined in the database for the route. The 3G data connection is used in various ways depending on the project. In some projects the data connection is simply used for remotely updating the timetable databases to the rolling stock. In some projects the rolling stock is communicating constantly with some operations software in the offices of the customer, receiving position and diagnostics from the rolling stock. Some projects also use this connection for showing real-time updates to trains schedules, news and weather forecasts.

PICTURE 3: Mitron driver control panel

The driver or conductor uses a driver control panel (DCP) to control the passenger information system. These touch panels are normally mounted in each driver cabin. The driver panel software is running on the Linux operating system. The driver uses the panel for example to select the correct route, play a specific prerecorded announcement or display custom text announcements on the displays. Some other functionality on the control panel are changing the active databases, viewing system diagnostics, changing volume levels and playing test announcements. Depending on the project, the USB port on the front of the panel can be used to update the software and/or timetable databases on the rolling stock.

The passenger information system can consist of a variety of displays types.

PICTURE 4: Mitron single sided wall mounted TFT display

PICTURE 5: Mitron double sided hanging TFT display

Normally there will be TFT displays mounted on the inside of the rolling stock in the passenger areas, displaying some kind of route information, often referred to as a pearl chain. The TFT contains a computer running the Linux operating system and then project specific software with the layouts of the specific customer. When the route is selected by the driver, the display controller software on the DCU keeps the state of the information on the displays up to date. Mitron older projects uses a Java application to generate and display the pearl chain on the display. Mitron newer projects uses web technology to show the pearl chain with Chromium browser running in kiosk mode.

PICTURE 6: Mitron LED display

The LED displays are mostly used on the sides and the front of the rolling stock to display route information to the people located on the outside of the train. The LED displays have their own proprietary software that are controlled from the train central unit. There is some projects which opted to use LED displays in the passenger areas of the rolling stock.

PICTURE 7: Mitron audio amplifier

Announcements are streamed from the audio controller on the train central unit to the audio amplifiers on board the rolling stock. The amplifiers plays the audio over the loudspeakers. The amplifiers have a Linux operating system. The driver microphones are connected directly to the amplifiers. The amplifiers have multiple audio inputs and outputs and allow for configurations where announcements can be targeted to specific areas on the rolling stock. The amplifiers also have digital inputs and outputs to which analogue relays can be connected. This can be used with relays for example to inform the Mitron system the state of the train doors.

PICTURE 8: Mitron emergency phone

The passenger information system can include also an emergency phone system. The emergency phones are located in the passenger areas and used to contact the driver or conductor of the rolling stock. The emergency phones contains proprietary embedded software. The emergency phone calls are routed via the train central unit to the driver cabins.

In some projects, the passenger information system have an interface to the rolling stock manufacturers' control and management system. The passenger information system can receive information like odometer readings, door openings and emergency brake activations via these interfaces. The passenger information system can for example use this information to show specific information on the displays, trigger announcements and/or activate surveillance cameras in the area where an emergency brake was activated. The passenger information system also provides information like GPS coordinates, route location and system time to the rolling stock over the interface to the rolling stock manufacturer control and management system.

A passenger information system can also include a CCTV system. IP cameras and dedicated hardware for video recording are then installed to the rolling stock. When there is cameras on the rolling stock, the driver control panel can for example be used to display camera views to the driver of door areas or an area where an emergency brake or emergency phone was activated.

PICTURE 9: Westermo railway certified switch

All the Mitron devices on the rolling stock are connected by Ethernet network through Rail certified switches. Mitron doesn't manufacture their own network switching components but uses 3rd party components from Hirschman, Moxa and Westermo.

The software running the Mitron passenger information system is mostly written in Java and the Java Messaging Service is used in the system to communicate between the software process and software on the devices. The passenger information system uses the PostgreSQL as the database server on the rolling stock. Most devices are running the

Linux operating system which allows for configuration through Bash Shell and Python scripts.

1.2 PIS Simulation

The customer that requested and ordered the PIS Simulator is a tram operator in France. The trams were refurbished and fitted with a Mitron solution consisting of public address, intercommunication, CCTV and infotainment. The tram operator desired a way to test the tram routes and infotainment events defined in the databases created with Mitron Passenger Information Creation System (PICS) software before the new databases are distributed to the individual trams.

Mitron offered two solutions to the tram operator. A hardware based test system with the minimal physical hardware necessary to be able to do the desired testing or a software based simulator solution with the software needed to accomplish the testing. The operator decided upon the software only simulator solution and out of this decision came the subject of this document.

An initial design was created for a solution based on VirtualBox. A design and functional specification document was written and delivered to the customer for approval. The initial design that was offered can be seen in Chapter 2. The functionality of the PIS Simulator and the rationale behind each functionality are shown in Chapter 3.

In Chapter 4 the actual implementation will be shown and explained. The rationale behind decisions and reasons for parting with the original design will be given.

Chapter 5 will discuss the simulator implementation, the uses and where improvements can be made.

2 OFFERED DESIGN CONCEPT

The following chapter shows the initial design concept document that was written and offered to the tram operator after it was decided to go forward with a software only simulation solution. Some clarifications are added here to the original document text for the benefit of the reader of this thesis.

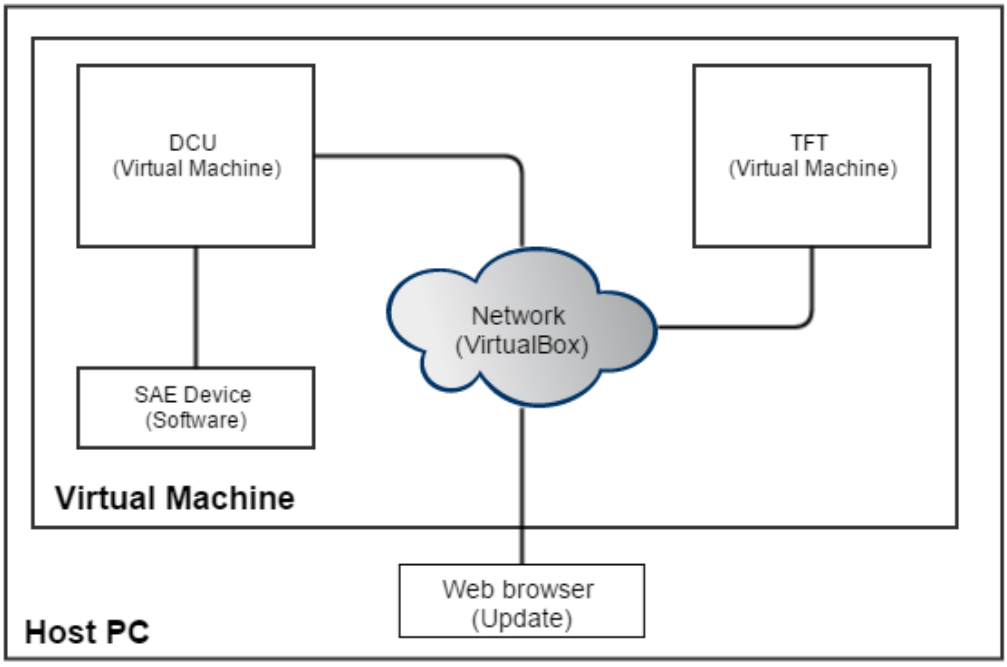
The purpose of the simulated passenger information system in the tram is for testing the tram routes and event triggers defined in the route and timetable databases in the head office of the tram operator. This is to allow for testing and verification of the information in the databases before these databases are distributed to the trams in operation. These databases are created with Mitron Passenger Information Creation System (PICS) software and is outside the scope of the simulator.

As the customer decided on a software only solution, the basis of the PIS Simulator design is a VirtualBox virtual machine containing the minimal necessary virtual devices needed for simulating the Mitron passenger information system functionality running on a tram when driving on route. In other words, the simulated tram environment will have to:

- allow the customer to import and update timetable databases
- allow the customer to simulate driving a tram route
- allow the customer to see a render of an internal display
- allow the customer to confirm visually and audibly that events (like announcements and advertisements) are triggered

For accomplishing the points mentioned above, the simulated tram environment will need to consist of at least the following three devices:

- DCU running all the main passenger information software processes
- TFT displaying the passenger information display layouts
- Society of Automotive Engineers vehicle bus device (SAE Simulator) for sending the route events



PICTURE 10: PIS Simulator Design

As seen in Picture 10, the simulated environment will only have one piece of physical hardware, a standard personal computer hosting the VirtualBox virtual machines. The recommended hardware for the PC is the same as the environment on which the simulator was designed and tested on at Mitron.

Processor	Reasonably powerful **x86 hardware** Minimum: Quad Core 2,0 GHz
HDD	Reasonably sized hard drive (PIS Simulator uses 50GB) Minimum: 320 GB
Memory	8 GB or more (4 GB for host + 4 GB for PIS Simulator machine) Minimum: 8GB
Operating System	Windows, Linux or Apple running VirtualBox (www.virtualbox.org) should work.
VirtualBox	Version 4.3.12 (Minimum and tested version)

TABLE 1: Recommended PC hardware

The PC, also referred to as the host PC, will have the VirtualBox software installed and will run a Linux guest virtual machine.

Inside the Linux guest virtual machine there will be a virtual machine simulating the Mitron DCU hardware and running the main passenger information software. There will be another virtual machine simulating a Mitron TFT and running the software for rendering the internal route display of the passenger information system. The SAE Simulator for simulating driving and route events will be running in the guest virtual machine environment and not as its own virtual machine.

Networking between the devices will be accomplished by using VirtualBox own networking virtualization software for allowing the Mitron virtual devices to communicate. Communication between the tram and PIS system is done with the Society of Automotive Engineers vehicle bus communication standard. SAE simulation can be done using a virtual serial port of the DCU virtual machine. Mitron will provide the software to simulate the SAE communication in order for the client to change the tram location and display all the screens of a route.

The VirtualBox audio system will be used to play the audio from the virtual DCU software to the host PC audio card.

The Web browser of the host virtual machine will be used to update the databases and virtual DCU software.

The PIS Simulator will be distributed as a VirtualBox appliance that can be imported into the VirtualBox virtualization software running on any PC where VirtualBox is installed and running.

3 SIMULATOR FUNCTIONALITY

During analysis, design and planning, the basic functionalities/requirements and acceptance criteria for the PIS Simulator were established and agreed upon with the customer. The agreed upon functionality and acceptance criteria for the PIS Simulator will be presented point by point in this chapter as a background reference. A brief explanation to the rationale behind the functionality is explained as well.

3.1 SIMU-01: PIS Simulator VirtualBox appliance

The requirement is that the PIS Simulator virtual machine is distributed as a VirtualBox appliance .ova file. The acceptance criterion is a PIS_Simulator.ova file that is available for download and installation from the Mitron FTP server.

A VirtualBox appliance was decided upon as it provides an easy way for the end user to install a very complex solution with very little configuration. The virtual machine will be preconfigured and a snapshot of the machine will be exported as the appliance.

3.2 SIMU-02: Import PIS Simulator VirtualBox appliance

The requirement is that the PIS_Simulator.ova VirtualBox application can be imported into any VirtualBox instance and provide the PIS Simulator virtual machine. The acceptance criterion is that the PIS_Simulator.ova can be imported by VirtualBox without any errors.

The requirement is to emphasize that the end user that uses or installs the tool might not be a software or information technology professional and that the solution should be easy to take into use. Emphasis is on very little configuration as to avoid places where errors can occur when the user needs to change settings.

3.3 SIMU-03: Starting up the PIS virtual machine

The requirement is that the PIS Simulator virtual machine can be powered up after installation. The acceptance criterion is that the PIS Simulator virtual machine starts and runs as a normal VirtualBox virtual machine.

The requirement again emphasizes that the threshold to start using the PIS simulator virtual machine should be very low and without relying on any end user technical abilities.

3.4 SIMU-04: Starting up the PIS Simulator devices

The requirement is that the PIS Simulator virtual devices can be started by the PIS simulator user. The acceptance criteria are that the user can log into the PIS Simulator virtual machine and start the PIS Simulator with a script or icon.

For the simulation to work there are a lot of processes that needs to be running and virtual connections that need to be established. These technical details should be hidden from the user and this requirement is for emphasizing this.

3.5 SIMU-05: Updating the PIS Simulator database

This requirement is for ensuring that the PIS Simulator database can be updated in a web browser via the PIS Dashboard Update Manager from the host computer. The acceptance criteria are that the user can connect with a web browser on the Host PC to the virtual DCU and PIS Dashboard update manager and install a new database.

Database testing is the main reason why the customer ordered the PIS Simulator, as the user might not be a database specialist, this functionality has to be easy and user friendly. This also verifies that the user created database can be installed to the rolling stock without problems as rolling stock updates uses the same application interfaces.

3.6 SIMU-06: Route simulation

The requirement is for the PIS Simulator to be able to simulate driving a route defined in the database. The acceptance criteria are that the user has an easy way to set the train on a desired route and simulate the drive.

Route simulation is necessary as it is the only way to trigger the events defined in the database. Most events like visual changes and audio announcements are triggered by waypoints on the route.

3.7 SIMU-07: Virtual TFT Display

The requirement is for the PIS Simulator to have a virtual TFT display, which can display the route information. The acceptance criteria are that a virtual TFT is visible and show the correct route information and events for the route that was selected and simulated.

This is a key feature of the PIS Simulation as the customer wants to verify that the events they defined in the database are displayed correctly. This is also the only way to visually verify that the defined route has all the waypoints and events defined.

3.8 SIMU-08: Audio Events

The requirement is for the PIS Simulator to play audio announcements. The acceptance criterion is that audio events are played and heard over host PC audio system.

This is a key feature of the PIS Simulation as the customer wants to verify that the audio events they defined in the database are played correctly.

3.9 SIMU-09: PIS Simulator User Manual

The requirement is for a user manual describing how to install and use the PIS Simulator. The acceptance criterion is a user manual provided with the PIS Simulator appliance containing the VirtualBox installation and configuration instructions, the PIS Simulator appliance import instructions and the PIS Simulator usage instructions.

Instructions on how to install and use the simulator as intended is needed to support the end user. With a manual there is also a reference point when problems occur during installation or usage of the PIS simulator that can be used for providing email or phone support to the end user.

3.10 SIMU-10: Update software

The requirement is that the simulator DCU software can be updated via INOI remote access or the PIS Maintenance Dashboard. The acceptance criterion is that the user can connect to the Virtual DCU and update the PIS software.

As issues are found and fixed with the PIS software, it would be beneficial to also update these to the PIS Simulator in the same way that the rolling stock is updated. However during development of the simulation environment it was found to be a very challenging point which will be discussed in detail in the implementation chapter.

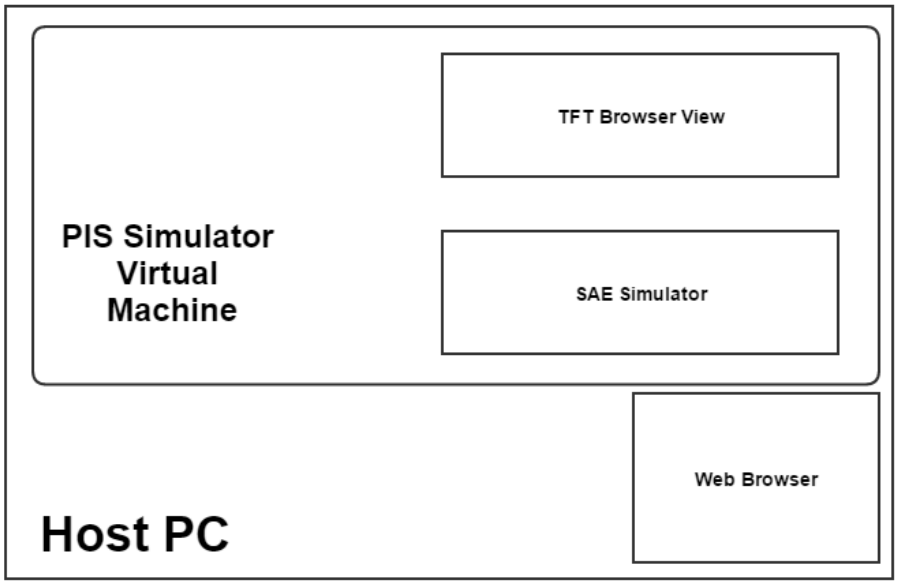
3.11 SIMU-11: Time synchronization

The requirement is for the PIS Simulator software to use the Host PC for synchronizing time from network time servers. The acceptance criterion is that the PIS Simulator and the virtual devices show the correct time.

The databases are for train driving schedules and having the correct time set to the environment is important for testing these date and time dependent schedules.

4 IMPLEMENTATION

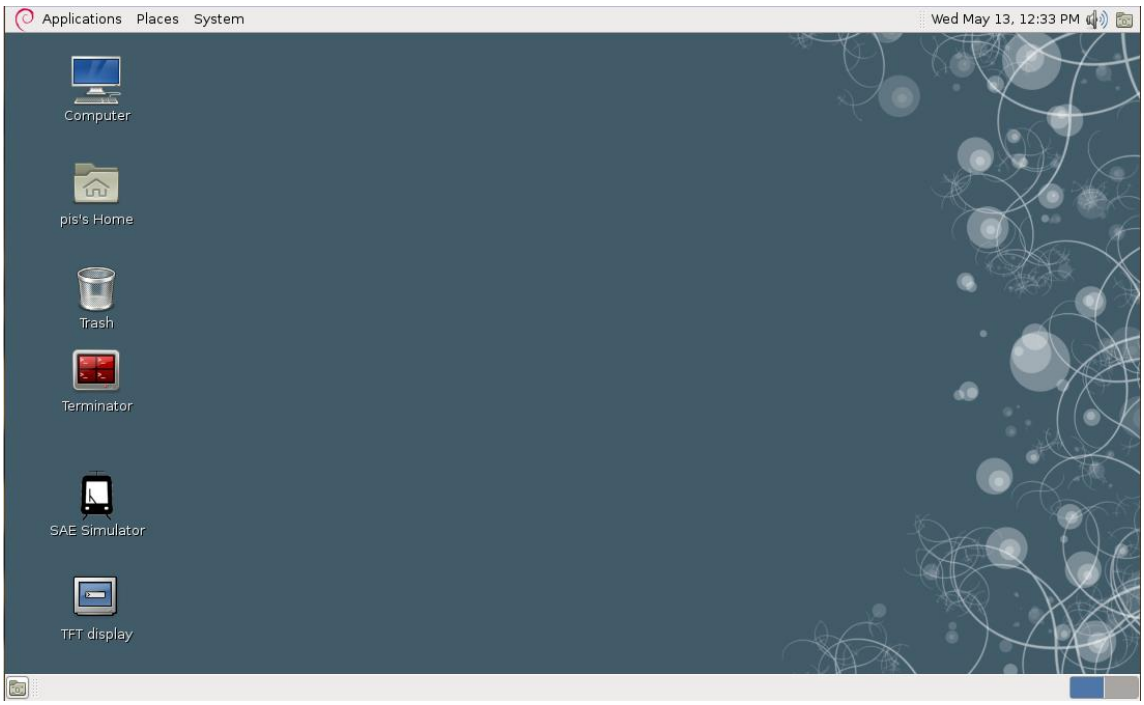
The initial architectural design of the PIS Simulator can be seen in Chapter 2. During the implementation phase of the project the design implemented can be seen in Picture 11 below. The following chapter will describe the changes made to the original design and also how some implementation challenges were solved.



PICTURE 11: Implemented PIS Simulator design

The PIS virtual machine is made up of a Debian Squeeze Linux distribution. Debian was a logical choice for the guest virtual machine as it is the Linux distribution used in Mitron on all the devices and by most developers for developing the passenger information system software. When the user starts the PIS Simulator virtual machine it is starting up the Debian Linux operating system and taken to the login screen. After successful login the user is able to work on the Gnome desktop.

Icons were added to the desktop to start the software related to the simulator. The actual PIS software processes are started during the Linux operating system boot up sequence. The user only has to start the actual simulation software after the login.



PICTURE 12: PIS Simulator virtual machine Debian desktop

4.1 Multiple virtual machines vs single virtual machine

In the original design it was decided that each device would be running in its own virtual machine. This was a logical approach as this simulates the individual devices separately like it would be in the real environment. This design soon ran into performance issues and it became clear that virtual machines running inside virtual machines are not optimal and that the hardware requirements for this will outweigh the usefulness.

During a demonstration of this solution, the simulator became totally deadlocked and it became the decision point that another approach would be needed.

It was decided not to virtualize each device anymore. This means that the main virtual machine is running all the passenger information system software in the background and as such become the guest virtual machine and the virtual DCU. In other words, the guest virtual machine is the DCU with a graphical user interface.

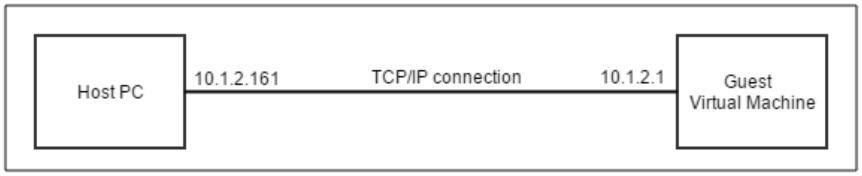
From a performance perspective this made a difference, as on a multiple processor host PC, a processor can be dedicated to the simulator and up to half of the memory of the host PC can be assigned to the guest virtual machine. In the original design the virtual resources are halved each time a virtual machine is added.

The decision to move to one virtual machine simulating all the devices also solved many other potential problems which are explained in this chapter.

4.2 Networking

VirtualBox include many types of different networking solutions. Networking the virtual devices was not as such a problem.

The difficult part would have been to transfer the simulation data from the host PC to the virtual DCU where it can be run from the SAE simulator used for simulating the routes. It needed a two-step upload process using sftp or scp to transfer the files into the guest virtual machine and then again using sftp or scp to transfer it to the virtual DCU. This solution is not very straight forward and user friendly.



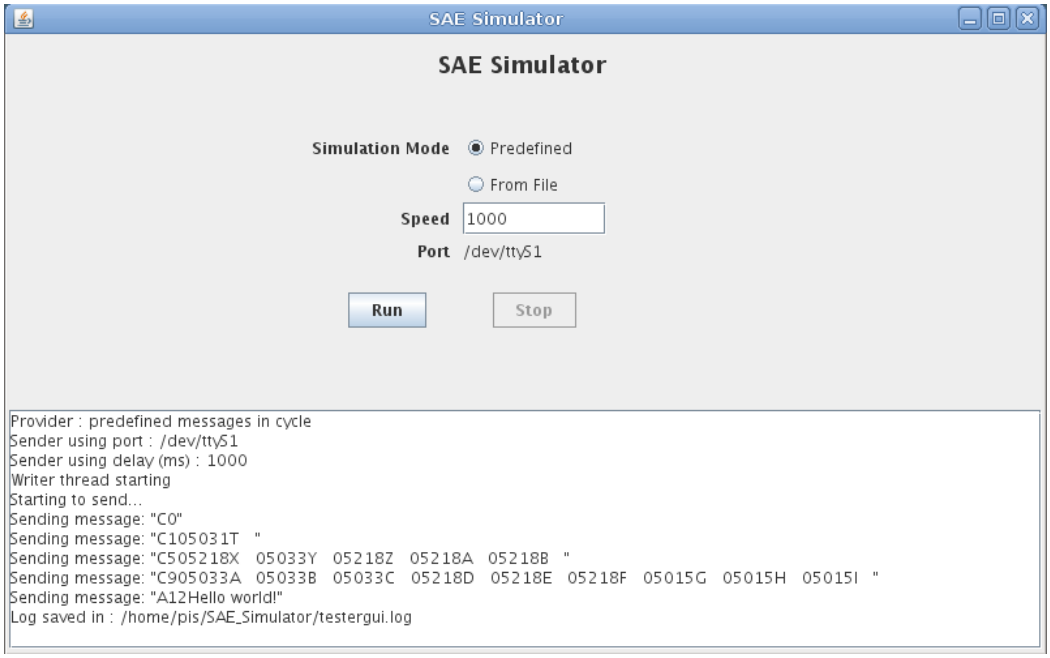
PICTURE 13: Host PC and guest virtual machine network connection

Moving to the one virtual machine solution indirectly solved this issue and using a normal FTP application like Filezilla can be used to transfer the data into the simulator environment with one step.

4.3 SAE Simulator

On the tram, there is a SAE vehicle bus communication device providing the passenger information system with route information and vehicle information. This device is connected to the Mitron passenger information system via a serial port on the train central unit. To simulate any driving on the PIS simulator, software simulating the SAE device would be needed. Fortunately during the development phase of the project, the project team developed such a tool that can be used from the command line.

Command line tools are not very user friendly for people who are not familiar with using it. As part of the PIS Simulator project a graphical user interface was developed for the tool as seen in Picture 14 in Java.



PICTURE 14: SAE Simulator user interface

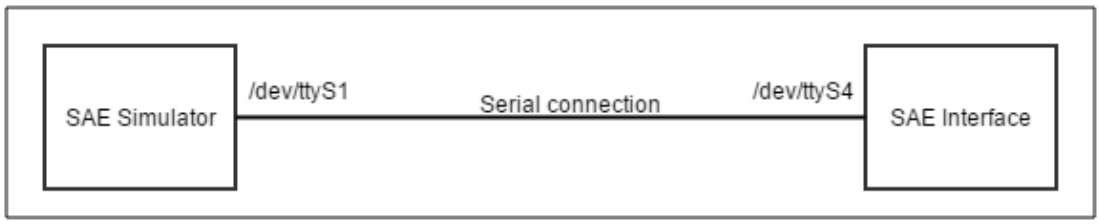
The tool can run a set of predefined test messages or a real route can be loaded from a file. The route information can be then played back at the desired speed. The tool displays the messages as they are read and sent to the PIS in the large text area. This provides a visual indication that the simulation is running.

4.4 Serial connection

Initially it was thought that the VirtualBox virtual serial devices could be used to connect the SAE Simulator to the SAE Interface software running on the virtual DCU. The connection between the main virtual machine and virtual DCU couldn't be established with only VirtualBox tools. It was needed to look for an alternative method to virtualize the serial connection and the communication between the two ports.

A Linux command line tool called socat was found. Socat is a command line based utility that establishes two bidirectional byte streams and transfers data between them. The following command would link the serial 1 port to the serial 4 port on the PIS simulator:

```
socat -d pty,raw,echo=0,link=/dev/ttyS1 pty,raw,echo=0,link=/dev/ttyS4
```



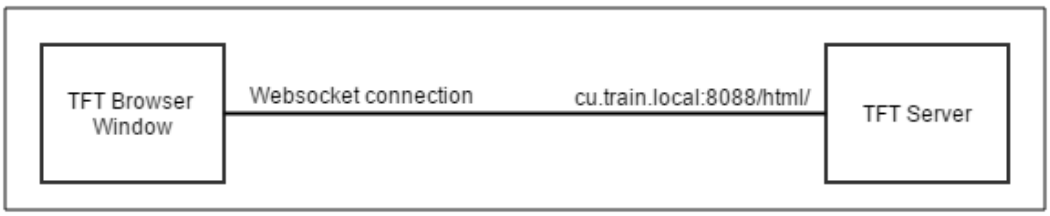
PICTURE 15: Serial connection between the SAE simulator and interface software

It was decided then that the SAE simulator would run on the virtual DCU so that the serial ports can be connected with socat and the SAE simulator X-window would be forwarded to the VirtualBox guest window system. This solution worked but the X-window was at times very slow and unresponsive.

Moving to a single virtual machine environment solved also the X-window forwarding performance. X-window forwarding became unnecessary when the SAE Simulator and DCU software are running in the same environment. The serial ports are still connected with socat and it has proofed to be an excellent tool for this purpose.

4.5 TFT View

The TFT views are essentially SVG generated graphics that are displayed in a Chrome browser running in kiosk mode. The TFT connects to a websocket server application that is running on the DCU and receive commands to refresh the view from the server.



PICTURE 16: Websocket connection for TFT route display render

The same views could be opened by opening the Chrome browser on the virtual machine guest and connecting to the websocket server address. This essentially eliminated the need for running the TFT display in its own virtual machine as it would add no value.

PICTURE 17: TFT Route view

In the simulator the TFT cannot run in full screen kiosk mode as it will cover the simulator application as well as be in the wrong display size. A Chrome application link was added to the desktop that starts the Chrome application with the correctly scaled dimensions.

PICTURE 18: TFT Station arrival view

4.6 Audio

Mitron developed IP amplifiers that listen on an internet multicast port for an audio stream and play the stream directly over the loudspeakers. The stream is played with Mitron announcement software and this is the standard solution. After investigation it was found that there was no way to change the destination of the audio stream with any configuration file or startup parameters. As the PIS Simulator doesn't include any IP amplifiers and simulating the amplifier would be difficult, a change to the announcement software was needed.

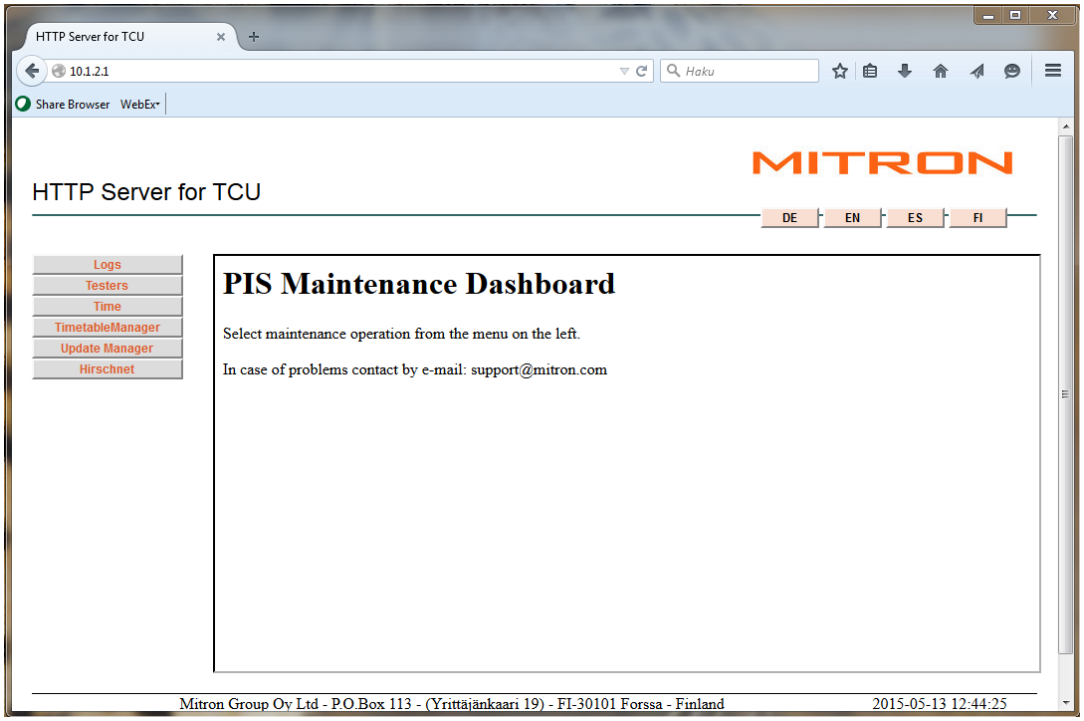
The announcement software was changed to play the audio stream to the ALSA player of the guest virtual machine. Using the ALSA player allowed for the audio be played

directly in the guest virtual machine audio system which is connected by VirtualBox to the host PC audio system. This change allowed for the announcements to be audible and verifiable.

Finding this solution has been beneficial, this version of the announcement software has also been used already in another project where some audio verification was needed without the use of all the passenger information system test devices.

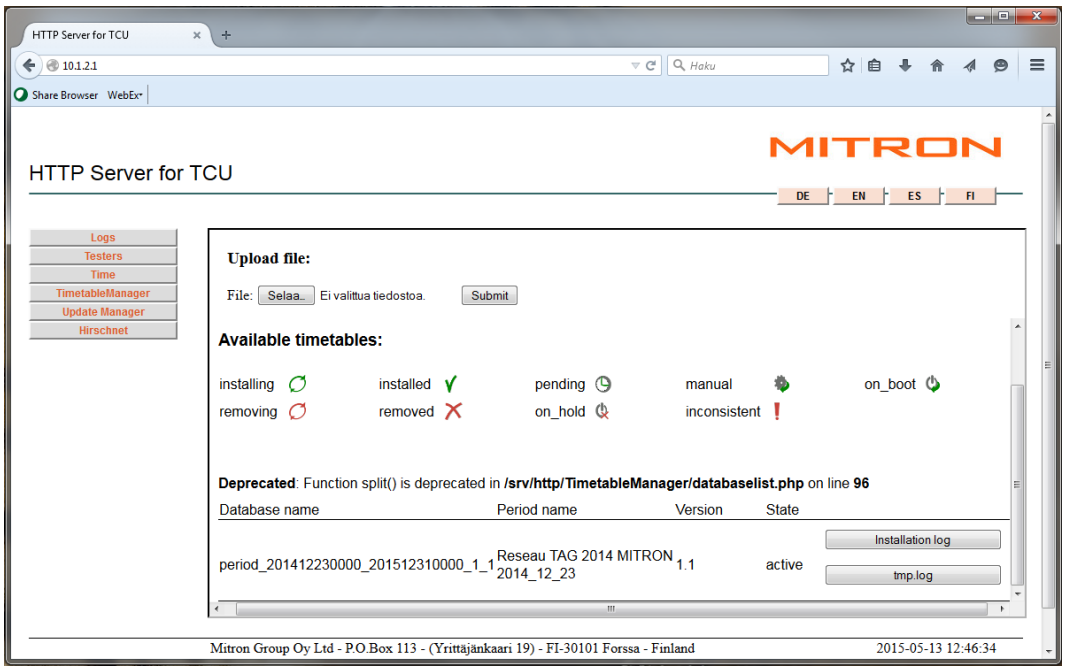
4.7 Updates

The DCU has a web application running for doing some of the maintenance tasks of the passenger information system. This is called the PIS Maintenance Dashboard. From the dashboard the user can download log files, activate test sounds and views, set the time of the passenger information system, update timetables, update passenger information system software and configure the network.



PICTURE 19: PIS Maintenance Dashboard

The timetable database updates via the maintenance dashboard work as planned and designed. The timetable databases is updated and managed via the timetable manager software on the DCU.



PICTURE 20: Timetable Manager

Updating the passenger information system software turned out to be more challenging and this requirement could not be met by the PIS Simulator. There are two issues that causes the passenger information system software installation to fail when updating the software with a standard project generated software update package.

The Mitron DCU hardware has a Field-programmable gate array (FPGA) device embedded. The update package contains software and drivers for this FPGA device. The installation of the FPGA software and drivers fails as there is no FPGA device in the virtual DCU. The result is that the installation can either froze while searching for the nonexistent FPGA device or then the update continues but will always be seen as failed on the update manager as the FPGA device update scripts always fails. The FPGA device doesn't control anything essential when doing route simulation. On the Mitron devices the FPGA is running for example a watchdog script that reboot the device after a set time when no network communication are detected. Additionally, on the TFT displays the FPGA is used for example to set the backlight of the display on and off.

The project generated software update package for the DCU also contains the original announcement software that stream the audio to the IP amplifiers. When this update package is installed, it breaks the audio solution mentioned in chapter 4.6.

It was considered to change the DCU update scripts to recognize when being installed to the simulator or to the actual DCU hardware. The risk of something not installing on the real environment because of these additions was considered not worth taking at this stage of the project. As the project is now in maintenance phase, it was decided that Mitron will update the simulator software to the latest releases version of the software and release a new version of the PIS_Simulator.ova when needed.

Generating a simulator specific update package was also considered. It was decided that a simulator specific update package that can be pushed by the remote update system to the trams by accident was also seen as an unnecessary risk. It was therefore accepted by the customer that this requirement would not be met and that Mitron will always provide an updated simulator.

4.8 Acceptance testing

After the design concept of the simulator was given to the customer, the actual functional specification was written. Below is an example of how the document is written.

4.5 SIMU-05: Updating the PIS Simulator database

ID	SIMU-05	Priority	MUST
Description	The PIS Simulator database can be updated in a web browser via Update Manager via own computer.		
Acceptance Criteria	1. Open web browser on Host PC. 2. Open a page to the address of the Virtual DCU (10.1.2.1) 3. Update Manager web page is available for use. 4. Database update can be uploaded and installed to the Virtual DCU.		

PICTURE 21: Functional specification excerpt

When the functional specification was written, the acceptance test criteria for a functionality was described and given to the customer. The customer approved the functionality and the acceptance criteria.

During the implementation this acceptance criteria was used to test if the functionality is implemented and completed.

There was no automated functional or acceptance testing in the simulation project.

No other testing was done in relation to the actual specific project PIS software. The project software release was installed and used in the simulator with the knowledge that the software release has passed the testing and quality control of the project team.

5 DISCUSSION

A whole simulator for a project in Mitron haven't been done like this before. This was the first attempt where a large part of the passenger information system functionality has been tried to be simulated on one machine and where the end product was delivered to the end customer as a product. This chapter will discuss how this simulator project can be used and further developed in the future.

5.1 Simulation

PIS simulation haven't been ever seriously considered or developed in Mitron because normally the software development teams have access to equipment racks with the actual hardware. Most of the development and testing are done directly with the hardware and this should not be changed. Parts of the passenger information system functionality have indeed been simulated with software, for example route simulators that send GPS coordinates and odometer readings to the passenger information system for simulating driving but not the actual devices like the train central units, displays or audio.

The PIS Simulator as a tool for the end customer to test the database events on, is a perfect use for such a simulator. The customer has been using the simulator since May 2015 and has filed no issues or complaints with the project team.

Although this PIS Simulator are working perfectly in this setup, it is missing two essential parts when it comes to other Mitron projects: driver control panels and LED displays.

Most projects will have some kind of user interface that a driver or conductor uses for controlling the PIS. In future simulation projects, the best way will have to be found for simulating the driver control panels and making it part of the PIS Simulator. The driver control panel normally has its own IP address that is also used for uniquely identifying the device. This means that the driver control panel will somehow have to be in its own virtual machine or environment where a unique IP address can be assigned to it. Placing the driver control panel in a Linux Container on the virtual machine guest might be the

best way to resolve this without having to add the overhead of another virtual machine inside a virtual machine.

Many projects have LED displays on the front and side of the trains and some projects might also have LED displays inside the passenger areas. It could be beneficial to have some textual or visual representation of what are visible on the displays during a drive simulation. This would require some software development, as currently such a tool does not exist. The LED display handling in Mitron is currently under heavy development, the developers are adding a testing application interface that will allow for a simulation tool to read at least the display values for verification. Making simulation software that creates a pixel for pixel visual render of the LED display is an option but it would not essentially add much value as the more important part would be to know the text that are set to be shown.

5.2 Software development

Many developers in Mitron have used a virtual DCU as part of their software development process. This normally consists of only a VirtualBox virtual machine running the DCU Linux base image with the specific project software installed. This normally runs on the developers' machine where the NetBeans IDE can connect to the software processes running on the DCU for debugging.

A software development environment which includes a PIS Simulator could add some value to the software development process. It can provide a quick way to see if a software change work as expected right through the whole PIS. It can provide a way for developers to work remotely when needed, as the developers are not as dependent on the hardware. It can also provide a test environment for the developer while onsite at a customer for testing changes before installing to the rolling stock.

As software development for projects come to an end, the hardware test devices are released for use in other projects which are in the development phase. The PIS simulator of a project can also provide a quick way to investigate issues reported by customers after the software development phase has ended, without having to immediately rebuild a hardware based test system.

5.3 Testing

The area where the PIS Simulator probably has the most potential is in the testing area. The PIS Simulator could be developed further and setup into each project as an automated test tool and part of the continuous integration or nightly build processes. It can also be used for integration testing to insure that the software processes are working together like intended.

With test tools like the Robot and Sikuli framework, automated tests can be run and various ways used to verify that the basic areas of the software works and continue to work after changes were made by the development team. For example, the simulator can be set to a route and the Sikuli framework can be used to compare images of the TFT display and verify that these are updated. The Robot framework can check the log outputs for the various processes for inconsistencies and errors.

A benefit of using the VirtualBox simulator would be that when the simulator is set up and working for the project, a snapshot can be taken and before the tests are run, the virtual machine can be reverted to the clean snapshot environment. This ensures that only the correct and desired software are tested.

6 CONCLUSION

The PIS Simulator project was a very interesting project. From a customer perspective, based on the fact that the PIS Simulator was delivered in May 2015 and the customer haven't filed any issues or complaints with Mitron until now, it could be considered a success.

From a company perspective, the PIS Simulator project is a success. It showed the potential of such a simulator not only as a development and testing tool but also as a Mitron product to add to the Mitron product line for the end customers to purchase and use alongside the PIS installations on the rolling stock.

From a personal development perspective, the initial plan was to use Linux containers for virtualization. As the learning curve was found to be too steep for the time frame in which the customer desired the solution, it was decided to stick to VirtualBox as the only virtualization solution. Although Linux containers were not used there was enough challenges and issues to solve and personal development and learning most definitely happened. Discovering tools like socat and the limitations of virtual machines inside virtual machines might be also beneficial knowledge for future projects.

REFERENCES

TFS-26137-170-036 Proposition Simulateur

1475 - PIS Simulator Functional and Technical Description

1475 - PIS Simulator User Manual