Intelligent Verification/Validation for XR Based Systems

Research and Innovation Action

Grant agreement no.: 856716

D5.1 – Basic Integration of the Pilots

iv4XR – WP5 – D5.1

Version 1.6

September 2020

Project Reference	EU H2020-ICT-2018-3 - 856716
Due Date	30/09/2020
Actual Date	30/09/2020
Document Author/s	Joseph Davidson, Premysl Paska (GA), Ian Saunter, Jason Lander (GWE), Jean-Yves Donnart (THA-AVS), Rui Prada (INESC-ID)
Version	1.6
Dissemination level	Public
Status	Final

This project has received funding from the European Union's Horizon 2020 Research and innovation programme under grant agreement No 856716

Document Version Control			
Version	**Date**	**Change Made (and if appropriate reason for change)**	**Initials of Commentator(s) or Author(s)**
1.0	30/04/2020	Initial document structure and contents	JD
1.1	04/09/2020	GWE report included	JL, IS
1.2	10/09/2020	GA report started, Intro expanded on	JD
1.3	14/09/2020	GA report finished	JD, PP
1.4	16/09/2020	THA-AVS report added, conclusion and exec summary written	JYD, JD
1.5	29/09/2020	Comments addressed, conclusion expanded	JD
1.6	30/09/2020	Final comments addressed and general polish	JD, JL, IS, JYD

Document Quality Control			
Version QA	**Date**	**Comments (and if appropriate reason for change)**	**Initials of QA Person**
1.0	30/10/2019	Comments on the initial structure	RP
1.4	26/09/2020	Request some details	RP
1.6	30/09/2020	Final review	JD

Document Authors and Quality Assurance Checks		
Author Initials	**Name of Author**	**Institution**
JD	Joseph Davidson	GA
IS	Ian Saunter	GWE
JL	Jason Lander	GWE
PP	Premsyl Paska	GA
JYD	Jean-Yves Donnart	THA-AVS

RP	Rui Prada	INESC-ID

TABLE OF CONTENTS

1 EXECUTIVE SUMMARY

This report details the progress of the integration of the iv4XR framework with the pilots supplied by the industrial partners. It explains what basic integration entails for each of the pilots, presents an overview of the technical details concerning each pilots' connection to the framework, and finally describes the current integration status and framework capabilities for each pilot.

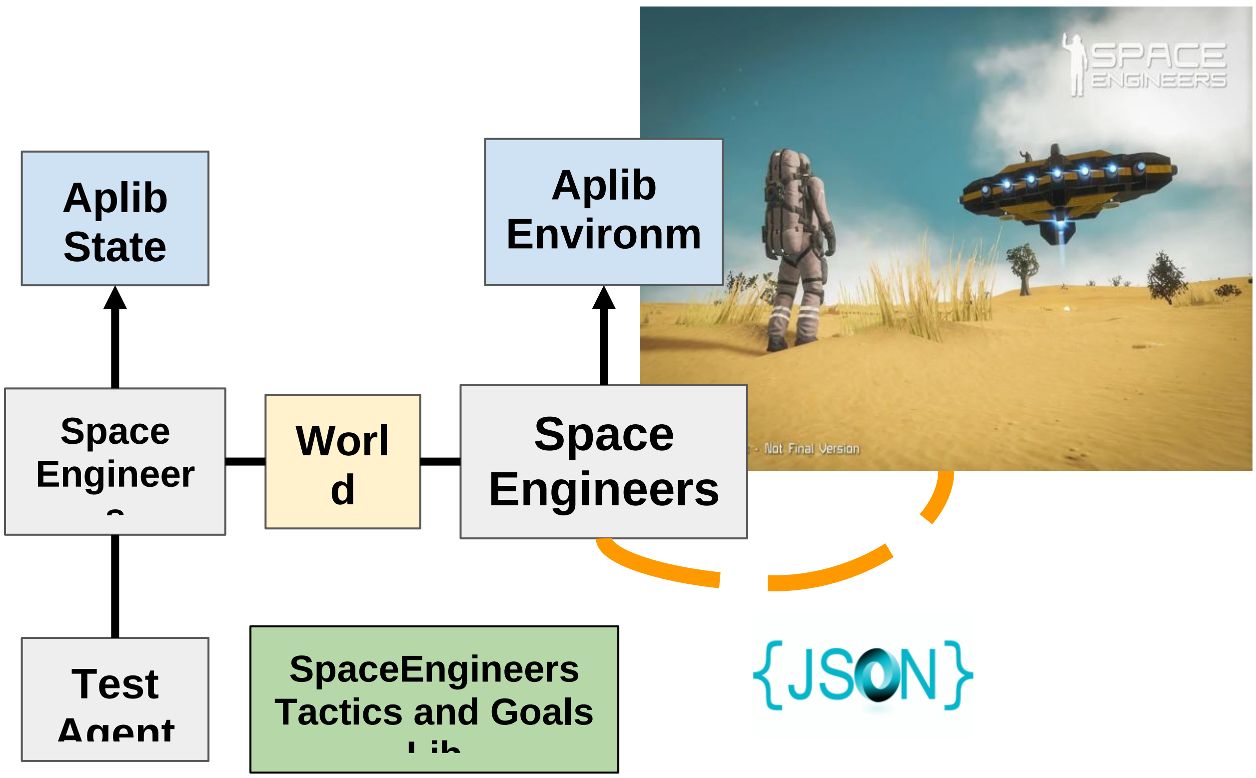
Basic integration for each of the pilots broadly consists of the definition and implementation of the coupling interface which can connect to the iv4XR framework. For two of the pilots, integration with iv4XR will allow it to control agents within the pilot to perform specified tasks, so a two-way communication method has been implemented to transmit information to the framework and receive commands from it.

All of the pilots can open a connection with the iv4XR system and transmit data. The integration process is continuing at each of the partners according to the original plan while also incorporating feedback from the rest of the consortium.

2 Introduction

One of the objectives of the iv4XR project is to permit external organisations to use the framework so that their extended reality environments can be tested or monitored with less human interaction than is required by the testing methods of today. For adoption to be effective, prospective developers need 1.) Some demonstration of the benefits of using iv4XR, and 2.) A measure of guidance on how to integrate the framework into their development lifecycle. The pilots are one of the methods that the consortium is using to deliver this knowledge.

There are three integration deliverables in this project, of which this is the first. The work in this deliverable consists of modifying the pilot systems to communicate with the iv4XR framework and connecting them together, allowing for information to flow between the two systems.



Block Diagram of how the iv4XR system would interface with Space Engineers

3 PILOTS

There are three pilots upon which the capabilities of the iv4XR framework are to be demonstrated: 1.) Space Engineers, a 3D game. 2.) A simulation scenario where a Nuclear power plant is being infiltrated by hostile forces. 3.) LiveSite, a building and infrastructure management system.

3.1 SPACE ENGINEERS (GOOD AI)

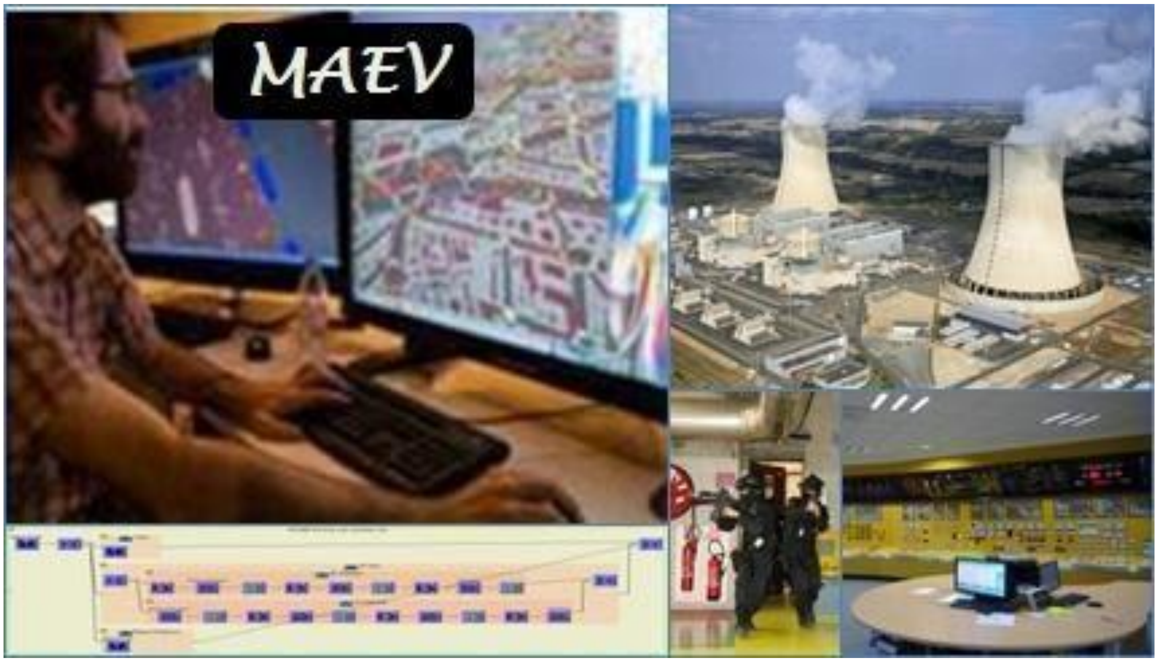
Space Engineers (SE) is a complex open world game with volumetric physics developed by Keen Software House which is a sister company of GoodAI (GA). In a world with volumetric physics, objects behave like real physical objects with mass, inertia and velocity, allowing a more realistic simulation of the physics of the real world. In SE players can build any object from blocks and the construction works just like if it would in the real world in a similar fashion to LEGO Technic.[1]


Integration with iv4XR is intended to streamline and improve the testing process for both in-development features and for pre-release testing. The current testing process consists of a team of testers who manually test the functionality and visuals of aspects of the game, whether it be tools, the environment, or the construction materials available to the player. Given the complexity of the game, there are more than 10,000 tests which have to be performed for each major release of the game.

3.2 NUCLEAR PLANT INTRUSION SIMULATION (THALES AVS)

THALES AVS develops very sophisticated simulation systems that involve not only virtual representations of real environments but also the simulation of virtual actors in addition to the participation of real actors (trainees and/or animators). The animation of these virtual actors is realized by specific components called CGE (Computer Generated Entities) for civil applications and CGF (Computer Generated Forces) for military applications that allow, thanks to Artificial Intelligence algorithms, the construction of very complex scenarios.

[1] Lego Technic is a type of Lego set which incorporates mechanical components such as gearing and electric motors. Sets are typically of vehicles and mechanical structures such as windmills or waterwheels, and such items constructed in technic exhibit these mechanisms. (http://www.technicopedia.com/1982.html)

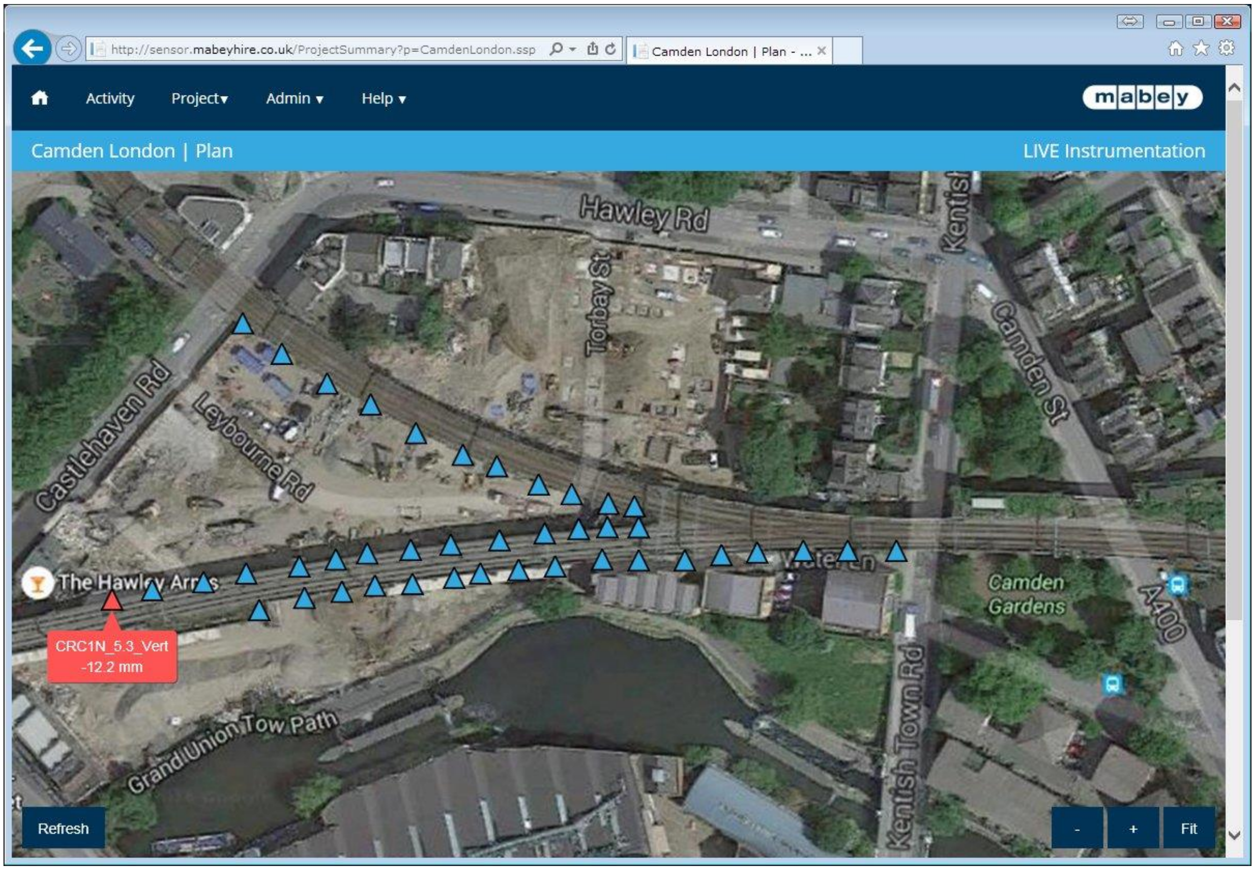


In this pilot we want to study how iv4XR can improve the current process simulation scenario verification of our CGE MAEV by:

1. Automating the process, hence reducing the amount of manual work,
2. Improving the quality of the verification by delivering more test coverage,
3. Extending the scope of the verification by including UX assessment which is currently lacking due to the absence of a feasible methodology for doing it.

The pilot involves the simulation of an attack by intruders trying to gain access to a nuclear plant. It is a study simulation whose objective is to assess the defense strategy of the plant. This defense strategy will be implemented in the simulation by a CGE scenario that includes the positioning of a set of fixed sensors disseminated into the infrastructure and the assignment of missions to the defense unity groups (static position, uniform rotation, patrol on a circuit, etc.).

3.3 LiveSite (Gameware)



Livesite is a real-time instrumentation and monitoring system for the building industry.

Sensors are installed on building and construction sites to monitor ground movements, building movements, temperature variations, rainfall, water levels, vibrations, dust and other environmental factors to ensure safety of both the site being worked on, and the neighbouring area and structures.

The various sensors connect to network hubs and the internet directly and upload readings to the Livesite servers. Readings can be uploaded 100s of times per second in some instances, and a large volume of data is produced.

The Livesite servers analyse and process the data, providing real-time graphs, numerical tables, and 3D visualisations of the sensor readings, allowing engineers to monitor the site activity and behaviours.

Ensuring the integrity and accuracy of the readings is critical to providing a safe building environment and prevent works or ground issues from causing problems with the structures involved. Currently, we can monitor a single stream of data with some set of heuristics and alert the managers should the heuristics be triggered. The iv4XR framework will assist in this by hosting agents which are able to correlate multiple streams of data from a diverse set of sensors and learn rules for alerting engineers.

4 INTEGRATION DEFINITION

Given the differing natures of each pilot, the term “Basic Integration” means different things. This section will describe what basic integration entails for each of the individual pilots.

4.1 SPACE ENGINEERS (GOOD AI)

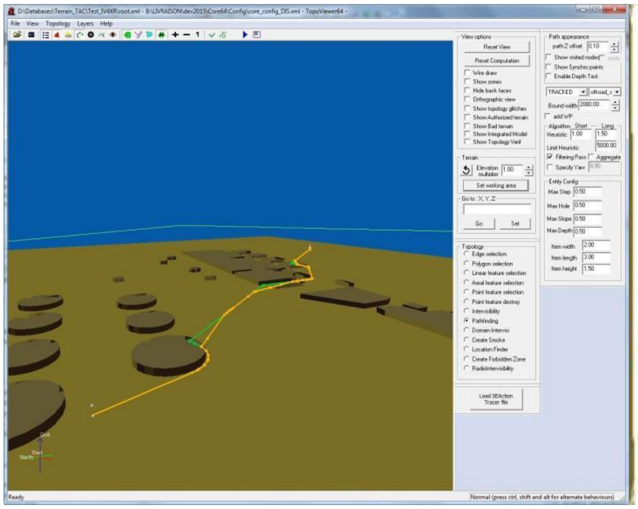
The main objective of this phase was to design and test high-level architecture of integration of the Space Engineers game with the framework. The integration enables testing of the game using the framework.

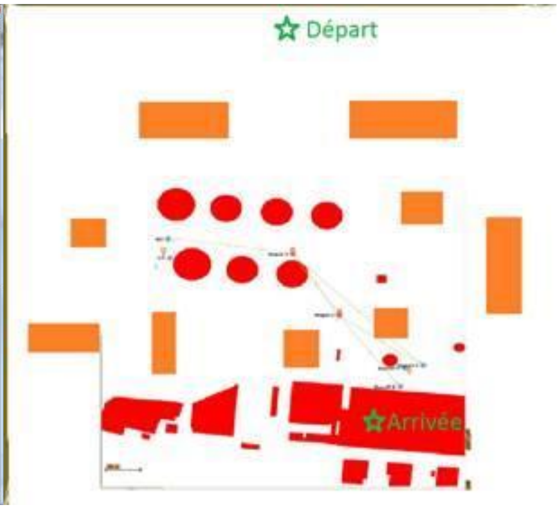
When performing tests on the game, the tester is embodied in the game as one of the titular space engineers. So they are given the same control as any other player and use this control to conduct their tests. Therefore, the agents who will be testing the game will also be embodied as engineers in the game environment. To facilitate this, integration for Space Engineers consists of allowing the iv4XR system to connect to the game through an interface which publishes and transmits the player control commands to the game.

As a result the interface, observations, and the ability to transmit movement commands to the game so that the character movement can be controlled by an iv4XR agent.

4.2 NUCLEAR PLANT INTRUSION SIMULATION (THALES AVS)

Today, the verification of a simulation with artificial agents (non-player characters) often relies on human operators which play the opposite force (e.g. the intruders) with the objective to challenge the scenario integrated into the simulation (the defense of the nuclear plant).
Because this task is cost consuming, we sometimes use the scenario functionalities of the CGE to model the different strategies of the opposite force and analyze, afterwards, the results of the different confrontations. In most cases, this kind of verification is not exhaustive enough to be efficient.
Our main objective is iv4XR Project is to test an alternative solution for scenario verification where AI agents can learn how to challenge the CGE scenario.
These AI agents will have access to all the necessary information (the perceived situation of each intruder) and will be allowed to decide which actions will be performed by their corresponding artificial agents into the simulation.





In terms of architecture integration, it means that:

- We need to define an API plugin that will allow the CGE to communicate with AI agents, whatever AI technology they rely on,
- The communication between the AI agent module and the CGE should be performed through a generic framework (e.g. iv4XR framework) in order to ensure that both modules could be replaced in the future by, respectively, another AI module or another simulation tool.

The CGE should be able to run the simulation much quicker than the real time if we want to exploit the efficient discovery capacities of the Machine Learning algorithms.

4.3 LIVESITE (GAMEWARE)

For IV4XR, we have set up an additional server (API Server) which can access readings from certain Livesite projects and provides a web-API to access and parse the readings and their associated meta-information.

The meta-information accompanying a project includes a wide variety of parameters that can be used to assist in verification of the validity of the readings themselves.

For instance the metadata for a sensor will include what type of sensor it is, and where it is installed.

It can then be deduced that a temperature sensor on the outside of a building for example, should never give a temperature reading significantly higher than the ambient temperature of the area on that particular day.

The API Server can provide the readings from sensors with transformations and filtering applied, allowing the IV4XR framework to detect inconsistencies or problems.

The objective is to allow the IV4XR framework to be able to access and analyse readings from Livesite projects, and deduce, in the first instance, any obvious errors or omissions, providing an automatic method of reading verification for what is an important aspect of large buildings works.

5 Connecting the Pilots

5.1 Space Engineers (Good AI)

The game provides a plugin API which allows access to some internal data structures without modifying the source code of the game itself. The selected design takes advantage of this plugin support to provide network API for the iv4XR framework. (Notably a game “mod” would not be as versatile as a plugin because the Mod API is limited and mods do not allow binary dependencies.)

We have developed a basic version of the Space Engineers plugin and verified it provides sufficient access to the internal state of the game to be able to provide the API for the testing framework.

The plugin now supports basic observation of the agent’s surroundings and control of the agent’s movement.

The source code of the plugin along with a testing binary release can be found in the project’s GitHub page: https://github.com/iv4xr-project/iv4xr-se-plugin

API

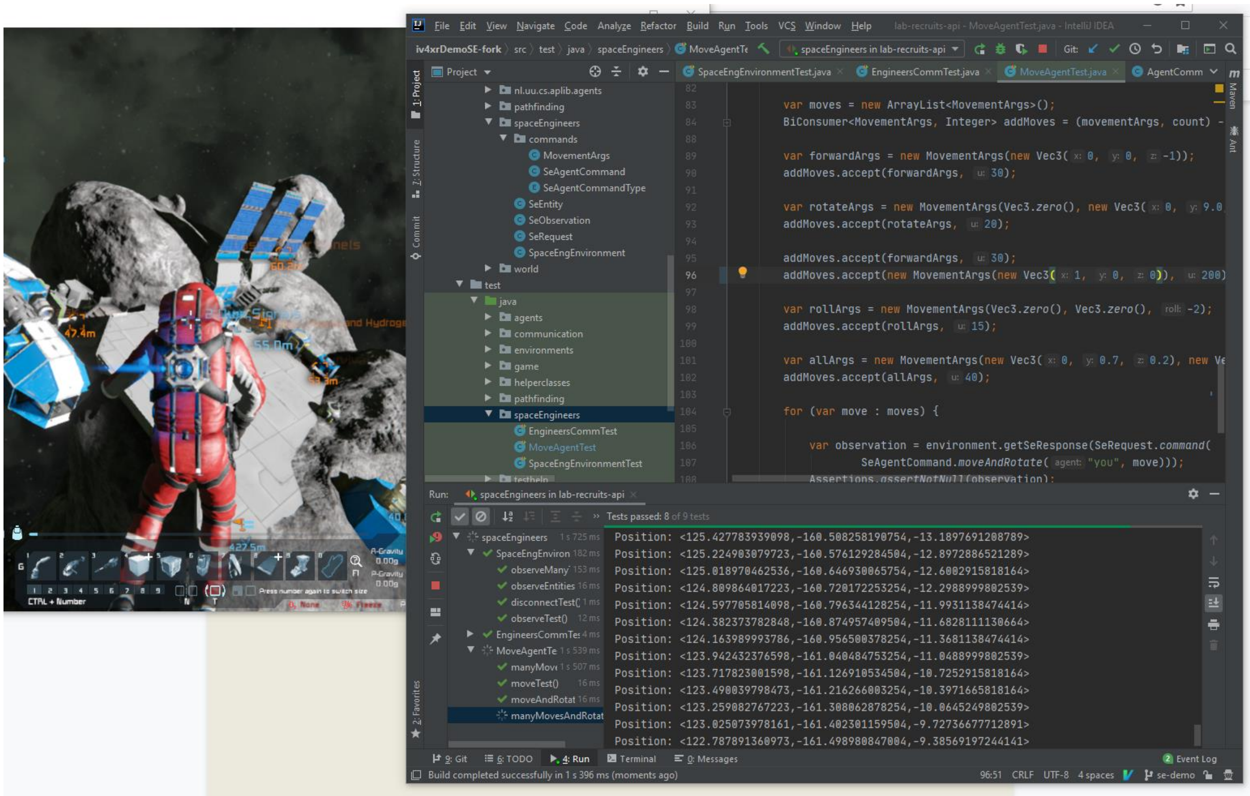
The network protocol is based on the Lab Recruits demo. It consists of JSON commands split by newlines. The protocol may still change if it proves necessary in the subsequent development stages.

Currently implemented commands:

- Observe – basic version; returns list of entities in the agent's surrounding and their location.
- MoveAndRotate – allows to move and and rotate the agent in all directions.
- Disconnect

There's a Java project derived from the Lab Recruits demo that contains a demo client in the form of unit tests. The repository is also available at the project's GitHub page.

On the screenshot below, you can see a Java unit test using the iv4XR framework prototype to control the agent in Space Engineers. In the test output pane in the bottom part, you can see agent position updates received from the game.



5.2 NUCLEAR PLANT INTRUSION SIMULATION (THALES AVS)

The basic integration phase consists in defining a first version of the API plugin and in developing the communication modules both in the CGE and in the framework.

In the CGE, we already have a communication module, developed in C++ and called AIEngine, which allows the transfer of basic commands to CGE agents. For the project, we are enhancing this module in order to allow an external module to:

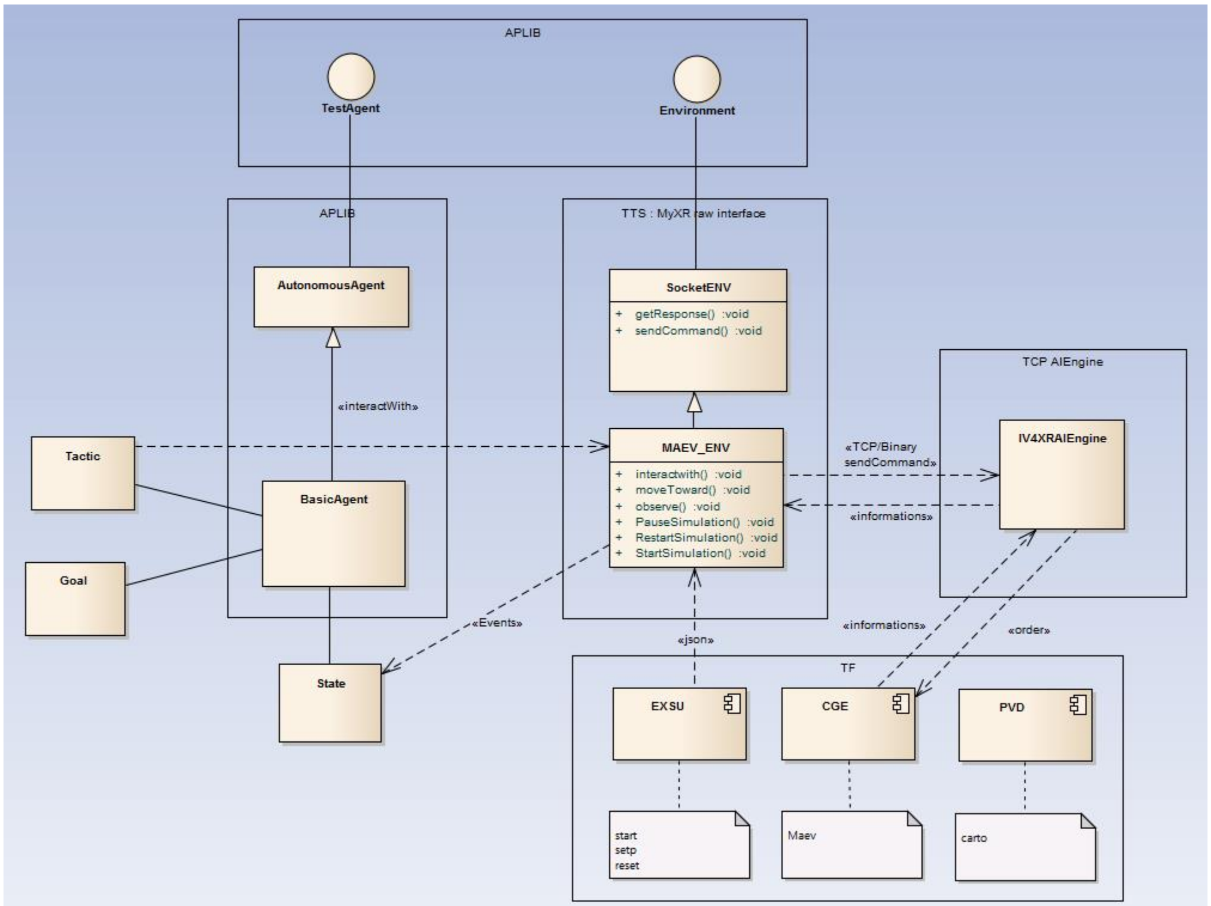
- Control the agents by giving orders as if they were played by an human operator,
- Get access to the state, memory and perceptions of each controlled agent.

To make the connection with the platform, our approach was the following:

- Implementing specific classes derived from the AppLib (iv4XR framework) high-level classes,
- Setting the sockets that will allow the "myXR raw interface" module of the framework to communicate with the CGE (communication with the AI Engine plugin through a TCP socket, similar to how the LabRecruit connects with Unity).

- Test and validate the interactions through the framework's interface (the Reinforcement Learning library from Thales SIX will be used to evaluate the whole performance of the architecture).

The next figure presents a detailed view of the architecture implemented during the basic integration.

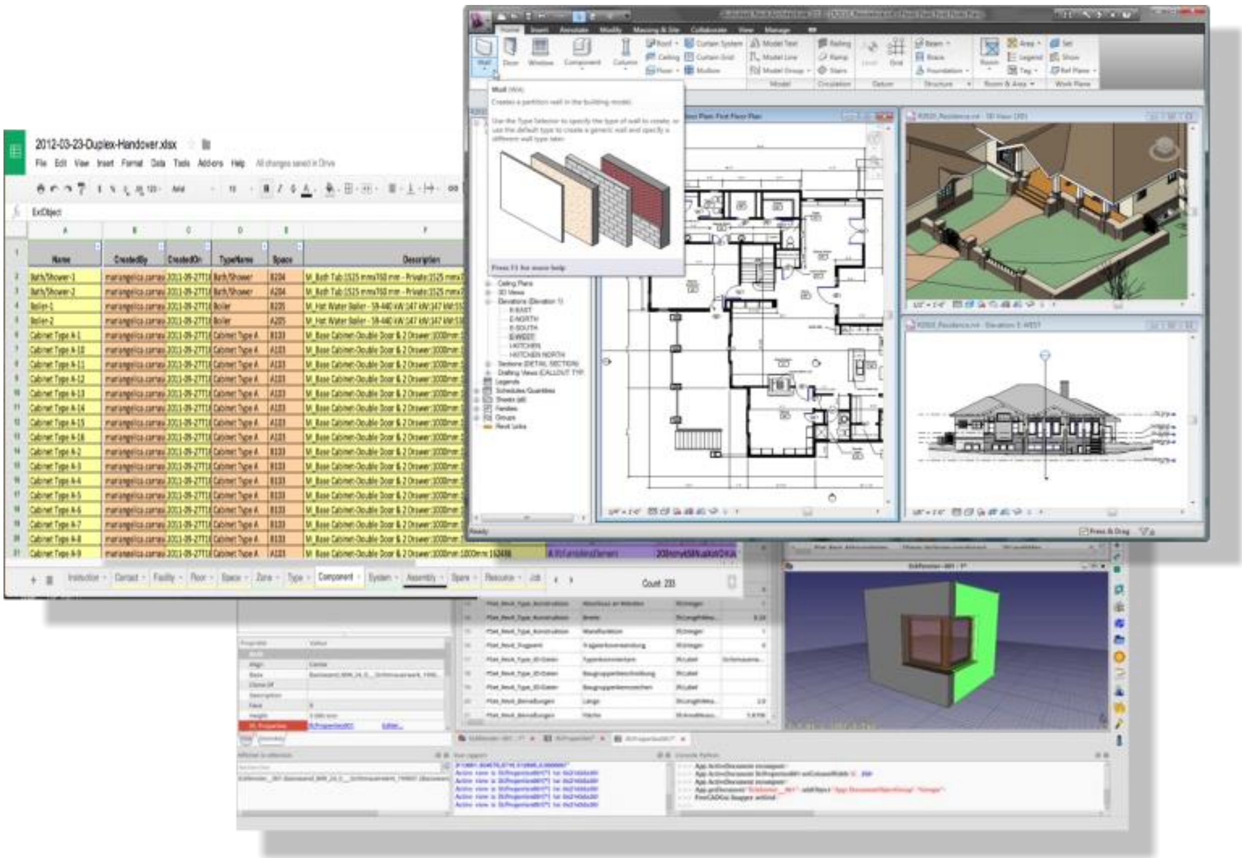


Note that "myXR raw interface" mainly uses two classes SocketENV and MAEV_ENV.

SocketENV contains two important functions: sendCommand() to send commands such as "Move to" orders from the Agent to the environment and getReponse() to receive information about the updated world such as entities position and detections. MAEV_ENV contains several interaction functions and, in particular, StartSimulation(), PauseSimulation() and RestartSimulation() to control the exercise driven by our EXSU component. MAEV_ENV is also responsible for the communication, through the TCP socket, with iv4XRAIEngine which is the C++ interface that will allow MAEV to communicate with the framework.

We still have to implement the world entities in the module by using the WOM of iv4XR framework (as seen in the iv4XR demo). And the code of the plugin will soon be available on the project's GitHub page: (https://github.com/iv4xr-project).

5.3 LiveSite (Gameware)



For basic integration, the objective is to have the IV4XR framework detect some obvious errors automatically. Such as values out of range or missing readings.

The API server can now be used to retrieve tables of readings with meta-information (including BIM related fields) from Livesite projects which can then be analysed by the IV4XR framework.

The meta and readings can be downloaded in JSON, XML and raw formats.

We are currently working on visualisation capabilities for the API server, so it can display in a visual way the data with the required transformations and filtering applied, and highlight the errors as reported by the IV4XR framework.

The API Server provides a Web API interface returning and using JSON format data, the documentation for the Web API is here http://217.174.253.215/APIDocs

6 CONCLUSIONS

This report detailed the efforts of GoodAI, Gameware, and Thales-AVS to start the process of integrating the iv4XR framework into their chosen pilot software. This deliverable mainly focused on the development of the coupling interfaces and their architecture which can be built upon in subsequent deliverables to expose the functionalities of the pilots to the agents hosted on the framework.

Each of the pilots can currently connect with the iv4XR framework and exchange information. With feedback from the rest of the consortium, the industrial partners will continue to integrate their pilots with the framework.

GoodAI will expand the repertoire of actions that the agent can perform and will increase the scope and detail of the objects in the game space that are observed and convert them into the generic World Object Model format. Thales-AVS will continue to connect MAEV to the framework to bring it up to a state suitable for the use by the academic partners. They will also modify the simulation to permit the training of reinforcement learning agents. Gamewares integration will proceed with adding visualisation tools, and in allowing iv4XR agents more access to the datastreams from areas managed by Livesite so that iv4XR agents can detect non-trivial errors indicated by the interactions of the various sensors and their data.

This deliverable marks the first stage of integration with the iv4XR framework. For each partner, it permits basic interaction between agents hosted by the framework and the pilot itself. Integration will continue to add more capabilities for the agents and will incorporate feedback from the rest of the consortium as they begin to make use of the systems.