FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

From the Ground to the Cloud: Towards an Integrated Transportation Simulation Platform

Tiago Manuel Lourenço Azevedo

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: Rosaldo J. F. Rossetti, PhD (Assistant Professor)

Co-Supervisor: Jorge G. Barbosa, PhD (Assistant Professor)

July 14, 2015

From the Ground to the Cloud: Towards an Integrated Transportation Simulation Platform

Tiago Manuel Lourenço Azevedo

Mestrado Integrado em Engenharia Informática e Computação

July 14, 2015

Abstract

Nowadays, universities and companies all around the world have a huge need for simulation and modelling methodologies. In the particular case of traffic and transportation, making physical modifications in the real traffic networks could be highly expensive, dependent on political decisions and could be highly disruptive to the environment. Therefore, simulation is broadly used in such scenarios.

However, while studying a specific domain or problem, analysis through simulation may not be trivial and very often requires several simulation tools, with different resolutions and domain perspectives, hence raising interoperability issues. With the recent evolutions in cloud computing and Software-as-a-Service (SaaS), there is a new paradigm where simulation software is used in the form of services. So, Simulation Software-as-a-Service (SimSaaS) is very beneficial to better exploit the huge amount of platforms and storage that simulation needs *per se* - and Cloud Computing is able to provide such resources.

To address issues arising in this novel perspective the main goal of this dissertation was to present the current state of the art in the field and to propose an agent-directed transportation simulation platform, through the cloud, by means of services. It was used the IEEE standard HLA (High Level Architecture) for simulator interoperability and agents for controlling and coordination.

To do so, it was necessary to build, through a systematic literature review, the body of knowledge needed to develop such platform. The reviewed studies were compared and summarised leading to the creation of a taxonomy of the research work, which represent the front research opportunities for the next years. The main scenarios and architecture of the platform were detailed. The proof of concept's implementation was further explained including the used software (OpenStack, Pitch pRTI, SUMO and EBPS) and the chosen simulation scenario. Finally, some experiments were made about the best approach to manage and launch VMs (Virtual Machines). Such analysis is very important to have better performance in simulations under the developed infrastructure.

Keywords: SimSaaS; cloud computing; HLA; agent-directed simulation; M&S

Resumo

Actualmente, as universidades e as empresas de todo o mundo têm uma enorme necessidade de metodologias que permitam simular e modelar. No que diz respeito ao tráfego e transportes, fazer mudanças físicas nas redes reais de trânsito poderia ser altamente dispendioso, estando dependente de decisões políticas e podendo ser altamente prejudicial ao meio ambiente. Por isso, a simulação é muito usada em tais cenários.

No entanto, o uso de simulação para estudar ou analisar um domínio ou problema específico pode não ser trivial e podem ser necessárias diversas ferramentas, com diferentes resoluções e perspectivas de domínio, causando o aumento de problemas relacionados com interoperabilidade. Com as recentes evoluções no âmbito do *cloud computing* e do *Software-as-a-Service* (SaaS), existe um novo paradigma onde o *software* de simulação é usado sob a forma de serviços. Assim, o *Simulation Software-as-a-Service* (SimSaaS) é muito benéfico para melhor explorar o grande número de plataformas e armazenamento que a simulação precisa, e que o *Cloud Computing* pode fornecer.

Para ultrapassar os problemas supra mencionados, o principal objetivo desta dissertação foi apresentar o atual estado da arte na área e propor uma plataforma de simulação de transporte direcionada a agentes, através da *cloud*, por meio de serviços. Utilizou-se o *standard* HLA (*High Level Architecture*) da IEEE para interoperabilidade de simuladores e agentes para controlo e coordenação.

Para que tal seja possível, foi imperativo construir, através de uma revisão sistemática da literatura, o conhecimento necessário para desenvolver a plataforma. Os estudos revistos foram comparados e sumarisados na forma de uma taxonomia do trabalho de pesquisa que representa as oportunidades de pesquisa mais importantes para os próximos anos. A arquitectura e os principais cenários de utilização da plataforma foram detalhados. A partir daí, o subconjunto de características mais importantes foi seleccionado na forma de uma prova de conceito. A sua implementação foi explicada indicando o *software* utilizado (OpenStack, Pitch pRTI, SUMO e EBPS) e o cenário de simulação escolhido. Por fim, foram conduzidas algumas experiências para se perceber a melhor abordagem no controlo e lançamento de máquinas virtuais. Esta análise é importante para se obter uma melhor *performance* em simulações utilizando a infraestrutura desenvolvida.

Palavras-chave: SimSaaS; computação em nuvem; HLA; simulação direcionada a agentes; M&S

Acknowledgements

Harvey Mackay once said that “none of us got to where we are alone”. Finishing my master’s dissertation and looking back at the years that have passed, I could not agree more. In my opinion, this document is not merely the culmination of one semester of work and writing, but also a culmination of every learning accomplishment and relationship experienced throughout my life either personally and professionally. So, I have to thank some people not only because they supported me directly in the conclusion of my degree, but also because they supported me indirectly by helping me to be the person I am today.

First of all, to all my family and their support since the beginning (yeah, since the very beginning!): my parents, my grandparents, my brothers, my uncle... I am very fortunate for having you in my life! Of course, you are the foundations, the basis, and without you nothing would be possible!

I must thank two teachers that were so important before I got into university: Maria Carlos Oliveira and Rui Ramos. I think I will never forget you for being so important to my intellectual growing process. Maybe without knowing, you emphasised to me how the profession of teaching can be so beautiful, important and striking in people’s life.

Music played and plays a crucial and decisive role in my life. It is like living two lives: one with informatics, another with music! I truly believe music gave me a different way to look at life and even at informatics. So, I am sincerely grateful to Luís Almeida (oh those Saturdays!) and Ana Marta (I will keep tracking on those music movements!). Also, to my wonderful friends in the flute section of the Crestuma Wind Band, Maria and Joana! My time in FEUP wouldn’t be the same without you, believe me. Music can really save me in the worst moments.

In parallel to music and informatics I have that “old” friends that give (and gave!) me so much: Cris, Eduardo, Diogo, Luísa, Filipa, Inês and Catarina! I have to say this: you are looovely! A special thanks to Luísa, responsible for the creation of Oculum. Luísa, you know how much I learned with you and how grateful I am... Thank you so much.

In FEUP, I have to acknowledge the ones who made my learning process so pleasant and funny: Soneca, Filipe, Paulo, Alexey and Quim, with whom I did most of my work. I spent many long hours with you guys being anti-social in front of a computer but I don’t think you can even imagine how much I learned with you. Soneca and Filipe, a special thanks for your patience and friendship, you are genuinely unforgettable! Beyond knowledge, I also carry in my heart good memories! I think you know how much I like you and because of that I would have adored to say some words but maybe it would be inappropriate... Well, you know... As well, a sincere thanks to Inês who turned into an amazing surprise, and to Malini, who helped me with the English .

I would like to finish by thanking those who directly supported me while working for my master’s dissertation. Thanks to the invaluable suggestions and availability of Zafeiris Kokkinogenis at the beginning of this adventure! Likewise, thanks to Dr. Jorge Barbosa for his tips, discussions and important expertise.

Finally, my tremendous gratitude to Dr. Rosaldo Rossetti, who was one of the very first lecturers that I've met in FEUP and with whom I have always kept in contact with throughout these last five years. Thanks for your key presence, experience, advices, encouragement and pushing attitudes. Thanks for the opportunities you gave me. Thanks for what I learned. Thanks for your paternal friendship. Really, thanks!

I think I am a very lucky man because of you all. As someone once said (I couldn't figure out who), I also want to emotionally say:

I would thank you from the bottom of my heart, but for you my heart has no bottom!

“Scientists discover the world that exists;
engineers create the world that never was”

Theodore von Kármán

“The ability to simplify means to eliminate the unnecessary
so that the necessary may speak.”

Hans Georg Albert Hofmann

Contents

List of Figures

LIST OF FIGURES

List of Tables

Abbreviations

API	Application Programming Interface
ATS	Artificial Transportation Systems
CPU	Central Processing Unit
EBPS	Electric Bus Powertrain Subsystem
FOM	Federation Object Model
HLA	High Level Architecture
HTTP	Hypertext Transfer Protocol
IaaS	Infrastructure as-a-Service
IEEE	Institute of Electrical and Electronics Engineers
IP	Internet Protocol address
M&S	Modelling and Simulation
MSaaS	Modelling and Simulation-as-a-service
NTP	Network Time Protocol
OMT	Object Model Template
PaaS	Platform as-a-Service
PADS	Parallel and Distributed Simulation
PC	Personal computer
RTI	Run-Time Infrastructure
SimSaaS	Simulation Software-as-a-Service
SaaS	Software-as-a-Service
SUMO	Simulation of Urban MObility
VLAN	Virtual Local Area Network
VM	Virtual Machine
XML	eXtensible Markup Language

Chapter 1

Introduction

1.1 Context

Nowadays, universities and companies all around the world have a huge need for simulation and modelling methodologies. The objectives are varied, but simulation is widely used for decision making and what-if analysis, as well as for performance optimisation, testing and training. In the particular case of traffic and transportation, making physical modifications to the real traffic networks could be highly expensive, dependent on political decisions and could be highly disruptive to the environment. Therefore, simulation is broadly used in such scenarios.

However, while studying a specific domain or problem, analysis through simulation may not be trivial and very often requires several simulation tools, with different resolutions and domain perspectives, hence raising interoperability issues. Thus instead of helping, simulation could be a headache! Transportation problems are usually complex and fall within this category of problems.

Until present, to the best of our knowledge, there are not many solutions for traffic that make full use of the intelligent agent concept. However, the multi-agent system metaphor has become recognised as a convenient approach for modelling and simulating complex systems [MT07]. Also, it has grown enormously not only for being applied to traffic but also to transportation in general terms [BK13], contributing to the advent of new concepts in transportation, such as Artificial Transportation Systems (ATS) [RLT11], and, more recently, an increasing interest in digital games applied to transportation [RAKG13].

With the recent evolutions in cloud computing and Software-as-a-Service (SaaS), there is a new paradigm where simulation software is used in the form of services. Indeed, such evolutions have been more significantly seen in the business world with information technology solutions moving to the SaaS paradigm [Sha10]. So, Simulation Software-as-a-Service (SimSaaS) is very beneficial to better exploit the huge amount of platforms and storage that simulation needs *per se* - and Cloud Computing is able to provide such resources. By this way, researchers do not need to have the simulation software installed on their own computers or directly access servers which host this kind of software.

1.2 Objective and Expected Contributions

To address issues arising in this novel perspectives of Section 1.1, the main objective of this dissertation was to present the current state of the art in the field and to propose an agent-directed transportation simulation platform, through the cloud, by means of services. It was intended to use the Institute of Electrical and Electronics Engineers (IEEE) standard High Level Architecture (HLA) for simulator interoperability and agents for controlling and coordination. To do so, it was necessary to build, through a systematic literature review, the body of knowledge needed to develop such a platform. The motivations regarding these objectives are to allow multiresolution analysis of complex domains, to allow experts to collaborate on the analysis of a common problem and to allow co-simulation and synergy of different application domains.

This dissertation is expected to fulfil three main contributions. Firstly, a technological contribution because one will have a cloud-based simulation platform for transportation using HLA and agents where simulations are offered in the form of services. Secondly, a scientific contribution since it will enable the collaboration among experts of the Modelling & Simulation (M&S) field with the agent-directed paradigm. Finally, an applied contribution with an agent-oriented platform for scientific simulation, through the cloud, by means of services. Basically, it will be a virtual laboratory.

1.3 Research Questions

The systematic literature review was led by six research questions. These questions were defined in order to make the objective of this work *S.M.A.R.T.*. A full description of *S.M.A.R.T.* objectives was made by Meyer [Mey03], but basically it is an acronym for Specific, Measurable, Attainable, Relevant and Time-bound, which are characteristics intended for the goals previously described.

The defined research questions are the following:

1. How can simulation in the cloud, by means of services, be beneficial to modelling and simulation?
2. What would be an appropriate architecture for a simulation platform, in the cloud, by means of services?
3. How to extend HLA to support agent-directed simulation in the cloud?
4. How to verify whether an agent-directed approach has advantages over the traditional HLA approach?
5. Are there any similar approaches to such a platform and what are the resources needed?
6. Which stakeholders would benefit from a cloud-based simulation platform?

These questions illustrate the context, motivation and objectives previously outlined. In order to understand how these questions make the objective *S.M.A.R.T.*, an explanation is now carried out.

When the first three questions are answered, these will make the objective specific because it will be possible to answer the questions *What do I want to accomplish?*, *Why is the platform important?* and *Which are the requirements and constraints of such a platform?*. The “who” and “where” are irrelevant taking into account the context of a master's dissertation. Furthermore, it will also make the objective relevant as, deep down, it is intended to see if this work is interesting/relevant.

In the case of the fourth question, the answer will lead to the measurement of the objective as it will be known how it is possible to verify progress during and at the end of development. "How much?" and "How many?" do not apply in this case.

The objective's attainability is met with the fifth question. Basically, it will be possible to see what is already done, what went right and wrong and what were other difficulties.

Finally, with the last question it is another way to verify the relevance of the objective. In the context of a master's dissertation, the objective is already time-bounded.

1.4 Document Structure

Besides this chapter, the document will have six more chapters. It is intended to illustrate this document structure with the metaphor presented in the title: "from the ground to the cloud". The *ground* represents the current state of things and the main concepts of where this dissertation will start from. The platform that is intended to develop and related work are based on the cloud computing paradigm and thus it is necessary to go *to the cloud*.

So, this document will start *from the ground* in Chapter 2, that is to say, from the main concepts related to the present dissertation and that are needed for a better understanding of the literature review.

With this background of concepts, it is possible to go *to the cloud* in Chapter 3, that is, to leave this ground of concepts and to further explore related studies by means of a literature review. This literature review was conducted by means of a systematic method, which will be explained. All the considered research will be compared and summarised, which will allow for the creation of a taxonomy of the research work, and how the platform should be inserted in it.

Having this literature review in mind, the proposed platform is presented in Chapter 4, explaining how it should be and what are the necessary development milestones to achieve a successful proof of concept. A formal definition regarding models is also presented.

Chapter 5 details the implementation procedure and Chapter 6 describes the preliminary results obtained from some experiments with virtualisation, which is needed to the platform.

Finally, Chapter 7 presents the main conclusions of the work done also suggesting some work guidelines for the future.

Chapter 2

Preliminary Background: From the Ground...

This Chapter starts *from the ground*, that is, from the main concepts related to the present dissertation and that are required for a better understanding of the Literature Review presented in Chapter 3. It is necessary to have a good and solid back**ground** to be possible to go farther in the pursuit of knowledge.

2.1 M&S in Traffic and Transportation

In the context of M&S, a system is defined as a collection of entities, for example people or machines that act and interact together towards the accomplishment of some logical end [ST70].

There is not a unique definition of M&S in the literature, depending on each domain and scientific field. According to the Merriam-Webster Online Dictionary[1], simulation is “the imitative representation of the functioning of one system or process by means of the functioning of another” or, in other words, the “examination of a problem often not subject to direct experimentation by means of a simulating device”. Therefore, in simulation there is not only the idea of representation but also of experimentation without the direct intervention of a human.

Modelling is basically an abstract and simplified representation of a system. It is similar to the system, but simpler [Rob08], as it should be an approximation to the real system with the most relevant features, but simple enough to be understandable. As Sharif [Sar05] points out, a good model is a judicious trade-off between realism and simplicity.

As it can be seen, simulation of a system is mainly the execution of a model. Indeed, simulation is widely used for decision making and what-if analysis, as well as for performance optimisations, testing, training, and so forth.

[1] www.merriam-webster.com/

Figure 2.1 shows the different methods for studying a system. Most times it is not possible to make direct experiments with the actual systems, and it is necessary to make simplifications using modelling. In this case, there is the question of whether it actually reflects the system, but analysing this issue is outside the scope of this dissertation. If the model is simple enough, it is possible to study the system using an analytical solution through, for example, calculus, algebra and probability theory. Otherwise, if the system is highly complex, simulation is performed using computational means.

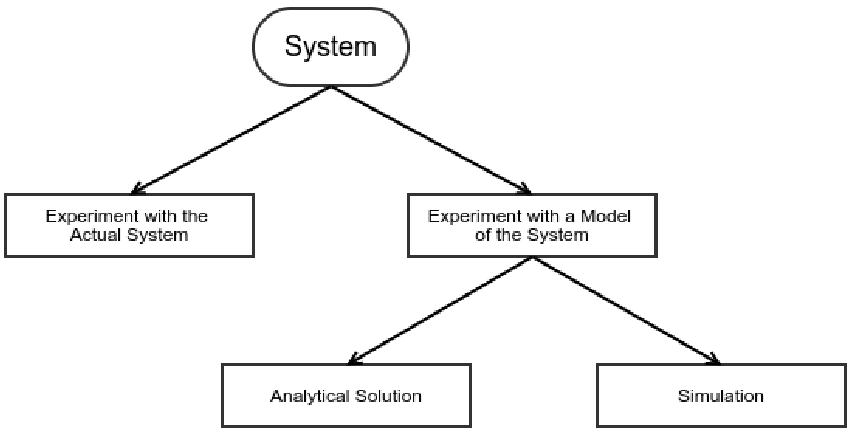

System
Experiment with the Actual System
Experiment with a Model of the System
Analytical Solution
Simulation

Figure 2.1: Different methods for studying a system

In the particular case of traffic and transportation, making physical modifications to the real traffic networks could be highly expensive, dependent on political decisions and could be extremely disruptive to the environment. Therefore, M&S is widely used in such scenarios and related tools can provide better and more concise data for analysis.

There are several ways to model traffic depending on the level of detail in which they describe the traffic dynamics. Figure 2.2 illustrates the major four categories of modelling in traffic, namely, macroscopic, mesoscopic, microscopic and nanoscopic.

Macroscopic models describe traffic in terms of flows [2] or densities, without considering their entities such as vehicles. Therefore, these models are good to analyse large or complex networks. As an example, the Lighthill–Whitham–Richards Model [LW55] uses differential equations to formulate relationships among traffic flow density.

Unlikely macroscopic models, microscopic models describe both the behaviour of each entity and their interactions, with each other as well as with the network. For that, these models incorporate the vehicles' behaviour rules such as acceleration, breaking, lane changing, and so forth. The Lane-change model [BACT06] and Route-choice model [Pra09] are methods used to determine vehicle's behaviour.

Mesoscopic models fill the gap between macro and micro models. They normally describe traffic entities at a high level of detail, whereas their behaviour and interaction are briefly described.

[2]Flow is the number of vehicles that pass through a certain road per hour

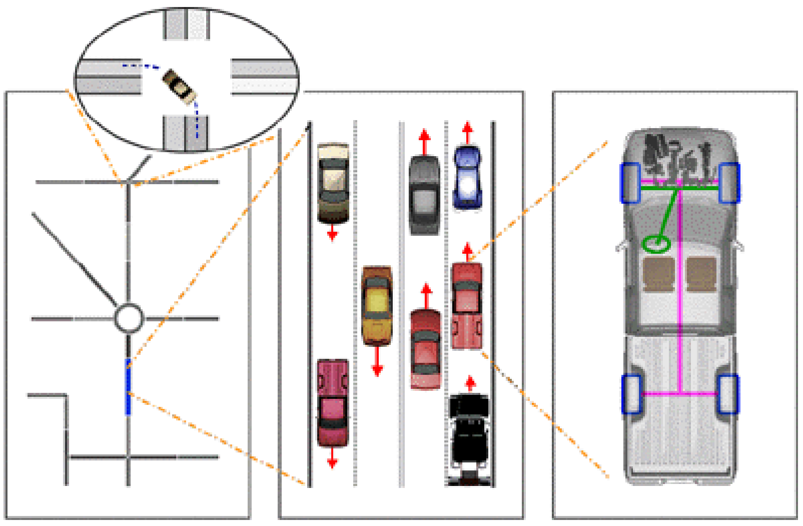


Figure 2.2: The major four categories of modelling in traffic (from left to right): macroscopic, microscopic, nanoscopic (mesoscopic within the circle) [KHRW02]

In these models, vehicles can be grouped in packets, which are routed throughout the network and are treated as one single entity [BKA05].

A new trend in traffic simulation though is the nanoscopic model which extends the capabilities of three basic components of microscopic simulation: vehicle specific modelling, vehicle movement modelling, and driver behaviour modelling [DP08]. For example, in the case of traffic, it intends to describe the components of a vehicle. Attempts at integrating these different perspectives in a more transparent way through modelling has already been reported in the literature [RB99] [FERO08a].

2.2 Agent-directed Simulation

The introduction of *intelligent demons* (called intelligent agents today) to control simulation experiments was introduced by the MISS Hungarian Center [JF12]. This notion of *intelligent agent* may not be consensual, mainly because of the difficulty of defining what *intelligence* is. However, it is considered an agent as an autonomous and proactive computational entity whose rational process is based on concepts such as Knowledge, Belief, Intention, Commitment, Goal, Desire and Emotion [WJ95].

The intelligent agent concept brings a genuine metaphor to represent autonomous entities as it is equipped with sensors and actuators[3] as well as with reasoning and decision-making abilities. Thereby, there are entities with high-level communication skills who also provide endless possibilities for system coordination and controlling. Otherwise, system coordination and controlling would be reduced to automated scripts, which do not bring so many advantages.

To date, there are not many solutions for traffic that make full use of the intelligent agent concept [DRO06][FERO08b][RL05a][RL05b]. However, the multi-agent system approach has become recognised as a convenient one for modelling and simulating complex systems [MT07]. Also,

[3]In this context, an actuator is something that can produce a change in the environment perceived by the agent

it has grown enormously, not only when applied to traffic but also when applied to transportation in general terms [BK13]. Nevertheless, just a few simulation tools truly support the concept of agents and multi-agent systems in traffic simulation; MATSim-T [BRM+09] and ITSUMO [BdASB10] are good examples to be mentioned. Also, besides agent-based traffic simulation, the MAS-Ter Lab platform introduces the concept of expert agents in charge of the simulation analysis process [ROB07].

The agent metaphor encompasses some other concepts, that Yilmaz and Ören [YÖ07] unify in the Agent-Directed Simulation paradigm. The authors indicate that the paradigm consists of three distinct, yet related areas that can be grouped into two categories: Simulation for Agents (agent simulation) and Agents for Simulation. The first is about simulation of systems that can be modelled by agents, that is, the simulation model is an agent or, in other words, simulation of agent systems. The latter can still be divided into two categories, namely agent-based simulation and agent-supported simulation, as follows.

- **Agent-based simulation** is the use of agent technology to generate model behaviour or to monitor generation of model behaviour. The perception feature of agents makes them pertinent for monitoring tasks. Agent-based simulation is useful for having complex experiments and deliberative knowledge processing such as planning, deciding, and reasoning.
- **Agent-supported simulation** deals with the use of agents as a support facility to enable computer assistance by enhancing cognitive capabilities in problem specification and solving. Hence, agent-supported simulation involves the use of intelligent agents to improve simulation.

A lot of researchers do not take into account the contribution of agents to simulation. Thus, in such cases, *agent simulation* and *agent-based simulation* are seen as the same principle. In this dissertation it is adopted the same perspective in which the two principles are seen indistinguishably.

2.3 HLA and Distributed Simulation

Parallel and distributed simulation (PADS) relies on partitioning the simulation model across multiple execution units. Each execution unit manages only a part of the model and handles its local event list, but locally generated events may need to be delivered to remote execution units [DM14].

Distributed simulation facilitates the reuse of heterogeneous simulation systems but has issues regarding interoperability of simulators. HLA is an IEEE software standard developed to provide a common technical architecture for distributed M&S, trying to provide the structural basis for interoperability among simulators.

In HLA, every participator of the simulation is called federate, and these federates can interact with each other within a federation. The baseline components of HLA include (1) Federate Interface Specification, (2) Framework and Rules, and (3) Object Model Template (OMT) Specification.

The Federate Interface Specification has the services which federates can use for communication [IEE10c]. This communication between simulators is managed by a Run-Time Infrastructure (RTI). In order for the interaction between federates and the RTI to be possible , there is the concept of ambassador. Basically, federates communicate with the RTI using their ambassador as an interface.

The Framework and Rules of HLA are the set of rules which must be obeyed in order to ensure the proper interaction within a federation. There are five rules for federates and also five rules for federations, detailed in [IEE10a].

OMT describes the format and syntax of the data exchanged among federates. Thus, it defines the object template data that all simulation units use in order to exchange data with each other [IEE10b].

Figure 2.3 depicts the main HLA components. It is possible to see that inside a federation every federate have the same Federation Object Model (FOM), but each one has its own specificities in the form of a Simulation Object Model (SOM). While the FOM describes the shared objects, attributes and interactions, a SOM describes the same but for a single federate. The interactions between a federate and the RTI through the ambassadors is also more clear in the Figure.

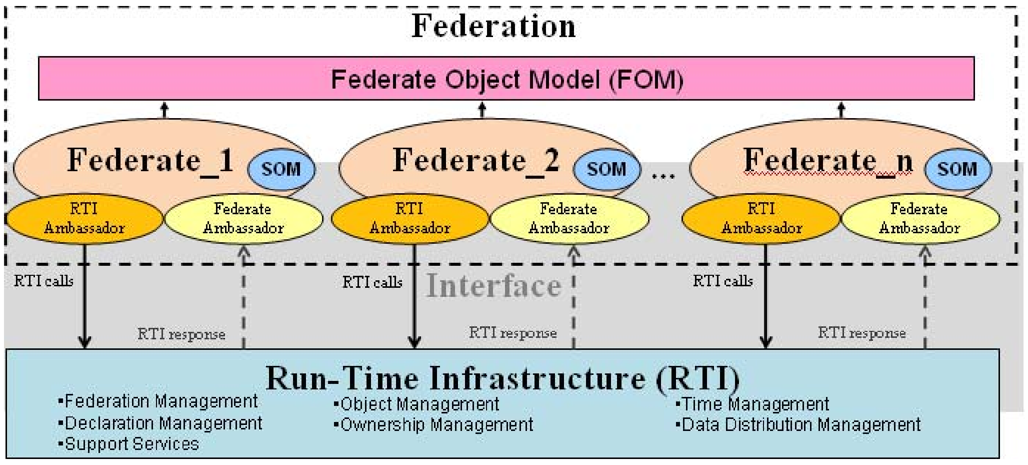


Figure 2.3: HLA's Functional Architecture [NAT13]

Other studies on distributed simulation in traffic and transportation domain can also be mentioned, mainly because of the emergence of the Intelligent Transportation Systems (ITS), and more recently as well as ATS [RL14][RFBO08]. One focus of this domain is in distributing mechanisms, with decision making capabilities within simulation (e.g. pedestrian, traffic lights or driver agents) [WUW+12][FCD12][VO11].

Another focus is on distributing models among different simulations. Sewall et al. [SWL11] present a real-time algorithm for modelling large-scale traffic using both a continuum macroscopic model and an agent-based microscopic model. Zegeye et al. [ZDSHB09] describe a traffic control framework for fuel emissions by integrating macroscopic traffic flow models with a microscopic emission and fuel consumption model.

Finally, an integrated framework that aims at coupling robotics and a traffic simulator is presented by Pereira and Rossetti [PR12]. This work developed an integrated framework enabling autonomous vehicles to be deployed in a rather realistic traffic flow while at the same time simulating all its sensors and actuators.

2.4 The Cloud Computing Paradigm

In 1969, Kleinrock [Kle05] said: “As of now, computer networks are still in their infancy, but as they grow up and become sophisticated, we will probably see the spread of ’computer utilities’ which, like present electric and telephone utilities, will service individual homes and offices across the country”. After so many years, this is becoming true, and computing is even more managed and delivered in the form of traditional utilities such as water, electricity, gas and telephony. Cloud Computing is helping to leverage this utility vision.

Consequently, Cloud Computing has become another buzzword and more and more work is being done in the field, not only in simulation but in several other domains, such as Information Systems [Sha10]. Nevertheless, there are dozens of different definitions for Cloud Computing and there is little consensus on that [Gee09]. Indeed, Foster et al. [FZRL08] point out that the term was co-opted by industry just as a marketing term for clusters. So, the authors ask “So is *Cloud Computing* just a new name for Grid?”, giving an interesting answer: **yes**, the vision is the same - to reduce the cost of computing increase reliability and increase flexibility; **but no**, things are really different now than they were years ago. We have a new need to analyse massive data, thus motivating greatly increased demand for computing. We also have low-cost visualisation and we are operating at a different scale!; **nevertheless, yes**, the problems are mostly the same in Clouds and Grids. There is a common need to be able to manage large facilities.

Another problem in defining cloud computing is that it overlaps with other domains. Figure 2.4 tries to illustrate the main fields of Distributed Systems and their overlaps. Web 2.0 covers the spectrum of service-oriented applications, opposing the Supercomputing and Cluster Computing, which have been more focused on traditional local applications. Cloud Computing lies at the large-scale side, being more scalable than Grid Computing. Grid Computing overlaps with all these fields, and because of that wrong definitions exist regularly. Foster et al. [FZRL08] make a more detailed comparison of Cloud and Grid Computing. Likewise, Buyya et al. [BYV$^+$09] perform a detailed comparison of Cloud Computing and other similar paradigms.

Despite this vision, it is possible to differentiate cloud computing from grid computing. In this research, it is adopted the definition provided by The National Institute of Standards and Technology [MG11], as it is an Institute responsible for standards and because no other standard definition exists so far:

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

It also states that this cloud model is composed of five essential characteristics, three service models, and four deployment models, which are briefly explained below.

Essential Characteristics:

- *On-demand self-service.* A consumer can unilaterally provision computing capabilities as needed automatically without requiring human interaction with each service provider.

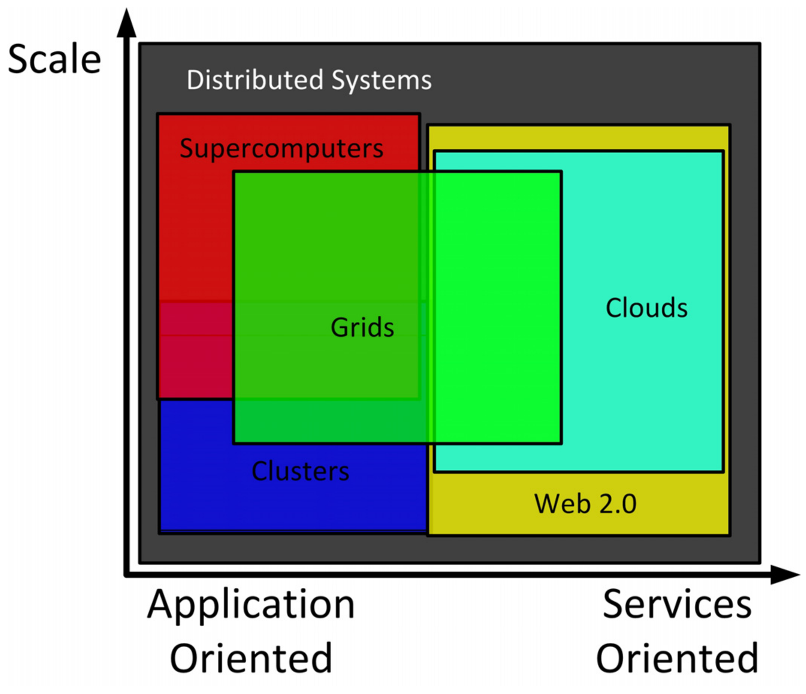


Figure 2.4: Fields of Distributed Systems according to scale and domain [FZRL08]

- *Broad network access*. Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous client platforms, either thin or thick.
- *Resource pooling*. The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand.
- *Rapid elasticity*. Capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward with demand.
- *Measured service*. Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized service.

Service Models:

- *Software as a Service (SaaS)*. It is provided to the consumer the capability of running provider's applications on a cloud infrastructure.
- *Platform as a Service (PassS)*. It is provided to the consumer the capability of deploying onto the cloud infrastructure consumer-created or acquired applications created using programming languages, libraries, services, and tools supported by the provider.
- *Infrastructure as a Service (IaaS)*. It is provided to the consumer the capability to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications.

Deployment Models:

- *Private cloud*. The cloud infrastructure is provisioned for exclusive use by a single organization comprising multiple consumers.

- *Community cloud.* The cloud infrastructure is provisioned for exclusive use by a specific community of consumers from organizations that have shared concerns.
- *Public cloud.* The cloud infrastructure is provisioned for open use by the general public.
- *Hybrid cloud.* The cloud infrastructure is a composition of two or more distinct cloud infrastructures (private, community, or public) that remain unique entities, but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load balancing between clouds)

As also shown by Armbrust et al. [AFG$^+$10], there are three aspects that are new in cloud computing: (1) the appearance of infinite computing resources available on demand, (2) the elimination of an up-front commitment by cloud users, and (3) the ability to pay for the use of computing resources on a short-term basis as needed. The authors conclude indicating ten obstacles and opportunities for growth of cloud computing, which are briefly outlined in Table 2.1.

Obstacle	**Opportunity**
Availability/Business Continuity	Use Multiple Cloud Providers
Data Lock-In	Standardize Application Programming Interface (APIs); Compatible software to enable Surge or Hybrid Cloud Computing
Data Confidentiality and Auditability	Deploy Encryption, VLANs, Firewalls
Data Transfer Bottlenecks	FedExing Disks; Higher Bandwidth Switches
Performance Unpredictability	Improved VM Support; Flash Memory; Gang Schedule VMs
Scalable Storage	Invent Scalable Store
Bugs in Large Distributed Systems	Invent Debugger that relies on Distributed VMs
Scaling Quickly	Invent Auto-Scaler that relies on Metalanguage; Snapshots for Conservation
Reputation Fate Sharing	Offer reputation-guarding services like those for email
Software Licensing	Pay-for-use licenses

Table 2.1: Top ten obstacles and opportunities for growth of cloud computing [AFG$^+$10]

Some of these obstacles and opportunities are relevant in the context of this work. For example, it is very important to have availability and keep track of bugs as well as to decide what will be the best software licensing to offer. All these obstacles are currently being researched, and show that Cloud Computing is a fresh and on-going hot topic not only for the industry but also among academics. Indeed, even recently new buzzwords emerged from Cloud Computing, trying to extend it even further: Fog Computing [BMZA12] and Cloud 2.0 [Mil14] are examples.

There are four main representative Cloud platforms with industrial linkages [BYV$^+$09] (the authors also analysed Sun Cloud, but it is now discontinued): Amazon Elastic Compute Cloud (EC2)[4], Google App Engine[5], Microsoft Azure[6] and Aneka [CNJ$^+$07]. Each one has it own focus, including one or more of SaaS, Platform-as-a-Service (PaaS) and IaaS.

[4]http://aws.amazon.com/ec2/
[5]https://cloud.google.com/appengine/
[6]http://azure.microsoft.com/

For a wider state-of-the-art perspective on Cloud Computing with important research directions there are several recent sources to analyse [GMBLGSCP15] [ZCB10] [SDH$^+$14] [AJG$^+$15] [HX14] [ASS$^+$15].

Virtualisation is now indispensable for almost every cloud, mainly because of the abstraction and encapsulation capabilities it brings. Foster et al. [FZRL08] make an analogy about virtualisation: just like threads were introduced to provide users the "illusion" as if the computer were running all the threads simultaneously, and each thread were using all the available resources, Clouds need to run multiple user applications, and all the applications appear to the users as if they were running simultaneously and could use all the available resources in the Cloud. The authors also refer that there are four main reasons that Cloud tend to adopt virtualisation:

- *Server and application consolidation*, as multiple applications can be run on the same server, resources can be utilized more efficiently.
- *Configurability*, as the resource requirements for various applications could differ significantly, virtualisation is necessary as this is not achievable at the hardware level. Virtual machines can be configured to use different parts of resources on the same physical machine.
- *Increased application availability*, as virtualisation allows quick recovery from unplanned outages.
- *Improved responsiveness*, as resource provisioning, monitoring and maintenance can be automated, and common resources can be cached and reused. Furthermore, multiple virtual machines can be started and closed on-demand to meet accepted service requests.

Some may say that virtualisation brings overheads. However, processor manufacturers such as AMD and Intel have been introducing hardware support for virtualisation. Thus, application performance on virtualised infrastructures is getting closer to application performance directly on a machine. Indeed, for a Central Processing Unit-intensive (CPU-intensive) PADS application, the performance loss in the virtualised environment is less than 5% [IOY$^+$11].

2.5 Summary

Many concepts were clarified and analysed in this chapter. After clarifying M&S, different methods to study a system are presented, showing that traffic and transportation are usually complex problems and that is why they are studied through simulation. A very important and essential concept in traffic and transportation analysis is the resolution it can have: macroscopic, mesoscopic, microscopic or nanoscopic.

The agent-oriented paradigm has a panoply of associated concepts which were clarified. The IEEE's standard for simulator interoperability, the HLA, is generally presented, also showing some related works.

There are several attempts to define the Cloud Computing Paradigm, which mainly lies on the large-scale side, being more scalable than Grid Computing. Some obstacles and opportunities exist, being highlighted the most relevant for this work. In summary, Cloud Computing is a fresh and on-going hot topic in industry and universities.

Chapter 3

Literature Review: ...To the Cloud

In the previous chapter a background of concepts was clarified. So, now it is possible to go *to the cloud* departing from such a back*ground* of concepts and by exploring related studies. All the reviewed research will be compared and summarised, which will allow for the creation of a taxonomy of the research work, and enable to know where about this body of knowledge the platform should be placed.

3.1 Search Method for Systematic Literature Review

The systematic literature review presented in this document will be conducted using the methodological framework proposed by vom Brocke et al. [vSN+09]. This framework has five phases: (1) definition of the review scope, (2) conceptualisation of topic, (3) literature search, (4) literature analysis and synthesis, and (5) research agenda.

The following sections describe these phases clearly referring to the literature review presented.

3.1.1 Definition of the Review Scope

In order to specifically define the scope of a literature review, vom Brocke et al.[vSN+09] refer to an established taxonomy presented by Cooper in 1988 [Coo88]. This taxonomy has six characteristics:

- **Focus**. It concerns the material that is of central interest to the reviewer, concerning one or more of four areas: research outcomes, research methods, theories, and practices or applications. This literature search focus on all types of papers;
- **Goal**. It concerns what the author hopes the review will accomplish and may relate to integration, criticism or central issue. Integration can include formulating general statements, resolving the conflict between contradictory ideas or statements of fact, or bridging the gap between theories and practices. Criticism is a judgement about some work. Central issue

may involve what has been studied in the past, what researchers will study in the future or methodological problems that have prevented a topic from progressing. It is wanted to identify the central issue of the literature review about an integrated simulation platform. To do so, the research questions of Section 1.3 will lead the research;

- **Organisation**. It is about how the review is arranged. A review can be organised historically (that is, in the chronological order in which topics appeared in the literature), conceptually (that is, works relating to the same abstract ideas appear together) or methodologically (that is, works employing similar methods are grouped). This literature is sorted conceptually;
- **Perspective**. It is the point of view of the reviewer while discussing the literature. This point of view could be neutral or espousal/advocacy of a position of perspective. There is no need to adopt a specific position in this work, thus the author will take a neutral point of view;
- **Audience**. It concerns to whom the review is written, for example specialised researchers, general researchers, practitioners, policy makers or general public. The audience of this literature are specialised researchers of the field;
- **Coverage**. It is the extent to which reviewers find and include works. The coverage can be exhaustive (including the entirety or at least most of the literature on a topic), exhaustive with selective citation (considering all the relevant sources, but describing only a sample), representative (including only a sample that typifies larger groups of articles), and central (reviewing the literature pivotal to a topic). This work's coverage will be reasonably representative.

Table 3.1 sums up the taxonomy of this Literature Review using the one proposed by Cooper.

Characteristic	**Cooper's options**	**Author's choice**
Focus	research outcomes, research methods, theories, practices or applications	All types of papers
Goal	Integration, criticism, central issue	Central issue
Organisation	Historically, conceptually, methodologically	Conceptually
Perspective	Neutral, espousal	Neutral
Audience	Specialised researchers, general researchers, practitioners, policy makers, general public, . . .	Specialised researchers
Coverage	Exhaustive, exhaustive with selective citation, representative, central, pivotal	Representative

Table 3.1: Cooper's taxonomy applied to the literature review

3.1.2 Conceptualisation of Topic

"A review must begin with a broad conception of what is known about the topic and potential areas where knowledge may be needed" [vSN+09]. Consequently, Chapter 2 presents the key concepts for a better conceptualisation of the work.

The choice of these concepts was straightforward while keeping in mind the objective set in Section 1.2. In order to make an agent-directed simulation platform, it is imperative an overview on M&S and clarification about agent-directed simulation. Then, one should look for any standard in the field, and that is why HLA is also presented. Finally, the platform will be in the cloud, being essential to present the Cloud Computing paradigm.

In order to clarify these concepts, several papers were looked into detail. The search of these papers did not follow any methodological approach as the goal was simple: to clarify concepts. However, the search process was not completely disorganised: ideas were frequently crossed between papers so conclusions could be more trustworthy, and searched database sources varied.

3.1.3 Literature Search

This phase "involves database, keyword, backward and forward search, as well as an ongoing evaluation of sources" [vSN+09]. Hence, four decisions must be made regarding these four issues.

First, the database sources selected were Scopus[1], Engineering Village[2] and ACM[3]. These databases are commonly known to contain a vast amount of literature and have been used by many researchers in software engineering. They usually include prominent scientific journals and conference proceedings related to the scope of this study. Due to the restrictions of each database, the search fields chosen for each one were title/abstract/keywords, subject/title/abstract (in all databases) and anywhere in the article, respectively.

Secondly, the suitable *keywords* must be chosen. As previously pointed out, four main concepts are closely related to this work and must be analysed further in order to find related work: SimSaaS, Cloud Computing Paradigm, HLA and agents. "Traffic" and "transportation" are not used because although some work is related to traffic and transportation, these terms sometimes do not appear in the text. Table 3.2 shows the queries built from the keywords, and the reasons behind that construction. A time frame from 2004 to 2015 was considered. The evolution of knowledge and technology in the software engineering field is tremendous every year. Thus, a time frame of a decade seems sufficient.

Thirdly, it is necessary to decide whether backward or forward search[4] is applied. In this case, the amount of scientific articles is considered appropriate for a representative literature review. Hence, it has been decided not to apply either backward or forward search.

Finally, the evaluation "in all phases means limiting the amount of literature identified by keyword search (...) to only those articles relevant to the topic at hand" [vSN+09]. Indeed, from each database query result, duplicated papers (compared with other databases), keynote speeches, thesis and papers that actually were not related to the topic were removed. After the removal process, every information was stored using JabRef [Jab15], in which each BibTeX record corresponded

[1]http://www.scopus.com/
[2]http://www.engineeringvillage.com/
[3]http://dl.acm.org/
[4]Backward search consists in search work by looking to the references of a document. Forward search consists in looking for who cites a document

Query identification	**Query strings**	**Reasons**
#1	*SimSaaS OR "Simulation Software as a service"*	Trying to find general works related to SimSaaS, making an overview of what is done in several fields
#2	*(SimSaaS OR "Simulation Software as a service" OR "Simulation as a service") AND cloud*	Sometimes researchers mention SimSaaS as *Simulation-as-a-service*. However, extending the search to such a term would give to many results, being necessary to filter them for cloud related works
#3	*(SimSaaS OR "Simulation Software as a service" OR "Simulation as a service") AND (hla OR "High Level Architecture")*	Same reasons as the previous one, but now related to HLA
#4	*"agent-supported simulation" AND (hla OR "High Level Architecture" OR cloud)*	Trying to find works where agents coordinate or control the simulation, but relating with HLA or Cloud Computing in order to focus
#5	*"agent-directed simulation" AND (hla OR "High Level Architecture" OR cloud)*	The same reasons as the previous one, but now with the general agent-directed approach

Table 3.2: Query strings used for literature search

to an article. In addition to each record information, it was also characterised by the following attributes:

- Abstract
- Tag with the domain of application (whether Traffic/Transportation, Industrial/Business, Biomedical, Crowd, Collaboration, Others or Generic)
- Tag with the category, which could be more than one (whether SimSaaS, HLA, Cloud, Agent-directed simulation or Agent-supported simulation)
- Tag with the type of paper (whether Theoretical, Practical or Case study)

3.1.4 Literature Analysis and Synthesis

"After collecting sufficient literature on a topic it has to be analysed and synthesised" [vSN+09]. With the help of JabRef and the defined tags for each record, it was possible to investigate:

- *Time analysis* since Jabref allows the sorting of records by publication year;
- *Domain analysis*, since JabRef can filter the records by selecting the intended tag;
- *Category analysis*. To achieve this, when selecting a record, JabRef would show the respective associated tags. So, it was easy to have a clear understanding of the mixture of categories;
- *Paper detailed analysis*. In JabRef, it is possible link a record to the respective file in disk. So, with just a simple click it was possible to quickly read the contents of each record.

3.1.5 Research Agenda

“The research agenda provides the basis for extending the review” [vSN+09]. After all the analysis it was possible to define a taxonomy of the literature review (cf. Section 3.5). The proposed taxonomy illustrates the gaps found and suggests where more research should be done in order to evolve the current scientific knowledge. Indeed, this systematic literature review showed that there is a general lack of work in the SimSaaS field.

3.2 SimSaaS in the Cloud

Simulation Software-as-a-service (SimSaaS) is a relatively new paradigm where simulation software is used in the form of services. Thanks to the significant attention on the Cloud computing paradigm and taking advantage of the several resources provided, it is possible to set simulations into the cloud. This way, it is possible to offer simulations through the cloud by means of services.

Figure 3.1 shows the time distribution of the 30 scientific articles found about SimSaaS. It is clear that SimSaaS is a very recent topic. Just five papers exist prior to 2011 and the first three [YO04][XTCT05][MRZ07] do not specifically use the term SimSaaS, vaguely mentioning *simulation* and *web*. Besides, there is an increased number of studies over the years.

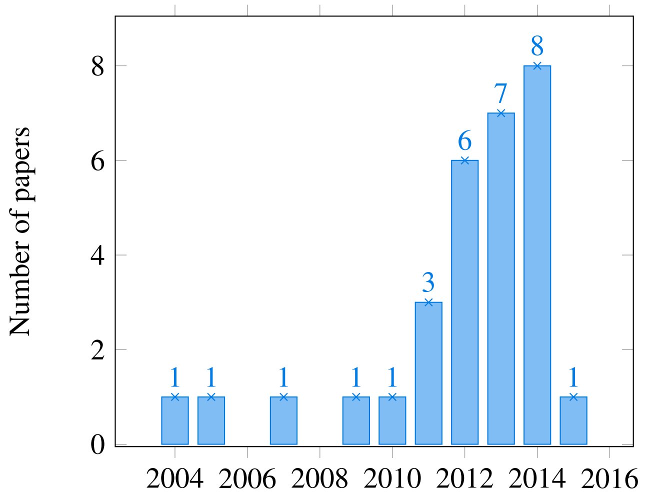


Figure 3.1: Temporal distribution of the papers found about SimSaaS

Although the amount of articles found about SimSaaS is not large, it is wide in which concerns application domains. In the biomedical domain there is a system devoted to simulations of electromagnetic field inside the human body [SCSS12]. In crowd and pedestrian M&S there is also one study proposing a method based on a distributed architecture with simulation in the cloud, where the authors indicate that there is a lack of automation and integration of tools for crowd M&S [WW15]. As Sliman et al. [SCS13] suggest, research dissemination methods suffer from a major drawback: they do not allow publishing simulation code and scripts along with the published papers. Thus, the authors demonstrate an ongoing project in which scientists can openly share their underlying code and data.

In the traffic and transportation domain, there is only a demo using SimSaaS for ITS applications [HKT+10]. In the industry there is a small work made in a contest for HP regarding a migration from a private cloud to a public one [OVKZ14]. Also in the industry, there is a study of cloud simulation in manufacturing [TKT+14] and business process analysis [BDG13].

Finally, a lot of other specific application domains use SimSaaS, mostly in the cloud, and more specifically research articles regarding ontology learning [WW14], truth consistency in data exchanged between services [TM14], an ontology-based layered simulation service description framework [LCHL13], scheduling parallel discrete event simulation jobs [LQC+12] and multi-tenancy architecture on the cloud [TLBE11].

In addition to these specific domain studies on SimSaaS, there are also general ones.

Tsai et al. [TLSS11] proposed a SimSaaS framework incorporating multi-tenancy architecture and scalability for simulation, also presenting a simulation run-time infrastructure. Guo et al. [GBH11] build a framework on top of this work, by proposing a layered framework to support simulation research with Discrete Event System Specification. Previously, a study discussed a test and development environment also using the Discrete Event System Specification Modeling Language and the Service Oriented Architecture framework [MRZ07].

Horuk et al. [HDG+14] propose a framework that wraps source code in a web service and tasking system, deploying the respective applications over different cloud infrastructures or local clusters, automatically configuring virtual machines to execute the applications. A web UI is provided for users to parameterise their applications.

Cayirci refers to SimSaaS using the term *Modelling and Simulation-as-a-service* (MSaaS), also mentioning the top threats [Cay13c]. Besides, he talks about the notions and relations of accountability, risk and trust modelling [Cay13a], as well as MSaaS composition in multi-data center or multi-cloud scenarios [Cay13b]. Cayirci is not the only one using the term MSaaS: Siegfried et al. [SVDBCH14] illustrate potential benefits that may be achieved by MSaaS and challenges that remain to be solved.

Liu et al. [LHQ+12] present work on planting existing simulation software into the cloud, proposing the needs and the architecture of Cloud Simulation, the development of Cloud Simulations services, a modified modelling method and a novel simulation execution mode. As Liu et al. [LQCH12] indicate, pioneers such as Richard M.Fujimoto, Bohu Li, Gabriele D'Angelo and so forth, made considerations on the Cloud Simulation but none has given an overall picture of Cloud Simulation to its full extent. Consequently, the authors propose the general architecture of a Cloud Simulation, which is a SaaS type cloud. This architecture is illustrated in Figure 3.2.

In general terms, it is possible to verify the infrastructure at the bottom of the architecture, where virtualisation plays a vital role. Before reaching the end users, who can be very diverse, the architecture shows a great detail in the specification of the offered services. Authors divide services into three self-explanatory groups: Modeling as a Service, Execution as a Service and Analysis as a Service. During the provision of these services, the available simulation resource can be reused with the aid of the Simulation Resource as a Service. Managing and connecting the baseline infrastructure and services there is the so-called Cloud Operating System.

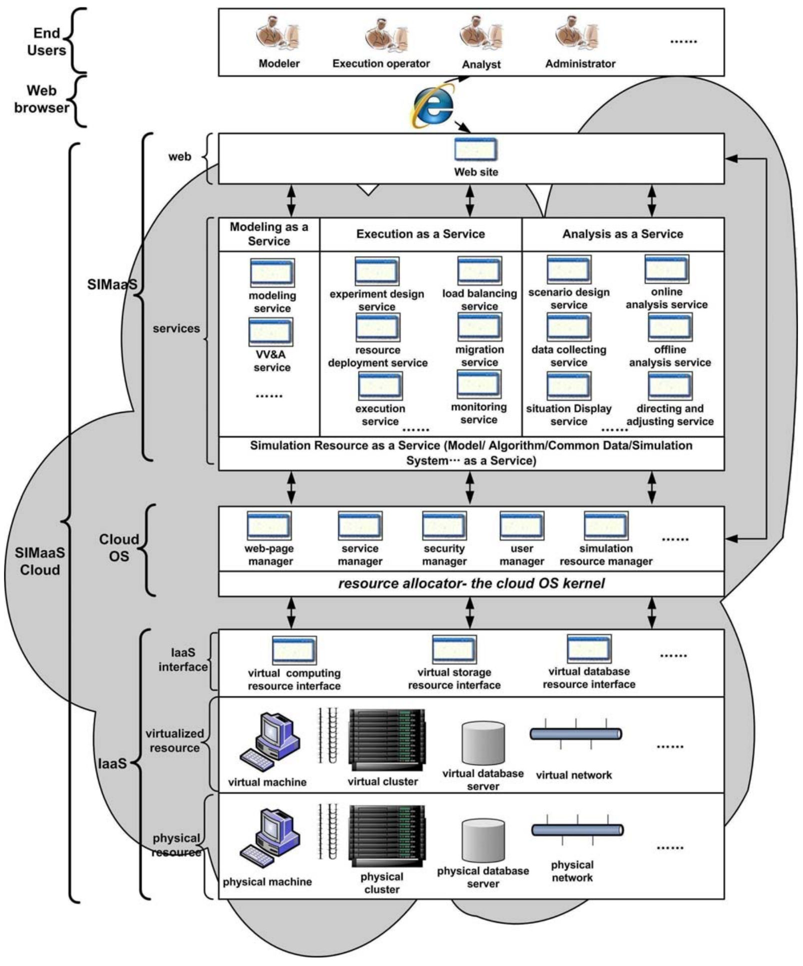


Figure 3.2: The architecture of the Cloud Simulation [LQCH12]

3.3 HLA and services

In the previous Section 3.2 it was possible to present several studies regarding SimSaaS. However, maybe due to being such a recent topic, while specifically discussing SimSaaS, there are almost no references about interoperability among simulators. However, HLA, the current standard for simulator interoperability, has been used in many different studies, such as in agent-based simulations. In the context of this dissertation, it is important to show how it is possible to extend HLA to enable simulations offered in the form of services.

A first approach can be seen in a study by Xie et al. [XTCT05]. Here, the authors propose a framework to extend the HLA to support the Grid-wide distributed simulation. Here, a remote proxy acts on behalf of the federate in interacting with the RTI. It hides the heterogeneity of the simulators, their execution platforms, and how they communicate with the RTI.

A more particular approach proposes a model-driven QoS-aware framework for simulation-based quantitative analysis of business processes [BDG13]. The framework adopts a distributed simulation approach that replicates the service-oriented infrastructure of a business process into the corresponding simulation infrastructure based on the HLA-Evolved standard. It is not clear, but it seems the framework uses a simple server to do so. In another work by the same authors [BDGG13] *HLAcloud*, a model-driven and cloud-based framework was introduced to support both the implementation of a distributed simulation system from a SysML[5] specification and its execution over a public cloud infrastructure.

Perhaps, the most relevant article found is one that presents a possible way for HLA to be integrated with a Service Oriented Architecture (SOA) in the context of a smart building project [DBTS12]. This article discusses the design of an HLA federate for the inclusion of a service oriented smart building controller in the simulation loop. In Figure 3.3 it is possible to see the Simulation Engine Service which is exposed as a RESTful service with several modules.

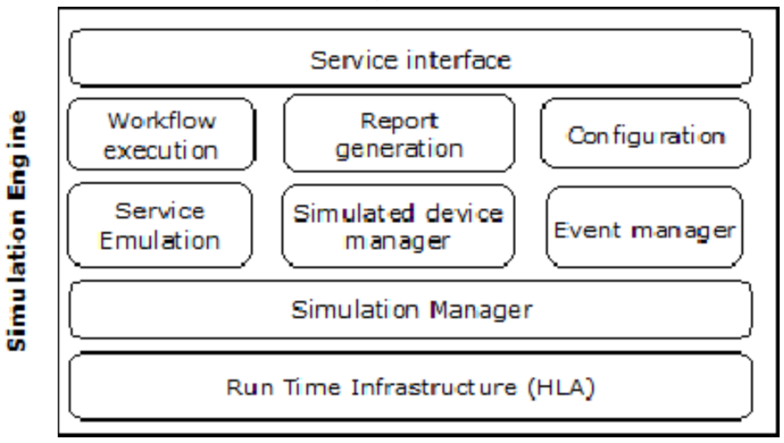


Figure 3.3: The Simulation Engine Service architecture with HLA [DBTS12]

It is important to mention that the Simulation Manager module is a service wrapper on top of the RTI that exposes access to the RTI's federation management via a RESTful API. It deals with the creation, initialisation, deletion, starting, stopping, and execution of simulations. It is not referred to in the picture, but there is also the Simulated Device Federate that is hosted internally by the Simulation Engine. Its role is to manage virtual devices and participate in the HLA federation execution. The result of events and interactions sent to the Simulated Device Federate are routed to the Simulated Device Manager module.

In the same paper, the authors cite Wang et al. [WYL$^+$08] who presented a comparison between HLA and SOA concluding that:

- HLA has good interoperability, synchronisation, and an effective and uniform information exchange mechanism between the communicating components (federates), but lacks several features of web services, such as: the integration of heterogeneous resources, web-wide accessibility across firewall boundaries;
- SOA benefits from loose coupling, component reuse and scalability but lacks a uniform data exchange format and time synchronisation mechanisms;
- The combination of HLA and SOA can extend the capabilities of the two technologies and thus enable integrated simulated and services.

[5]Systems Modelling Language dialect of the Unified Modelling Language (UML)

3.4 Agent-directed Simulation-as-a-service

Agent-directed simulation and, more specifically, agent-supported simulation is used in a huge variety of fields. Nevertheless, in the field of Simulation-as-a-service, just a few examples exist, and even fewer concern the cloud.

Albeit there is such a lack of work regarding agents and the cloud, Jávor and Fűr [JF12] have delineated the main trends of the development of distributed simulation over a grid with intelligent agents, especially over the Internet through Web-based applications, highlighting the concepts of service-based simulation system approach. These trends could be easily extrapolated for the cloud. However, most of the focus is on model sharing, which is not very relevant to this study. Yilmaz et al. [YOA06] also mentioned the importance of agents to simulation (even in gaming) by exploring the relationship of software agents with simulation and games.

An agent-supported simulation is seen in a work by Guo et al. [GHW12], where *handling agents* were used for composition of simulation services with different time and event granularity. In this case, the composition consisted of wildfire and weather simulation services, but no cloud was used.

Two papers try to formalise languages for ambassador agents[6] [TD10][TDPHZ11]. The authors suggest that most current simulation interoperability standards are insufficient as they focus exclusively on information exchange to support the federation of solutions without providing the necessary introspective. HLA is a bit more flexible, as the information to be exchanged is not standardised (it only says how to structure the data), but the focus remains on information that can be exchanged with a system. So, the formal approach to simulation interoperability (using agent-supported simulation) tries to solve this problem.

JAVA Agent DEvelopment Framework (JADE)[7] is a common software framework that simplifies the implementation of multi-agent systems. Web Services Integration Gateway (WSIG) is an add-on for JADE which performs two-way translations between service requests and responses and JADE agent requests and responses. Thanks to this, it was possible to design a service-oriented simulation software framework as part of a broader approach towards generating improved levels of actionable views of situation awareness [SM09]. Figure 3.4 illustrates an application of this framework.

The authors referred that the great benefit of using JADE as the underlying agent development framework is that JADE agent entities can invoke web service functionality hosted outside the JADE runtime environment using normal JADE agent protocols, and that external entities can invoke JADE agent functionality from outside the JADE environment using normal web service protocols. Although the framework is very relevant, the applications were not in the cloud or in a grid.

The only study found that truly implements agents in the cloud indicates that cloud computing can speed-up significantly agent-based M&S to facilitate more accurate and faster results, timely

[6]In these papers an ambassador agent represents simulation services that are candidates to contribute to the solution of a problem

[7]http://jade.tilab.com/

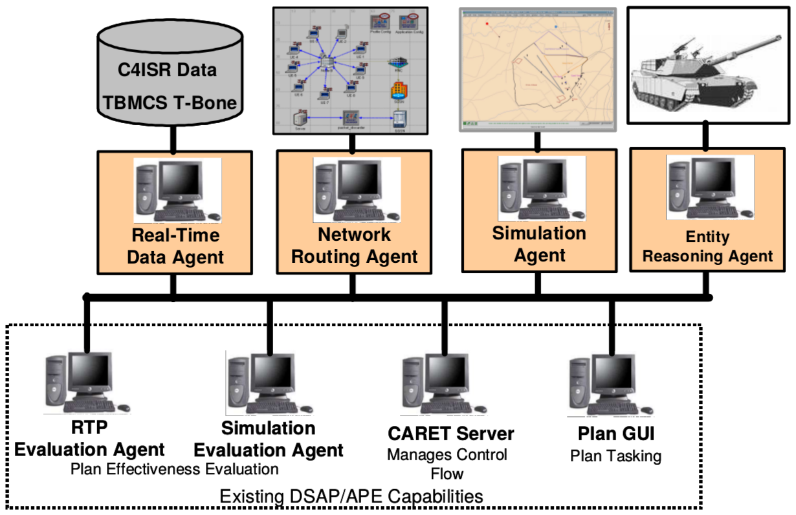


Figure 3.4: Application of the framework described in [SM09]. DSAP/APE is another application of the same framework

experimentation, and optimization [TAK+14]. However, the many different clouds, cloud middleware and service approaches make the development of agent-based M&S in the cloud highly complex.

To conclude this section, in 2004 it was already pointed out that the existing federated simulation environments had limitations in supporting dynamic model and simulation updating [YOA06]. HLA federation development, for instance, requires complete specification of object models and information exchanges before the simulation begins. They also observed and argued that the lack of machine processable formal annotations describing the behaviour, assumptions and obligations of federates is a fundamental roadblock. After a decade, it seems that these problems still remain practically the same.

3.5 Taxonomy of the Research Work

The results of applying a Systematic Literature Review search process were shown in this Chapter.

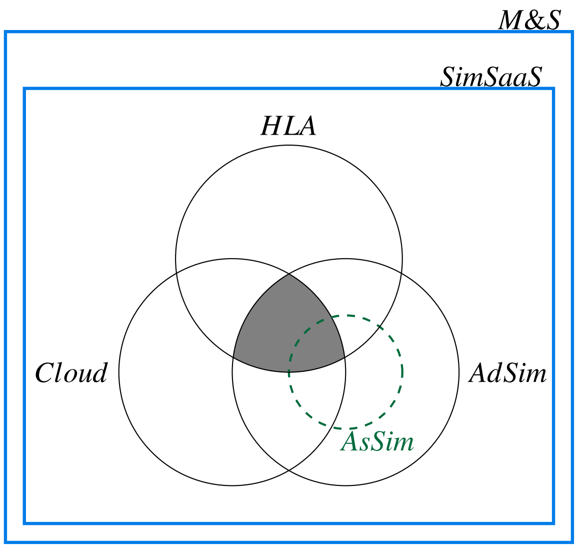
SimSaaS is a trendy term with a high potential to grow further. Thus, it is the ideal time to take advantage of this hype. To wait longer could mean another person doing this, and the contributions of this dissertation would be delivered late. However, there are problems in SimSaaS: there is a lack of automation and integration of tools in M&S [WW15], and research dissemination methods suffer as they do not allow publishing simulation code and scripts along with the published papers [SCS13]. But not everything is bad! The papers found about SimSaaS are wide concerning application domains, despite they are few.

Simulator interoperability is a very explored subject in general, but when it comes to SimSaaS, almost nothing focuses on this. Indeed, Yilmaz et al. [YTFD14], in their panel about the future of research in M&S refer to the distribution of SimSaaS in the cloud as a future research topic. Again, this work comes at the ideal time as SimSaaS is still a very vague topic, lacking specificity in the different domain applications.

HLA is another term referenced a lot in the literature since the first complete version (HLA 1.3) was published in 1998. However, again, there is a few work regarding the extension of HLA to allow simulation services in general and in the cloud. Nevertheless, HLA solely has some disadvantages [YOA06][TD10].

Although cloud computing can speed-up significantly agent-based M&S to facilitate more accurate and faster results [TAK+14], agents are not an exception and there is an absence of work involving putting agents in the cloud to support SimSaaS. Nonetheless, an example was shown where it was possible to develop an agent-supported simulation to bear out SimSaaS [SM09].

In summary, there are a lot of gaps in the literature concerning SimSaaS, SimSaaS in traffic and transportation, SimSaaS in the cloud, HLA in the cloud, solutions to HLA restrictions, agents to support SimSaaS and agents in the cloud. Therefore, these will be on the front line of this work.



(AdSim: Agent-directed Simulation, AsSim: Agent-supported Simulation)

Figure 3.5: Diagram representing the taxonomy of the research work. Grey area is where this dissertation will focus

A taxonomy of the research found make the gaps identified more allows for gaps to be identified more clearly. Figure 3.5 illustrates the taxonomy in the form of a Venn Diagram. Every article is about M&S, more precisely SimSaaS. So, there are studies that only mention SimSaaS, which was the case with some articles of Section 3.2. Then, inside the SimSaaS topic, research can focus on Cloud (cf. Section 3.2), HLA (cf. Section 3.3) or Agent-directed simulation (cf. Section 3.4). In the particular case of Agent-directed simulation, there is a subset regarding Agent-supported simulation. As some works can address more than just one term, the representation in the form of a Venn Diagram was chosen to illustrate these junctions.

Having this taxonomy in mind, and knowing that there are few studies regarding SimSaaS and junctions of the other topics, this work will precisely aim to strike the junctions of every topic, as described in Chapter 4. It should be noted that the platform that will be developed not only covers the Agent-supported simulation, but also the general Agent-directed simulation as, for instance, agent models could be simulated.

3.6 Summary

A literature review is presented, focusing on four distinct yet related topics: SimSaaS, Cloud Computing, HLA and Agents. This literature was conducted through systematic method which is explained. Finally, a taxonomy of the research work is presented in the form of a Venn Diagram for a clear visualisation about which topics can (and should) have synergies among them. This taxonomy could be used as a conceptual framework for future developments. In the next chapter a platform which makes a synergy with all these concepts is presented.

Chapter 4

Proposed Solution: Oculum

After the preliminary background of Chapter 2 and the gap analysis of the Literature Review presented in Chapter 3, it is possible to design the general architecture of an agent-directed transportation simulation platform.

This platform will be called *Oculum*, which means *eye* in Latin: the eye that observes, studies and analyses a virtual and parallel reality existing with the simulations. In Appendix A it is possible to see the *Oculum*'s logo. The different sized circles try to illustrate that *Oculum* is made of different elements (the simulators) that are part of a whole. The circles also resemble the planets around us, and that is because *Oculum* is a cloud-based platform.

This chapter presents a general overview about what the platform and the main involved concepts should be, specifying some more concrete scenarios later on. For the sake of clarification, a formal definition regarding models is presented. Finally, the subset of functionalities selected for implementation as a proof of concept are shown, as well as the development milestones needed to achieve them.

4.1 Platform Overview

According to the gaps previously found in the literature review, it is proposed an integrated transportation simulation platform relying every term (SimSaaS, HLA and Agent-directed simulation), as it will offer simulation in the form of services, using HLA for interoperability of simulators and agents for collaboration. The general architecture of this platform will be similar to the one described in Figure 3.2. The main differences in relation to the latter are that this platform will be adapted in order to support HLA in the *Virtual Resources* tier and Agents in the *Cloud Management* tier. Figure 4.1 shows the generic architecture of the proposed platform considering the three main tiers that differ from the generic cloud simulation.

The scientific community needs collaboration in its pursuit of multidisciplinary achievements. Thus, the scientific community has developed a first sharing approach by creating public repositories

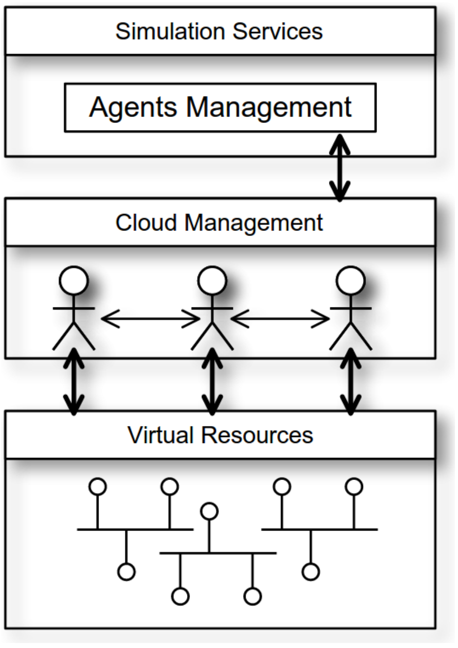


Figure 4.1: The main tiers of the general architecture of *Oculum* that are different to the ones mentioned in Figure 3.2

of datasets, which are sustainable, shared and ever-evolving. However, there are other roles/stakeholders that would also benefit from such datasets, such as public decision-makers and the industry alike. Nevertheless, while they are all interested in testing their own algorithms/calculations, it is only the scientific community that generally adopts the philosophy of data sharing. Besides data, there are also processes, methods and plans, which are even less commonly shared.

In this multi-stakeholder and multi-resource philosophy, HLA already supports the inclusion of resources because a federate is sufficiently general to consider not only simulators but also databases, data loggers, and other resources. Therefore, the so-acclaimed agent-directed approach could be used to enable collaboration between experts, sharing not only data but also processes. But why the interest in sharing? What is the interest in creating this shared community above the cloud? The answer may be simple and consists in generating knowledge and innovation.

Simple sharing methodologies are, for example, personal pages in the platform for each researcher with the created models, similar suggested models, integration of models of other researchers and performed simulations with obtained results. Nevertheless, the agent-directed paradigm could provide better options. For instance, some researchers have their own legacy tools, which run some complex and less optimised simulations, producing outputs. It would be much better if these tools were implemented in the platform with the possibility to deploy agents that act on behalf of the researcher, that initialise these tools and even generate graphics from the resulting output. It would be like an avatar. In practice, these are services that exist in agents: the possibility to start and stop everything, collect data, perform data analysis and even choose the output to serve as the input to other tools.

How can the researcher implement and deploy their own agents? The platform itself in the cloud allows that! As there is a methodology like HLA to interoperate simulators, a design methodology would also exist for each researcher to implement their own agent/avatar. In summary, this is more than just a scientific environment for empirical science. There is the need for it to be more than that in order to value stakeholders like public decision makers and the industry in general, as they could implement their own simulations in the platform.

There are many fields where such a platform can enhance simulations. For instance, cloud computing is becoming increasingly deep-seated in our lives, and as Mark Weiser once said, "The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it" [Wei91]. In fact, living labs can be supported by simulation, as the people in these cases would be receiving stimuli from other virtual realities, supported by the platform.

In order to formally represent *Oculum*, its conceptual model is presented in Figure 4.2. In short, it has three main concepts: Simulation, Personal Page and User. A Simulation has associated at least one simulator, which could have a Map and executes inside a virtual machine. A Simulation (finished or still running) is a concretion of one or more models and so a simulator represents one or more model. The User has its own personal page where, as previously described, all the used models and finished simulations are presented. A user, which only has one personal page, can deploy several agents, which are characterised by their cognition, effectors and sensors. A ternary association is necessary in the diagram: there is a simulation for a pair of one output and personal page, there is none or several outputs for one simulation and personal page, and only one personal page for a simulation and respective output.

4.2 Platform Scenarios

The general concepts and architecture of *Oculum* are now more clear. Therefore, it is possible to reference some more scenarios which are evidences of its potential. It is now evident that it enables easier visualisation and steering of any simulation at any point in the world. In fact, it can leverage integrated multi-domain simulations in the area of Artificial Transportation Systems, Artificial Societies or even Symbiotic Simulations[1].

The principal scenario in *Oculum* is launching simulations by choosing models. As mentioned previously, a simple view of HLA could be a whiteboard of parameters which are changed (published) and read (subscribed) during simulation. Hence, when a user launches a simulation it has to choose, for each model, the attributes that are published and subscribed. It also has to select other options depending of the nature of the simulation, such as whether it wants logging, the map used, the quantity of vehicles and their routes, or attributes to track.

In *Oculum*, the concept of *model* is very generic: for example, it can be a microscopic model of a driver or even a nanoscopic model of a car engine. It can also be a FOM, a route, car flows, etc. This generalisation of models requires some constraints in the platform, since a model could be

[1] In a symbiotic simulation real time data is collected from a physical system. Simultaneously, results from the simulation experiments are used to control the same physical system

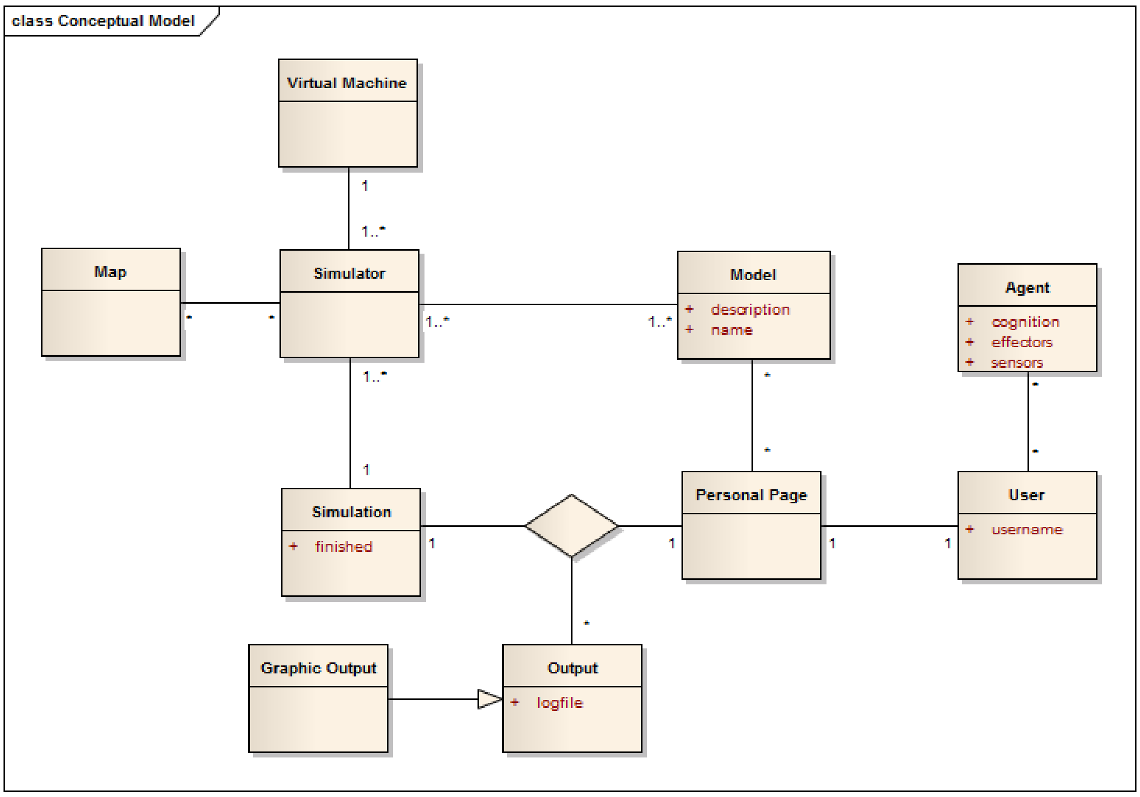


Figure 4.2: *Oculum*'s conceptual model

used just in some simulators. If a researcher wants a model to be implemented in other tools, it will be possible to download the classes and generic code that *Oculum* understands and which the researcher will need to extend in a programmatic way.

The integration of models is a process in which a user chooses different models in order to conduct a simulation. Figure 4.3 clarifies this process by presenting its main components in the form of a class diagram. On the one hand, there are the models related to HLA: the *FOMInteraction*s and the *FOMObjectClass*es that have specific attributes needed for HLA interoperability. On the other hand, there are the others which are rather related with the simulators and that can be the four previously described models in Section 2.1. The grey classes are not real models, but only important information: attributes are needed for a FOMObjectClass specification and a model may work with some simulators but not with others and hence need implementation (code) associated.

Agents can serve simple functionalities such as recovering from failures or exploring the platform to see whether the outputs or maps used by a user are being used by others. Even so, it is necessary to explicit how deployment of agents could work. In order to be easy for researchers, it should enable visual programming language[2] facilities or regular programming languages for more advanced users. In such facilities three main items need to be explored:

[2]Visual Programming languages allow users to create programs by manipulating program elements graphically instead by writing them textually

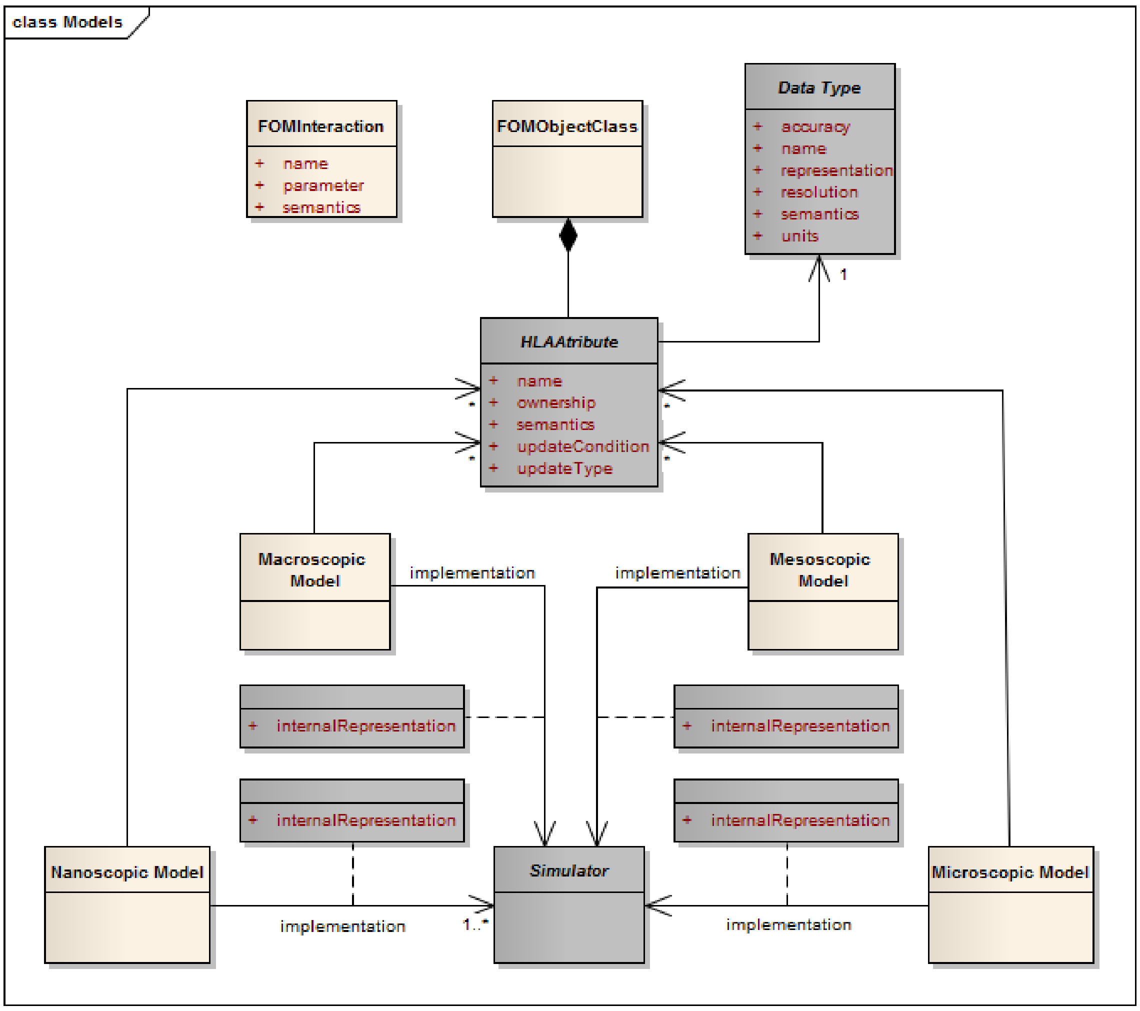


Figure 4.3: Main components in integration of models

- **Perceptions** of the environment, that could be the existing models in the platform and their attributes, simulations' parameters and respective outputs;
- **Actions** on the environment such as launching tools and simulations, graphics generation, screen captures at some simulation step, integration of models;
- **Cognition** is decided by the researcher and is expected to implement his or her own expertise. For instance, with simple reactive agents, cognition may be written in form of what-if rules in which for some kind of perceptions some actions must be executed.

Collaboration experience can be enhanced by using collaborative tools such as real time chats, online model editors/integrators and real-time experiences visualisation. Two or more researchers could beforehand agree on collaborative simulations, where each one can change them while still running. In these cases, when a change occurs, a simulation could be forked for future analysis in order to see whether the change was of interest.

In relation to general panels in the platform, some must be pointed out. One for simulation management concerning the instances being prepared, running and in queue. Another for administration tasks such as user management and platform consumption. And finally one for map selection where it is possible to choose parts of the map to include in the simulation (e.g. traffic lights, points of interest and streets).

4.3 Formal Definition Regarding Models

Some formalisation can be made regarding the problems about models.

Let S be the set of all running or finished simulations in the platform and M the set of all models. A simulation can be seen as several models running. If there are n simulations:

$$S = \{S_1, \dots, S_k, \dots, S_n\}, k, n \in \mathbb{N}, S_k \text{ is simulation k}$$
$$S = \left\{ \bigcup_{k=1}^{n} m_k \middle| m_k \in M \wedge \#M \leq n \right\}, \#S \leq n$$

A model is a simplification of reality and so it is like an aggregation of characteristics. It can output some characteristics of what is representing (C_o) to a simulation or it may need some other characteristics in order to properly represent the reality (C_i). With M being the set of all models and C the set of all characteristics that can be used by a model:

$$\forall m \in M, m = (C_i, C_o), C_i, C_o \subseteq C, \#C_o > 0$$

When a model m_z needs input characteristics ($\#C_i > 0$), a model m_w is said to $help$ it by the following:

$$help : M, M \rightarrow \{0, 1\}$$
$$\forall m_w, m_z \in M, \forall C_{i_w}, C_{o_w}, C_{i_z}, C_{o_z} \subseteq C,$$
$$helps\left[m_w = (C_{i_w}, C_{o_w}), m_z = (C_{i_z}, C_{o_z})\right] \Leftrightarrow C_{o_w} \subseteq C_{i_z} \wedge C_{i_w} = \varnothing \wedge m_w \neq m_z$$

Given this view on models, formalisation of a simulation S must be corrected:

$$S = \left\{ \bigcup_{k=1}^{n} m_k \middle| m_k \in M \wedge \#M \leq n \wedge \#S \leq n \right|$$
$$\left. \nexists m_k = (C_{i_k}, C_{o_k}) \in M \middle| \#C_{i_k} > 0 \wedge \nexists m_w \in M \middle| \neg helps(m_w, m_k) \right\}$$

4.4 Proof of Concept

Due to time constraints it is not possible to implement the whole platform in the context of this master's dissertation. It is then imperative to define a subset of significant and relevant functionalities to implement as a proof of concept of the platform. Based on the taxonomy presented in Figure 3.5, a good proof of concept should use all the terms. Therefore, the proof of concept will try to prove the following concepts:

- **The platform**, which must have an infrastructure for simulation in the cloud in form of services;
- **Interoperability of simulators**, by enabling HLA in the infrastructure previously defined;
- **Agent-supported simulation**, by creating simple agents which create simulations automatically;
- **Integration of models**, by allowing the user or the agent to choose different models to launch a simulation. According to the taxonomy, this is more related with the broader concept of Modelling & Simulation.

4.5 Development Milestones

The necessary methodology to achieve the proof of concept is now presented. It is briefly presented in form of phases which will be detailed in Chapter 5.

Phase 01: Infrastructure installation
Create a private cloud which can launch Virtual Machines (VMs). Verify that their components are properly working by establishing communication between different VMs and also with the internet.

Phase 02: Run a multi-resolution simulation
Run a simulation using two different simulators of distinct resolutions to verify whether they produce a valid output. Confirm that inside one VM it is possible to gather information from where the simulation is launched.

Phase 03: Create an agent to launch simulations
Deploy a simple agent in the form of a script which is able to launch the simulation of the previous phase with different input parameters.

Phase 04: Integrate models and generate graphics
Allow a user to also choose the different input parameters of simulations by integrating models. Enable automatic graphics generation from simulation outputs.

4.6 Summary

In this chapter *Oculum* was presented as an agent-directed transportation simulation platform. A subset of features to be implemented as a proof of concept as well as the main phases required were also described. In the following chapters the implementation processes and results performed to successfully create the proof of concept are explained in detail.

Proposed Solution: Oculum

Chapter 5

Implementation

After proposing the general architecture of *Oculum* in Chapter 4, this chapter describes the implementation and development process of the phases mentioned in Section 4.5

This Chapter starts by making an overview of the software chosen to implement the proof of concept by describing OpenStack and Pitch pRTI. The former is presented as the software that will be used for cloud management and the latter will be used for simulator interoperability with HLA. Finally, Simulation of Urban MObility (SUMO) and Electric Bus Powertrain Subsystem (EBPS) will be used to enable the multi-resolution simulation indicated in Section 4.5. They were chosen for this dissertation because they were already used in previous studies that took place in the Artificial Intelligence and Computer Science Laboratory (LIACC) regarding interoperability with HLA.

Next, it describes in detail the development milestones required to achieve the proof of concept. The process in which SUMO and EBPS interoperate through Pitch pRTI is explained. Afterwards, the implementation is explained in two parts: first, the OpenStack's infrastructure which enables the existence of *Oculum*, and subsequently, the way to enable the agent-supported simulation paradigm. Finally, an illustration of the obtained results using agents is shown.

5.1 Development Software Overview

5.1.1 OpenStack

OpenStack is an open-source and free software platform for cloud computing management firstly released on 21st October 2010. Briefly, it is deployable as an IaaS solution and aims to deliver solutions for all types of clouds by being simple to implement and massively scalable. A series of interrelated programs, or services, support the overall OpenStack solution with APIs available. Table 5.1 provides a list of OpenStack services as stated in the official Installation Guide for the Juno release[1].

[1]http://docs.openstack.org/juno/install-guide/install/apt/content/index.html

Service	**Project Name**	**Brief Description**
Dashboard	Horizon	Provides a web-based self-service portal to interact with underlying OpenStack services
Compute	Nova	Manages the lifecycle of compute instances in an OpenStack environment.
Networking	Neutron	Provides Network-Connectivity-as-a-Service for other OpenStack services and an API to define networks and the attachments into them.
Object Storage	Swift	Stores and retrieves arbitrary unstructured data objects via a RESTful, HTTP based API.
Block Storage	Cinder	Provides persistent block storage to running instances. Its pluggable driver architecture facilitates the management of block storage devices.
Identity Service	Keystone	Provides an authentication and authorization service for other OpenStack services.
Image Service	Glance	Stores and retrieves virtual machine disk images. OpenStack Compute makes use of this during instance provisioning.
Telemetry	Ceilometer	Monitors and meters the OpenStack cloud for billing, benchmarking, scalability, and statistical purposes.
Orchestration	Heat	Implements an orchestration engine to launch multiple composite cloud applications based on text file templates readable and writable by humans.
Database Service	Trove	Provides scalable and reliable Cloud Database-as-a-Service functionality for both relational and non-relational database engines.

Table 5.1: OpenStack services

Usually there are around two releases every year and OpenStack is ever-evolving and used both in private and public clouds from personal to business environment. Although its ecosystem is increasing every year, it is very robust and there are several ways to joining it, such as by being a simple individual member or even by becoming a sponsor. Other ways include participating in the tools provided by OpenStack: a site for asking and answering questions, a wiki with the core projects, a bug reporting website, mailing lists and even social networks.

More than 500 hundred companies and more than 25 400 people in 165 countries currently support the OpenStack which already has more than 20 millions lines of code. It usually has two summits every year and a lot of other events to allow users and developers to meet personally and stay in touch. For all these reasons OpenStack was chosen as the software for cloud management instead of others.

5.1.2 Pitch pRTI

Pitch pRTI is a software developed by Pitch Technologies (Pitch), a company founded in 1991 which is a leading provider of interoperability products, services and solutions for development of distributed systems. Pitch's family of products is being used by complex simulation programs in governments and industries worldwide such as BAE SYSTEMS, NASA, Boeing and MITSUBISHI.

It plays an active role in the M&S Community, and is an active member of the Simulation Interoperability Standards Organization (SISO).

Pitch pRTI is a commercial RTI which is compliant with all the HLA versions including the last one, HLA Evolved. It provided APIs for C++, Java and even Web services through an add-on. It works on most of the operating systems, including the major versions of Windows, Linux and MAC OS. Its interface has some important features such as a local graphical user interface with visualisation of federations in the form of a graph, FOMs, network statistics, federate details and time graphs.

It gives 400 000 updates per second and 0.13 milliseconds latency on standard computers having a proven robustness in NATO Training Network projects and in the Viking 14 C2 exercise[2]. It provides complete material to users who want to use it. These materials include a free tutorial and samples with federates and FOMs to have an overview of the software. In Figure 5.1 it is possible to see the graphical user interface of Pitch pRTI.

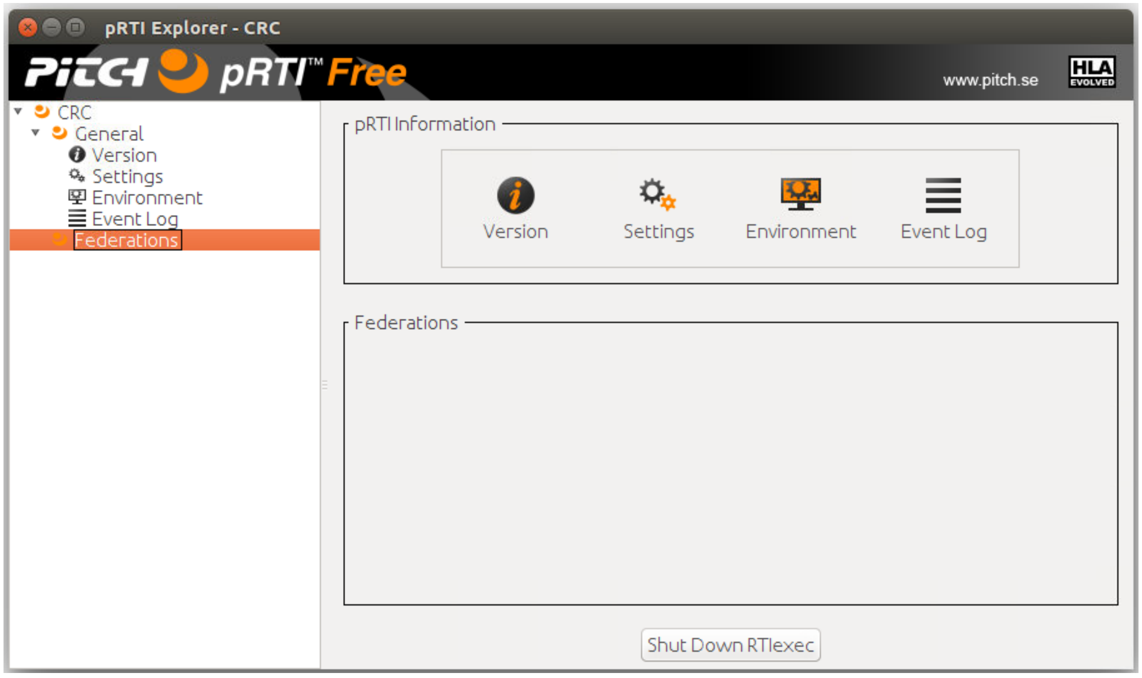


Figure 5.1: Pitch pRTI interface

Apart from Pitch pRTI, there are several other tools with RTI implementations but most of them either do not implement the full HLA specifications, have limitations for free use or are discontinued. The only software that seems to stand against Pitch pRTI is the open-source The PoRTIco project[3]. However, it is currently in a migration process of servers and documentation, without the full HLA Evolved specification. Consequently, Pitch pRTI was chosen.

[2]VIKING 14 is a training platform designed to prepare civilians, military and police together for deployment to a peace or crisis response mission area. The exercise is multidimensional, multifunctional and multinational

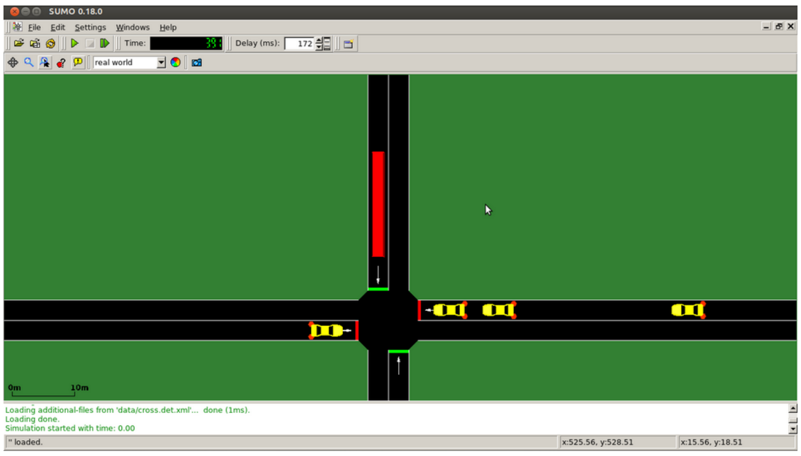
[3]http://www.porticoproject.org/

5.1.3 SUMO

SUMO (Simulation of Urban MObility) is an open-source program (licenced under GPL[4]) for microscopic traffic simulation. It is mainly developed by the Institute of Transportation Systems, located in the German Aerospace Center.

Among other features, it allows for the existence of different types of vehicles, roads with several lanes, traffic lights, graphical interface to view the network and the entities that are being simulated, and interoperability with other applications at runtime through an API called TraCI. Moreover, the tool promotes itself as fast, still allowing a version without a graphical interface where the simulation is accelerated putting aside visual concerns and overheads [BBEK11].

In Figure 5.2 it is possible to visualize the SUMO's graphical interface with a running simulation. It is possible to point out almost all specified features: vehicles stopped at the traffic light as well as a long vehicle entering an intersection.



```
Loading additional-files from 'data/cross.det.xml'...  done (1ms).
Loading done.
Simulation started with time: 0.00
" loaded.                                  x:525.56, y:528.51    x:15.56, y:18.51
```

Figure 5.2: SUMO working

SUMO is widely used in the scientific community with a high number of papers referencing it. Nevertheless, SUMO was also used to provide traffic forecasts for authorities of the city of Cologne during the Pope's visit in 2005 and during the 2006 FIFA World Cup, as well as in several national and international projects. Table 5.2 shows the packages that are installed with SUMO as indicated in the official website[5].

5.1.4 EBPS (Electric Bus Powertrain Subsystem)

The Electric Bus Powertrain Subsystem (EBPS) is a mathematical model of an electric bus subsystem implemented in MATLAB Simulink [PRRA12].

MATLAB (MATrix LABoratory) is a proprietary high-level language and interactive environment used all around the world by many engineers where models are expressed in familiar mathematical notation. It has a huge variety of add-ons for maths, statistics, optimisation, signal

[4]GNU **G**eneral **P**ublic **L**icense, a free, copyleft license for software and other kinds of work
[5]http://www.sumo-sim.org

Component	Brief Description
SUMO	command line simulation
GUISIM	simulation with a graphical user interface
NETCONVERT	network importer
NETGEN	abstract networks generator
OD2TRIPS	converter from O/D matrices to trips
JTRROUTER	routes generator based on turning ratios at intersections
DUAROUTER	routes generator based on a dynamic user assignment
DFROUTER	route generator with use of detector data
MAROUTER	macroscopic user assignment based on capacity functions

Table 5.2: SUMO packages

processing, communications, control systems, image processing, computer vision, computational finance, parallel computing and application deployment.

Simulink is built on top of MATLAB and is a block diagram environment for multi-domain simulation and model-based design. As with MATLAB, Simulink also has the possibility to be extended with add-ons like physical modelling, control system design, code generation, real-time simulation and testing, simulation graphics and reporting.

EBPS is a nanoscopic model with several subsystems representing the vehicle's powertrain, the energy that is recovered from regenerative braking and the energy that is absorbed from the braking in the batteries and super-capacitors. Although it was initially modelled in continuous time, this system was modified to a discrete model [MKS+13]. Figure 5.3 shows the main susbystem of EBPS in the Simulink graphical interface.

This model receives values of velocity as input. The main outputs it produces are:

- **Power** required for each instant of the simulation;
- **Total Energy** spent to complete a simulation;
- **Battery Charging**, that is, the total energy absorbed by the braking system during the simulation.

5.2 Detailed Development Milestones

Now that all the software that will be used in the proof of concept is presented, the development milestones of Section 4.5 are described in detail.

Phase 01: Infrastructure installation

1. Install OpenStack
2. Launch two simple instances and verify connectivity between them and the internet using a *ping* command

Phase 02: Run a multi-resolution simulation

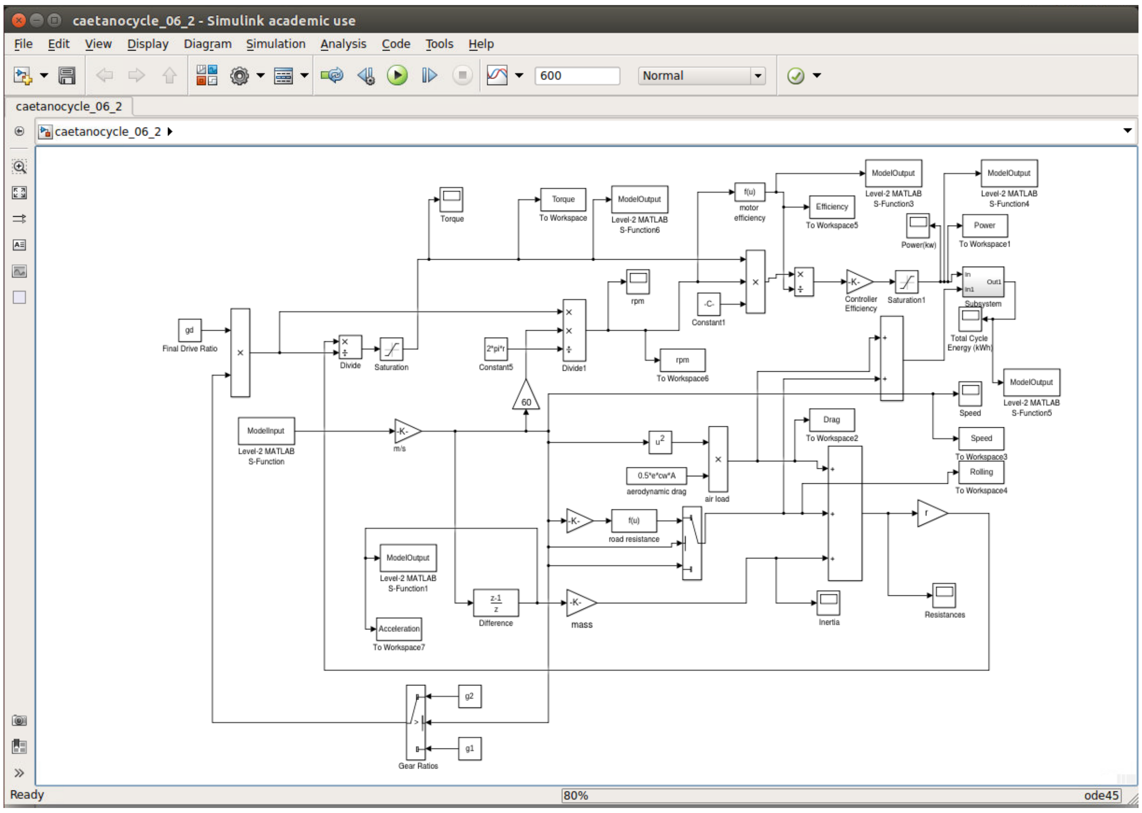


Figure 5.3: The main subsystem of EBPS model in Simulink graphical user interface

1. Create a VM with Pitch pRTI linking SUMO and EBPS in order to run a multi-resolution simulation
2. Adapt the VM to enable its usage by OpenStack
3. Launch an instance with this VM using OpenStack's metadata service which enables it the possibility to retrieve instance-specific data. Confirm that this data is correctly read inside the VM

Phase 03: Create an agent to launch simulations

1. Create a script which automatically launches two simulations with instance-specific data that make the simulations different
2. Adapt the VM to read the instance-specific data so that the two simulations are in fact different

Phase 04: Integrate models and generate graphics

1. Make an interface to the user so it is him which chooses the differences between the automatic simulations launched by the script
2. Change the script so it can generate graphics with simulation's outputs

5.3 Simulation Scenario

EBPS and SUMO will be used in the proof of concept to help prove the possibility to interoperate simulators. Every detail about the specification of the FOM objects and other specificities with HLA can be seen in the work of Macedo et al. [MKS+13]. This section will just briefly outline the simulation process for a quick understanding.

Figure 5.4 shows the main interactions during the simulation. The SUMO Federate is the one responsible for creating the federation in pRTI and that which starts/stops the simulation. It is also responsible for saving the outputs of EBPS for later analysis. To do so, the SUMO Federate publishes the interactions in HLA responsible for starting and finishing the simulation while EBPS just subscribes to these interactions. This means that the SUMO Federate can start and stop the simulation while EBPS can only receive the orders to start and stop. EBPS will only need the velocity of the bus during simulation. Consequently, before the simulation starts, SUMO informs pRTI that it will be able to change the attribute velocity during the simulation with the *publishAttribute* method. On the other side, EBPS will just need to receive the updates when the attribute is changed. EBPS informs pRTI of it with the *subscribeAttribute* method. Similarly, all the other attributes outputted by EBPS are subscribed by SUMO so it can save them.

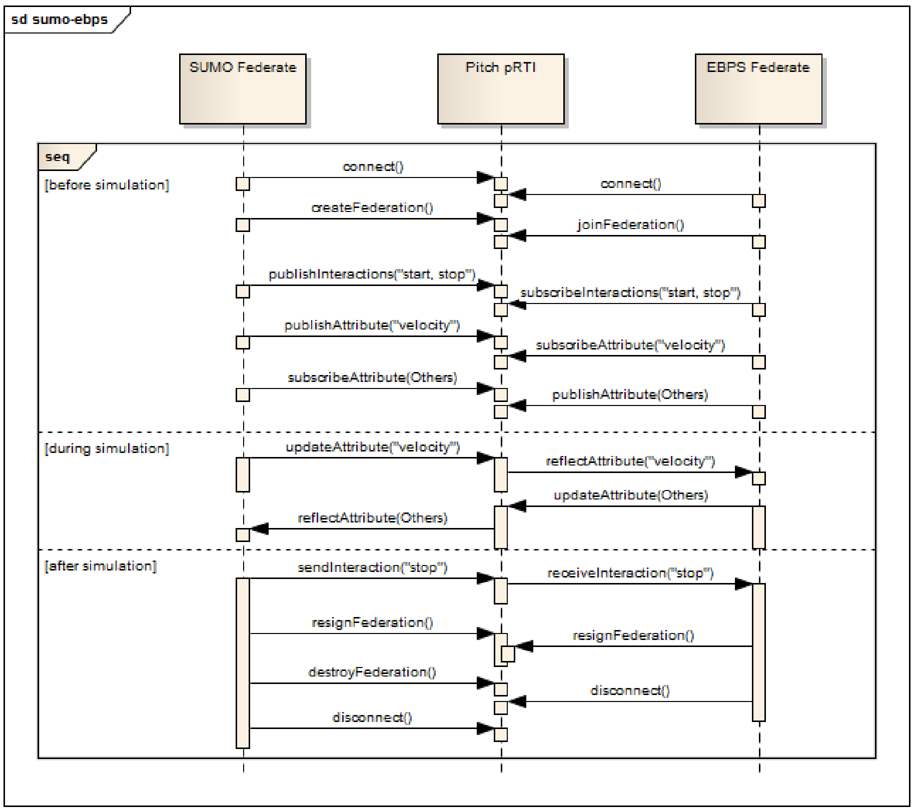


Figure 5.4: Sequence Diagram with the main interactions of SUMO, EBPS and Pitch pRTI during the simulation

Whenever SUMO changes the velocity of the bus, its Federate will call the *updateAttribute* method. pRTI is responsible for making a callback to EBPS with the new value of the attribute. Then, EBPS will be able to make its calculations and in the same way update the other attributes.

When SUMO sees that the simulation is finished, it sends the *stop* interaction so the EBPS Federate will be able to disconnect from pRTI. The SUMO Federate will also disconnect from the pRTI but it will also destroy the federation.

5.4 OpenStack's Infrastructure

OpenStack is very versatile and allows its installation to occur with different types of configurations. The minimal architecture that is recommended is OpenStack installed on top of three physical nodes with some specifications regarding, for instance, network interface controllers. However, due to some constraints, such physical nodes were not available. Although there are some non-official installation guides for just one node, none of them worked as there was always some error concerning one or more services, mainly the Networking service. Hence, VirtualBox[6] was used to virtualise the three physical nodes, which is near to the reality of OpenStack.

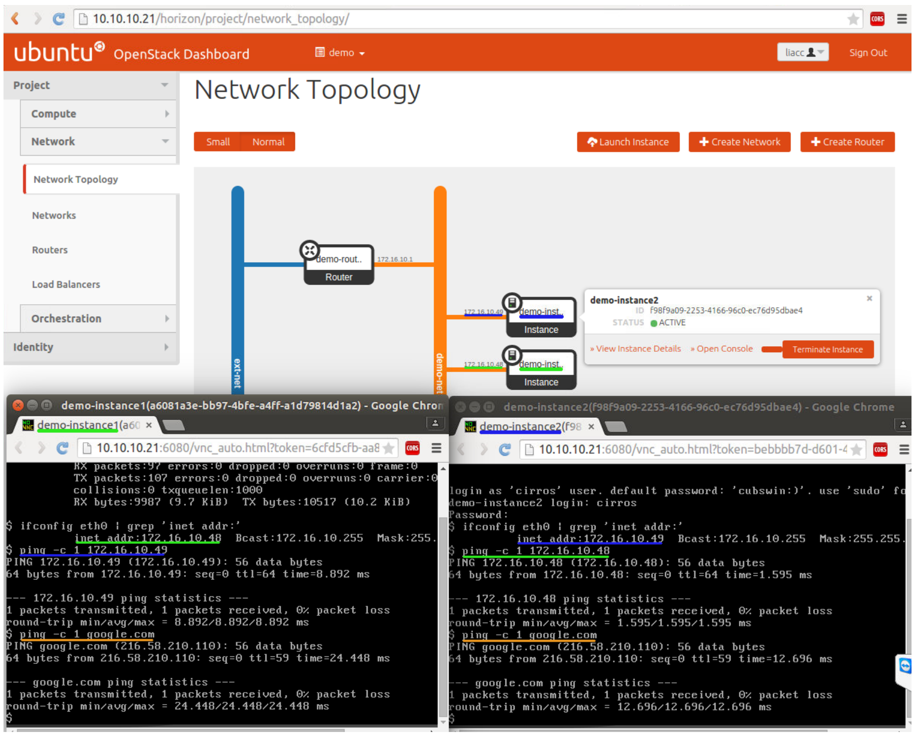
Not every service is needed to successfully achieve the proof of concept. Therefore, Annex B.1 describes the services implemented in each node as well as their characteristics. The control node has the most services running and some management portions of Compute and Networking as it is the main node where interaction with user occurs.

After this, installation is necessary to make some more configurations for using OpenStack. The important ones are enabling the CirrOS image[7], configuring networks and updating default security groups. Regarding the network, it was created inside OpenStack, an external network and an internal network where every image will be launched. In order for each image to have access to the internet, a router is necessary to connect the internal and external networks. These configurations are easily achievable by accessing the Control Node terminal and running the commands depicted in Annex B.2.

After checking that every service is running in OpenStack, accomplishment of *Phase 01* is finished after launching two instances and verifying connectivity. By accessing OpenStack's Dashboard, the CirrOS image previously created was launched in two instances, *demo-instance1* and *demo-instance2*. As it is possible to see in Figure 5.5, they are shown in Horizon's Network Topology view with automatically assigned IPs and are able to *ping* each other and the internet.

After the installation of OpenStack, the next step is to create a VM with SUMO, EBPS, Pitch pRTI and every code needed to enable a multi-resolution simulation. Due to the OpenStack constraints, that image must be created outside of it with tools such as *virt-manager*[8], and also by installing *cloud-init*[9] inside the image. After that, it is only necessary to upload it to OpenStack through the Image Service (Glance). With the Compute Service (Nova) it was possible to launch the image and run the simulation successfully.

[6]http://www.virtualbox.org
[7]CirrOS is a minimal Linux distribution designed as a test image for cloud environments
[8]*virt-manager* is a desktop application for managing virtual machines
[9]*cloud-init* is a multi-distribution package which handles early initialisation of a cloud instance



```
          RX packets:97 errors:0 dropped:0 overruns:0 frame:0
          TX packets:107 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:9987 (9.7 KiB)  TX bytes:10517 (10.2 KiB)

$ ifconfig eth0 | grep 'inet addr:'
          inet addr:172.16.10.48  Bcast:172.16.10.255  Mask:255.
$ ping -c 1 172.16.10.49
PING 172.16.10.49 (172.16.10.49): 56 data bytes
64 bytes from 172.16.10.49: seq=0 ttl=64 time=8.892 ms

--- 172.16.10.49 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 8.892/8.892/8.892 ms
$ ping -c 1 google.com
PING google.com (216.58.210.110): 56 data bytes
64 bytes from 216.58.210.110: seq=0 ttl=59 time=24.448 ms

--- google.com ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 24.448/24.448/24.448 ms
$
```

```
login as 'cirros' user. default password: 'cubswin:)'. use 'sudo' fo
demo-instance2 login: cirros
Password:
$ ifconfig eth0 | grep 'inet addr:'
          inet addr:172.16.10.49  Bcast:172.16.10.255  Mask:255.255.
$ ping -c 1 172.16.10.48
PING 172.16.10.48 (172.16.10.48): 56 data bytes
64 bytes from 172.16.10.48: seq=0 ttl=64 time=1.595 ms

--- 172.16.10.48 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 1.595/1.595/1.595 ms
$ ping -c 1 google.com
PING google.com (216.58.210.110): 56 data bytes
64 bytes from 216.58.210.110: seq=0 ttl=59 time=12.696 ms

--- google.com ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 12.696/12.696/12.696 ms
$
```

Figure 5.5: Two instances communicating in OpenStack and their visualisation in Horizon's Network Topology

OpenStack has a metadata service which allows instances to retrieve instance-specific data through an IP address and also injection of files into the instance file system. All these informations are specified in the moment of launching an instance. Using the *cURL* command[10] it was possible to confirm that the instance had access to the intended data. Thereafter, *Phase 02* is also validated.

5.5 Enabling Agent-supported Simulation

An agent can be implemented in several ways and concerns many issues regarding whether communication with other agents in a multi-agent system is necessary. However, in this case, implementation of an autonomous agent without concerning communication issues with other agents is intended. To do so, a PHP script created because it can be executed in the server-side, just as an agent should be: an autonomous entity working on the side of OpenStack.

The created agent has to make HTTP requests to the OpenStack's API in order to launch the intended simulations. The first request is to authenticate and obtain a token which will be used in all the subsequent HTTP requests. All the data is in the JSON format. The next one is done to actually launch the instance specifying the metadata with the simulation specifications. Figure 5.6 summarises the main interactions required by the agent to launch one simulation. After the instance is launched, it requests the models information and it is only then that the simulation can be run. When it finishes, the resulted outputs are sent to the *Oculum* API which later notifies the agent. The

[10] *curl* is terminal tool which enables, besides other things, to transfer data from a server IP

same process described in the Figure is repeated to launch the same simulation but with different input parameters.

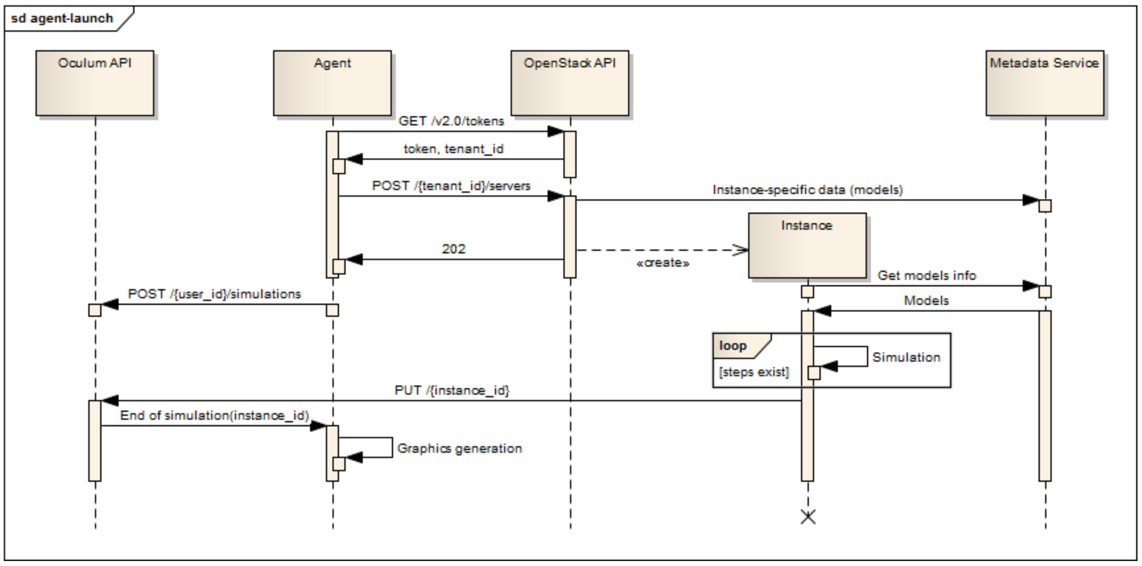


Figure 5.6: Sequence Diagram with the main needed interactions to launch an instance

With *Phase 03* concluded, it is necessary to enable an easy customisation of input parameters by a user or, in other words, to integrate different models, in order to achieve *Phase 04*. From Figure 4.3 it is possible to instantiate a specific group of models needed for the proof of concept and regarding integration of EBPS and SUMO. Figure 5.7 clarifies the instantiation made and the respective models' hierarchy, in the form of a class diagram. The brown entities are the actual models than a user can choose. It is possible to see two *FOMInteraction* models, *Start* and *Stop*, and a *FOMObjectClass* model, the *BusEngine*. Basically, all these models together represent the integration explained in Section 5.3.

The *Bus* model is characterised by its route and its driver behaviour. A driver may be described with different attributes such as acceleration and maximum speed, but while an aggressive driver accelerates more and can reach higher speeds, a calm driver is the opposite, and even so they are all described by the same attributes.

As described in Figure 4.3, every model has an internal representation. The models regarding HLA functionality have parts of a FOM specification, which is represented in the form of a eXtensible Markup Language (XML) file. The internal representation of EBPS and Bus is the code itself. Finally, for the car flows and driver behaviours, the representation is an XML file which is understood by SUMO.

In summary, to illustrate the process of integration of models, let us take the proof of concept. Concerning HLA, a user would choose every *FOMInteraction*s and the *BusEngine* which has already the needed attributes to exchange in HLA. Next, it chooses the *EBPS* model as well as the *Bus* model and a Car Flow Model (for example intense traffic with aggressive drivers). After all the models are chosen, the platform needs to verify if the models match in terms of HLA attributes and to ask the user which attributes will be published/subscribed by each model. With this, the platform

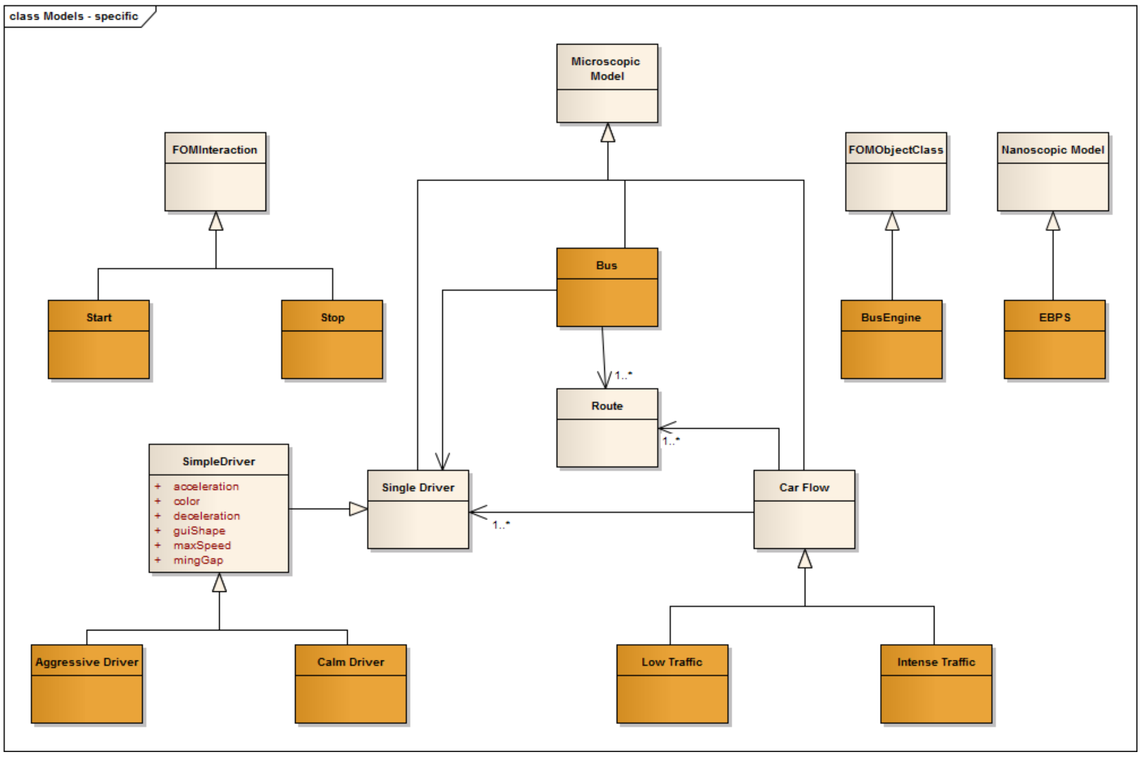


Figure 5.7: Models (in brown) used in the proof of concept and respective hierarchy

is finally able to launch the simulations as the user wants. This process is the first approach needed for an easy deployment of agents because the integration is translated in different actions that the agent will perform.

Although gathering the instance-specific data was explained previously, it should also be mentioned that inside each instance a bash script will be needed to coordinate the simulation. It is only after everything is ready that the SUMO Federate may start the simulation. The following code snippet shows how it waits for pRTI.

```bash
# (...)
screen -d -m -L pRTI1516e-cmdline-gui &
echo "pRTI initiated. Waiting for it..."

while sleep 1
do
  if fgrep --quiet "pRTI>" screenlog.0
  then
    echo "pRTI is ready!"
    break
  fi
done
echo "Going to SUMO..."
# (...)
```

Launching the pRTI's command in a bash script forces the existence of a terminal screen. This way, the command *screen* was used in line 2. The "*-d -m*" flags make *screen* to start in detached mode which is useful for scripts just like this one. The "*L*" flag enable the output of pRTI to go to a file named *screenlog.0*. When pRTI is ready it will output "*pRTI>*" to the console and that is why in line 7 the script is searching for this specific output inside *screenlog.0*. When found, the cycle is broken and the script continues to SUMO.

Waiting for SUMO and EBPS is different from waiting for pRTI, as they do not need a terminal screen attached. Although the bash script waits in the same manner from SUMO and EBPS, they have some differences. The following code snippet shows the process for EBPS:

```
 1  # (...)
 2  java main.SumoFederate < input &
 3  # (...)
 4
 5  java main.EBPSFederate > "output.log" 2>&1 &
 6  echo "EBPS initiated. Waiting for it to be ready..."
 7  while sleep 1
 8  do
 9    if fgrep --quiet "EBPS ready" "output.log"
10    then
11      echo "s"$'\n' >> input
12      break
13    fi
14  done
15  # (...)
```

First, in line 2 it is necessary to turn a file into an input to the SUMO Federate. SUMO Federate will be waiting for a "*s*" character to run. In line 5 the EBPS Federate is started, redirecting its standard output stream to the file *output.log*. The EBPS Federate will output "EBPS ready" when it is ready, thus in line 9 the script is searching for this in the file. When finally it appears in the file, the script will output a "*s*" character to the file *input*, which will be read by SUMO Federate, and the simulation can finally start.

5.6 Results Illustration Using Agents

After the simulation ends, as depicted in Figure 5.6, the agent generates graphics for a better analysis by the user who launched the simulation. To illustrate the kind of graphics that an agent can produce, let's consider that it launches two simulations, one with an aggressive driver and another with a calm driver (cf. Figure 5.7). In both cases the bus travels through the same route and the traffic flow is equal.

The agent can present the several outputs in the form of average, standard deviation or even how they changed during simulation. It can also cross several outputs for a comparative analysis.

The graphics are generated with the help of Google Charts[11].

In Figures 5.8 and 5.9 it is possible to see that the different models for drivers behaviour influence the average of the acceleration and velocity. As it would be expected, an aggressive driver had higher accelerations and velocities. Converting the values of Figure 5.9, the velocity average of the calm driver was approximately $30.74km/h$ while the average velocity of the aggressive driver was approximately $36.41km/h$, which is in consistency with the simulation scenario of Section 5.3.

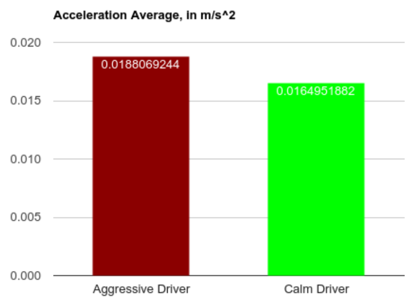


Figure 5.8: Graphic generated by an agent with the acceleration average during two simulations with different driver behaviours (in m/s^2)

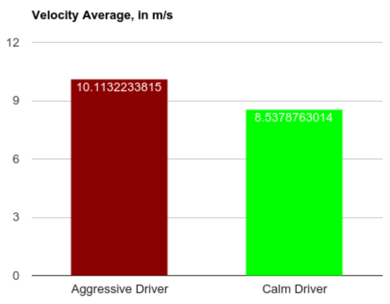


Figure 5.9: Graphic generated by an agent with the velocity average during two simulations with different driver behaviours (in m/s)

As it was mentioned, the agent can also generate graphics with the variations of a variable during the simulation time. Likewise, a different analysis between the calm and aggressive driver behaviour can be performed. In Figure 5.10 it is possible to compare the acceleration profiles. Analysing these graphics, a user could see that occurred what was supposed, namely, the aggressive driver having higher peaks of acceleration and performed in shorter periods of time.

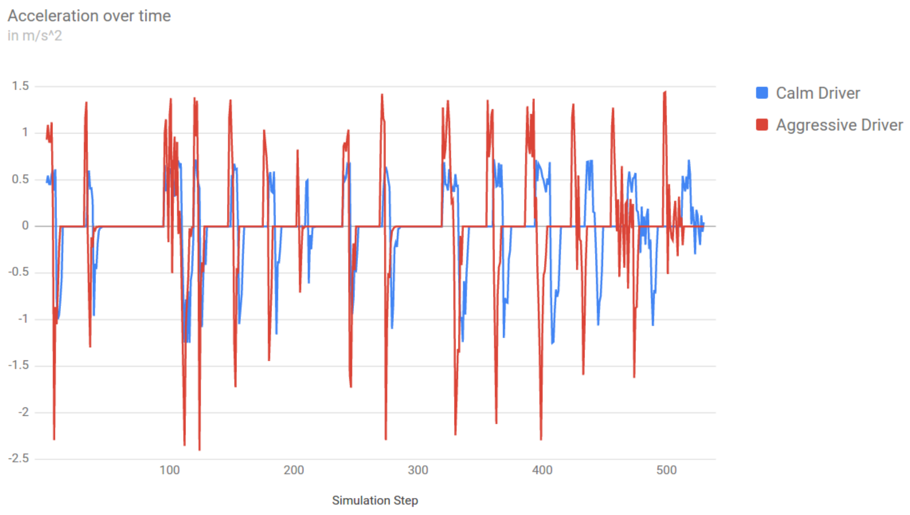


Figure 5.10: Graphic generated by an agent with the accelerations of a calm and aggressive driver during a simulation

[11]https://google-developers.appspot.com/chart

5.7 Summary

In this chapter the development milestones and the way SUMO and EBPS interoperate are described in detail. The implementation process carried out is presented by first mentioning the infrastructure and then by describing the agent-supported simulation paradigm. Some outcomes of using agents are illustrated. In the next chapter results regarding experiments with the proof of concept are shown.

Chapter 6

Discussion of Experiments and Results

In the platform, a common scenario is the launch of VMs with simulations that users want. For them that is transparent: they launch and just have the results at the end, just like a service should be. However, it is necessary to experiment how the simulations should run in order to have the best performance possible at the backend. This chapter shows some preliminary results regarding the way VMs should be launched.

To analyse performance, the simulation scenario of Section 5.3 is used without OpenStack. As it was mentioned in the previous chapter, OpenStack is running on top of three VMs which are simulating a cloud and hence it does not make sense to use a virtualised OpenStack to evaluate performance issues. When OpenStack launches a VM, it only serves as a cloud management middleware between the user and the hypervisor. So, the direct use of virtual machines in a physical node will serve to guide the usage of OpenStack in terms of what should be the characteristics of a compute node and how it could affect simulation.

The simulation scenario is used in two experiments. In the first experiment, the performance of one simulation directly in the physical node and in a VM with different CPU memory allocations is evaluated. The second, to evaluate the performance degradation in a set of different scenarios in which many VMs are running in a single physical node. The physical node is a computer with the characteristics shown in Table 6.1.

Characteristic	**Value**
CPU	Intel® Core™i7-2600K
Clock Speed	3.40GHz
RAM Size	9.7GB
Graphics Card	Gallium 0.4 on NVE4
Operating System	Ubuntu 14.04 64-bit
Number of cores	8 logical (hyperthreading)

Table 6.1: Computer characteristics used in the experiments

6.1 First set-up: Running one simulation

In Table 6.2 it is possible to visualise the results from running one simulation directly in the Personal Computer (PC) or in a VM. Each VM had Ubuntu 14.04 as operating system, which could explain the impossibility to run the simulation with 128MB of RAM, as it is less than the recommended minimum system requirements. The boot time corresponds to the time it takes to initialise all the tools (SUMO, EBPS and pRTI) while the simulation time is the time of the simulation itself.

Infrastructure	**Boot time**	**Simulation time**	**CPU percentage average during simulation**
Directly in PC	01:26	02:52	≈17%
VM with 128MB of RAM	–	–	–
VM with 256MB of RAM	03:32	04:23	≈13 %
VM with 512MB of RAM	01:44	02:52	≈13 %
VM with 1024MB of RAM	01:34	02:51	≈13 %
VM with 2048MB of RAM	01:28	02:52	≈13 %
VM with 3072MB of RAM	01:23	02:50	≈13 %
VM with 4096MB of RAM	01:23	02:51	≈13 %

Table 6.2: The general results for the first set-up. Time is in the MM:SS format

There is no clear difference between the performance of running the simulation directly in the PC or within a VM with 512MB or more of RAM. The constant value of 13% of CPU is explained as the VM allocates one core of the eight available regardless of the memory allocated. Indeed, there are some cases where the simulation can run in less time than directly in the PC. This information supports what was already said: performance on virtualised infrastructures is getting closer to performance directly on a machine [IOY$^+$11]. The simulation only performs clearly worse performance when available memory is about and beneath 256MB of RAM.

This experiment solely serves as a baseline to understand the minimum requisites of the given simulation scenario in order to be compared hereafter. Furthermore, it is shown that the given simulation scenario is quicker enough to be used in the test experiments of the next section.

6.2 Second set-up: Running multiple simulations

A common situation of a simulation platform in the cloud is its constant usage with users asking to launch simulations. One person may think that the best way to launch and maintain VMs in a single physical node is to take advantage of the multi-core architecture that the computer under testing has by having several VMs running simultaneously.

Thus, it was experimented how the computer behaves with multiple running simulations at the same time. To do so, VMs with 2048MB of RAM were chosen. With this value it was possible to quicker test the allocation of all the memory of the PC: around 10 GB. The process was simple: firstly, two VMs were launched and both simulations were started at the same time; secondly, three

VMs were launched and the three simulations were started at the same time, and so forth. Table 6.3 shows the overall results.

Simultane-ously running VMs	**Boot time average**	**Boot time standard deviation**	**Simulation time average**	**Simulation time standard deviation**	**Total execution time average**
2 VMs	02:50.50	00:00.71	02:50.50	00:00.71	05:41.00
3 VMs	04:17.33	00:12.66	02:51.67	00:00.58	07:09.00
4 VMs	06:02.50	00:11.56	02:52.00	00:00.82	08:54.50
5 VMs	14:09.40	03:51.11	02:52.80	00:00.84	17:02.20
6 VMs	25:30.17	08:32.08	02:59.83	00:10.41	28:30.00
7 VMs	88:07.29	24:21.25	03:52.57	00:28.14	91:59.86

Table 6.3: Results of running multiple simulations in VMs of 2048MB of RAM at the same time. Time is in the MM:SS.00 format

There are two relevant situations in Table 6.3. Firstly, with five VMs, when these VMs try to allocate all the memory available in the CPU. It is when the boot time average and standard deviation increase substantially. Secondly, with seven VMs, where all the eight logical cores in the PC get full (seven for the VMs and one for the computer itself). In this case boot and simulation times increase too much and the time to run a complete simulation is too much high.

So, what should be the best configuration for a compute node used by a cloud management software such as OpenStack? Considering that one simulation takes a total of 4 minutes and 20 seconds (cf. Figure 6.2), it would take 26 minutes to execute six simulations in a row and 21 minutes and 40 seconds to execute five simulations in a row. As it was shown, with six VMs running simultaneously the average time of a single simulation is about 28 minutes and 30 seconds which corresponds to approximately 10% more time spent in simulation. Thus, there is no advantage of running six simulations simultaneously. With five VMs running simultaneously the average time of a single simulation is about 17 minutes and 2 seconds, which is 21% quicker. There are advantages of running simulations simultaneously up to five VMs, in this specific case.

Nevertheless, from four VMs to five VMs running simultaneously there is a tremendous growth in the standard deviation and in the boot time which more than duplicates. The simulation time remains almost unchanged for the different configurations being the boot time the most affected. As the 4 VMs configuration has significant better times, it should be preferred to the 5 VMs configuration. Indeed, with 4 VMs there are still more or less 2GB left in the processor for other necessary tasks than could be needed by the operating system.

To better understand what could be a more generic case, Table 6.4 shows the results when running multiple VMs with 1024MB of RAM simultaneously.

To the contrary of the VMs described in Table 6.3, with VMs of 1024MB it is obviously possible to have more simulations running simultaneously with better times. Similarly, the average time and standard deviation grow more when coming to the limit of 10GB of the processor. Nonetheless, up

Simultane-ously running VMs	Boot time average	Boot time standard deviation	Simulation time average	Simulation time standard deviation	Total execution time average
2 VMs	02:09.00	00:11.31	02:53.50	00:00.71	05:02.50
3 VMs	01:59.67	00:07.37	02:51.33	00:00.58	04:51.00
4 VMs	04:28.75	01:39.44	02:52.75	00:00.96	07:21.50
5 VMs	09:30.80	01:07.13	02:57.40	00:03.36	12:28.20
6 VMs	13:14.50	01:29.53	03:01.33	00:05.69	16:15.83
7 VMs	18:24.71	03:09.91	03:13.14	00:07.27	21:37.86
8 VMs	24:57.13	04:08.03	03:21.38	00:10.06	28:18.50
9 VMs	38:48.11	08:37.11	03:26.33	00:24.59	42:14.44
10 VMs	113:26.47	23:59.38	04:00.80	00:29.26	117:27.27

Table 6.4: Results of running multiple simulations in VMs of 1024MB of RAM at the same time. Time is in the MM:SS.00 format

to eight VMs simultaneity makes simulations quicker than in sequence. The total execution time of 8 simultaneously VMs is 20% quicker than running them in a row.

Although it is possible to have better times using simultaneity, those values are still very bad. Due to resource concurrence, there is congestion in accessing the shared resources. A common way to avoid this is to defer in time the execution of the simulations, just like a pipeline. Figure 6.1 illustrates the idea of pipeline applied to the launching of three VMs. As the boot time is the one with the worst results, simulations are deferred considering this attribute.

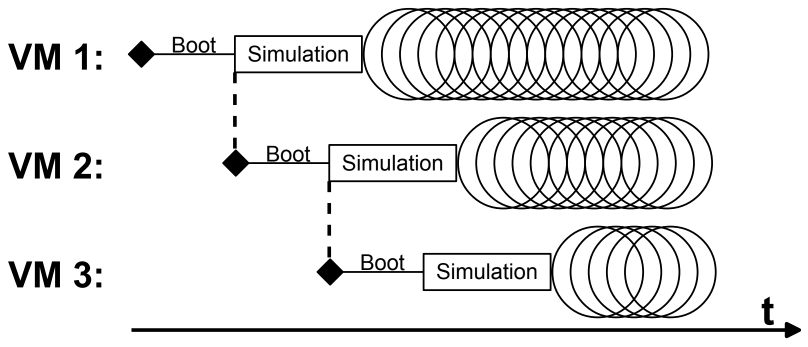


Figure 6.1: Example of pipeline execution with three VMs

The advantage of this process is that, for example, in the first VM of Figure 6.1, after the simulation has ended, there is still plenty of time to be spent in other simulations while the other VMs are coming after. Thereby, the boot time is executed just once and the subsequent simulations have the tools always ready to start. In this dissertation only one proof of concept/simulation scenario was developed hence, to recreate something similar, after the simulation ended the *stress* package[1] was executed, representing high computational simulations.

[1]The *stress* package is a terminal tool which can impose load on and stress test systems by making high floating points computations, constant calls of *malloc()* and *free()* for memory allocation and even the *sync()* system call which commits to disk all data in the kernel filesystem buffers

In Table 6.5 a comparison is made between the results of pipeline and simultaneously executions in the limit cases of Table 6.3. From the user point of view, the average execution time with pipeline is much better in general. However, from the platform point of view, the execution time of all the simulations using 4 VMs (the approach pointed out as being the best in this context) are greater than with simultaneity. Even so, pipeline brings one advantage that simultaneity does not bring in the way it was presented: more time to execute other simulations. Consequently, the launching of VMs should be made using the pipeline process.

Running VMs	**Execution time average with pipeline**	**Execution time average simultaneously**	**Execution time of all simulations with pipeline**	**Execution time of all simulations simultaneously**
4 VMs	05:05.25	08:54.50	11:48.00	09:10.00
5 VMs	06:15.80	17:02.20	19:24.00	23:41.00
6 VMs	07:09.33	28:30.00	28:33.00	42:05.00

Table 6.5: Comparing results between the pipeline and simultaneously approach with VMs with 2048MB. Time is in the MM:SS.00 format

6.3 Discussion

These analysis are indicative and can orient the distribution of workload in the cloud, where there are several physical nodes and it is necessary to distribute simulations in each one. It seems that the number of cores in a processor does not substantially influence the simulation times as opposed to the allocated RAM.

One thing pointed out is that the tools could be initiated just once, hence removing the main problem in these analysis, the boot time. By this way, some may think that pipeline is not necessary because most of the times VMs will not be launched. This could not be true. Each VM has a specific set of tools to be used and a physical node may not have sufficient memory to run all the possible set of tools every time. Besides, there are some set of tools than can be used more often than others and consequently a physical node could balance and have more than one VM running with the same set of tools. In these cases, when launching more than one VM it is necessary to consider pipeline to reach a better performance.

The problem can be seen as follows. Each physical node has two queues: the queue for launching new VMs using the pipeline process and the queue to launch new simulations with a set of tools that are already booted in some VM. If this last queue grows to much, some balancing can be made and thus simulations go to the other queue.

6.4 Summary

This chapter started by presenting the experiments that were conducted in order to orient the future distribution of work in the cloud. To do so, some experimental scenarios were set up using the developed proof of concept to analyse the best way to manage the simulations in the infrastructure. The chapter concludes with a brief discussion which tries to bring some clarification about how this analysis could help.

Chapter 7

Conclusions

7.1 Overview

The recent evolutions in Cloud Computing and SaaS brought a panoply of challenges and opportunities to M&S and even to the agent-directed simulation paradigm. Nevertheless, it seems that this *cloud* SimSaaS is still very *sparse* because it is not fully explored yet. This dissertation appeared as an attempt to *densify* it.

To do so, this document started to explain its context, motivation and objectives. From that, it was necessary to show the six research questions that guided the systematic literature review, necessary to build the body of knowledge required to achieve the intended objectives.

Next, for the overall document comprehension, it started *from the ground* in Chapter 2 or, in other words, with the preliminary back*ground* of concepts which are covered in the subsequent chapters.

With the concepts clarified, it was possible to go *to the cloud* in Chapter 3 by verifying the current research level necessary to create a *cloud*-based simulation platform, that is, regarding SimSaaS, Cloud Computing, HLA and Agent-directed simulation. From this, it was possible to conclude that there is a lack of published research regarding these terms. All the reviewed studies were compared and summarised leading to the creation of a taxonomy of the research work. This taxonomy allowed a clear visualisation of the research gaps that should have synergies in the fields of M&S and the front research opportunities for the next years.

With these gaps perfectly identified, it was proposed a general architecture of an agent-directed transportation simulation platform which establishes a synergistic relationship between all involved fields so as to bridge such gaps. *Oculum* is the name of this platform and it is generally presented in order to have a perfect understanding about how it should be used and what use cases it has. For a clear and effective explanation of the problem, a formal definition regarding models is presented. Finally, the subset of relevant features is selected in the form of a proof of concept that was developed in the context of this dissertation.

The implementation of this proof of concept was further explained by first indicating the used software (OpenStack, Pitch pRTI, SUMO and EBPS) and the chosen simulation scenario. Next, it was explained how the main infrastructure was implemented using OpenStack and how the agent-supported simulation paradigm with integration of models was applied. The type of graphics and analysis that the agents enable is also shown.

Some experiments were performed on the implemented platform, in order to determine the best approach to manage and launch VMs. Such analysis is very important to achieve better performances in simulations by using the best set-up with the developed infrastructure.

7.2 Contributions and Discussion of the Research Questions

This dissertation fulfilled three main contributions:

- **Technological contribution**. OpenStack, agents and HLA were put together with Pitch pRTI, SUMO and EBPS to bridge all the gaps mentioned in the proposed taxonomy of the research work. With these technologies, synergies were promoted.
- **Scientific contribution**. The developed platform provides collaboration among experts of M&S with the support of the agent-oriented paradigm. Comparatively to literature and related work, this extends the knowledge already published by emphasising on synergies less explored.
- **Applied contribution**. More generally than the technological contribution, a platform was developed. Such a platform enables scientific simulation, through the cloud, by means of services. Thus, universities and companies previously described with interoperability issues, benefit from such a practical platform without concerning details connecting them.

To support the scientific contribution, two articles were accepted in conferences during the dissertation development. In “A state-of-the-art integrated transportation simulation platform”, a paper accepted in the *4th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, the general architecture of *Oculum* was proposed [ARB15a]. In “Densifying the sparse cloud SimSaaS: The need of a synergy among agent-directed simulation, SimSaaS and HLA”, a paper accepted in the *5th International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH)*, the sparse cloud SimSaaS is shown by the literature review and the taxonomy also presented in this document [ARB15b]. With the front research opportunities for the next years and current research work, it is expected that this paper can leverage the scientific activity in the field, with researchers actually finding useful to make the jump to the cloud. In fact, it will bring advantages, not only to each researcher in particular, but also to the overall simulation scientific community seeking for more knowledge.

Another contribution of this dissertation deals with the discussion of the Research Questions that guided the Literature Review process:

1. How can simulation in the cloud, by means of services, be beneficial to modelling and simulation?
2. What would be an appropriate architecture for a simulation platform, in the cloud, by means of services?
3. How to extend HLA to support agent-directed simulation in the cloud?

4. How to verify whether an agent-directed approach has advantages over the traditional HLA approach?
5. Are there any similar approaches to such a platform and what are the resources needed?
6. Which stakeholders would benefit from a cloud-based simulation platform?

It is impossible to answer the first question in an objective way. However, several ideas can be pointed out. In Section 2.4 it was possible to show how cloud has grown enormously in so many fields, the same way SimSaaS did in Section 3.2. If these areas are growing it cannot mean they are bad, supposedly. So, as Cloud Computing provides huge amount of platforms and storage that simulation needs *per se* with less complexities, joining SimSaaS and cloud computing may benefit M&S.

It was possible to find an objective answer to the second question. Figure 3.2 shows how an architecture of Cloud Simulation by means of services should be. This general architecture first appeared as an attempt to give an overview of Cloud Simulation, which has never been done before [LQCH12]. In Chapter 4, an extension of this architecture is done taking into account all the drawbacks seen in the Literature Review.

It was not possible to find a suitable answer to the third question. Nevertheless, part of the question may be already answered. HLA was already integrated with SOA [DBTS12], and there are now tools that can make agents support services [SM09]. Indeed, the whole Section 3.4 shows studies regarding the use of agents in SimSaaS. Thus, as it was proposed in Chapter 4, the answer may be found in the combination of both rather on the extension of HLA.

The forth question is another one without any final solution in the literature review. The only clue found in there is that several times HLA is described as having many disadvantages, and agents are characterised as having a lot of advantages. Even tough, with this dissertation, it was proposed the use of agents to support some features that would be not possible with just a traditional HLA approach. In this sense, there are advantages over the traditional HLA approach.

Chapter 3 comprises an answer to question five. Several related works with similar approaches were shown and summarised.

The answer to the last question may be subjective. There is not any work which clearly says the stakeholders that would benefit from a cloud-based simulation platform, neither from just a simulation platform. So, the answer to this question must involve the common sense. The most obvious stakeholders are the academic and company professionals which use simulation in their daily researches. However, in a last instance, everyone in the M&S community would benefit from such a platform: with faster and simpler simulations, professionals could work quickly and better. Thus, the outcomes of their research would most likely have better and more relevant quality.

7.3 Further Developments

Many things can be pointed as further developments to the platform. Basically, the ones who will turn the proof of concept into the general view of *Oculum* presented in Chapter 4. Being more specific, the platform should be extended and the general website should be developed, which will bring the main interface to the end user. Likewise, the agents may be further developed with more intelligent features such as learning capabilities, possibility to recommend models, validation/calibration of models, abilities to apply data mining and more intelligent visualisation adaptable to each researcher. The platform should also have collaborative tools such as model editors and results analysis. The creation of agents by a user should be further studied in order to be easy and straightforward.

To leverage the concept of Simulation Software-as-a-Service, emphasising the provided services, some visualisation capabilities of on-going simulations should be provided. For example, the attributes of traffic lights and vehicles can be visualised through time. In what concerns the SUMO simulator there is already an API providing communication with microscopic simulators [TARO10]. A relatively recent work shows how it was possible to retrieve information such as of vehicles and traffic lights in real-time using TraSMAPI [AdARR14]

Regarding OpenStack, it should be explored other services which were not used in this work but that could be useful: Heat, for example, provides some capabilities which make easy to launch different instances in a more automatic way. OpenStack also has a service for monitoring activity that could be helpful if used. Despite Pitch pRTI being very robust and having very good documentation, it is a commercial software with very strong restrictions for the free license. Open-source alternatives such as The PoRTIco project should be tried.

The integration of models is intended to be very generic. However, in the proof of concept just two different simulators were used, which could have led to a solution which is not sufficiently generic. Some other simulators and simulations scenarios should be tried in the platform in order to study this.

It should be better analysed the best way to launch and run VMs in the platform. The actual solution relies in every software already installed in a single VM but this brings some constraints. One the one hand, as the number of supported simulators grow, more different VMs will be needed, even whether some VMs could be bigger and with more simulators installed. On the other hand, if it is necessary more than one simulator instance, that could bring some issues of shared configurations of those simulators. In this sense, a single VM for a single simulator with HLA enabled in the networks could solve these problems and bring more flexibility: at any time, a simulator could join a simulation, which could be useful for collaborative simulations. However, this means more VMs and more virtual networks and hence more overheads with operating systems and network communications.

Passos et al. [PRK11] evaluate several simulation tools in the optic of Future Urban Transport. Indeed, the extension of *Oculum* to provide several other simulators should be done taking into account whether their characteristics are in consonance with current needs such as it was mentioned in Passos et al.'s work.

Finally, a good cloud should be prepared to high peeks of utilisation. The platform infrastructure should be prepared to these situations and scalability tests should be done. To evaluate the acceptance level among the end users, a survey is needed. This could even lead to more good ideas of new and better features.

References

[AdARR14] Tiago Azevedo, Paulo J. M. de Araújo, Rosaldo J. F. Rossetti, and Ana Paula Rocha. JADE, TraSMAPI and SUMO: A tool-chain for simulating traffic light control. In *Proceedings of the 8th International Workshop on Agents in Traffic and Transportation, ATT'14, held at the Thirteenth International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS'14*, volume 1, pages 8–15, 2014. Cited on page 58.

[AFG$^+$10] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, et al. A view of cloud computing. *Communications of the ACM*, 53(4):50–58, Apr 2010. Cited on page 12.

[AJG$^+$15] Abdelzahir Abdelmaboud, Dayang N.A. Jawawi, Imran Ghani, Abubakar Elsafi, and Barbara Kitchenham. Quality of service approaches in cloud computing: A systematic mapping study. *Journal of Systems and Software*, 101(0):159 – 179, 2015. Cited on page 13.

[ARB15a] Tiago Azevedo, Rosaldo J. F. Rossetti, and Jorge G. Barbosa. A State-of-the-art Integrated Transportation Simulation Platform. In *Proceedings of the 4th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS 2015)*, 2015. Cited on page 56.

[ARB15b] Tiago Azevedo, Rosaldo J. F. Rossetti, and Jorge G. Barbosa. Densifying the Sparse Cloud SimSaaS: The need of a Synergy among Agent-directed Simulation, SimSaaS and HLA. In *Proceedings of the 5th International Conference on Simulation and Modeling Methodologies, Technologies and Applications (SIMULTECH 2015)*, pages 172–177, 2015. Cited on page 56.

[ASS$^+$15] Saeid Abolfazli, Zohreh Sanaei, Mohammad Hadi Sanaei, Mohammad Shojafar, and Abdullah Gani. Mobile cloud computing: The-state-of-the-art, challenges, and future research. 2015. Cited on page 13.

[BACT06] Moshe E. Ben-Akiva, C. Choudhury, and Tomer Toledo. Lane changing models. In *Proceedings of the International Symposium of Transport Simulation*, 2006. Cited on page 6.

[BBEK11] Michael Behrisch, Laura Bieker, Jakob Erdmann, and Daniel Krajzewicz. SUMO–Simulation of Urban MObility. In *The Third International Conference on Advances in System Simulation (SIMUL 2011), Barcelona, Spain*, pages 63–68, 2011. Cited on page 38.

[BdASB10] Ana L. C. Bazzan, M. de Brito do Amarante, Tiago Sommer, and Alexander J. Benavides. ITSUMO: an agent-based simulator for ITS applications. In *Proc. of the 4th Workshop on Artificial Transportation Systems and Simulation. IEEE*, 2010. Cited on page 8.

[BDG13] Paolo Bocciarelli, Andrea D'Ambrogio, and Daniele Gianni. 4SEE: A Model-driven Simulation Engineering Framework for Business Process Analysis in a SaaS Paradigm. In *Proceedings of the Symposium on Theory of Modeling & Simulation - DEVS Integrative M&S Symposium*, DEVS 13, pages 31:1–31:8, San Diego, CA, USA, 2013. Society for Computer Simulation International. Cited on pages 20 and 21.

[BDGG13] Paolo Bocciarelli, Andrea D'Ambrogio, Andrea Giglio, and Daniele Gianni. A SAAS-based Automated Framework to Build and Execute Distributed Simulations from SysML Models. In *Proceedings of the 2013 Winter Simulation Conference: Simulation: Making Decisions in a Complex World*, WSC '13, pages 1371–1382, Piscataway, NJ, USA, 2013. IEEE Press. Cited on page 22.

[BK13] Ana L. C. Bazzan and Franziska Klügl. A review on agent-based technology for traffic and transportation. *The Knowledge Engineering Review*, pages 1–29, 2013. Cited on pages 1 and 8.

[BKA05] Wilco Burghout, Haris N. Koutsopoulos, and Ingmar Andreasson. Hybrid mesoscopic-microscopic traffic simulation. *Transportation Research Record: Journal of the Transportation Research Board*, 1934(1):218–255, 2005. Cited on page 7.

[BMZA12] Flavio Bonomi, Rodolfo Milito, Jiang Zhu, and Sateesh Addepalli. Fog computing and its role in the internet of things. In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16. ACM, 2012. Cited on page 12.

[BRM$^+$09] Michael Balmer, Marcel Rieser, Konrad Meister, David Charypar, Nicolas Lefebvre, Kai Nagel, and K. Axhausen. MATSim-T: Architecture and simulation times. *Multi-agent systems for traffic and transportation engineering*, pages 57–78, 2009. Cited on page 8.

[BYV$^+$09] Rajkumar Buyya, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Brandic. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation computer systems*, 25(6):599–616, 2009. Cited on pages 10 and 12.

[Cay13a] E. Cayirci. A joint trust and risk model for MSaaS mashups. pages 1347–1358, Washington, DC, 2013. Cited on page 20.

[Cay13b] E. Cayirci. Configuration schemes for modeling and simulation as a service federation. *Simulation*, 89(11):1388–1399, 2013. Cited on page 20.

[Cay13c] E. Cayirci. Modeling and simulation as a cloud service: A survey. pages 389–400, Washington, DC, 2013. Cited on page 20.

[CNJ$^+$07] Xingchen Chu, Krishna Nadiminti, Chao Jin, Srikumar Venugopal, and Rajkumar Buyya. Aneka: Next-generation enterprise grid platform for e-science and e-business applications. In *e-Science and Grid Computing, IEEE International Conference on*, pages 151–159. IEEE, 2007. Cited on page 12.

[Coo88] Harris M. Cooper. Organizing knowledge syntheses: A taxonomy of literature reviews. *Knowledge in Society*, 1(1):104–126, 1988. Cited on page 15.

[DBTS12] Monica Dragoicea, Laurentiu Bucur, Wei-Tek Tsai, and Hessam Sarjoughian. Integrating HLA and Service-Oriented Architecture in a Simulation Framework. In *Proceedings of the 2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (Ccgrid 2012)*, CCGRID '12, pages 861–866, Washington, DC, USA, 2012. IEEE Computer Society. Cited on pages 22 and 57.

[DM14] Gabriele D'Angelo and Moreno Marzolla. New trends in parallel and distributed simulation: From many-cores to Cloud Computing. *Simulation Modelling Practice and Theory*, 49(0):320 – 335, 2014. Cited on page 8.

[DP08] Hussein Dia and Sadka Panwai. Nanoscopic traffic simulation: enhanced models of driver behaviour for ITS and telematics simulations. In *INTERNATIONAL SYMPOSIUM ON TRANSPORT SIMULATION, 8TH, 2008, SURFERS PARADISE, QUEENSLAND, AUSTRALIA*, 2008. Cited on page 7.

[DRO06] Nuno Duarte, Rosaldo J. F. Rossetti, and Eugénio Oliveira. A communication-based model for perception and action in car traffic simulation. In *18th European Meeting on Cybernetic Science and Systems Research*, pages 731–736, 2006. Cited on page 7.

[FCD12] Sébastien Faye, Claude Chaudet, and Isabelle Demeure. A distributed algorithm for multiple intersections adaptive traffic lights control using a wireless sensor networks. In *Proceedings of the first workshop on Urban networking*, pages 13–18. ACM, 2012. Cited on page 9.

[FERO08a] Paulo A. F. Ferreira, Edgar F. Esteves, Rosaldo J. F. Rossetti, and Eugénio C. Oliveira. A Cooperative Simulation Framework for Traffic and Transportation Engineering. In Yuhua Luo, editor, *Cooperative Design, Visualization, and Engineering*, volume 5220 of *Lecture Notes in Computer Science*, pages 89–97. Springer Berlin Heidelberg, 2008. Cited on page 7.

[FERO08b] Paulo A. F. Ferreira, Edgar F. Esteves, Rosaldo J.F. Rossetti, and Eugénio C. Oliveira. Extending microscopic traffic modelling with the concept of situated agents. In *Proceedings of the 5th Workshop on Agents in Traffic and Transportation (ATT), 7th International Conference on Autonomous Agents and Multi-Agent Systems*, pages 87–93, 2008. Cited on page 7.

[FZRL08] Ian Foster, Yong Zhao, Ioan Raicu, and Shiyong Lu. Cloud computing and grid computing 360-degree compared. In *Grid Computing Environments Workshop, 2008. GCE'08*, pages 1–10. IEEE, 2008. Cited on pages 10, 11, and 13.

[GBH11] S. Guo, F. Bai, and X. Hu. Simulation software as a service and Service-Oriented simulation experiment. pages 113–116, Las Vegas, NV, 2011. Cited on page 20.

[Gee09] Jeremy Geelan. Twenty-one experts define cloud computing. *Cloud Computing Journal*, 4:1–5, 2009. Cited on page 10.

[GHW12] Song Guo, Xiaolin Hu, and Xiaoming Wang. On Time Granularity and Event Granularity in Simulation Service Composition (WIP). In *Proceedings of the 2012 Symposium on Theory of Modeling and Simulation - DEVS Integrative M&S Symposium*, TMS/DEVS '12, pages 38:1–38:6, San Diego, CA, USA, 2012. Society for Computer Simulation International. Cited on page 23.

[GMBLGSCP15] José A. González-Martínez, Miguel L. Bote-Lorenzo, Eduardo Gómez-Sánchez, and Rafael Cano-Parra. Cloud computing and education: A state-of-the-art survey. *Computers & Education*, 80:132–151, 2015. Cited on page 13.

[HDG$^+$14] C.B. Horuk, G. Douglas, A. Gupta, C. Krintz, B. Bales, G. Bellesia, B. Drawert, R. Wolski, L. Petzold, and A. Hellander. Automatic and portable cloud deployment for scientific simulations. pages 374–381. Institute of Electrical and Electronics Engineers Inc., 2014. Cited on page 20.

[HKT$^+$10] J. Härri, M. Killat, T. Tielert, J. Mittag, and H. Hartenstein. DEMO: Simulation-as-a-service for ITS applications. Taipei, 2010. Cited on page 20.

[HX14] Wu He and Lida Xu. A state-of-the-art survey of cloud manufacturing. *International Journal of Computer Integrated Manufacturing*, (ahead-of-print):1–12, 2014. Cited on page 13.

[IEE10a] IEEE. IEEE Standard for Modeling and Simulation (M &S) High Level Architecture (HLA)– Framework and Rules. *IEEE Std 1516-2010 (Revision of IEEE Std 1516-2000)*, pages 1–38, Aug 2010. Cited on page 9.

[IEE10b] IEEE. IEEE Standard for Modeling and Simulation (M &S) High Level Architecture (HLA)– Object Model Template (OMT) Specification. *IEEE Std 1516.2-2010 (Revision of IEEE Std 1516.2-2000)*, pages 1–110, Aug 2010. Cited on page 9.

[IEE10c] IEEE. IEEE Standard for Modeling and Simulation (M&S) High Level Architecture (HLA)– Federate Interface Specification. *IEEE Std 1516.1-2010 (Revision of IEEE Std 1516.1-2000)*, pages 1–378, Aug 2010. Cited on page 8.

[IOY$^+$11] Alexandru Iosup, Simon Ostermann, M. Nezih Yigitbasi, Radu Prodan, Thomas Fahringer, and Dick H.J. Epema. Performance analysis of cloud computing services for many-tasks scientific computing. *Parallel and Distributed Systems, IEEE Transactions on*, 22(6):931–945, 2011. Cited on pages 13 and 50.

[Jab15] JabRef Development Team. *JabRef*, 2015. Cited on page 17.

[JF12] András Jávor and Attila Fűr. Simulation on the Web with Distributed Models and Intelligent Agents. *Simulation*, 88(9):1080–1092, September 2012. Cited on pages 7 and 23.

[KHRW02] Daniel Krajzewicz, Georg Hertkorn, C. Rössel, and P. Wagner. Sumo (simulation of urban mobility). In *Proc. of the 4th middle east symposium on simulation and modelling*, pages 183–187, 2002. Cited on page 7.

[Kle05] L. Kleinrock. A Vision for the Internet. *ST Journal for Research*, 2(1):4–5, November 2005. Cited on page 10.

[LCHL13] Tan Li, Xudong Chai, Baocun Hou, and Bohu Li. Research and Application on Ontology-based Layered Cloud Simulation Service Description Framework. In *Proceedings of the 2013 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, SIGSIM-PADS '13, pages 391–396, New York, NY, USA, 2013. ACM. Cited on page 20.

[LHQ$^+$12] X. Liu, Q. He, X. Qiu, B. Chen, and K. Huang. Cloud-based computer simulation: Towards planting existing simulation software into the cloud. *Simulation Modelling Practice and Theory*, 26:135–150, 2012. Cited on page 20.

[LQC$^+$12] X.C. Liu, X.G. Qiu, B. Chen, Q. He, and K.D. Huang. Scheduling parallel discrete event simulation jobs in the cloud. volume 2012, London, 2012. Cited on page 20.

[LQCH12] Xiaocheng Liu, Xiaogang Qiu, Bin Chen, and Kedi Huang. Cloud-based simulation: The state-of-the-art computer simulation paradigm. In *Proceedings of the 2012 ACM/IEEE/SCS 26th Workshop on Principles of Advanced and Distributed Simulation*, PADS '12, pages 71–74, Zhangjiajie, 2012. IEEE Computer Society. Cited on pages 20, 21, and 57.

[LW55] Michael J. Lighthill and Gerald Beresford Whitham. On kinematic waves. II. A theory of traffic flow on long crowded roads. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 229(1178):317–345, 1955. Cited on page 6.

[Mey03] Paul J. Meyer. What would you do if you knew you couldn't fail? Creating SMART Goals. In *Attitude Is Everything: If You Want to Succeed Above and Beyond*. Meyer Resource Group, Incorporated, 2003. Cited on page 2.

[MG11] Peter Mell and Tim Grance. The NIST definition of cloud computing. 2011. Cited on page 10.

[Mil14] Emiliano Miluzzo. I'm Cloud 2.0, and I'm Not Just a Data Center. *Internet Computing, IEEE*, 18(3):73–77, 2014. Cited on page 12.

[MKS$^+$13] J. Macedo, Z. Kokkinogenis, G. Soares, D. Perrotta, and Rosaldo J.F. Rossetti. A HLA-based multi-resolution approach to simulating electric vehicles in simulink and SUMO. In *Intelligent Transportation Systems - (ITSC), 2013 16th International IEEE Conference on*, pages 2367–2372, Oct 2013. Cited on pages 39 and 41.

[MRZ07] Saurabh Mittal, José L. Risco, and Bernard P. Zeigler. DEVS-based Simulation Web Services for Net-centric T&E. In *Proceedings of the 2007 Summer Computer Simulation Conference*, SCSC '07, pages 357–366, San Diego, CA, USA, 2007. Society for Computer Simulation International. Cited on pages 19 and 20.

[MT07] Lisa Jean Moya and Andreas Tolk. Towards a taxonomy of agents and multi-agent systems. In *Proceedings of the 2007 spring simulation multiconference-Volume 2*, pages 11–18. Society for Computer Simulation International, 2007. Cited on pages 1 and 7.

[NAT13] Athar Nouman, Anastasia Anagnostou, and Simon J.E. Taylor. Developing a distributed agent-based and des simulation using poRTIco and Repast. In *Distributed Simulation and Real Time Applications (DS-RT), 2013 IEEE/ACM 17th International Symposium on*, pages 97–104. IEEE, Oct 2013. Cited on page 9.

[OVKZ14] F. Oblea, A.E. Votaw, S. Kothari, and J. Zeng. Migrating simcloud to HP helion. *HP Laboratories Technical Report*, (36), 2014. Cited on page 20.

[PR12] José LF Pereira and Rosaldo J.F. Rossetti. An integrated architecture for autonomous vehicles simulation. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pages 286–292. ACM, 2012. Cited on page 9.

[Pra09] Carlo Giacomo Prato. Route choice modeling: past, present and future research directions. *Journal of Choice Modelling*, 2(1):65 – 100, 2009. Cited on page 6.

[PRK11] Lúcio Sanchez Passos, Rosaldo J.F. Rossetti, and Zafeiris Kokkinogenis. Towards the next-generation traffic simulation tools: a first appraisal. In *Information Systems and Technologies (CISTI), 2011 6th Iberian Conference on*, pages 1–6. IEEE, 2011. Cited on page 58.

[PRRA12] Deborah Perrotta, Bernardo Ribeiro, Rosaldo J.F. Rossetti, and João L. Afonso. On the potential of regenerative braking of electric buses as a function of their itinerary. *Procedia-Social and Behavioral Sciences*, 54:1156–1167, 2012. Cited on page 38.

[RAKG13] Rosaldo J.F. Rossetti, Joao Emilio Almeida, Zafeiris Kokkinogenis, and Joel Gonçalves. Playing Transportation Seriously: Applications of Serious Games to Artificial Transportation Systems. *IEEE Intelligent Systems*, 28(4):107–112, 2013. Cited on page 1.

[RB99] Rosaldo J. F. Rossetti and Sergio Bampi. A Software Environment to Integrate Urban Traffic Simulation Task. *Journal of Geographic Information and Decision Analysis*, 3(1):56–63, 1999. Cited on page 7.

[RFBO08] Rosaldo J.F. Rossetti, Paulo A.F. Ferreira, Rodrigo A.M. Braga, and Eugénio C. Oliveira. Towards an artificial traffic control system. In *Intelligent Transportation Systems, 2008. ITSC 2008. 11th International IEEE Conference on*, pages 14–19. IEEE, 2008. Cited on page 9.

[RL05a] Rosaldo J.F. Rossetti and Ronghui Liu. Activity-based analysis of travel demand using cognitive agents. In Harry Timmermans, editor, *Progress in Activity-Based Analysis*, pages 139–160. Oxford: Elsevier, 2005. Cited on page 7.

[RL05b] Rosaldo J.F. Rossetti and Ronghui Liu. An agent-based approach to assess drivers' interaction with pre-trip information systems. In *Intelligent Transportation Systems*, volume 9, pages 1–10. Taylor & Francis, 2005. Cited on page 7.

[RL14] Rosaldo J. F. Rossetti and Ronghui Liu, editors. *Advances in Artificial Transportation Systems and Simulation*. Academic Press, 2014. Cited on page 9.

[RLT11] Rosaldo J.F. Rossetti, Ronghui Liu, and Shuming Tang. Guest editorial special issue on artificial transportation systems and simulation. *IEEE Transactions on Intelligent Transportation Systems*, 2(12):309–312, 2011. Cited on page 1.

[ROB07] Rosaldo J. F. Rossetti, Eugénio C. Oliveira, and Ana L. C. Bazzan. Towards a specification of a framework for sustainable transportation analysis. In *Proceedings of the 13th Portuguese Conference on Artificial Intelligence*, pages 179–190. APPIA, 2007. Cited on page 8.

[Rob08] Stewart Robinson. Conceptual modelling for simulation Part I: definition and requirements. *Journal of the Operational Research Society*, 59(3):278–290, 2008. Cited on page 5.

[Sar05] Robert G. Sargent. Verification and validation of simulation models. In *Proceedings of the 37th conference on Winter simulation*, pages 130–143. Winter Simulation Conference, 2005. Cited on page 5.

[SCS13] L. Sliman, B. Charroux, and Y. Stroppa. A new collaborative and cloud based simulation as a service platform: Towards a multidisciplinary research simulation support. pages 611–616, Cambridge, 2013. Cited on pages 19 and 24.

[SCSS12] B. Sawicki, B. Chaber, J. Starzyński, and R. Szmuro. Internet application concept to trivialize EMF biomedical computing. *COMPEL - The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, 31(4):1190–1197, 2012. Cited on page 19.

[SDH$^+$14] Le Sun, Hai Dong, Farookh Khadeer Hussain, Omar Khadeer Hussain, and Elizabeth Chang. Cloud service selection: State-of-the-art and future research directions. *Journal of Network and Computer Applications*, 45(0):134 – 150, 2014. Cited on page 13.

[Sha10] Amir M. Sharif. It's written in the cloud: the hype and promise of cloud computing. *Journal of Enterprise Information Management*, 23(2):131–134, 2010. Cited on pages 1 and 10.

[SM09] Gary Shao and Robert McGraw. Service-oriented Simulations for Enhancing Situation Awareness. In *Proceedings of the 2009 Spring Simulation Multiconference*, SpringSim '09, pages 48:1–48:7, San Diego, CA, USA, 2009. Society for Computer Simulation International. Cited on pages xi, 23, 24, 25, and 57.

[ST70] Joseph William Schmidt and Robert Edward Taylor. *Simulation and analysis of industrial systems*. RD Irwin, 1970. Cited on page 5.

[SVDBCH14] R. Siegfried, T. Van Den Berg, A. Cramp, and W. Huiskamp. M&S as a service: Expectations and challenges. pages 248–257. SISO - Simulation Interoperability Standards Organization, 2014. Cited on page 20.

[SWL11] Jason Sewall, David Wilkie, and Ming C. Lin. Interactive hybrid simulation of large-scale traffic. In *ACM Transactions on Graphics (TOG)*, volume 30, page 135. ACM, 2011. Cited on page 9.

[TAK$^+$14] Simon J. E. Taylor, Anastasia Anagnostou, Tamas Kiss, Gabor Terstyanszky, Peter Kacsuk, and Nicola Fantini. A Tutorial on Cloud Computing for Agent-based Modeling & Simulation with Repast. In *Proceedings of the 2014 Winter Simulation Conference*, WSC '14, pages 192–206, Piscataway, NJ, USA, 2014. IEEE Press. Cited on pages 24 and 25.

[TARO10] Ivo J. P. M. Timóteo, Miguel R. Araújo, Rosaldo J. F. Rossetti, and Eugénio C. Oliveira. TraSMAPI: An API oriented towards Multi-Agent Systems real-time interaction with multiple Traffic Simulators. In *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, pages 1183–1188, 9 2010. Cited on page 58.

[TD10] Andreas Tolk and Saikou Y. Diallo. Using a Formal Approach to Simulation Interoperability to Specify Languages for Ambassador Agents. In *Proceedings of the Winter Simulation Conference*, WSC '10, pages 359–370. Winter Simulation Conference, 2010. Cited on pages 23 and 25.

[TDPHZ11] Andreas Tolk, Saikou Y. Diallo, Jose J. Padilla, and Heber Herencia-Zapana. Model Theoretic Implications for Agent Languages in Support of Interoperability and Composability. In *Proceedings of the Winter Simulation Conference*, WSC '11, pages 309–320. Winter Simulation Conference, 2011. Cited on page 23.

[TKT$^+$14] S.J.E. Taylor, T. Kiss, G. Terstyanszky, P. Kacsuk, and N. Fantini. Cloud computing for simulation in manufacturing and engineering: Introducing the CloudSME simulation platform. volume 46, pages 89–96, Tampa, FL, 2014. The Society for Modeling and Simulation International. Cited on page 20.

[TLBE11] Wei-Tek Tsai, Wu Li, X. Bai, and Jay Elston. P4-SimSaaS: Policy specification for multi-tendency simulation software-as-a-service model. pages 3067–3081, Phoenix, AZ, 2011. Cited on page 20.

[TLSS11] Wei-Tek Tsai, Wu Li, Hessam Sarjoughian, and Qihong Shao. SimSaaS: Simulation Software-as-a-service. In *Proceedings of the 44th Annual Simulation Symposium*, ANSS '11, pages 77–86, San Diego, CA, USA, 2011. Society for Computer Simulation International. Cited on page 20.

[TM14] Andreas Tolk and Saurabh Mittal. A Necessary Paradigm Change to Enable Composable Cloud-based M&S Services. In *Proceedings of the 2014 Winter Simulation Conference*, WSC '14, pages 356–366, Piscataway, NJ, USA, 2014. IEEE Press. Cited on page 20.

[VO11] Matteo Vasirani and Sascha Ossowski. A computational market for distributed control of urban road traffic systems. *Intelligent Transportation Systems, IEEE Transactions on*, 12(2):313–321, 2011. Cited on page 9.

[vSN$^+$09] Jan vom Brocke, Alexander Simons, Björn Niehaves, Kai Riemer, Ralf Plattfaut, and Anne Cleven. Reconstructing the Giant: On the Importance of Rigour in Documenting the Literature Search Process. In *Proceedings of the 17th European Conference on Information Systems*, Verona, Italy, 2009. Cited on pages 15, 16, 17, 18, and 19.

[Wei91] Mark Weiser. The computer for the 21st century. *Scientific american*, 265(3):94–104, 1991. Cited on page 29.

[WJ95] Michael Wooldridge and Nicholas R. Jennings. Intelligent agents: theory and practice. *The Knowledge Engineering Review*, 10:115–152, 6 1995. Cited on page 7.

[WUW$^+$12] Tichakorn Wongpiromsarn, Tawit Uthaicharoenpong, Yu Wang, Emilio Frazzoli, and Danwei Wang. Distributed traffic signal control for maximum network throughput. In *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on*, pages 588–595. IEEE, 2012. Cited on page 9.

[WW14] Sixuan Wang and Gabriel Wainer. Semantic Mashups for Simulation As a Service with Tag Mining and Ontology Learning. In *Proceedings of the Symposium on Theory of Modeling & Simulation - DEVS Integrative*, DEVS '14, pages 25:1–25:8, San Diego, CA, USA, 2014. Society for Computer Simulation International. Cited on page 20.

[WW15] Sixuan Wang and Gabriel Wainer. A Simulation As a Service Methodology with Application for Crowd Modeling, Simulation and Visualization. *Simulation*, 91(1):71–95, January 2015. Cited on pages 19 and 24.

[WYL$^+$08] Wenguang Wang, Wenguang Yu, Qun Li, Weiping Wang, and Xichun Liu. Service-oriented High Level Architecture. In *Proceedings of the 2008 Summer Computer Simulation Conference*, SCSC '08, pages 16:1–16:12, Vista, CA, 2008. Society for Modeling & Simulation International. Cited on page 22.

[XTCT05] Yong Xie, Yong Meng Teo, Wentong Cai, and Stephen John Turner. Servicing Provisioning for HLA-Based Distributed Simulation on the Grid. In *Proceedings of the 19th Workshop on Principles of Advanced and Distributed Simulation*, PADS '05, pages 282–291, Washington, DC, USA, 2005. IEEE Computer Society. Cited on pages 19 and 21.

[YO04] Levent Yilmaz and Tuncer I. Ören. Exploring Agent-supported Simulation Brokering on the Semantic Web: Foundations for a Dynamic Composability Approach. In *Proceedings of the 36th Conference on Winter Simulation*, WSC '04, pages 766–773. Winter Simulation Conference, 2004. Cited on page 19.

[YÖ07] Levent Yilmaz and Tuncer I. Ören. Agent-directed simulation systems engineering. In *Proceedings of the 2007 Summer Computer Simulation Conference*, pages 897–904. Society for Computer Simulation International, 2007. Cited on page 8.

[YOA06] Levent Yilmaz, Tuncer Ören, and Nasser-Ghasem Aghaee. Intelligent agents, simulation, and gaming. *Simulation & Gaming*, 37(3):339–349, 2006. Cited on pages 23, 24, and 25.

[YTFD14] Levent Yilmaz, Simon J. E. Taylor, Richard Fujimoto, and Frederica Darema. Panel: The Future of Research in Modeling & Simulation. In *Proceedings of the 2014 Winter Simulation Conference*, WSC '14, pages 2797–2811, Piscataway, NJ, USA, 2014. IEEE Press. Cited on page 24.

[ZCB10] Qi Zhang, Lu Cheng, and Raouf Boutaba. Cloud computing: state-of-the-art and research challenges. *Journal of Internet Services and Applications*, 1(1):7–18, Apr 2010. Cited on page 13.

[ZDSHB09] S. K. Zegeye, B. De Schutter, J. Hellendoorn, and E. A. Breunesse. Integrated macroscopic traffic flow and emission model based on METANET and VT-micro. *Models and Technologies for Intelligent Transportation Systems*, pages 86–89, 2009. Cited on page 9.

REFERENCES

Appendix A

Oculum’s logo

Oculum’s logo

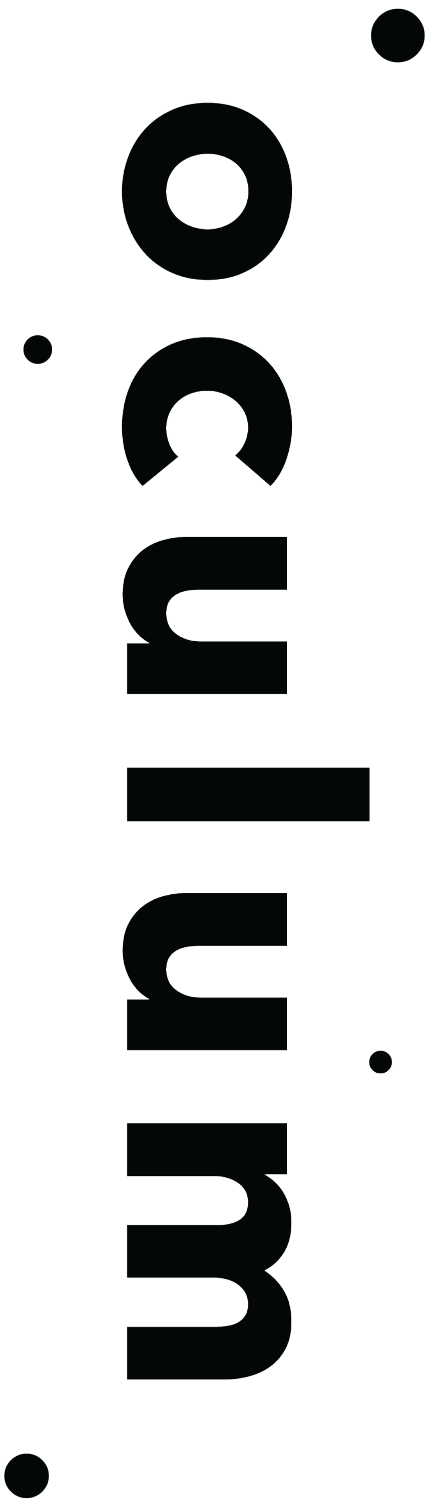


Figure A.1: The Oculum’s logo

Appendix B

OpenStack configurations

B.1 Nodes' characteristics

It is considered three virtual networks provided by VirtualBox: *vboxnet0*, *vboxnet1* and *vboxnet2*. *vboxnet0* is the Management Network, *vboxnet1* is the Instance/Guest Network and *vboxnet3* is the External Network for floating IPs.

Node	**Characteristics?**	**Services**
Control	2048MB of RAM. Ubuntu 14.04 server (64-bit). Connected to vboxnet0 and to exterior (via NAT)	NTP, RabbitMQ, MySQL, Keystone, Glance, Neutron, Nova, Cinder, Heat, and Horizon
Network	1024MB of RAM. Ubuntu 14.04 server (64-bit). Connected to vboxnet0, vboxnet1, vboxnet2 and exterior (via NAT)	NTP and Neutron
Compute	5120MB of RAM. Ubuntu 14.04 server (64-bit). Connected to vboxnet0, vboxnet1 and to exterior (via NAT)	NTP, Nova and Neutron

Table B.1: OpenStack's Services running in each node

B.2 Code run after installation

```
#CirrOS image
glance image-create --name 'CirrOS 0.3.2 x86_64' --is-public=true \
 --container-format=bare --disk-format=qcow2 \
 --location http://download.cirros-cloud.net/0.3.2/cirros-0.3.2-x86_64-disk.img
# External Network
neutron net-create ext-net --shared --router:external=True
neutron subnet-create ext-net --name ext-subnet \
 --allocation-pool start=192.168.100.20,end=192.168.100.254 --disable-dhcp \
 --gateway 192.168.100.2 192.168.100.0/24
```

```
#Internal Network
neutron net-create demo-net
neutron subnet-create demo-net --name demo-subnet --dns-nameserver 8.8.8.8 \
 --allocation-pool start=172.16.10.20,end=172.16.10.254 \
 --gateway 172.16.10.1 172.16.10.0/24
#Router
neutron router-create demo-router
neutron router-interface-add demo-router demo-subnet
neutron router-gateway-set demo-router ext-net
#Update default security groups
nova secgroup-add-rule default tcp 22 22 0.0.0.0/0
nova secgroup-add-rule default icmp -1 -1 0.0.0.0/0
```