Design of Low Cost PC-based Simulators for Education and Training Purpose Using DDS

Hakiri Akram[1,2], Berthou Pascal[1,2], Gayraud Thierry[1,2]
[1]CNRS; LAAS, 7, avenue du Colonel Roche, 31077 Toulouse, France
[2]Université de Toulouse; UPS, INSA, INP, ISAE; LAAS; F-31077 Toulouse, France
Email: {Hakiri, Berthou, Gayraud}@laas.fr

***Abstract*—The use of simulation in training and education enables to prepare personal in realistic environment. But the cost and the complexity to create and reuse simulations often limits their application. In this paper we investigate a low cost and high fidelity PC-based simulator based on Data Distribution Service (DDS) middleware. The main parts of the systems and the architecture, including the hardware and the software are introduced. Real-time networking between distributed simulators is achieved using a reliable distributed communication, which employs publish-subscribe middleware build using OMG-DDS. Result shows these methods could produce low cost, extensible, reliable and distributed simulators.**

I. Introduction

We have seen an inordinate amount of attention on novel 3D input devices, e.g. data gloves, data suits, space balls, "flying mice", trackers, etc. We have also seen the same level of attention on novel 3D output devices, e.g. 3D TV displays, etc. The second stage of this innovation is focusing on more important part of the problem, the cost and the real time communication. Indeed, design of real simulators remains domain specific and highly expensive for small companies, because the high performance of multi-core systems for graphics and capturing the distributed sensors and actuators to emulate a real conditions. Here, we investigate high performance, low cost PC simulators for education and training purpose: a real car driving cab, with a modular configuration to fit every need, from initial to advanced driving training.

The remainder of this paper is organized as follows: after this introduction, Section II uses two case studies to motivate common requirements of simulation for training purpose. The design and the architecture of the simulators, including hardware and the software are described in Section III. Section IV focalizes on the architectural overview of the DDS infrastructure. Section V show the networking part when connecting several simulators together in distributed environment using publish-subscribe paradigm build using OMG-DDS. Section VI compares our work with related researches. Finally, some concluding remarks and future works are presented in Section VII.

II. Scope and motivation

Although simulators have become more populated in flight and driving simulation to teach tactics, resources management and strategic thinking, computer simulation is still remarkably complex, high expensive and hard to evolve and extend. What is needed, therefore, is low complexity, high performance and low cost simulators for social interaction.

A. Challenge 1: Reducing the cost of computer simulation

Context. simulation is often used in the training of military personnel and in emergency response preparedness [2]. Simulators are used when it is very expensive or simply dangerous to allow trainees to use real world.
Problem. design of real simulators remains domain specific and highly expensive for small companies, because the high performance of multi-core systems for graphics and capturing the distributed sensors and actuators to emulate a real conditions.
Solution approach. our approach suggests a high performance, low cost PC-based simulators for save driving in car driving education: a real car driving cab, with a modular configuration to fit every need, from initial to advanced driving training.

B. Challenge 2: Allowing high performance for distributed simulators

context. Developing and deploying distributed applications in traditional pragmatic approach imposes significant challenge to most developers. It requires considerable learning curve for experts to be able to develop and deploy large scale distributed applications.
Problem. Both client-server and Peer to Peer architecture are limited in their usability when used in distributed real-time data-centric communication. Many details are mandatory for understanding the interaction between components. In the other hand, the existing software solutions are tightly integrated in a particular programming language. Before a distributed system can be implemented, developers should familiarize them self's with the new language syntax and network programming paradigm.
Solution approach. The current effort in improving the technology base for Human Interface Systems have been somewhat lacking with respect to distributed middleware. What is needed, therefore, are middleware-guided network QoS provisioning solutions that are not tied to a particular network QoS mechanism. These solutions should ideally operate across wide range of network QoS mechanisms without requiring modifying application code source. Thereby we show that

the newly developed technologies lead in favor of DDS-based solutions which provide distributed real-time communication.

III. HARDWARE AND SOFTWARE ARCHITECTURE

The driving simulator, is among the most sophisticated low cost systems: it is high performance, low latency, low cost, highly distributed and fulfills the most relevant QoS requirements of distributed architecture.

A. Hardware configuration

The hardware architecture is shown in Figure 1. It is composed of numerous functionalities embedded in specific modules. Each simulator is composed of the following components:

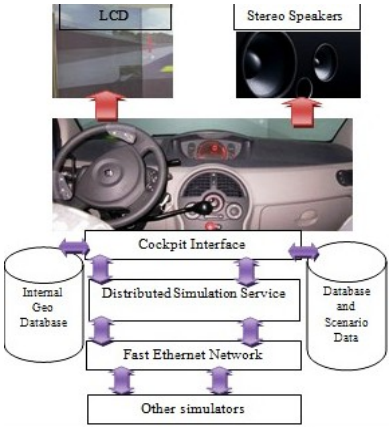

Fig. 1. Hardware architecture of the simulator

- an innovative and full-equipped cockpit,
- genuine parts and dashboard coming from the car manufacturer,
- left hand drive or right hand drive,
- adjustable seat, steering wheel, seat belt, rugged shifter and pedal assembly with force feedback
- steering wheel with motorized force feedback,
- small-size driving cab.

Real-time data are sent across field bus network based on CAN [9] to ensure the communication between the sensors and the actuators within the driving simulator. Further, camera is used to capture the movement of each participant and send them to teaching post. The flow issued from each camera is composed of sound and video streams. Both IP-Camera and any other webcam can be used within this system.

B. Modular Software Architecture

The design of the simulator is modular and extensible: several modules are developed separately and plugged within one more general component. Many libraries and databases were included to allow driving in real condition. Roads were designed to be fully compliant with realistic driving applications. Information such as lane topology, traffic attributes, crossroad modeling, road markings, road signs, urban furniture, etc. are integrated. As illustrated in Figure 2, night driving module is also developed to provide the training in many metrologies environment. In order to fit every need and

Fig. 2. City and night driving modules

to provide a safe driving, an emergency braking and safety distance module (total reaction time test and 2 seconds rule) is developed, including risk observation module with more than 40 situations. A Huge 3D database including city, country, highway and maneuvering area is provided, like shown in Figure 2. In addition, weather conditions and brightness like rain, snow, fog, daylight, dusk and night are added to meet the realistic condition, which every trainer can oppose. Further, several diverse traffics with more than 50 visible automate vehicles at the same time, and also pedestrian and animals can be confronted during the simulation exercise.

The scenario information is used for saving the exercise, and replying it when needed. These scenarios data files are containing not only computer data to be processed by the simulation, but also a descriptive text with teacher interpretation of the trainee mistakes. Replay function is provided with an external view and a printable report. An eco-driving module is provided for the trainees and permits accessing to all other modules in replay mode and allows the display of the following information for each driving: the driving time, the covered distance, the total consumption, the total CO2 emission, the average speed, the average consumption, the average CO2 emission, etc. moreover, these information are displayed within a graph with curves indicating the immediate values of consumption/speed/CO2 emission.

IV. KEY CONCEPTS BEHIND DDS

DDS [5] is ultra-low latency, real-time, and Data-Centric Publisher/Subscriber (Pub/Sub) middleware [8]: real-time means that all the right data should be delivered at the right place on the right time, all the time. Ultra-low latency means that it is able to distribute high volume of data with very low latency. Data centric means that it is able to ensure availability, reliability, safety and integrity [7]. It is policy driven and allows to configure all the QoS that relay matters when distributing information. DDS supports dynamic discovery (publisher and subscriber can join and leave the simulation at any point of time) and full time decoupling (publisher delivers some kind of data and subscriber is interested is any combination of those in any time).

The DDS specification implements a Global Data Space (GDS) to be fully distributed, avoid a single point of failure, and share information between participants [6]. Applications

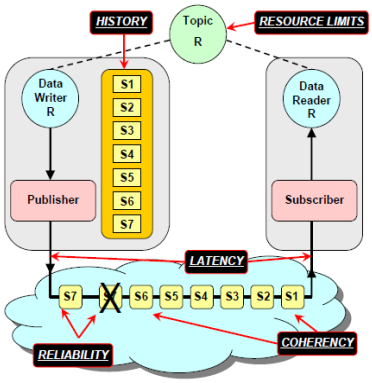

Fig. 3. Publisher-Subscriber communication architecture based on DDS

can read and/or write data objects addressed by their identifier fields: the Domain ID, the partition ID, the name of the Topic and a key. As shown in Figure 3, information are exchanged between distributed application using Pub/Sub paradigm with the aid of the following entities: *publisher* and *datawriter* on the sending side, *subscriber* and *datareader* at the receiving side.

Publisher: is the object responsible for sending of data, owning and managing the DataWriter. A Datawriter can be only owned by a single Publisher while a Publisher can own many Datawriter.

Subscriber: is the object responsible for the receipt of published data. The subscriber own and manages Datareader, which can be only owned by a single subscriber while a subscriber can own multiple Datareader.

Topic: represents the unit of information that is produced and/or consumed. A Topic is an association between a Type, a name and QoS policies used to control the properties of this Topic. As described in Figure4, Topic Types are expressed by means of structures and represented by the subset of OMG-IDL [12] standard.

Domain: It provides a virtual network linking all applications that have joined it (having the same Domain ID). The Domain consists of one or more DomainParticipant which isolate applications into several sub domains.

Partition: this entity represents a logical grouping of Topics associated with the Domain. It serves as a container and logical manager of DDS entities.

```
struct vehiculeSimule
{
        float carburant;
        float distanceParcourue;
        float vitArbre;
        float coefDerapT;
        float coefDerapL;
        float roulis;
        float consoInstantannee;
        float acc_angleChasse;
        float acc_rotation;
        float acc_Cebtrifuge;
        float accelArret;
};
```

Fig. 4. IDL definition of the driving car Type

V. NETWORK MODULE OF THE DDS BASED PC SIMULATOR

The architecture of the distributed simulation including many simulators in the same live simulation is depicted in Figure 5. The network module allows delivering the data issued from the virtual environment by sending data to all the simulators sharing the same simulation exercise. In order

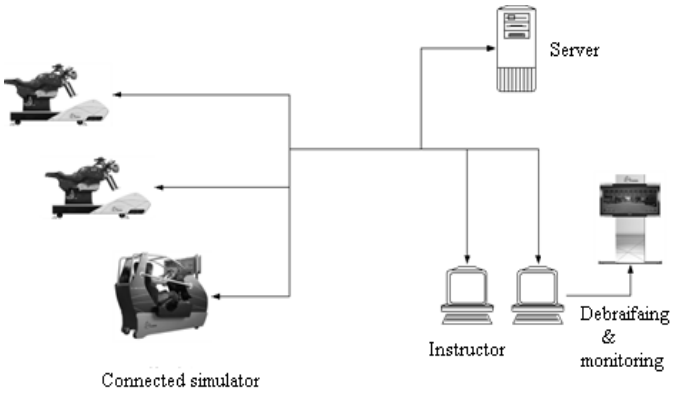

Fig. 5. Distributed simulators interconnected in live simulation

to model the different flows issued from each simulator we used 10 diffracts Topics, which have divers QoS policies to fulfill the QoS requirement of each data flow. The relationship between the Topics and all other DDS entities is given in Figure 6. Depending on the QoS requirements of each flow,

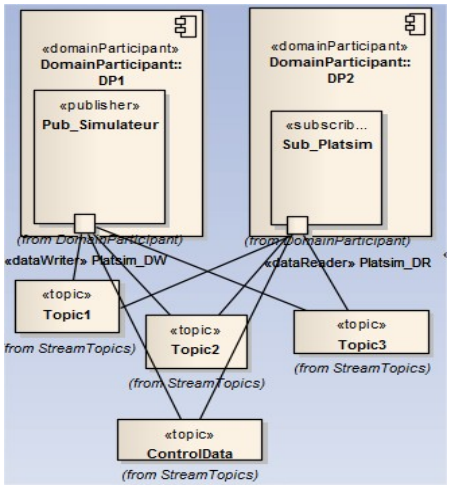

Fig. 6. Organization of Topics of the Simulator

described in Table I, the traffic profile is divided into three classes: signal, states (events) and streams: Signals represent data which change continuously, for example data taken from an speed sensor (accelerator). The reliability QoS policy is chosen as Best-Effort (BE) because many samples are sent during the simulation, so that can be lost without the need of retransmitting them. Signal producer indicate the speed at which it can publish by offering an update deadline equal to 50ms. Latency Budget can be chosen in the User QoS profile

TABLE I
QOS ASSOCIATION WITH THE KIND OF DATA

QoS Policies	Signal	Multimedia	Events
Reliability	BE	BE	R
Latency Budget	20	30	10
Durability	P	Vol	P
History	L	L	L
Deadline	0.05	0.033	0.01

to meet the requirement of each trainee, without external stress factors, especially for new trainees.

Events are in the most time very critical and they should be sent reliably (R). The durability QoS Specifies whether or not the middleware will store and deliver previously published data to new DataReaders, events use Persistent (P) durability.

Multimedia stream is less critical because they are defined using a precedence relationship depending on the codec used, but they can be sent/received unreliable (BE) to avoid the network latency.Further, data samples are sent continuously between the distributed simulators, so the QoS history setting is 'KEEP_LEAST' for the three flows.

Finally, in order to evaluate the performance of the distributed interactive simulation, we measured the network traffic transmitted during a simulation exercise as shown in Figure 7. The

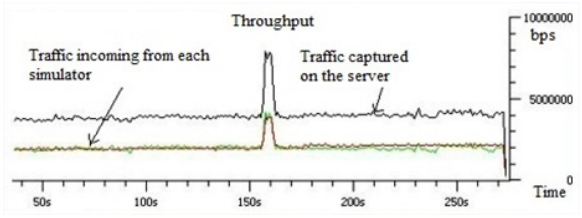

Fig. 7. Throughput at two distributed simulator

scenario used to evaluate the network module is composed of two simulators publishing and subscribing data during the simulation exercise, and we used another computer (for monitoring and debriefing) to collect all the traffic. Each simulator is considered at the same time as publisher and subscriber, and has a webcam for capturing multimedia (video and audio).

To ensure the inter-operability between the distributed simulators, the OMG-RTPS [10] which is a real time wired protocol build upon of UDP is used. Using the DDS middleware provided by RTI [11], the throughput at each simulator is up to 2Mbps. Hence simulators are able to deliver high volume data (multimedia), time critical data (signal in the simulation flow) and events without network bottlenecks. The experiments wa scaled to more six simulators and the traffic remains satisfying the QoS of all the participants.

VI. RELATED WORKS

This Section compare our activities on low cost PC-based simulators with related works.

Multi-domain and low cost simulation. Earlier work [1] investigated development methods to design an integrated fly simulator for research and training purpose. Our solution provides the way to everyone to be trained in safe condition with, among other, a real car driving cab using existing software and hardware.

high performance collaborative simulation. High Level Architecture (HLA) [3] has a wide applicability, across a full range of simulations areas, including education and training, analysis, engineering, and web-based distributed applications.However, HLA focuses on the inter-operability of simulators without giving any features for the QoS management. In our investigation, high performance and rich QoS policies offered by DDS-based architecture are used to ensure both network centric and inter-operability with a minimum development effort.

VII. CONCLUSION

In this paper, together hardware, software and network architecture design for low cost PC-based driving simulator is addressed. The successful implementation leads in favor of DDS, which is ultra-low latency middleware, providing data-centric and decoupled communication with a QoS provision and offers the possibility of the management, the online resources reconfiguration, and address the challenge of information exchange in high performance communication systems. The future works will focus intercontinental PC-based simulation using DDS.

ACKNOWLEDGMENT

This research is supported by the French FUI-DGE (Single Inter-Ministerial Fund of the Directorate General for Enterprize) program within the network simulation Platform (PLATSIM).

REFERENCES

[1] H. Smailai, M. Laban, and J. Dominicus, *New Integrated Modeling and Simulation Techniques for Reasearch and Training Applications*, AIAA Modelin and Simulation Technologies Conference and Exhibits, 1(-18 August 2005, San Fransico, California.

[2] J. Sanjay and C.R. McLean, *An Architecture for integrating Modeling and Simulation For Emergency Response*

[3] P. Wu, B. Cai, D. Zhang, Z. Zhou, and Y. Chen, *HLA-based Multi-aircraft Comabt Simulation System*, Conference on Informatics in Control, Automation and Robotics (CAR), 2010 2nd International Asia, 6-7 March 2010, pp 331-334

[4] L. Guangda, H. Qitao, and H. Junwei, *Architecture Development of Research Flight Simulator Based on COTS*', International Conference on Information Engineering and Computer Science, 2009. ICIECS 2009.

[5] OMG. Data Distribution Service for Real-Time Systems Specification. DDS v1.2, http://www.omg.org/spec/DDS/1.2/

[6] Lu, X et al., *A Novel QoS-Enable Real-Time Publish-Subscribe Service*. In: Proceedings of ISPA, pp. 19-26 (2008)

[7] Xinjie L et al.,*QoS-Aware Publish-Subscribe Service for Real-Time Data Acquisition*. Lecture Notes in Business Information Processing. 2008

[8] Douglas C. et al. *Addressing the Challenges of Tactical Information Management in Net-Centric Systems With DDS*, The journal of Defense software Engineering. March 2008.

[9] Can in Automation: http://www.can-cia.de/

[10] Data Distribution Ser vice Interoperability Wire-Protocol Specification. DDSI v2.1, http://www.omg.org/spec/DDSI/2.1/

[11] Real Time Innovation: http://www.rti.com/

[12] Common Object Request Broker Architecture (CORBA/IIOP) v3.1, http:// www.omg.org/spec/CORBA/3.1/