People's Democratic Republic of Algeria. Ministry of Higher Education and Scientific Research.

Computer Science department Saad Dahleb Blida University.

System of systems simulation approach for crowd management: Case of sport event.

submitted for the degree of Master Software engineering.

By

Ghris Gadi

Keciour Aymen

Supervised by Imane Cherfa

Date: 09/12/2020

In front of the jury:

Mme. Mancer Yasmine: President

Mme. Chikhi Imane: Examiner

2019/2020

ملخص

الهدف الأسَاسي من هذه الأطروحة هو إقتِرَاح محَاكَاة لنظَام الأنظمة إِدَارة الحشود في الأحدَاث الريَاضية. هذه المحَاكَاة تهدف إِلَى تقيِيم النموذج المبني بوَاسطة لغة نمذجة الأنظمة (SysML) . التقيِيم سيكون علَى أسَاس تحليل السلوكيَات العَامة لنظَام الأنظمة في مختلف أمَاكن الحدث الريَاضي، الغرض الرئِيسي من تحليل هذه المحَاكَات هو الكشف عن ثغرَا ت في قُدُرَات الأنظمة علَى القيَام بِالمُهمة المَطلوبة، وفقًا لِلمعَايِّر التي تحدِد نجَاح المُهِمة (مِثل، سرعة الإِستِجَابة لخدمَات الطوَارِىء، معَايِّر السيطَرة علَى الحشد و التكلُفة، و مَا إِلَى غير ذلك). مع تمكين مهندِسي نظَام الأنظمة بِتوقُع نتَائِج غير مرغوب فيهَا التي يُمكن أن تَحدث في الحيَاة الحقيقِيَة، وبَاتَالي تَحدِيد كيفِية الرد الأمثل علَى تِلك التوقُعَات.

الكلِمَات المفتَاحِية: نظَام الأنظمة، محَاكَاة، نموذج، إِدَارة الحشود.

Abstract

The aim of this thesis is to propose a simulation approach, in context of systems of systems. The simulation approach have to be proposed in context of crowd management in sporting events SoS. The simulation aims at evaluating the SysML models that represent the concrete architecture and analysing the overall behaviors of SoS in different event venues. The main purpose of doing so is to detect capability gaps, or mission tactical gaps according to the overall mission effectiveness parameters (e.g.response time to an emergency, crowd control criteria, etc.). Simulation results help the SoS SE engineers to predict the effects of certain courses of action, and therefore to decide how to react optimally to a situation.

Keywords: System of systems, Simulation, Model, Crowd management.

Résumé

L’objectif de ce mémoire est de proposée une approche de simulation dans le cadre de la gestion des foules dans les SoS d’événements sportifs. La simulation vise à évaluer les modèles SysML qui représentent l’architecture concrète et à analyser les comportements globaux du SoS dans les différents lieux d’événements. L’objectif principal de cette opération est de détecter les lacunes de capacité ou les lacunes tactiques de la mission en fonction des paramètres globaux d’efficacité de la mission (par exemple, le temps de réponse à une urgence, critères de satisfaction de la foule, critères de contrôle des foules, critères de coût, etc.). Tout en tirant parti des résultats de la simulation, les ingénieurs de SoS permettent de prédire les effets de certaines pistes d’action, et donc de décider comment réagir de manière optimale à une situation.

Les mots clés : Système de systèmes, Simulation, Modèle, Gestion des foules

Acknowledgments

First and foremost, we would like to take this opportunity to express our deep and sincere gratitude to our supervisor Ms Imane Cherfa, for her guidance whenever we ran into a trouble or had a question about the research or the writing, she always steered us to the right direction without her it would be impossible for us to complete this thesis.

We also like thanks the examiners, for the time spent on carefully reading this thesis and their valuable suggestions to make the best version possible of this thesis. And all the professors of the computer science department of Saad Dahleb University and any contributor to the knowledge we have acquired in recent years.

Finally, we would like to thanks our families and friends, for all the support through the past years and what to come.

Contents

List of Figures

List of Tables

Notation and Acronym

AB: Agent based

ADE: Application Domain Expert

API: Application Programming Interface

BDD: Block Definition Diagrams

CS: Constituent Systems

DE: Discrete Event

DoDAF: Department of Defense Architecture Framework

DoD: Department of Defense

IBD: Internal Block Diagrams

JRCC: Joint Rescue Coordination Centre

LKP: Last Known Position

MBSE: Model Based Systems Engineering

MoDAF: Ministry of Defense Architecture Framework

MoD: Ministry of Defense

MoE: Measure of Effectiveness

NATO: North Atlantic Treaty Organization

NAF: NATO Architecture Framework

Ped: Pedestrian

SoS: System of Systems

SysML: System Modeling Language

SoS SE: System of Systems Systems Engineering

SE: Systems Engineering

SMA: Swedish Maritime Administration

SDI: Strategic Defense Initiative

SD: System Dynamic

UML: Unified Modelling Language

UAF: Unified Architecture Framework

2D: Two Dimensional

3D: Three Dimensional

General introduction

Context

Systems of Systems (SoS) encompass a group of independent and distributed systems which, through synergism between them, work together towards a common mission [1, 2]. The SoS SE literature addresses a wide variety of application domains such as commerce, transportation, healthcare and military [3], where independent and heterogeneous systems, often preexisting are implied to deliver a service. Crowd management is also an example of SoS [3]. It aimed at developing an integrated crowd control system during temporary events of mass transit, such as sports events or political meetings [3]. It involves several constituent systems such as organizations (Command and control center, fire brigade, Police, etc.), humans (stewards, decisionmakers, fire fighters, etc.) and technical systems (sensors, communication means, etc.)

Modeling and simulation represent a promising path for the development and analysis of systems of systems [2]. While modeling allows SoS engineers to control the overall complexity of SoS, to reveal and document its structure and behaviour, and to communicate these to stakeholders [4], simulation can be used to demonstrate these models from various point of view. Simulation in SoS SE can be used to engineer the integration of a system into SoS [1], to avoid the undesirable emergent behaviour, to evaluate the ability of an architecture configuration to accomplish the specified SoS mission [5], or to identify the key factors that influence overall SoS performance [6].

Problem Statement

Although several contribution in context of Modeling and simulation for SoS SE have been proposed, it is still an emerging research field. Several model based contributions have been proposed, which often are based on the SysML (System Modeling language) [7]. Unfortunately, few of these propositions consider simulation. We quote as an example the work of [7], which proposed a mission oriented model based approach to model an acknowledged SoS using the SoS SE wave model and the SysML language. The approach was validated by proposing different models for the crowd management in sporting events case study, but no simulation was proposed for these models. In case of crowd management, simulation is interesting for: (1) to ensure that the SoS is able to give a fast response and suggest the best way to evacuate crowds safely and efficiently when emergency (such as fires and earthquakes) occurs, (2) to assist designers in evaluating risk and maximizing safe design of architectural space, (3) to detect key factors that can influence the SoS performance.

Objectives

The aim of this work is to propose simulation approach in context of crowd management in sporting events. This work is to be done in context of an existing approach. The simulation aims at evaluating the SysML models that represent the concrete architecture, and analyzing the overall behaviors of SoS in different event venues. The main purpose of doing so is to detect capability gaps, or mission tactical gaps according to the overall mission effectiveness parameters.

Thesis structure

This manuscript will be presented in four main chapters as follow:

– Chapter 1: this chapter present a general idea about the history of SoSs, while focusing on its characteristics, types, and the different modeling and simulation techniques.

– Chapter 2: this chapter present a general idea on the crowd simulation methods used, its applications, and challenges.

– chapter 3: in this chapter we give an overview of our simulation approach, with a basic crowd management case of sport event SoS model.

– Chapter 4: this chapter we show the implementation details and the performance analysis of the different simulated scenarios .

Chapter 1

System of systems

1.1 introduction

In this chapter, we present an overview of Systems of Systems (SoS), from its historic appearance and definition, to its simulations approaches: DE (Discrete event), SD (System Dynamic), AB (Agent based), and multi-method. While passing by the important concepts that characterize a SoS, and its major types. Finally, we present the modeling approaches that were presented in a wide variety of publications, to capture SoS behaviours easily, in order to manage the complexity that the SoS and its constituents systems brings.

1.2 System of systems definition

There is no specific and broadly acknowledged definition of a SoS, between experts that bound SoS into a specific field, making it hard to pin point the field exactly. The literature is numerous, and there are many attempts to define and characterise System of systems to achieve some resemblance [8, 9, 10, 11]. Systems are a group of interacting entities natural or human made, they range from simple to complex defined by their structures and purposes that they express through their functions. Most of the literature defined the meaning of SoS as a set of systems that work together to create a more complex system that achieve goals none of the constituent systems can achieve alone [1, 2, 12]. Figure1.1 below shows an example of search and rescue SoS and the constituent systems that are involved.

We can see in the Figure1.1 that several constituent systems are included in the SoS, such as: the available resources of the Swedish Maritime Administration (SMA), a life raft that indicates

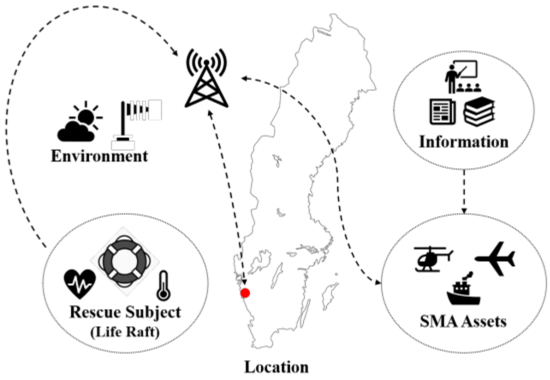


Figure 1.1: example of an SoS (search and rescue) [12]

a Last Known Position (LKP) and gives a distress call which is forwarded to SMA's Joint Rescue Co-ordination Centre (JRCC), and the different information circulating among the constituent systems.

1.3 Historical overview of System of systems definition

Boulding's publication on general systems theory, was one of the first adopter of the term "system of systems". Even though the early thought of SoS originated from different sources [13]. In Boulding's paper, the SoS concept is stated as "the arrangement of theoretical systems and constructs in a hierarchy of complexity". He also characterises SoS as dynamics, between systems structure that make the whole SoS adjust and evolve over time, as he describes SoS as a "open system" that can be affected by different events[13]. A lot of Boulding's work on systems theory state a lot of similarity, according to recent system of systems concepts, later on a bunch of Systems concepts started to be characterized as SoS like urban city planning, System science structure and biological systems [14, 15, 16].

In the late 80s (1980), the united states strategic defense initiative (SDI) turned into an important source in setting up the floor for SoS, to be an engineering concept focused on linking independent systems with each other, to create a more complex system that achieve new goals [17]. This advancement led to improved awareness toward system of systems in an academic and industry aspects, overall SoSE took 15 more years of maturity, to be considered as a discipline [2]. By the early 21st Century, SoSE is still viewed as being in its earliest stages [18].

1.4 System of systems Characteristics

Many characteristics were proposed in the literature to distinguish SoS from monolithic system. We will discuss them in the following.

1.4.1 Maier's characteristics

Maier characterisation of system of systems are represented in five major point with the acronym "OMGEE" [19]:

- **Operational Independence**. Each constituent system of the SoS is autonomous and have the ability to function outside of the SoS.
- **Managerial Independence**. Even though the constituent systems of a SoS cooperate with each other for a common goal, each cs have the ability to self-manage their operational purpose within a SoS.
- **Geographic Distribution**. The CS working together in a SoS are distributed over a geographical extent. In addition, the collaboration within systems can be by exchanging information and not a significant amount of their entities, or whatever those systems specifics are therefore, they remain different systems.
- **Evolutionary Development**. The development of an SoS is evolutionary in a way that its functionality can be under continuous change, these functionality can be added or removed with experience. Therefore, a SoS is never at it final form.
- **Emergent Behaviour**. The emergent behaviour is a result of the work of the constituent systems collaboration with each other to achieve new goal that none of the constituent system can do alone.

1.4.2 Boardman and Sauser characteristics

Boardman and Sauser, also worked on a characterisation that separate monolithic systems from a SoS and pointing to the ability to merge standalone systems with existing systems to form SoSs [20]. They identify five characteristics with the acronym "ABCDE":

- **Autonomy**. This characteristic looks a lot like Maier second characteristic “Managerial independence”, where each system is autonomous and independent with the ability to self-govern itself within the SoS.
- **Belonging**. where constituent systems cooperate to achieve a common SoS goal even if it means losing some their functions.
- **Connectivity**. To achieve SoS common goals the constituent systems must be linked with a powerful and dynamic network therefore connected with each other.
- **Diversity**. The constituent systems are diverse and different from one another with the openness for evolvement.
- **Emerging**. This characteristic also look a lot like Maier last characteristic “emergent behaviour”, where the collaboration and the interaction of the constituent systems bring on new functions and behaviour that its special to the SoS as whole.

1.4.3 Abbott’s characteristics

Abbott argues that the naming system of systems should be saved for systems that are unique and modern. In addition, a SoS should not be viewed as a chain of systems where some systems must be at the top of others, but as a place where systems exist and new ones can join to function and interact within it [21], Abbot defines Three characteristics of this place as follow:

- **Open at the top**. This characteristic is important so that SoS not viewed as a hierarchy, where some systems must be at the top because an open SoS at the top is able to add additional systems without any interference with any top level systems.
- **Open at the bottom**. Mean the ability of the SoS to change, or adapt some of its CS functionality in order to achieve the SoS purpose. like adding links for communications purposes between the systems.
- **Continually evolving**. Meaning that the SoS and its Constituent systems are continually evolving depending on the time technology, so the SoS is never in its final form.

1.5 System of systems dimensions

Even though SoS lack an exact definition, the literature offers a lot of SoS description properties like the Maier, Boardman and Sauser, and Abbott. The characterisation shown above in section 1.4, draws the first lines of a SoS space. These characteristic share some familiarity with each other. Claus Ballegaard Nielsen et al. Put out eight dimension for the SoSs based on the familiarity of the SoS characteristics that has been published in [2] (e.g. the term “independence” this characteristic can be found on Maier [19] “Managerial independence” and on Boardman and Sauser [20] ”autonomy” characterisation).

1.5.1 Autonomy of Constituents

Autonomy is the ability to self-manage or govern by a system specific rules, the system is free from any external other constituent systems rules within the SoS, that make each system ownership its own. This characteristic is on Maier as Managerial independence [19] which demand from the systems the ability to self-govern itself within the SoS with their own rules. In Keating et al [18] is stated that “the Autonomy of constituents systems is important to SoS engineering”. Boardman and Sauser [20], autonomy states to the ability of the constituents systems to be independent with the ability of self-control outside the SoS. Cook also admits the need of independence like Maier and the need of Constituent systems to be “purposeful” and have their own goals [22].

1.5.2 Independence

Independence is the ability of the constituent systems to function outside of the SoS. This function can relate the whole SoS goal, and other can be not related (independent) of it [2]. This characteristic is mentioned on Maier as operational independence [19], also mentioned by Krygiel as the ability of the cs to preform independent action outside the SoS [23], and by Jamshidi with the line “systems are independently operable” [24]. Sharawi et al. also mentioned that the independence of the cs activities and functions are critical to the SoS definition [10].

1.5.3 Distribution

Distribution is when the constituent systems are not sharing the same place and different from one another, so that some form of connection or medium is needed for communication and data sharing, distribution is described in Maier characteristic as geographic distribution [19] where systems are geographically distributed from each other's. Shenhar [25], describes this characteristic as the need of systems to be geographically distant within the SoS, as of Kotov in his characterisation he mentions the need of physical separation between the cs [26]. Finally, Manthorpe [27], describe this characteristic, as a spreading of data is necessary between cs, which could mean that each cs have to own his data and share it with other systems via some kind of connection.

1.5.4 Evolution

Most of all SoSs have a long timeframe of usage and are subjects to many modifications in its functionality or in the constituent systems involved. Maier [19] defined this characteristic as evolutionary development, and Carlock and Fenton [28] use the meaning the lack of permanent state of the SoS to identify evolution. Carney et al [29], Crossley [30] and Despotou et al [31] define evolution as continuous steps of changes and upgrades that the constituent systems goes through. Abbott [21] used the term continually evolving.

1.5.5 Dynamic Behaviour

Dynamic behaviour is the ability for a SoS to change its structure of functions without arranged interference, unlike the evolution dimension which is the ability to make a planned changes to the SoS. Only a few publication recognise this characteristic as an important one for a SoS especially when dealing with faults or threats that can hit our SoS. Boardman and Sauser [20] refer to this characteristic where he stated that "dynamic determination of connectivity" is necessary for the autonomy of the cs to connect and disconnect them in real-time. Crossley [30] looks at the SoS as a dynamic of entities, while Schneider and Trapp [32] present a real-time safety structure that dynamically reconfigure an open SoS.

1.5.6 Emergence of Behaviour

Emergent Behaviour refer to the new behaviour that emerge as a result of the collaboration between the constituent systems within the SoS, that none of the constituent systems could achieve on their own, Maier [19], and Boardman and Sauser [20], refer to this characteristic straight forward as emergent behaviour. In addition, Abbot [21] stated this characteristic as a need of an opens to the SoS.

1.5.7 Interdependence

Interdependence is the common dependency between the constituent systems to achieve the shared goal that the SoS provide, this goal is the main focus of the constituent systems that each cs can sacrifice some of its function for the reason to be within the SoS. This characteristic can be found on Krygiel publication with the term “Interdependence” [23], and “interdependencies” between the cs by Sage and Cuppan [33], and the same term is used from Bar-Yam et al [34]. These last two authors stated that even though the “independence” characteristic with the “interdependence” seem contradictory, but the SoS constituent systems must make the necessary trade-offs between the level of independence and interdependence to achieve the common goal.

1.5.8 Interoperability

Interoperability is the capability of the SoS to merge a diverse systems to it, this will bring up the need for a an interface able to link these heterogeneous systems together. Interoperability characteristic appears in Shenhar [25], as the ability of different systems to function with each other, by Manthorpe as “the integration of capabilities between the cs” [27], by Krygiel as “interoperability and integration“ [23], by Carlock and Fenton and Cook [28, 22] by the term “heterogeneity”, and by Abbott by the term “open at the bottom” [21].

1.6 System of systems dimension mapping

With the SoS characterisation and the dimension section it’s clear that the SoS space is vast and the publication is there to prove it. Nilsen et al [2] provide figure 1.2 below that show a table containing the eight dimension section as an image these dimension will be as stated by different authors publication of definition and characterisation(e.g. Maier, Abboot characterisation…) of the SoS concepts. As we can see from Nilsen et al survey the “Evolution”, the “Emergences behaviour”, and “Interoperability” characteristic are used more than the others characteristics. Nilsen et al [2] commented on this survey result as, “not unexpected”, and the reason for that can be that theses concept are hard to understand with SoS domains being in their earliest stages [18], which bring the necessity from different authors to give them more attention than the other characteristics. They also commented on the other results that contain low number of mentions

Author(s)	Autonomy	Independence	Distribution	Evolution	Dynamic Reconfiguration	Emergence of Behaviour	Interdependence	Interoperability
Boulding 1956	•	•					•	•
Ackoff 1971	•						•	•
Eisner et al. 1991			•			•	•	•
Noam 1994	•	•	•			•		•
Shenhar et al. 1994			•			•		•
Manthorpe 1996				•		•		•
Maier 1996	•	•	•	•		•		
Kotov 1997		•	•					
Lukasik 1998			•	•		•		•
Krygiel 1999			•	•		•		•
Roe 1999			•	•		•		•
Cook et al. 1999	•	•	•	•		•		•
Pei 2000			•		•			•
Carlock et al. 2001		•		•		•	•	•
Sage et al. 2001	•	•	•	•				•
Chen et al. 2003	•	•		•	•	•	•	•
Keating et al. 2003	•	•			•	•	•	
Bar-Yam et al. 2004		•		•		•	•	•
Crossley 2004		•		•	•			•
DeLaurentis et al. 2005	•	•	•	•	•	•	•	
Abbott 2006				•	•			
Boardman et al. 2006	•		•	•	•	•		•
Cocks 2006	•		•	•		•		
Boehm 2006				•		•		
Fisher 2006	•	•	•	•		•	•	
Sharawi et al. 2006	•	•	•	•	•	•		•
DoD SE Guide for SoS 2008	•	•	•	•	•	•	•	•
Karcanias 2010	•	•	•	•		•	•	
INCOSE 2015	•	•	•	•	•	•	•	•
Count	16	17	19	21	10	22	12	20

Figure 1.2: SoS dimensions mapping [2]

from other authors as, even though characteristic as “dynamic behaviour” and “distributed” are more knowledgeable between some authors, that does not mean they are not significant among the other characteristics that why some of these characteristic are mentioned in the publication regarded of the authors direction. (e.g. “evolution” and “emergence behaviour” are characteristic of a SoS as a whole, while “independence” and “autonomy” are characteristic of the constituent systems rather than of the SoS as a whole.). Nilsen et al also point out the variety in the characteristic that the different publications hold which prove the rich contributions on this domain, and the new publication incline to have more of this characteristics than the old ones [2].

1.7 System of Systems Categorisation

The US department of defence put out four types of SoSs [35]:

- **Directed**. Directed SoSs are made to realize specific purposes. Even though the constituent systems have the capacity to operate independently, but they are managed to achieve the SoS goals

– **Collaborative**. Collaborative SoSs does not have a clear central management, the constituent systems are independent but they cooperate voluntarily to achieve the common SoS goals.

– **Acknowledged**. Acknowledged SoS constituent systems are independent in their development and goals they are separate organization from the SoS, the success of the acknowledged SoS mission depends on the cooperation between the SoS and the constituent systems.

– **Virtual**. Virtual SoSs are characterise with the lack of central management or goals or both, which make the constituent systems behaviour or goals emergent.

Maier was the one responsible on defining three of these types [19], while later on the "acknowledged" SoS was defined by Dahmann and Baldwin[36] because it contain cases found in many military systems. Both these authors categorise SoS based on the degree of central management to help system of systems engineers select the best design approaches for SoSs [2], where in the "directed" SoS the constituent systems are one with the central management and the other types have none or some central management influence [17]. "Collaborative" SoS central management have no way to enforce decision because its barely exist within the SoS [17]. "Virtual" SoS central management doesn't exist [17]. While the "acknowledge" SoS tries keep the relationship between the central management and the constituent system somewhat separate, with maintaining the collaboration to achieve the SoS goals [36].

1.8 System of systems modeling

A model generally take many forms. It can be a graphical drawing, text, mathematical formalism, or all of the above. The main purpose in creating a model is to replicate and capture a specific characteristic of our study targets, in order to observe specific behaviours and easily manage complex systems [37]. The area that capture systems behaviour is known as Model-Based Systems Engineering (MBSE).

1.8.1 System of systems engineering

System of systems engineering (**SoSE**) concepts are being used to develop, design, analyse and capture SoS capabilities by the SoS engineering community. Modeling Languages like the unified modelling language (**UML**), and systems modelling language (**SysML**) can be found at the core of some MBSE approaches, and are used by different publication in creating models which capture concepts and relations for complex systems and SoSs like the study shown here [38, 7]. **Ontology** is also got adopted by many engineering field like SoSE, SE, and it being used to model systems by using different terminologies and viewpoint [39], ontology is the domain that represent knowledge and how to parse it through understanding its content [40].

The SoS complexity raises from its characteristics and types. Complex SoSs often come with high complexity, and a lot of viewpoints (e.g. Operational, services, strategic, capability…) [41], each of these views aim to facilitate the management of complex systems, with a lot of systems specific data and descriptions. These type of SoSs can be found in many military systems and require in-depth modelling approach [42]. Capturing these views is usually done with enterprise architecture frameworks. The department of defense architecture framework (**DoDAF**) is the architectural framework for the us department of defense (DoD) [43], the ministry of defense architecture framework (**MoDAF**) is for the uk ministry of defense (MoD) [44], there is also the NATO (North Atlantic Treaty Organization) architecture framework (**NAF**). Recently, a new architectural framework called the unified architecture framework (**UAF**) have emerge with the interoperability between nations and UML with SysML integration at its core [45]. With this many MBSE approaches, as stated above in the SoS categorisation section1.7, deciding on the perfect model approach by system of systems engineers is very dependent on the SoS scale and type [2].

1.8.2 System of systems engineering wave model (life-cycle)

The complexity of SoSs is known subject by SoS engineers, which make its development cycle a challenging process, that can't be done in a single wave. Trying to capture all of the SoS and its cs capabilities with all the upgrades, require continuous development cycle, that are represented in SoSE wave model (life-cycle). SoSE wave model is an iterative process that involve six steps in the first cycle [46]. First, **Initiate SoS**, initiating the SoS require from the engineers to provide essential SoS information that set the floor for SoS devolvement (e.g. SoS goals…). Second step, **conduct SoS analysis** where the engineers identify key points like the SoS performance measure, risks and mitigations that will be necessary on later steps. Third step is **developing SoS architecture** where engineers start developing the SoS architecture depending on the information gathered in earlier steps. Fourth is **planning SoS update** by evaluating SoS priorities and setting up the next SoS update wave. Fifth **implementing SoS update** where engineers implement and observe the cs implementation. Finally **continue SoS analysis**, by revisiting key information of the SoS and its states. This will conclude the first cycle as shown in figure 1.3. It represents the SoS wave model. The next cycle will start from the last step continue SoS analysis.

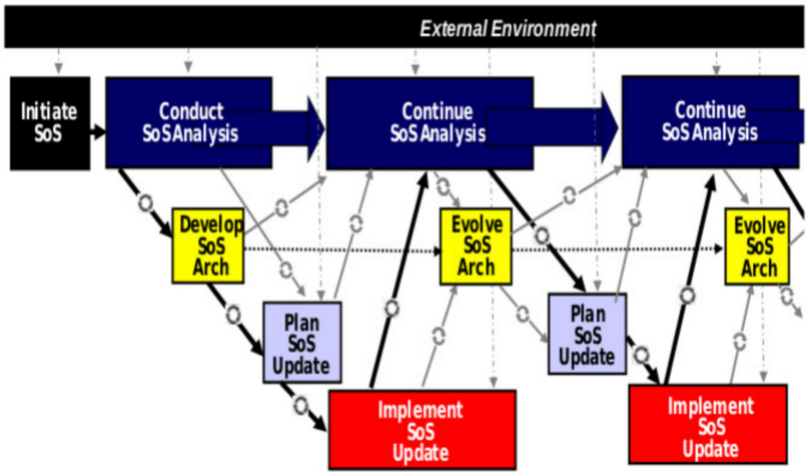


Figure 1.3: SoSE wave model [46].

1.9 Simulation and system of systems

Computer simulation is one of the most regularly utilized type of models testing. Brartley et al[47] describe it as the act of testing a model of a system, with a number of input and then studying the outcome. In addition, by Shannon[48] as the process of testing a model made from a system concept, and experiment it with simulation to understand the system, from different aspect like performance evaluation, strategy testing, discover emergent behaviour, reduce or mitigate risk from real life systems.

1.9.1 System of systems simulation approaches

In regards to simulation approaches, there are three main methods of simulation of SoS, which are: DES, SD, and ABS. Another way to simulate SoS is to combine the cited methods (multi-method). This part look at each one of these simulation approaches.

1.9.1.1 System dynamics (SD)

System dynamics simulation represent a top-down model structure, where the model represent a general system view in its complete abstract form, and overlook the great details like the individual, product, event involves within it [49]. Since SD is widely used on the strategic side of business simulation, its uses terms like "**stocks**", stock are usually the details component of a system that SD overlook and do not model in detail mentioned above (e.g. money, people, material, events…) [49, 50]. "**flows**" are diagrams that represent the relation and connection between the system component (e.g. financial transaction between people that involves money and materials, the effect of advertisement and the connection between ads cost and the new clients, the event where encouraging the employees may result in a better employees yields …). SD flow graphs usually comes with negatives and positives relation based on mathematical formalism, where the negatives could be a cost loss and the positive is the gains of a system [49, 50]. "**Causal loops**" are the presentation of the stock and flows connections with each other [49, 50]. The negatives component and the positives ones aspects of any business is where system dynamic excel at pointing, because since we are on a simulation time is not the problem but defining which individual system component we need to increase or decrease or even add is, without the need to model the system individual behaviour relationship. See figure 1.4 below show an example of SD simulation that represent and ad effectiveness on a sum of people simulation with clear view on stock, flows, and causal loops.

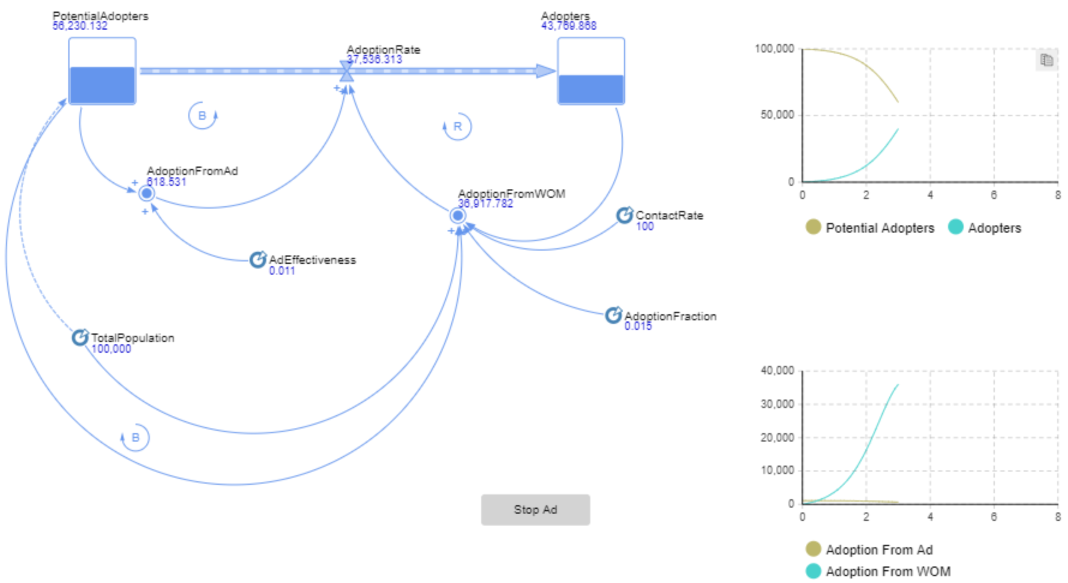


Figure 1.4: SD simulation example of ads effectiveness [51].

1.9.1.2 Discrete event simulation (DES)

DES approach is the most established model simulation between the others because it has been used by for a long time [52]. DES is mostly used in facilities systems to gather important data (e.g. queue waiting times and length, system bottlenecks, resources utilisation…). and like SD it also simulate kind of a top-down model, where the aim is the overall system but unlike SD system component are given more details and characteristic. In DES, system component are treated like object, an object can represent individuals, machine, cars, workers etc. In addition, each of these objects hold attributes (e.g. availability, time required to perform a mission…) unlike ABS where some of the overall model attributes can be stochastic. Most DES attributes are taken from real life example for accurate results [53]. In DES, The state of the system component (objects) changes over time in response to certain events, this event can be a work order, or a faulty machine etc. DES events can be triggered at a certain time, random time, or multiple times, and they will be indicator to start a set of process that will change the component state to do its designed job. This job can be an event, that will start another process and so on. That is why DES simulation is considered a one threaded computer simulation where the system component have no capacity to operate independently [52]. See figure 1.5 below that represent a DES simulation in a facility workshop.

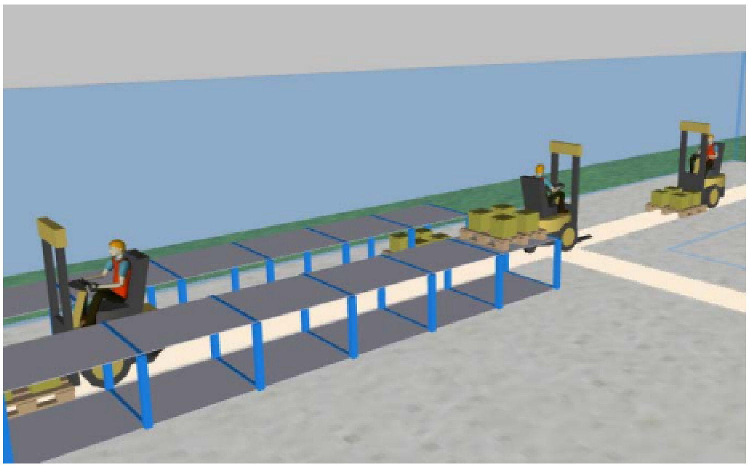


Figure 1.5: DES simulation example of a workshop [49].

1.9.1.3 Agent based simulation (ABS)

ABS simulation have seen a super-fast advancement development in recent years even though its young age compared to the other approaches, because of it capabilities at simulating complex systems models [54]. ABS is known for it uses to simulate models that represent a number of entities (agent), that share a common environment usually called "world". To represent these individual agents and their interactions, modeling these entities require a focus on detailing their characteristics, interaction rules, and behaviour. Thanks to the fast growing memory and CPU's (central processing unit) powers seen in the last 15 years and with the advancement in computer science whether its the UML language or Object oriented modeling. These concepts has been adopted to help make this micro-level detailed agent behaviour possible [54, 55]. Figure 1.6 below show an agent behaviour implementation in repast symphony ABS simulation platform, where the agent in hand transition between high alert and low alert mode with defining characteristic in each.

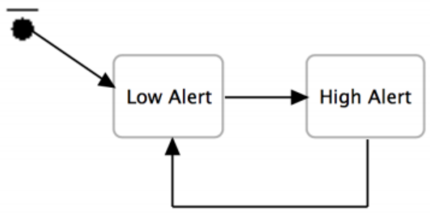


Figure 1.6: ABS agent behaviour example.[55].

The term agent in ABS does not have a specific meaning, an agent can represent people, vehicle, equipment, products, or even whole companies and systems. It all depends on the system or the SoS we are simulating. Russell[56] defined an agent as "anything that can be viewed as perceiving its environment (world) through sensors and acting upon that environment through actuators". With no clear rules on agents definition in ABS, the autonomous characteristic that agent interacting with each other in a multi threaded aspect brings, is becoming helpful in studying a SoS performance and its emergent behaviours, when SoS heterogeneous CS are modeled as agent that communicate with other agent with performance analysis characteristic [57]. See figure 1.7 below that represent an Agent based simulation of simple queuing service and waiting area place.

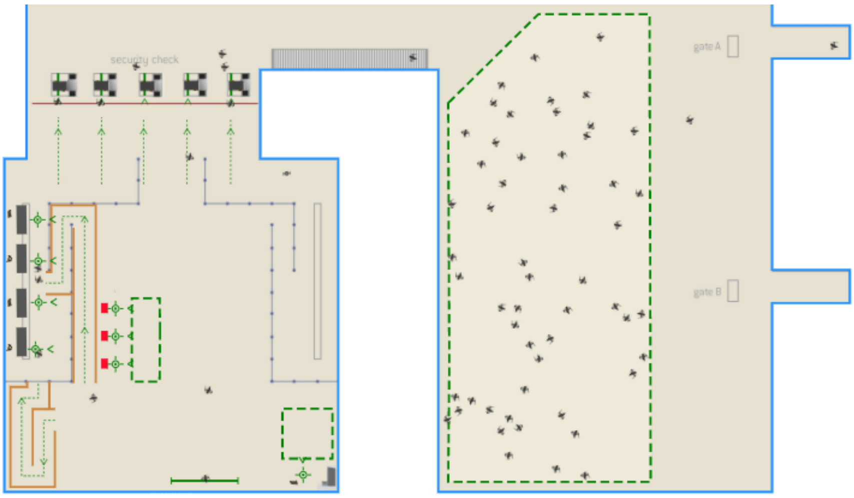


Figure 1.7: ABS example of a service area [49].

1.9.2 Comparisons of the Methods

Knowing each simulation approach strong points and limits, it is necessary for modelers to decide on the best approach according to the problem at hand.

1.9.2.1 DES and SD

DES and SD have a lot of thing different from each other, as shown in section.1.9.1.2 and section.1.9.1.1. Each of these approaches represent different situations. DES tend to be used on facilities systems that uses terms like queues such as the example show in figure 1.5 of a workshop where a queue is show as the worker take turns in grabbing resource's, while SD models uses term like flow, stock, and causal loops these term are perfect for simulating strategic business models as show in figure 1.4 where its clearly show the flow of ad adopters. The second point of difference is that even though both of these method uses a top-down model structure, DES models on general require some attention to details on its entities this leads to more time, and attention requirement to model DES. SD models on the other hand are top-down structures to capture the general perspective of the problem at hand. The last point of difference we can deduct from the two approach is the amount of stochastic variable used in modeling, DES require more random attributes in its entities than SD (e.g. one why to model the movement speed of a worker shown in figure.1.5 is a random interval variable between X and Y), thus trying capture DES characteristic can be different each simulation run on the same variable). SD on the other hand deciding on a random variable is final on each simulation run (e.g. as shown in figure 1.4 to model the ads effective we need a percentage that will turn our potential users to users this percentage is final on a simulation run, only when increasing or decreasing this variable the result Will change.).

1.9.2.2 ABS, DES, and SD

ABS is a unique approaches compared to DES and SD, that have been empowered by the recent advances in memory, CPU's, and computer science advancement. ABS is completely bottom-up approaches modeling structure, where the focus on each agent is done to the behavioural level, which make the model more stochastic in nature specially when the agents represent humans compared to SD or DES. Another point of difference between ABS and the other approaches is that each agent in the simulation is active, thanks to the multi threaded computer advancement where in DES entities relies on a certain event to become active. One of the strongest point of ABS is that with the wide range of the agent representation DES and SD models can both be simulated with ABS, but that come at a cost of a lot of time consuming time at modeling, that why modelers need to observe the system at hands then decide the optimal approach to simulate, as shown in figure 1.8.

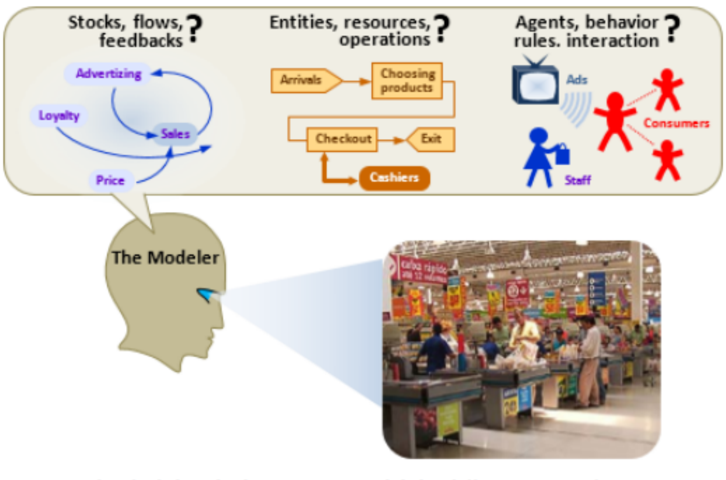


Figure 1.8: deciding on the optimal approach to simulate [49].

1.9.3 Multi-method simulation

Knowing an SoS scale and it's all constituent systems from the beginning is impossible for SoS engineers, that's why one of SoS characteristic state that SoSs are never in their final forms [21], and the developing of an SoS is complex and continuously evolving with its iterative development cycles steps [46]. That's why it is possible to combine several simulation approaches. It is interesting to analyze each CS while defining each approach to a specific system, then incorporate the different simulation approaches DE, SD, AB is necessary in some SoSs to overcome the limits that each simulation approach holds, and at the same time benefit the most from each one. Whether it is SD high strategic level problems management and demonstration shown in section.1.9.1.1, or DE high-level process operational description shown in section.1.9.1.2, or AB high micro level description of any agents shown in section.1.9.1.3. See figure 1.9 below that show a simple concept to multi-method simulation.

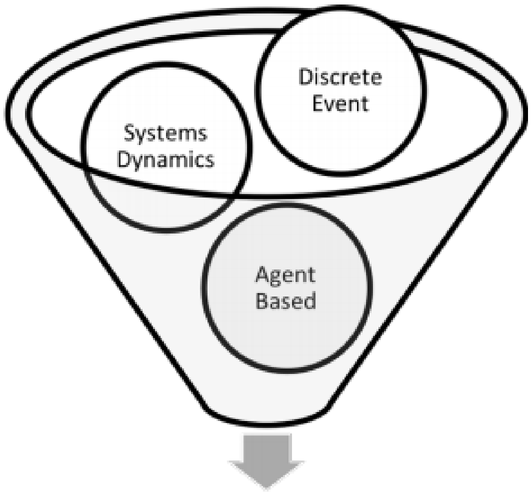


Figure 1.9: multi-method simulation concept [58].

1.10 Conclusion

We started this chapter with a overview of the term "System of systems" and its characteristic. Even thought the term System of systems is vague and its applications are open to debate, whether you call this or that an SoS. The literature is there and plenty with characteristic to point out SoS type and getting acknowledged by SoSE as a discipline [2]. Later, we show that even though SoSE is at its earliest stages [18], there is is multiple approach's to model and simulate SoS within the domain, where each have its strong point and limits depending on the problem at hand.

Chapter 2

Crowd management simulation

2.1 Introduction

In recent years, computer simulation and modeling technologies have been utilized for multiple of uses, concerning crowd both in its ordinary and crisis circumstances. like: entertainment , crowd management, traffic flow, architectural plan, evacuation and other comparable applications

In this chapter going to focus on the crowd management simulation aspect, we start by defining crowd management simulation. then we present the different simulations methods: macroscopic, mesoscopic and microscopic used to simulate a crowd. Finally we conclude this chapter with some crowd management simulation applications and its challenges.

2.2 Crowd management definition

Crowd is a large group of entities gathered in one place, these entities can be a group of animals, vehicles, or human etc. The **Pedestrian** (ped) aspect of human crowd is one of the most researched phenomenon that can be observed in several situation planned or not like the streets, stadiums, protests etc, where the crowd individuals are walking on foot.

During the gathering of a large amount of people for a long amount of time, Outbreaks are bound to happen between people, like fights, property damage, and crowd panic etc. Crowd management is the art of controlling the crowd and limit its unwanted behaviour through the time of a the mass gathering to ensure its safety, with the help of external systems such as police, emergency services, military in extreme out of control cases etc. Figure 2.1 below show a crowd example.

Figure 2.1: Crowd example [59].

2.3 Crowd simulation approach

Crowd simulation is the act of simulating a large amount of entities gathering together described in the section above, in computer simulation, crowds are usually displayed in two ways:

- **2D**, 2 dimensional space simulation is one of the most used display simulation due to its simplicity but high visualisation demonstration to the human eye. Its mainly used by computer scientist where realism is not needed but the overall process is of more important.

- **3D**, 3 dimensional visualisation on the other hand, is mainly used for its realism due to its power in displaying a high quality rendering and animation visuals, That's why we find it in a lot of entertainment focused work.

With these displays human crowd simulation methods take form of three major aspect where each has its own applications and benefits. These methods are based on the degree of the implementation of crowd behaviours, crowd behaviour in these methods can be classified into three main factors: [59]:

- **Physical Factors**, physical factor refer to the external characteristic, that each crowd individual can posses like position, movement speed, physical size, etc.

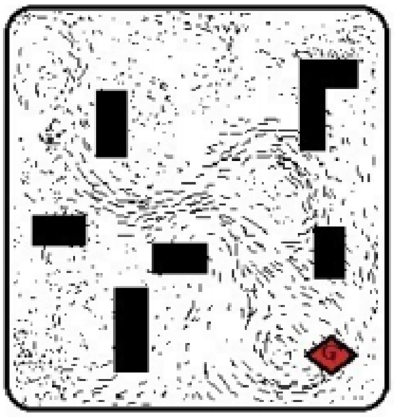
- **Social Factors**, the social factors refer to the human behaviour and how it can be influenced by its social surrounding, like culture, family ties, etc.

– **Psychological Factors**, psychological factors refer to to the human behaviour and how it can be influenced by his emotion, feelings etc.

2.3.1 Macroscopic (Flow-based) approach

Crowd simulations in this approach is the simplest one between the three, due to the feature that ignore all the human crowd aspect that can be seen in real life like, physical, social, and psychological factors. This approach tend to focus on the movement of the crowd as a whole, with a display that resemble a flow, fluid if the crowd is moving together as a whole or as a gas particle if its scattered [60]. This approach is considered the best method if all you need is to simulate super big sized crowd flow direction with no obstacles on the way, for it low computational cost since you are ignoring most of the crowd behavioural factors. Figure 2.2 show an example of flow-based simulation.

(a) Crowd flow as a whole [59].



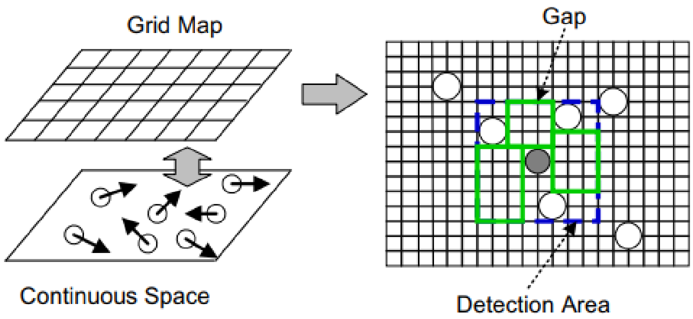
(b) Scattered crowd flow [60].

Figure 2.2: Flow-based crowd simulation example

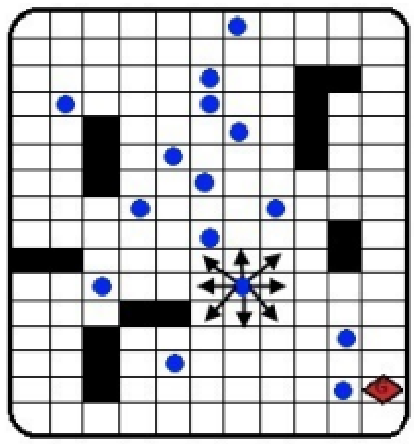
2.3.2 Microscopic (Agent-based) approach

Microscopic crowd simulation method takes completely opposite direction than the macroscopic. This methods has adopted the ABS Approach discussed in chapter.1.section.1.9.1.3 and used it to simulate a crowd, where each individual in the crowd is treated as autonomous agent with his own characteristic for more realism to the simulation. This approach aim at integrating the behavioural factors mentioned above to the crowd individuals, for the ability to interact with the world and react to it different dynamics, but that come at a High computational cost [61].

To this date capturing the behavioural crowd is a struggle specially in the social and psychological factors due to its complexity and the lack of understanding of the human mind, therefore each simulation that try to implement these behaviour tend to be immature [59] or use simple rules and stochastic variable. As for the physical factor numerous advancement is made with multiple working model's that simulate a crowd movement with consideration to its mass, movement speed and other physical factors before taking a step, where each publication has its unique algorithm to solve the problem. [61, 62, 63, 64]. Figure 2.3 below show a microscopic simulation concepts.



(a) Gap detection based on the agent size [62]



(b) Microscopic crowd simulation approach [60].

Figure 2.3: Microscopic crowd simulation concepts

2.3.3 Mesoscopic (Entity-based) approach

Entity based approach is a generalisation of the agent based approach. The idea of this approach is to take the microscopic approach and its important features, like the implementation of crowd behavioural factor and simplify it for a lower computing cost while simulating a large amount of crowd individuals. The simplification is done by by modeling one entity of the crowd with a global variable that define the behaviour of the crowd, and apply these variables to all the crowd individual [59, 60]. entity based approach are good if the simulation contain one type of the crowd, where all of its entities share the same characteristic like the evacuation scenarios. Figure 2.4 show an entity based simulation example.

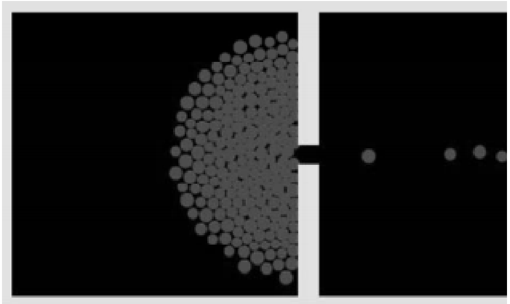


Figure 2.4: entity based simulation example [65]

2.4 Crowd management simulation application

Crowd simulation has been used for multiple application during the past decade, we can see it as a form of customers flow and architectural analysis in different places, such as airport, railway station, stadiums, malls and many more. the aim for such simulation is to improve customers satisfaction criteria by discovering the bottlenecks in the process of customer services and optimise them for a better queue times. Figure 2.5 show an example of such application.

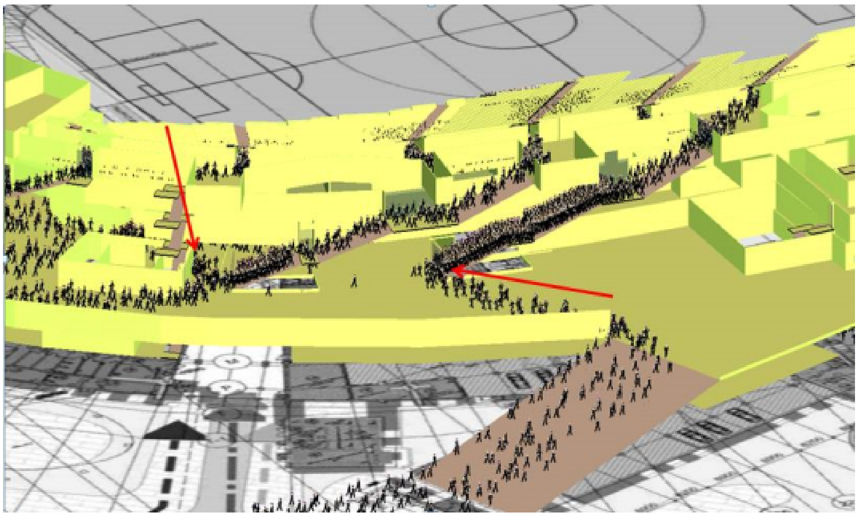


Figure 2.5: Crowd flow on a stadium example [66].

Another interesting crowd management application is traffic in roads, traffic flow like pedestrian flow in building can help detect bottlenecks, traffic causes, and save a lot of money by testing different road implementation and its effects. traffic simulation can be integrated with the major building we talked about earlier like stadium, airport, stations etc to detect the general impact of the flow of both traffic and pedestrian and get insight on how to make it better. figure 2.6 below show a traffic simulation example.

Figure 2.6: Traffic flow example [67].

Evacuation scenario in crowd simulation, is one of the most used application in the domain due to it importance. This application is used to capturing the architecture limits of any building, its usually do that without the need of introducing the psychology and social factor of the pedestrian, with the exception of his physical factors, which make its results more realistic and accurate. Figure 2.7 below show an evacuation example.

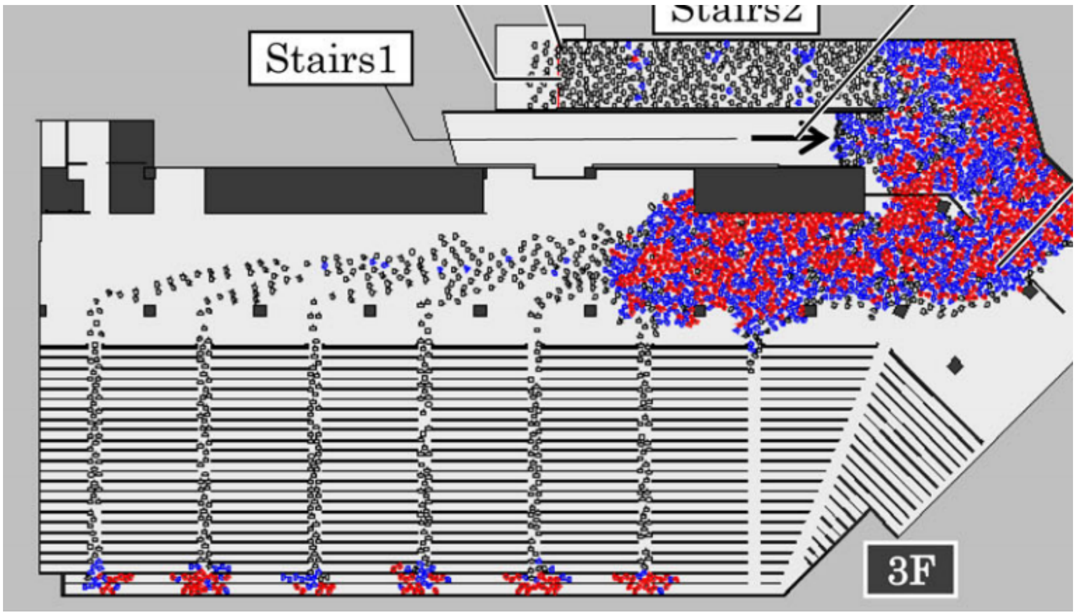


Figure 2.7: Evacuation on a stadium example [68].

Computer generated entertainment like movies and video games also implement crowd simulation concepts, for more emergent and realism to the consumer. Figure 2.8 show an example of a video game with a crowd riot in game implementation.

Figure 2.8: Crowd riot in game example [69]

2.5 Crowd management simulation challenges

Crowd management simulation is used for multiple scenarios, its helps the world test and solve complex process that involves a crowd, without the need for a real crowd. With all the algorithmic, and modelling advancement we have seen till now. crowd simulation is still open to many problems and concepts, the most important one is simulating a human social and psychological behaviour due to its complexity.

Crowd management passes through important real life issues, that simulation have not tackled yet like **the simulation of a panic crowd**, crowd panic happens often when danger arise, and causes many casualties within the crowd individual. Understanding the behaviour that causes the panic and its pattern can be of great help in public safety [70]. Another problem the simulation advancement lack to model like the real thing, is **Intelligent Behavior of Rescue Workers or the Authority**, many crowd management systems involves human led systems to control the crowd like police officer, emergency services etc. These worker play an important role in emergency cases to ease down the crowd panic, and ensure the crowd flow. Today simulation advancement also struggle in simulating these interactions [70]. The last point that we like to mention is **The simulation of Confrontational Crowd**, sometimes during an event a riot occur. The cause of this riot can be directly linked to psychology and sociology aspect of the crowd, today simulation advances also struggle to realistically simulate this phenomena using computer algorithms [70].

2.6 Conclusion

in this chapter we showed how crowd management is complex matter that burdens the world. Simulation was one of the methods picked up by computer scientists in an attempt to ease down its complexity or to reproduce it for entertainment reasons by simulating many real world scenarios that involves a crowd. We also mentioned the approaches used to simulate a crowd with the most used being Flow-based, entity based and agent based. And their limits.

Chapter 3

Crowd Management Simulation Approach

3.1 Introduction

This chapter discusses our contribution, which is to propose a simulation approach for the crowd management SoS in sport events. We start by presenting the modeling approach on which we relied, and present our extension, and our simulation approach. The modeling approach is based on SoS wave model shown in figure 1.3, with the SysML four pillars implementation. Then, we present an overview of the SoS at hand. We end the application of our chapter with the different diagrams modeling our SoS.

3.2 The SoS modeling approach and its extension

The modeling approach we used for our SoS is presented by cherfa et al [7]. In our work, we propose the extension of this approach by adding the simulation activity (See figure 3.1). The part we add implies two SoS engineering activities of the wave model: plan SoS updates and implement updates.

The method on which we relied is based on SysML, with the purpose to help the engineers in the first three steps of the SoS wave model, shown in figure 1.3 (initiate SoS, conduct SoS analysis and develop the SoS architecture), to generate the desired architecture. This Approach is well suited for acknowledged SoSs, where the main SoS organism is managed by the SoSEs, and each CS by its own SEs [46]. This approach defined two main actors for the architectural development in the course of seven steps:

- **Application domain expert**, ADE is the domain master, he/she, is responsible in defining the SoS mission, with the experience to balance the SoS mission with its CS goals. ADE is the actor in 5 steps of the seven mentioned in this approach as follow: (1) **Mission decomposition**, the purpose of this step is to provide the general view of the SoS mission and its submission, using the SysML requirement diagram. (2) **mission definition**, the purpose of this step is generate the behavioral view of the SoS goals using the SysML activity diagram. (3) **role definition** the purpose of this step is to design the hierarchy of the CS that achieve the SoS goals, using SysML BDD. (4) **role assignment** the purpose of this step is to assign each activity with its role within the SoS. (5) **capture measures** the purpose of this step is to capture the effectiveness measures required for the SoS performance, using the SysML parametric diagram.

– **System architect**, system architect main mission is to generate the SoS desired architecture, based on the ADE work. System architect is the actor in the last remaining steps. (6) **abstract architecture generation**, the purpose of this step is to generate the SoS structure that, represent the connection between the CS, using the SysML BDD and IBD. (7) **concrete architecture design**, the purpose of this step is to better the abstract design above. The figure below gives an overview the actors, steps, and tools involves in this approach.

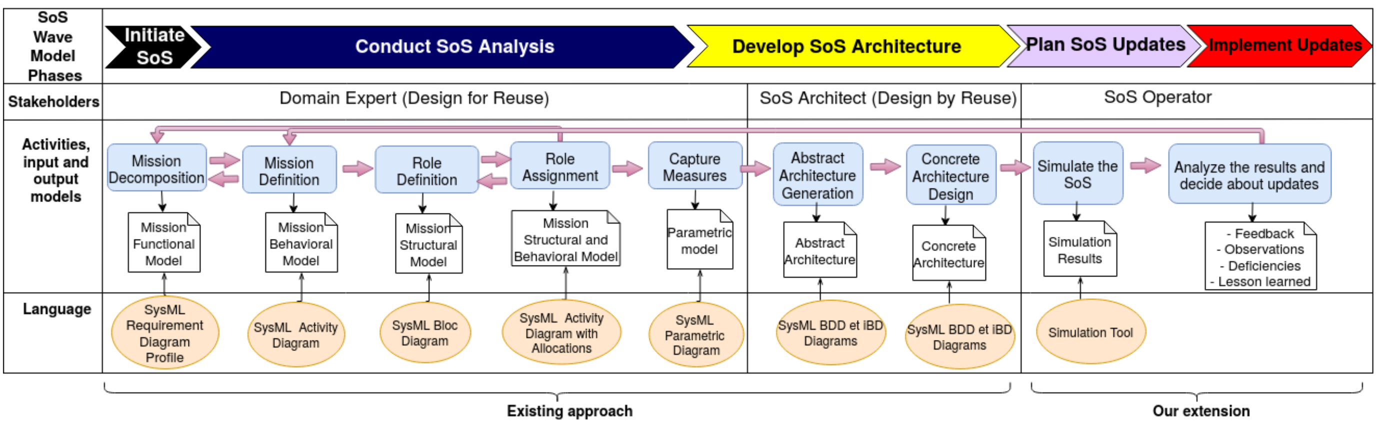


Figure 3.1: SoS simulation extension

In our work, the extension we propose implies one new role, which is the **SoS operator**. The later is responsible for managing, monitoring and controlling the correct execution of the SoS. Furthermore, he is responsible of the simulation. Simulation can reveal deficiencies and thus, he can plan updates and implement them according to the simulation results.

3.3 Our simulation approach

While modeling allows SoS engineers to control the overall complexity of SoS, to reveal and document its structure and behaviour, and to communicate these to stakeholders. Simulation can be used to demonstrate these models from various point of view: to avoid the undesirable emergent behaviour, to evaluate the ability of an architecture configuration to accomplish the specified SoS mission, or to identify the key factors that influence overall SoS performance.

In our work, we propose a simulation approach that aims to refine the SoS models, and to detect deficiencies in the concrete architecture. Furthermore, the SoS operator can test different SoS implementations, and updates like new CSs, new missions, performance results etc. This can help to decide about the optimal reaction to a situation.

Figure 3.2 below shows our simulation approach. It is based on the ABS method. The purpose of choosing the ABS, is that a SoS includes multiple constituent systems to work on one goal, and thus to interact, and the ABS is the best method for simulating interactions, and behaviors [71].

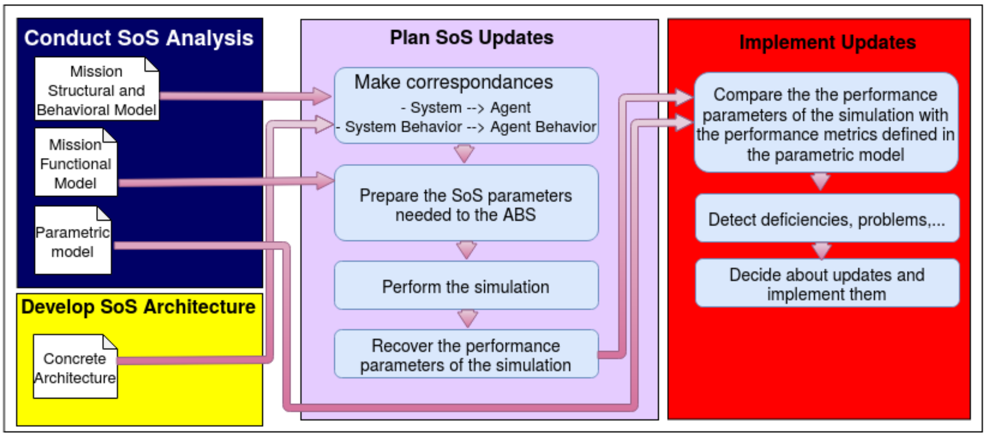


Figure 3.2: Our simulation approach

Our simulation approach is based on the models obtained from the analysis and design engineering activities. The mission structural and behavioral model with the concrete architecture contain the concrete constituent systems, and their behavior. Thus, every constituent system becomes an agent in the, and its behavior becomes the agent behavior. The mission functional model we give the mission requirement, so from this model we deduct the SoS parameters needed, for example: response time to an emergency=12 min as maximum.

From this data, we perform the simulation and measure the actual SoS performances. The simulation outcomes reveal the actual state of the SoS. The performance parameters of the simulation are compared to those defined in the requirement, and are calculated using the parametric model results. Then, the SoS operator from the simulation results detect deficiencies in the architecture, note the lesson learned, implement updates, and take different decisions,...

3.4 System of systems: Crowd management

In what follows, we will use our approach to model and simulate acrowd management in sporting event case study. During most of sport events scenarios, the numbers people attending is big enough to be called a crowd. and whenever there is a big crowd, a crowd management protocol need to put in place to prevent disastrous situations. Crowd management is recognised as an acknowledged SoS [3], where most of its CS are heterogeneous and independent (e.g. police officers, stewards, emergency services etc), but they work and communicate with each other to achieve the mission at hand. Figure 3.3 below shows an overview of crowd management case sport event SoS.

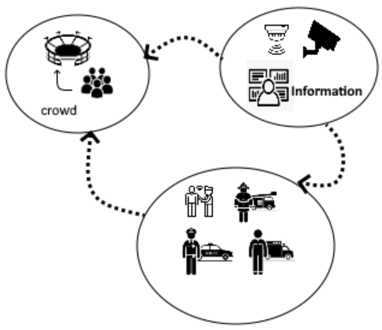


Figure 3.3: overview of crowd management case of sport event SoS.

3.5 Conceptual model description

This section present, the different diagrams implementation we used within the SysML environment, to create the SoS model.

3.5.1 Systems Modeling Language

SysML is a graphical modelling language like UML, but for systems engineering. SysML originally developed by an open source specification project, later on got picked and further developed by, OMG, INCOSE, and AP233. It supports specification, analysis, design, verification, and validation of multiple systems and systems of systems. SysML was devoloped as an extension of UML, therefore it uses a lot of UML diagrams (e.g. activity diagram, use case diagram...), while also adding its own diagrams. Figure 3.4 below, show SysmML diagram taxonomy specifying the diagrams that are common with UML and the new ones.

With the SysML four pillars we carried out the whole architecture design of our model, by following the approach discussed above in section.3.2. The expression the four pillars of SysML refers to the four essential diagrams of SysML. **Requirement**, for SoS missions. **Activity** from the behavior diagram, for the CS behaviour, **Block definition diagram** from the Structure diagram, for the SoS structure. **parametric diagram** for SoS analysis. Figure 3.5, show a preview of the diagram used according to SysML diagrams.

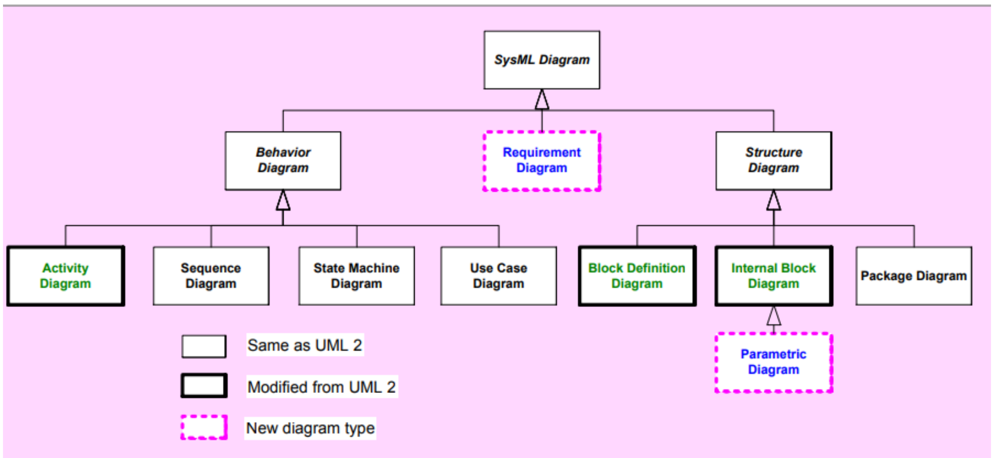


Figure 3.4: SysML diagram Taxonomy [72].

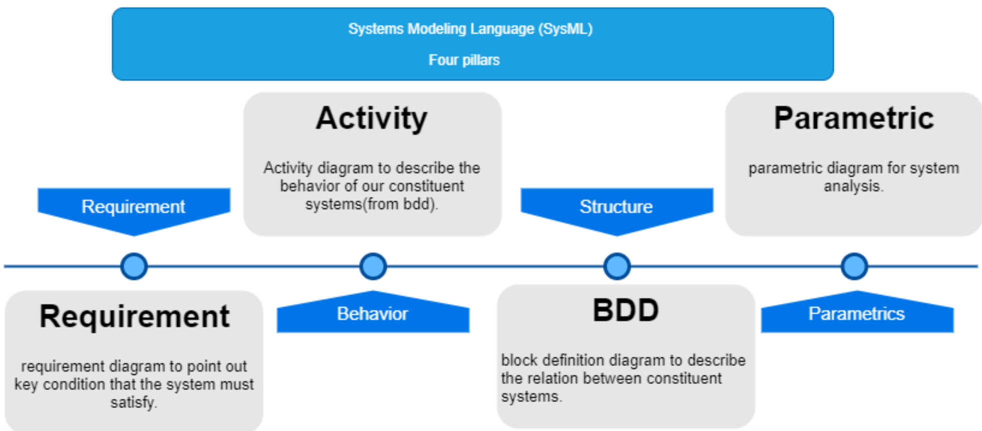


Figure 3.5: preview of SysML four pillars, and they uses in our model.

3.5.2 Requirement diagram

The requirement diagram is a structural diagram that show the relationships among requirement construct, each requirement (rectangle with «requirement» keyword) is a mission or a condition that a system must satisfy [72]. The main mission in our case, is to **maximise the safety of the crowd**. To achieve this mission multiple submission need to be achieved, you cant achieve this mission without **providing fast emergency response** and the ability to **mitigate risks**. For risk mitigation, the submission went with are: (1) **observing the crowd and risk detection**, this mission is necessary for risk detection , you cant mitigate the risk if you don't have ways to know if it exist. (2) **controlling the the crowd**, risk mitigation require a crowd control mechanism, if a CS sees unwanted behaviour he mus have the ability to change it. that why the submission for this mission is **provide security services**. Figure 3.6 below show a screenshot of the requirement diagram.

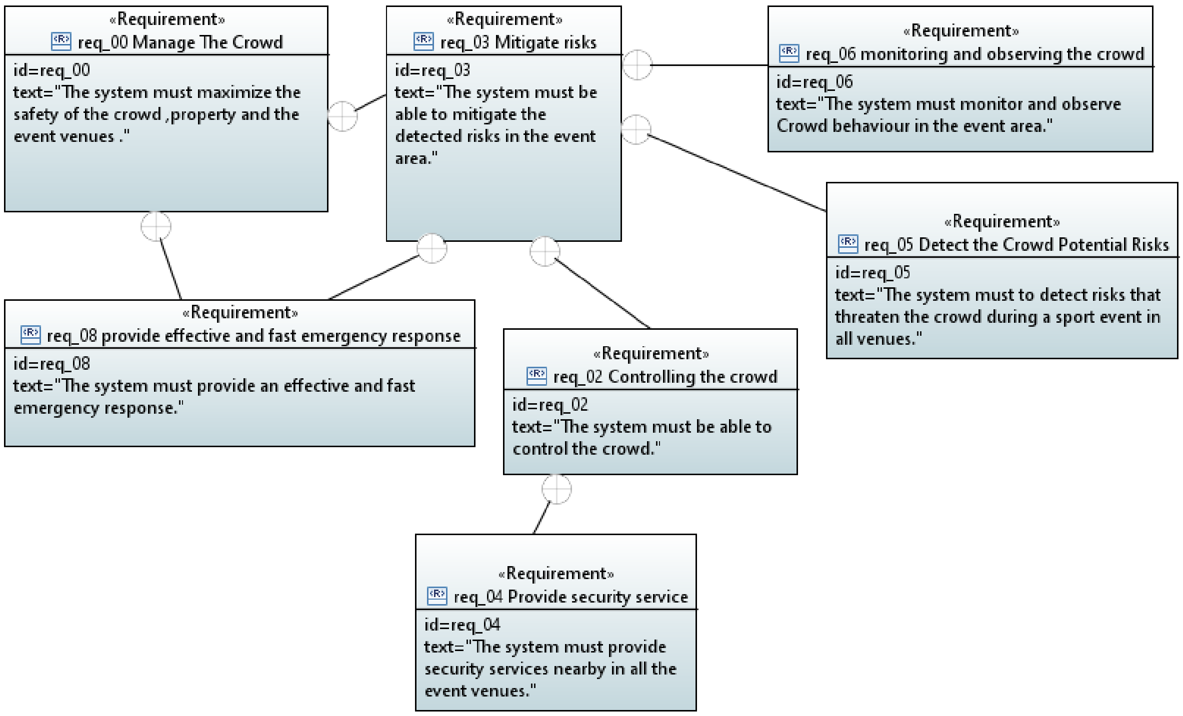


Figure 3.6: crowd management in sport event, requirement diagram.

3.5.3 Activity diagram

SysML Activity diagram is an extension of the UML Activity diagram. It represent the sequence of actions that describe the behavior of a block from a BDD or other structural element. The activity diagram can be described as an operation of a system. The sequence of action is defined using control flow, and the control flow is drawn from one operation to another, this flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join etc.

We used the activity diagram to show the behaviour of our SoS constituent systems, this behaviour is going to be used later on in our simulation. Sport event crowd management have a lot of systems, these systems must satisfy the mission we set earlier in the requirement diagram. The systems we went with are:

– **Control center**, in arena venues we find a lot of detection systems put in place like camera, smokes, gas leaks etc. the control center main activities is to observe the arena through all the different sensors and then notify the agents on the field if risk occur. And call for evacuation in extreme cases.

– **Security systems**, during a fan attendance at the event, providing security is an important task to achieve the missions of the SoS. One of the most utilised security systems in sport events are police officer and security stewards. These last two play an important part in mitigating potential risks, by observing the crowd and intervene to control the situation if risk is detected.

– **Emergency services**, providing fast and effective emergency is one of the important mission in crowd management. where the service at hand need to be expert at situation like this, and react according to the crowd safety. In sport event the most utilised emergency services are, ambulances and firefighters. Figure 3.7, for control center, security agents, and ES implementation.

– **Fans**, fans are the crowd we are managing. All the CSs main mission is provide the safety of this system. In sport event The crowd must enter the event venue and exit from it safely. **Crowd entrance**, consist of the crowd entering the arena to take a seat, it usually involves queues at the gate of the entrance for a search, parking lot if the fan have a car, and some venues, that the crowd uses to buy food, bathrooms etc. We implemented this behaviour as a fan going to get serviced and then taking a seat so he can enjoy the event. **crowd exit**, in most events crowd exiting the event venue it involves of him taking an exit path, and grab his car if he have one.

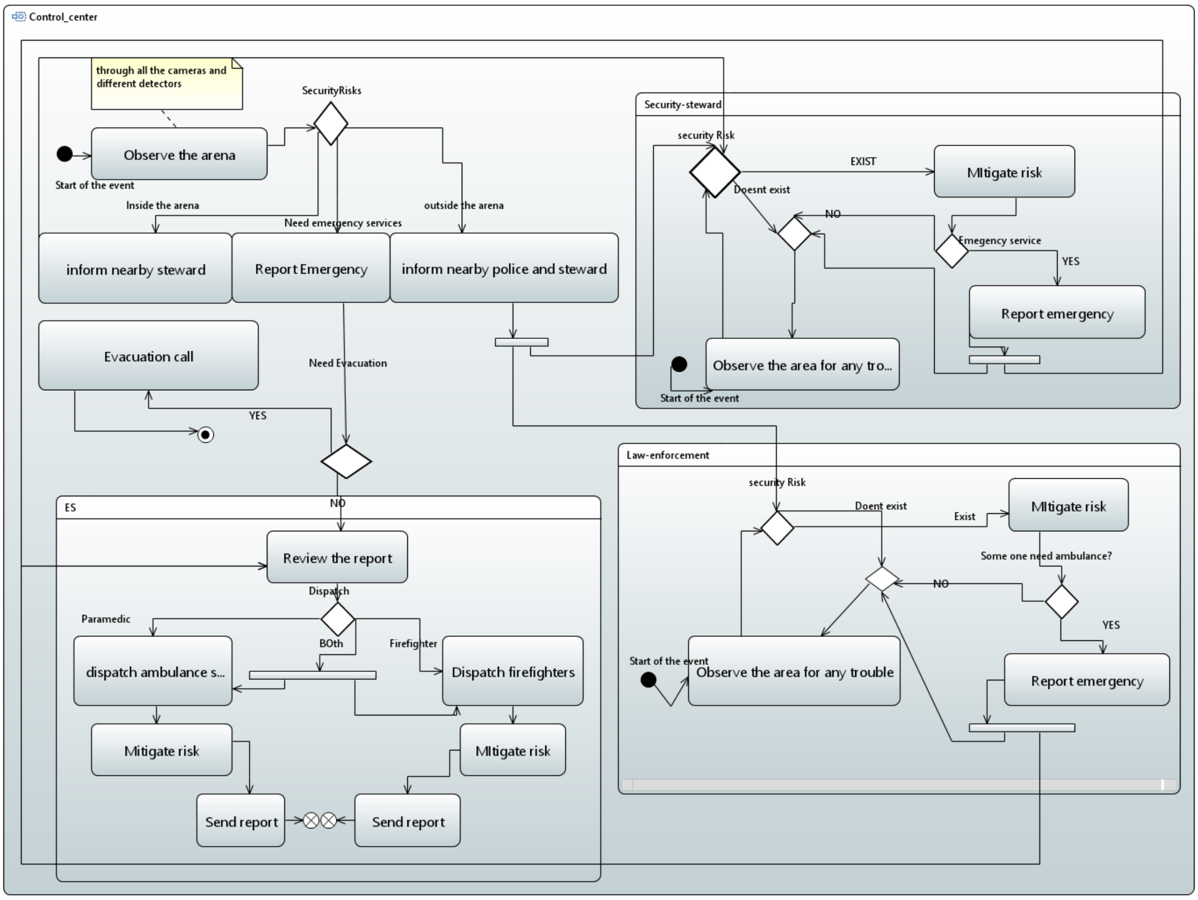


Figure 3.7: control center, security agents, and ES activity diagram.

– **Event venue services**, Sport event venues come with parking lots and other services like bathrooms, fast-foods etc. the people responsible for these service, main activity is to service a fan. Figure 4.7, for the activity diagram.3.10 show fan, car-park attendant, arena venues services, and entrance service implementation.

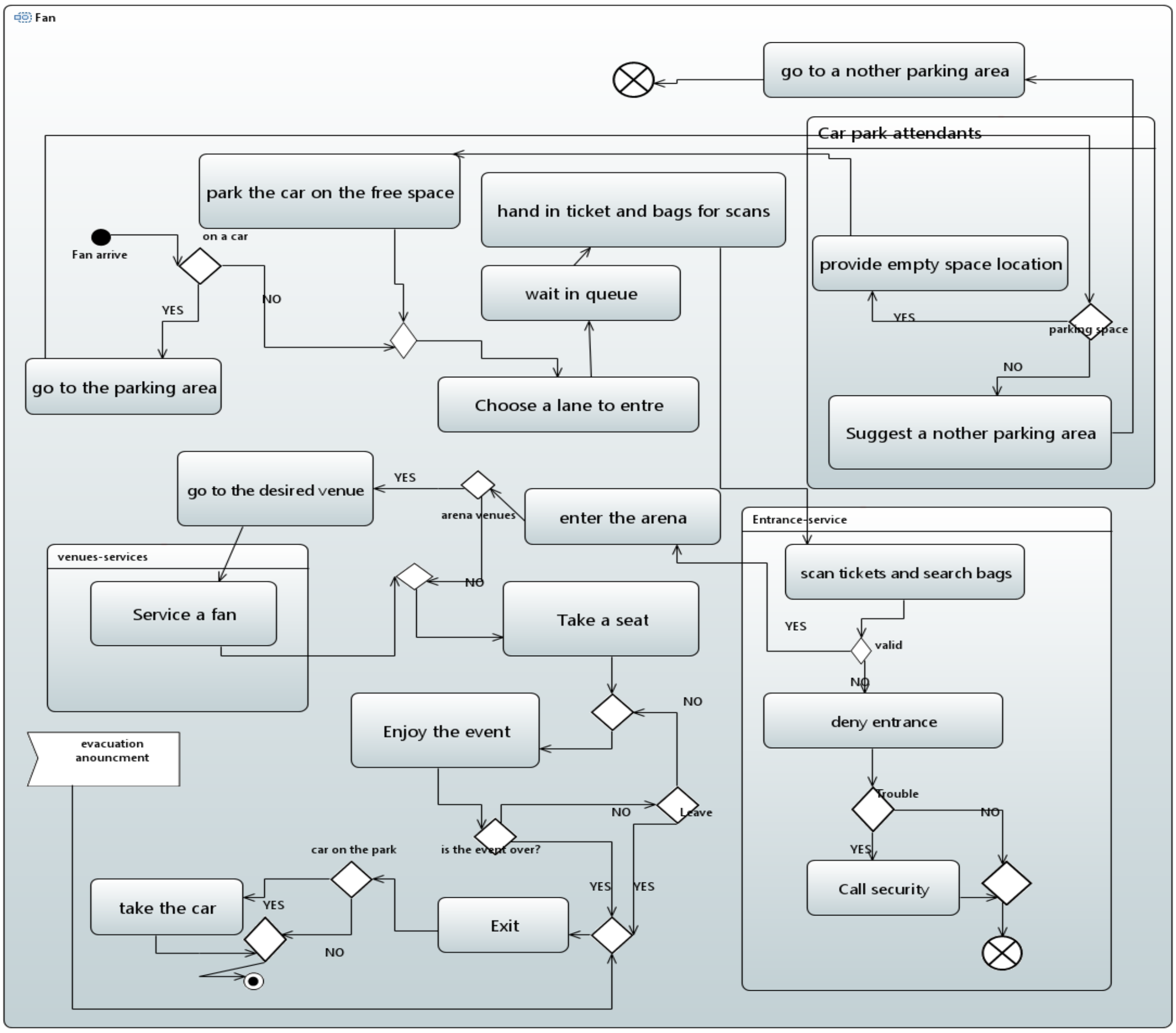


Figure 3.8: fan, car-park attendant, arena venues services, and entrance service activity diagram.

3.5.4 Block definition diagram

Block definition diagram (BDD) is based on UML class diagram from the UML composite structure, BDD describe the relationship among blocks (e.g. composition, association, specialization), each block from the block definition diagram provides a unifying concept to describe the structure of an element, system, hardware, software, data, procedure, facility, or a person etc [72].

We used the block definition diagram (BDD), to set the architecture for our SoS, that involve our constituent systems and the relation with each other's, where each block represent a constituent system. We definitely did not pick all the constituent systems involves in a sport event management, but we picked what we think are the necessary systems and obvious like stewards, emergency services, police etc, to satisfy our requirement. For more details see section.3.5.3. Figure 3.9, and figure 3.10 below show the SoS BDD implementation.

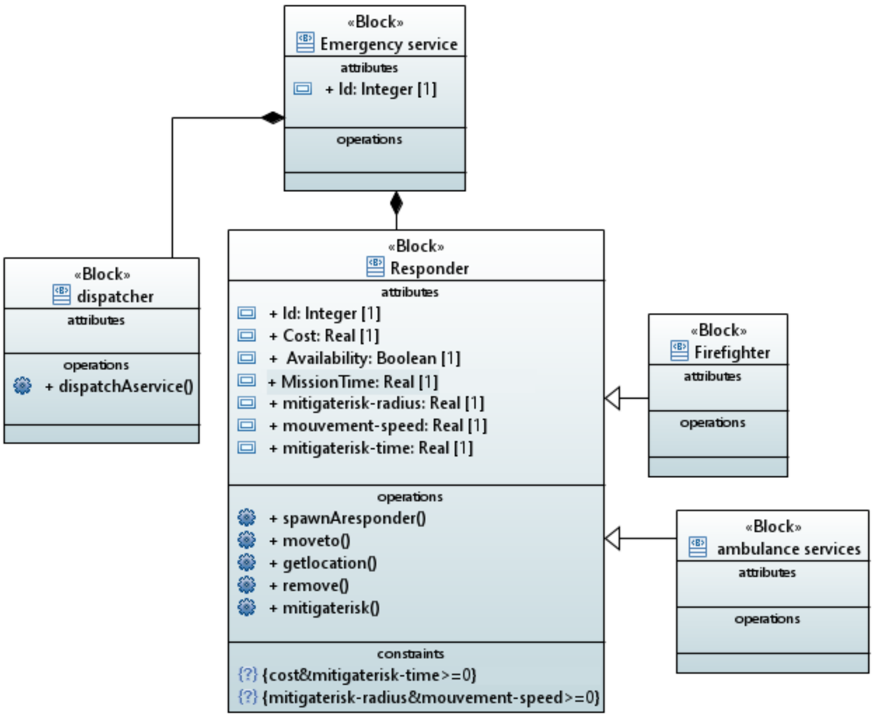


Figure 3.9: Emergency Services BDD.

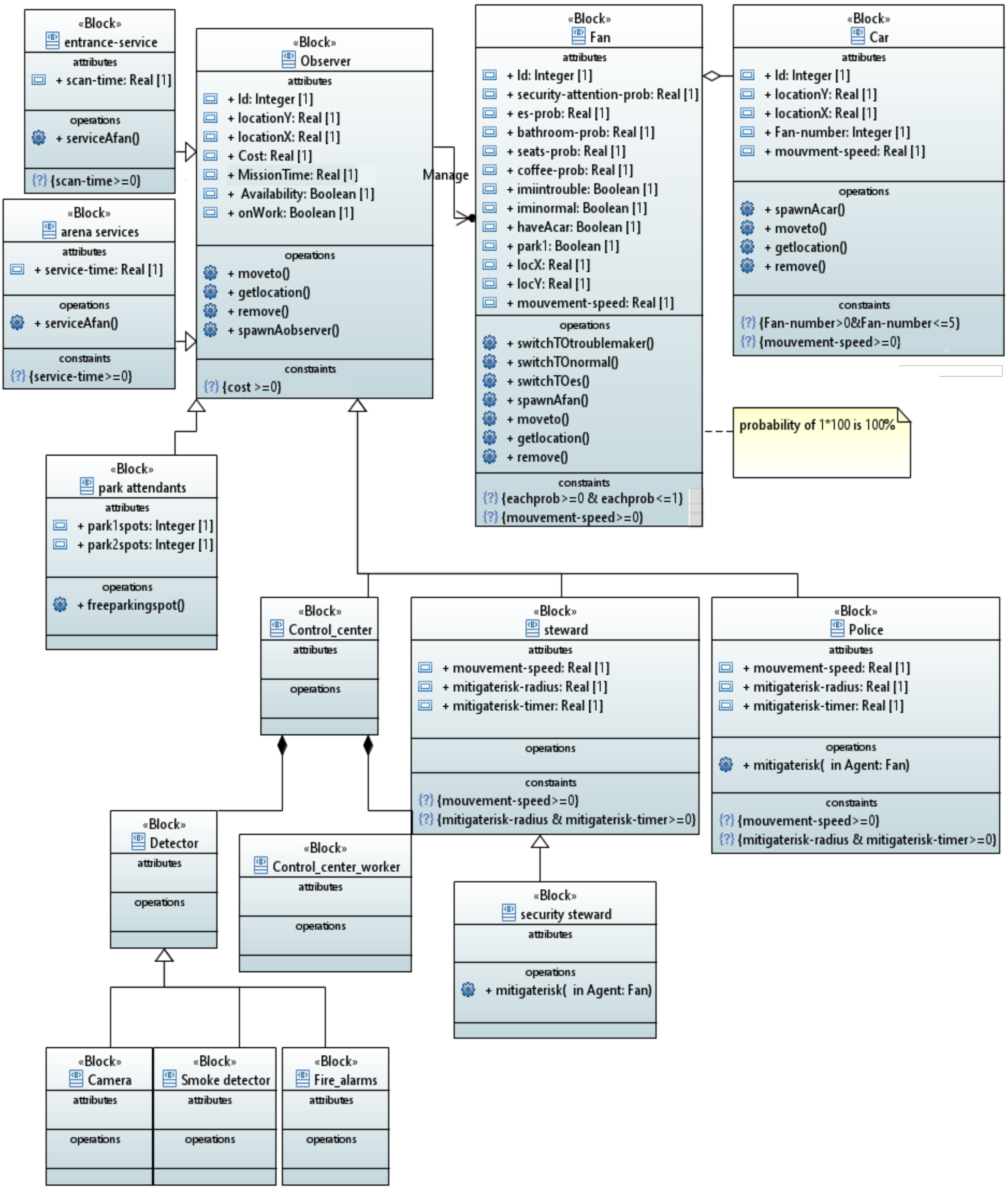


Figure 3.10: Sport event BDD.

3.5.5 Parametric diagram

SysML parametric models is a specialisation of a block diagram that uses mathematical formulas defined by constraint blocks to support the engineering analysis of critical system parameters, including the evaluation of key metrics such as performance, reliability, and other physical characteristics [72].

Performance analysis, is totally dependant on the expert and what they want to analyse from a specific mission, but one of the most used performance analysis, is cost to performance, where we see the time required to achieve a mission and the cost of that. for this analysis we need three main variable availability, cost, and the time require to achieve a mission. Figure 3.11 show parametric diagram implementation.

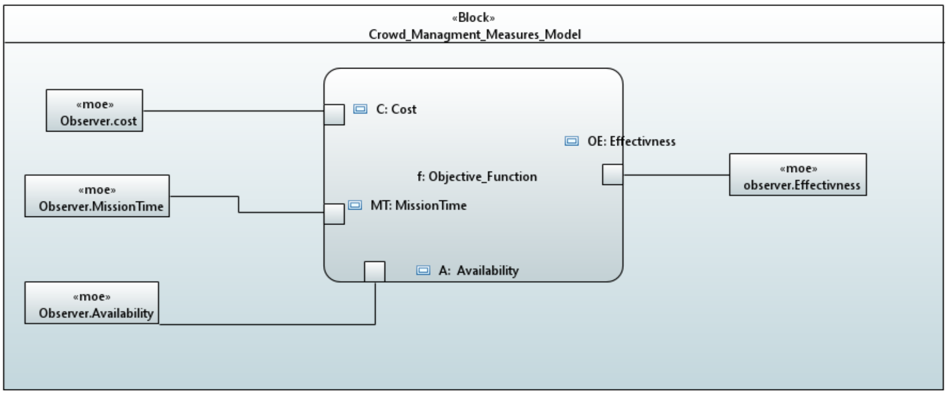


Figure 3.11: Parametric diagram for agent performance.

Another important performance metric for the safety of the crowd and arena performance. Is area density performance. It involves Two main variables **Fan number** on a given **space**. This measure is going to be used on the evacuation scenario to see the performance of the arena movement flow. Figure 3.12 below show the parametric diagram for the density performance.

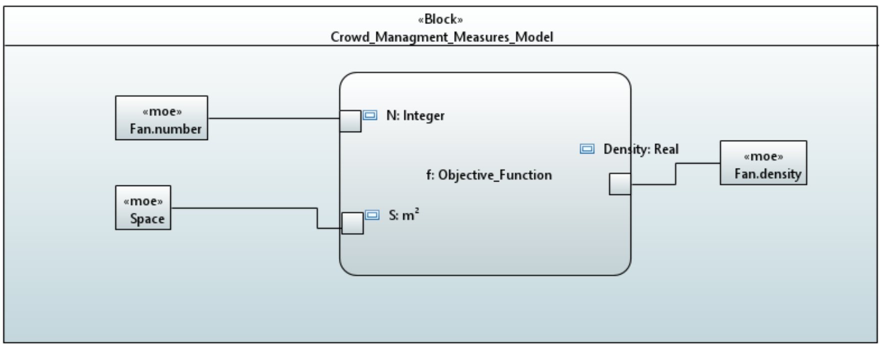


Figure 3.12: Parametric diagram for arena performance.

3.6 Conclusion

In this chapter, we presented our simulation approach for SoS, then we model the crowd management case study. We presented and discussed the the diagrams involved. Then we showcased how simulation can further more refine this approach, if we used it as way to analyse the model at hand and test different outcomes, to plan future SoS updates. The next chapter will show how the simulation was done.

Chapter 4

Sport event: crowd management simulation implementation & evaluation

4.1 Introduction

This chapter provide a detailed implementation of the models shown in chapter.3 in form of a simulation, including the tools used. we visualise the simulation on an architecture drawing of a sport arena, the simulation will focus on the entrance area with the parking lots, the inside of the arena and its different venues, finally the seating area. The simulation will carry out different scenarios in each area. After that we will investigate the scenarios driven, by capturing key data that will be compared to our performance target, in order to decide the optimal solution to achieve the SoS common purpose and goal.

4.2 Simulation approach and execution tool

In this section we will discuss the simulation approach, and platform we chose for our model simulation.

4.2.1 Agent based simulation approach

As shown in chapter.3 crowd management includes multiple systems to work on one goal which is to manage the crowd. Crowd management systems, are mostly led by humans in real world and ABS is the best method for human interaction, and behaviour simulation [71]. In our case, each agent in our ABS represent a law-enforcement, security steward, or a fan etc. ABS strong point are perfect for crowd management simulation, whether its the high detailed characteristic and behaviour of an agent that a bottom-up model structure present, or the multi threaded world that is shared with our systems, where each system need to be active and reacting on the fly according to each behaviour. As of for the crowd trying to mimic human behavior is a complex and hard to achieve. But the best approach to simulate a crowd in our case, is the microscopic crowd shown in section.2.3.2, where we need to preserve physical and behavioural features of each crowd individual, therefore we can benefit the most from our ABS model structure.

The **world** we are simulating is a sport arena venue since we are doing a sport event, and **the BDD diagram** blocks are seen as systems within the SoS and as agent within our simulation. **Activity diagram** is going to be the behaviour and the interaction that the agent represent in order to achieve the SoS missions shown in **the requirement diagram**. And **the parametric diagram** is to capture the performance with a specific function of each mission an agent is partaking in.

4.2.2 Anylogic execution tool

We have carried out our simulation using anylogic, anylogic is a multi-method simulation tool (discrete event, agent based, system dynamics) developed by the anylogic company. We tried out some ABS tools like netlogo and repast simphony, but we found anylogic to be the best tool for agent simulation that fits our needs with its rich simulation methods libraries (e.g. pedestrian library, road traffic library….). Anylogic is a java based platform, which means all of java features are present like the object oriented modeling, security, and the multi-threaded environment that java brings with it was also one of the reason to chose anylogic, getting used to it was easy since we already have a basic java knowledge and sharing the models on the fly between different platforms as long as that platform have a java java-enabled environment, was a nice addition because of java portability.Figure4.1 below show a screen shot of anylogic IDE startup page.

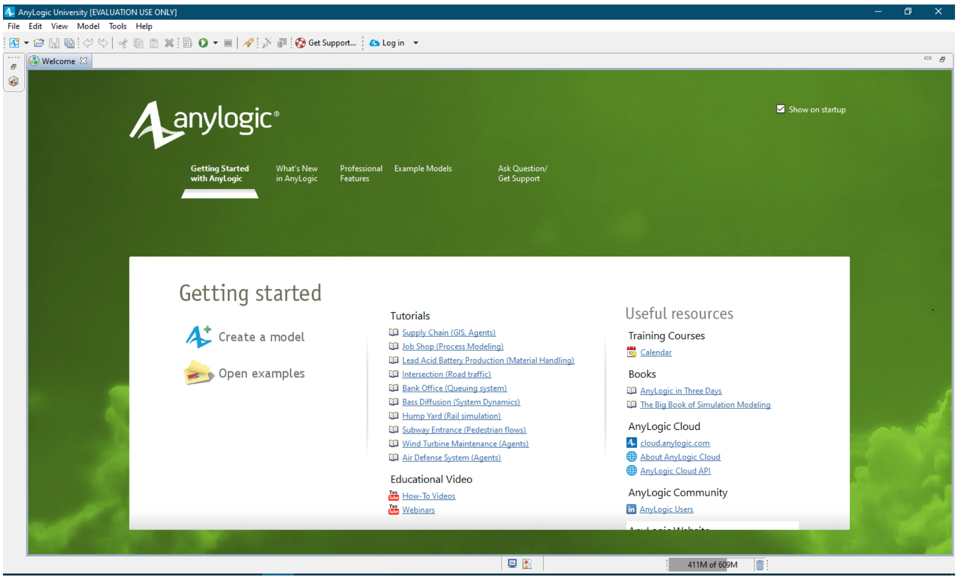


Figure 4.1: Anylogic IDE.

4.3 Simulation display implementation

In this section we present images of the display we chose to visualise our simulation and ease down its complexity to the reader. This section holds two sub section, in the first section we will show the arena target and its different venues, this is the place ”word” that we mentioned in section.1.9.1.3 where we showcased the ABS approach, and the second section will showcase the different CS and their display implementation.

4.3.1 Subject arena sport

To simulate a sport event we need a sport arena. The sport arena we are simulating on is not an official arena it could be a student’s cad (Auto-CAD) architectural drawing. Its was taken from an online cad-drawing website [73]. Our simulation is displayed on these images, so each scenario we are going to provide or discussed, will be based on these architectural drawing structure.The drawing consist of three images each represent a view inside, outside, and seats the arrows drown on these images are there to show agents flow on the map:

– **First**, is a general view of the outside area as shown in figure 4.2, by analysing it we notice that there are **two** parking-lots that can hold up to 243 car, with **two** gates at each one for car entry and another for exit, the gates are shown in a red arrows. we can also notice **two** more gates one at the left of the image which is the main gate for the pedestrian entrance and exit, and the other one is for special agent entrance like the teams players bus or ES, this gate located at the right side of the image shown in figure 4.2 with red arrows also. finally we can notice **four** ways to enter the arena, there is **two** gates shown in black arrow near the pedestrian entrance so they are for pedestrian, and another **two** near the two buses and the special gate show in blue arrows, these gates are special gates as well for special agents same agents as the gate above them. deciding on which gate is for special agent or pedestrian is merely by observation, as we can notice two buses near the special gate entrance with two separate room that contain facilities used by players mostly like gyms and cloth changing lockers, and direct access to the arena as show in figure 4.3.

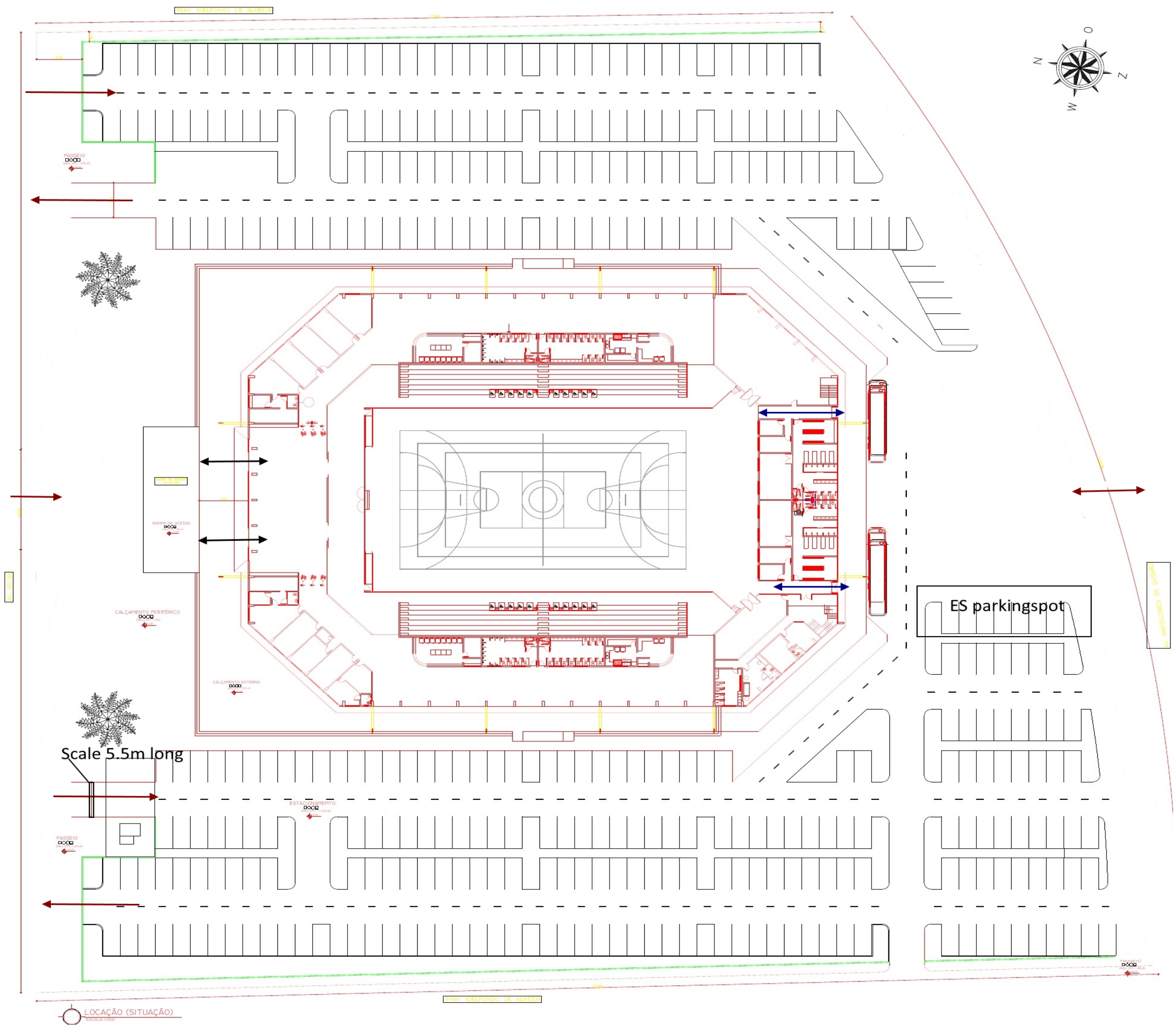


Figure 4.2: general view of the arena, outside area. [73].

– **Second**, Is a general view of the inside as shown in figure 4.3, by analysing the image

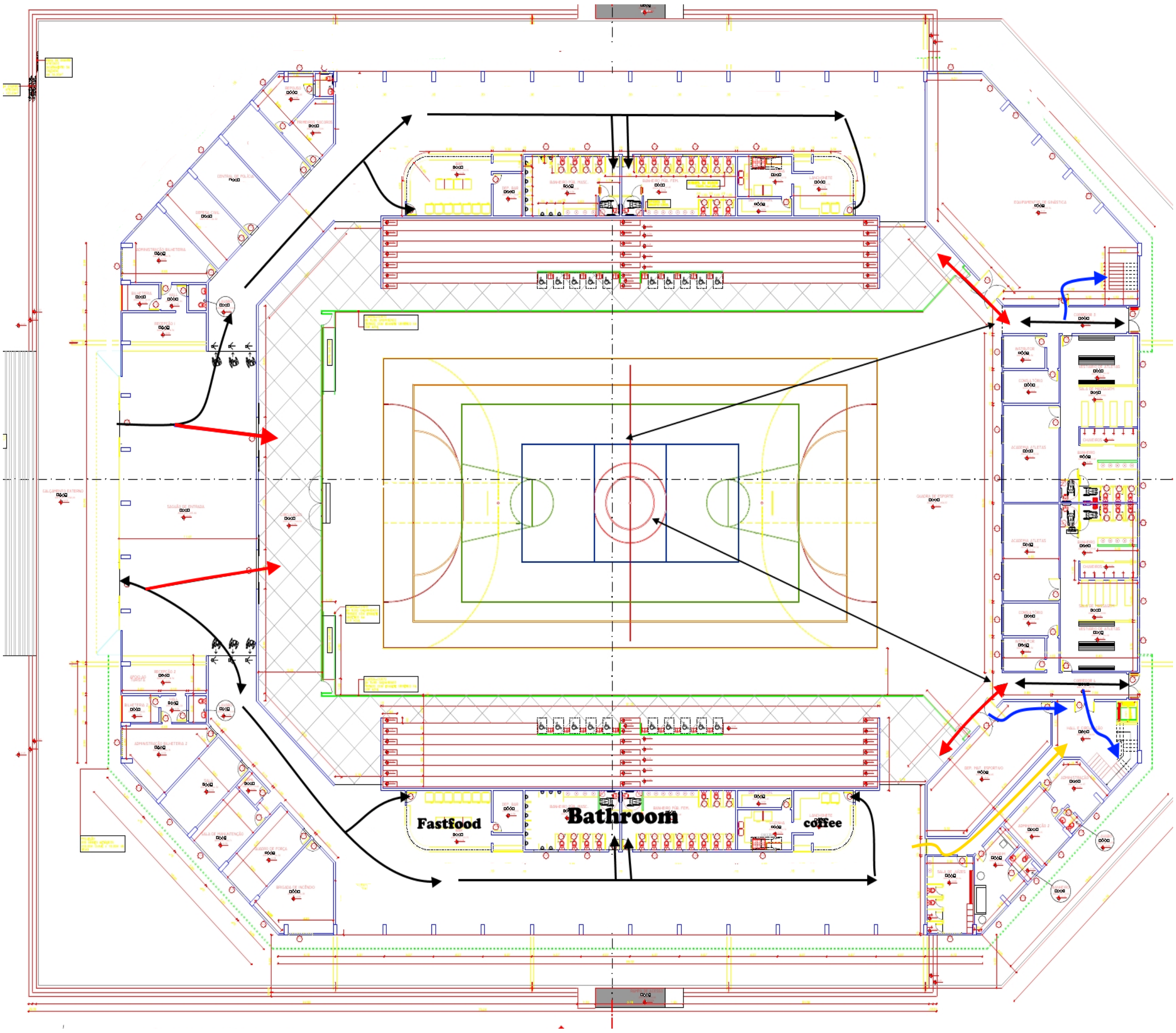


Figure 4.3: general view of the arena, inside area. [73].

we notice that, there is **six** venues that can be used by our pedestrian, these venues are marked with the names fast-food, bathroom, coffee with identical looks in the upper half of the image. the flow of the pedestrian is show by black arrow, we also notice that their are **two** stairs shown with blue arrows, that can get you to the seating area, but they are located near the special entrance, thus they are not for pedestrian entrance. finally we see that there are **four** ways to enter the seating area shown in red arrows.

– **Third**, is a general view of the seating area show in figure 4.4, this view is simple it contain 5248 seats, with multiple stairs to navigate the area. the exist from the seating to the outside is the same as the entrance rout where there is **two** stair plus the **four** gates.

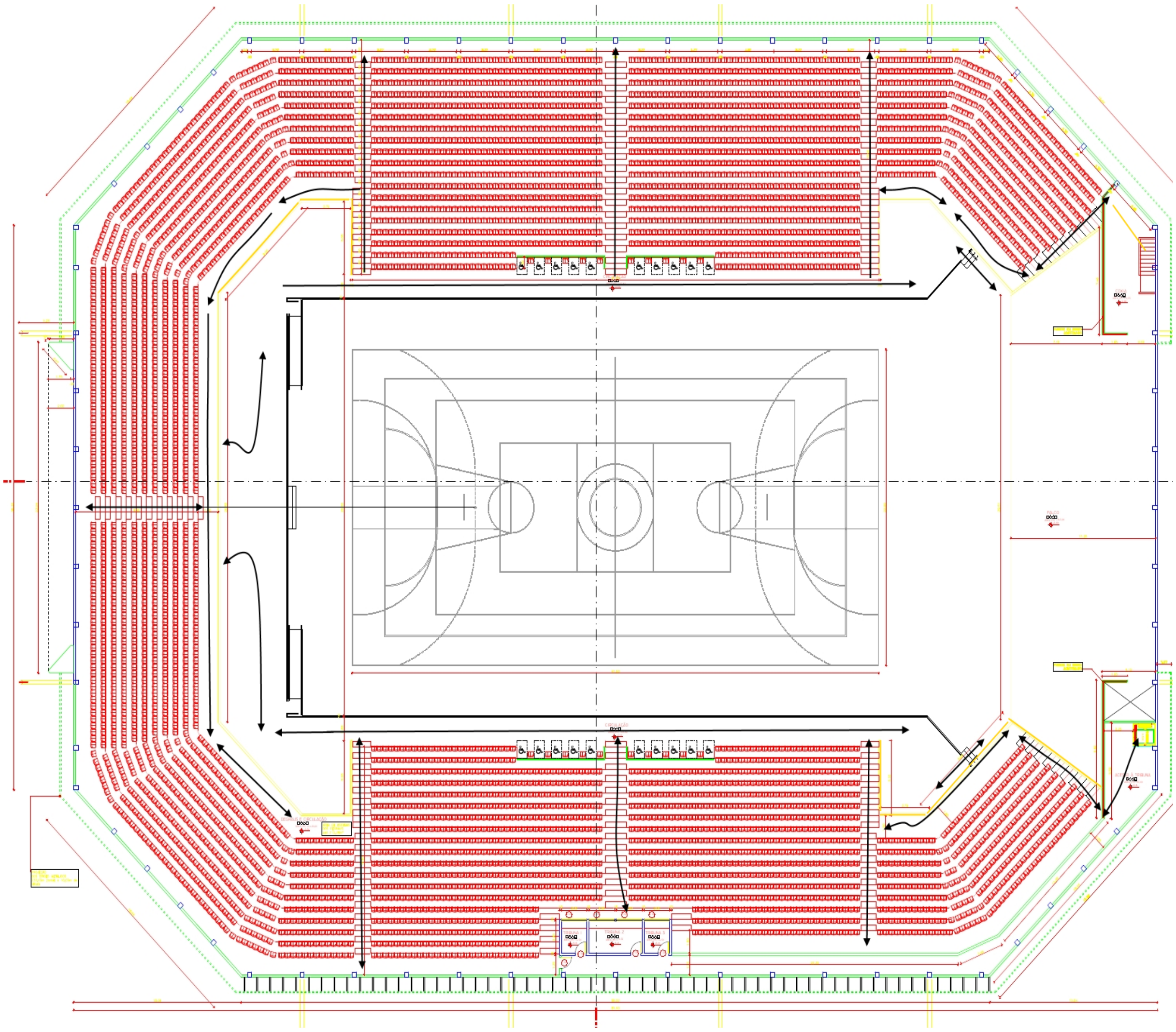


Figure 4.4: general view of the arena, seating area. [73].

4.3.1.1 Arena sport simulation implementation

We used Anylogic representation API (application programming interface) to visualise our simulation, with this API we had the ability to import these images to the graphic editor, then using the pedestrian libraries we are able to draw over our arena images to specify the coordination of the walls, or any obstacle that our pedestrian need to avoid, since these obstacle are drawn above our image, the pedestrian reaction to these obstacle will give the illusion of the crowd moving inside the architectural drawing. See figure 4.5 for an implementation of Anylogic space markup, the arrows are there to show case the gates, stairs and where they lead when entering, thus each ped enter one of these gates will be teleported to the according exit in our simulation. See figures.4.2, 4.3, 4.4 for reference.

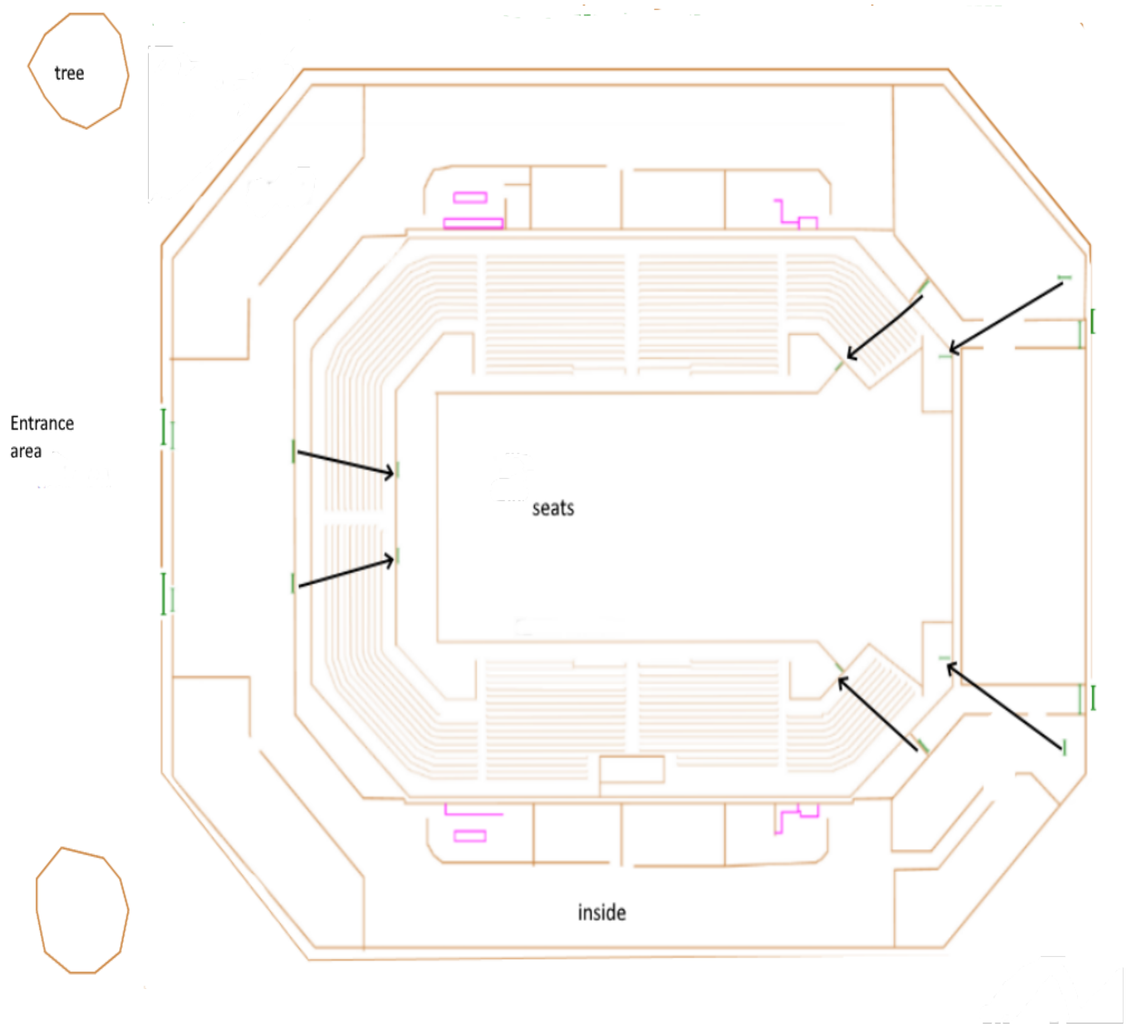


Figure 4.5: representation of the arena obstacles.

4.3.2 Agents simulation display implementation

Agents displays are made using Anylogic presentation API also, with it we made our 2D presentation agent drawing using simple shapes, like circles, rectangles, and lines etc. Anylogic presentation API come with a scale on each presentation display, this scale will be used to decide the perfect space that shapes take on our main simulation images. We decided our scale on the sport event pictures from our parking lots according the cad drawing. as shown in figure 4.2, where its show the scale that represent 5.5m for the road width. And figure 4.6 show anylogic implementation as you can see the presentation API calculated that 1 meter represent 11.87 pixels. This section represent the different agents display implementation, with their scale within a simulation.

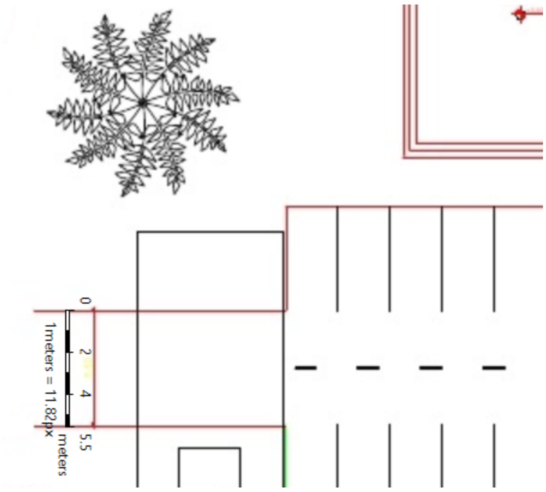


Figure 4.6: sport arena scale.

4.3.2.1 Crowd (fan)

Fans are presented with a simple green circle as shown in figure 4.7, we chose this display for its simplicity and visual clarity in the 2D space compared with the other shapes. we put 0.4 meter on the scale for the fans, because we think that a walking human being or sitting, if seen from the sky will average out on 0.4 meter, so each fan is represented by 0.4 meter green circle diameter on the arena. This value is important because it decide the distance of the crowd between each other, as stated in section.4.2.1 we are using a microscopic approach for our crowd, thus this variable will dictate the physical restriction that each fan will not cross.

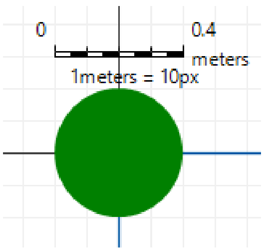
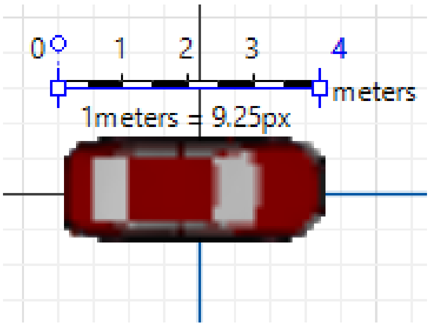


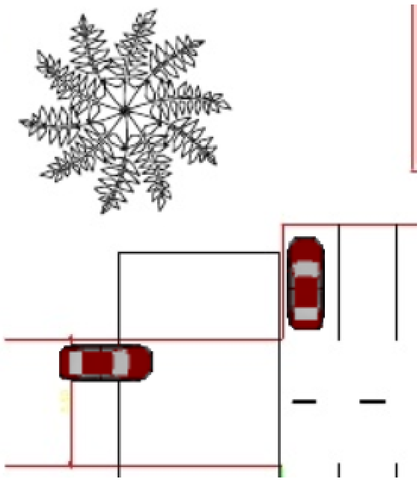
Figure 4.7: Fan simulation display

4.3.2.2 A fan car

Anylogic comes with some pre-built shapes for general use, one of them was a car shape, its looks good and simple so there is no need to change it. For the car scale we made it in a way that will fit on the parking lots, after putting the scale on the arena as shown in figure 4.6, the parking-lot length is around 4.5 meter, so we put the car length at 4 meter. See figure 4.17a for the car scale and its representation according to the parking-lot slot, and figure 4.17b for how it fits.



(a) car simulation display



(b) car in parking-lot

Figure 4.8: car simulation display according to our arena

4.3.2.3 Law-enforcement

police officers are common in sport event, they are used mostly outside the arena to keep it under control, since police officers are humans like our crowd we used the same display, but with a blue colors. police officers are reactive agent that react to the crowd bad behaviour, just modeling their bodies isn't enough remember we are modeling ABS approach, where agent are controlled at the same time in a multi threaded environment, that means when a police officer need to go to a specific pedestrian it does not mean that he will be there, we approach this problem by doing another circle with 2 meter diameter, at its center is the agent body. this circle acts as a range that if a pedestrian is within it the police officer have the ability to intervene to that pedestrian behaviour and change it.

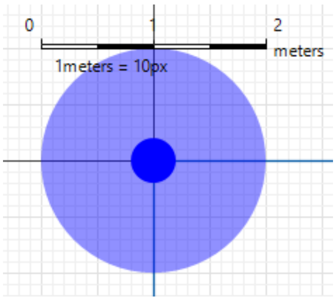


Figure 4.9: police simulation display

4.3.2.4 Control center

visualising a control center is not an easy task, the center give tremendous amount of aid for the agent that manage the crowd, this aid usually is behind the curtains (cant be seen). That why we decide to not visualise it, but with all the sensors and the workers monitoring them we decided to represent that, with the ability that each agent in the arena have instant reaction to different crowd behaviour.

4.3.2.5 Security-stewards

security stewards follow the same concept, that we used on police officers, 0.4 meter for the physical body, and 2 meters for the interaction range see section.4.3.2.3 for more detail. The main difference between the two in our simulation is that law-enforcement are used outside the arena and security-stewards are used inside, to distinguish between the two we used the yellow colors instead of blue.

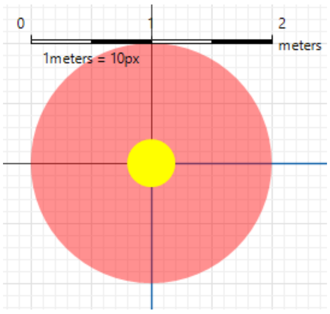


Figure 4.10: security stewards simulation display

4.3.2.6 crowd services

Sport event arenas usually comes with services to increase the crowd comfort ability (e.g.bathrooms, food services etc), or for security reason (e.g.checkup point etc). In our case the services that our simulation simulate are the entrance service for a search, bathroom service, fast-food, and bathrooms as shown in figure 4.3. On general these services in real life represent, a queue line with a point where the pedestrian go to get serviced (e.g a table on a fast-food place, a Turnstile for a verification, scanning place etc), our simulation services are not different it will be represented with a green line for queues and green point for a service place. See figure 4.11 for clear view on the queue line and service point.

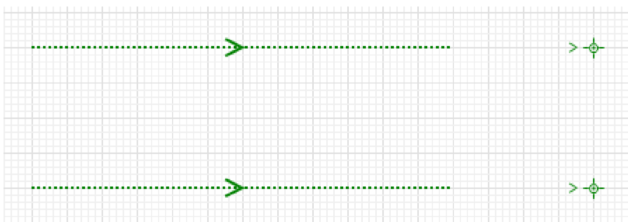


Figure 4.11: queue lines and service points.

4.3.2.7 Parking-lots

Monitoring parking-lot attendant in real life, usually represent of him allowing cars through the gate with no much movement involved. so we decide to implement this agent, with no visualisation like the control center. But this class will be the one responsible for the parking logic, parking logic is simple its consist of spawning a car and then moving it to a free parking slot until the lot is full.

4.4 simulation results and evaluation

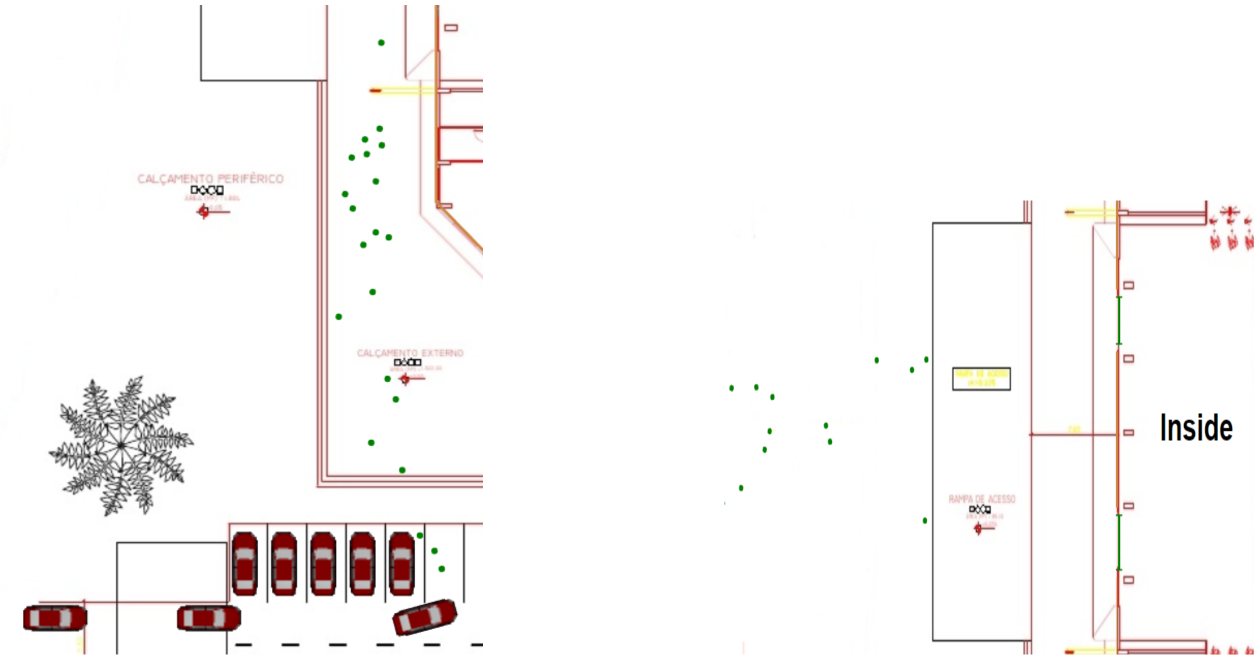
This simulation number one mission is managing the crowd as seen in the requirement diagram chapter2.section3.5.2, since we are doing a sport event simulation, managing the crowd mean a safe pedestrian entrance to the arena, inside the arena, and a safe exit. This simulation is based on three main scenarios: **Crowd entrance**, **Crowd behaviour during-after the entrance**, **Crowd exit.**. Each of these scenarios is represented in a subsections below.

System evaluation is represented in our Parametric diagram for more details see chapter.2 section.3.5.5, its involves three main variables **cost**, **availability**, and **mission time** to achieve the best cost performance in a successful mission. Using this measure of effectiveness (Moe) require knowing each of these variable. Unfortunately in our simulation knowing the cost of each agent mission is impossible since we have no real-life data to be compared to. Thus the evaluation is based on available agent mission timing alone. Since cost is not represented, adding or removing agents and services is based on our target time mission for each scenario (e.g. we can put a mission where we need to enter all the pedestrian into the arena in 3 hours, then we can test the optimal approach to achieve this mission, and the graph results are compared to the time, and the number of services agents, without their cost to time which is the best approach for a mission Moe according to our modelling approach .[7]).

4.4.1 Crowd entrance

The first scenario in our simulation is the crowd entrance, the crowd have one main gate for entrance, and the cars coming for the parking lots. therefore the pedestrian in our simulation will be spawned in the main gate, see figure 4.12b for an example, and for the cars each car will act as a walking gate where each car after it take a parking spot, it will spawn from 1-5 fans at a random variable, figure 4.12a for an example. For reference see figure 4.2 left red arrows.

After the pedestrian spawn they will follow the activity diagram shown in chapter2.section3.5.3. The pedestrian entrance to the stadium is done with the two gates as shown in figure 4.2 with the left black arrows, thus the entrance service will be there with the queue lines. The next activity according to our activity diagram is picking a venue whether its the bathroom, coffee, or fast-food this activity will also be at random percentage see figure 4.3 for reference. The next activity will be entering the seating area and taking a seat, That will conclude our crowd entrance simulation. the figure below show an example of ped spawn locations.



(a) lower parking-lot (b) main gate

Figure 4.12: peds heading to the entrance gate

4.4.1.1 Crowd entrance mission

One of the main problem in any events entrance is queue time, queue time are important factor in the crowd satisfaction criteria. That's why this simulation mission is focused on queue times. The average queue time for each ped will be gathered from gate one, two, and each of the inside venues (food, bathroom, coffee), using the average formula below:

$$A = \frac{1}{n} * \sum_{i=1}^{n} p_i \tag{4.1}$$

where:

- A: is the average queue time for all the pedestrian that entered a queue.

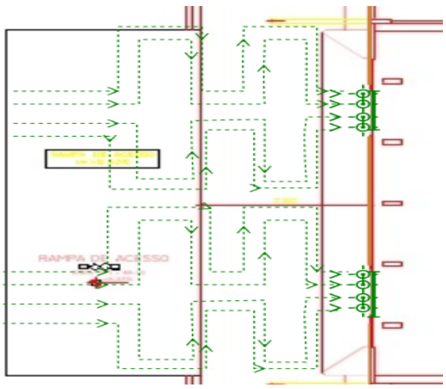
– n: is the number of pedestrian that entered the queue.

– i: i=(1,2,3,4.......n).

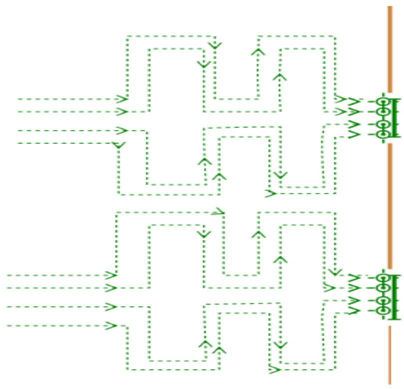
– p_i: is the pedestrian number i queue time.

The mission of this scenario is to service 5248 pedestrian (number of seats inside the arena). to the seating area as fast as we can, while keeping the queue time as low as we can in different venues. Since this is a simulation the spawn rate of pedestrian is up to us, we can spawn 5248 in an instant if we want to, but we need the maintain a realistic approach to the mission. With no real life data to model to, the data we can gather is unlimited for the average queue time, where a lot of variables come into play to achieve that, like the pedestrian spawn rate, the movement speed, the probability for a pedestrian to enter each venue , queue line numbers and so on, some are logical to play with and change and some are not(e.g.changing a pedestrian movement speed is not logical, as an average human being cant go beyond around 1.4m/s).

Because of the unlimited data we can possibly get, we decided to aim for the **limit** that our architectural drawing can bring. The main system that come into play in this scenario is the entrance service system, so we started with there to see how many service point this system can hold with respecting the fan scale see section.4.3.2.1. as you can see in figure 4.13 up to 4 service point each gate. and for the arena venues we can fit 22 service point for each fast-food venue, 11 service point for each coffee venue, and 30 service point for each bathroom side. shown in figure 4.14.

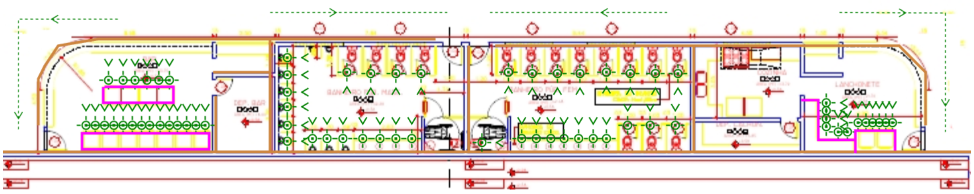


(a) entrance service points setup with the image background

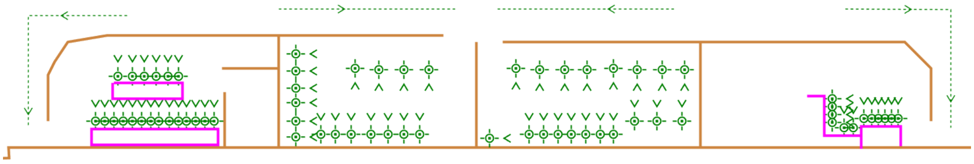


(b) entrance service points setup without background image

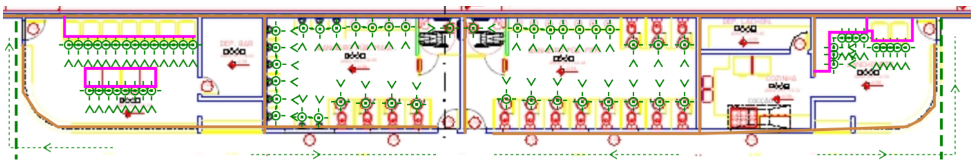
Figure 4.13: entrance service points setup



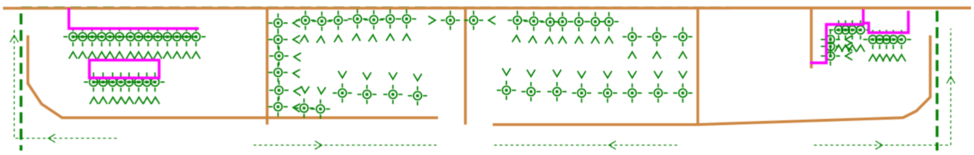
(a) arena upper venues service points setup with the image background



(b) arena upper venues service points setup without background image



(c) arena lower venues service points setup with background image



(d) arena lower venues service points setup without background image

Figure 4.14: arena venues service points setup

Running a simulation in this setup will help us detect its limits according to our targets times, as a start for example we can put the target at **10 Minute max**, queue time in entrance gates. with **2h max** entrance time. To achieve this we need a spawn rate for the pedestrian that can put out 5248 pedestrian in less than 2 hour, with the cars spawn and gate we can start the simulation with 60 ped a minute with each car 1 to 5 random spawn, that should be enough to generate 5248 ped in 2 hour.The idea here is to generate enough peds, to see if the setup above can hold around 5248 ped in less than 2 hour.

4.4.1.2 Crowd entrance result

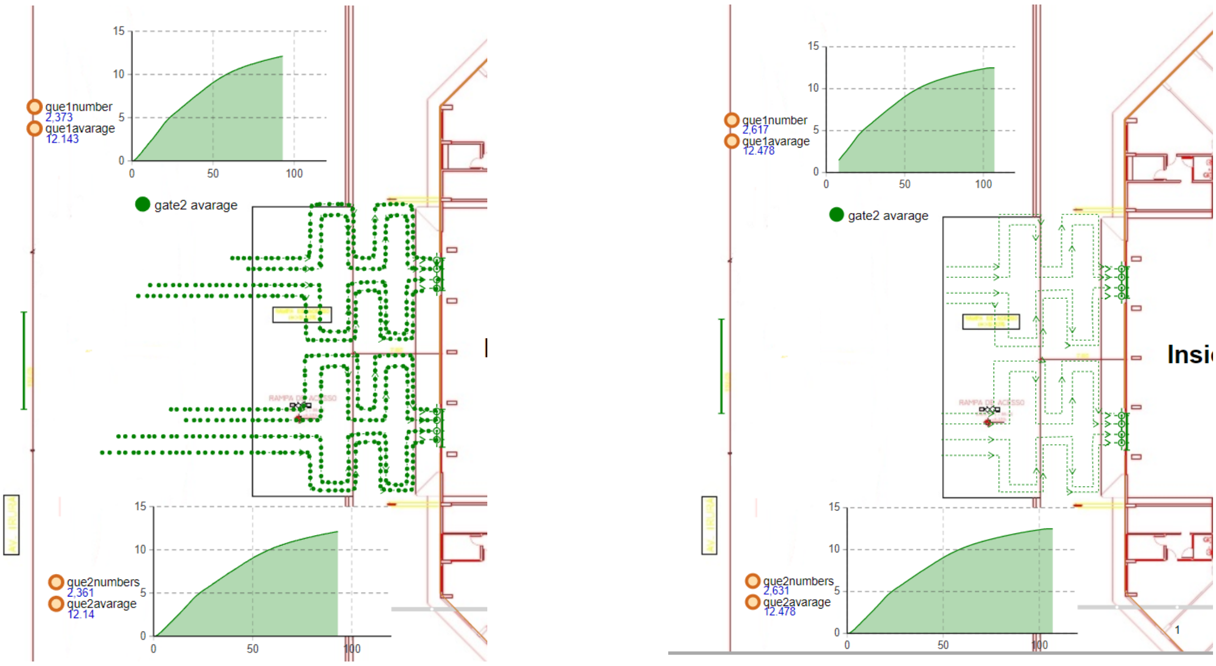
The table below summarise the initial variable used in our first scenario. Most of the characteristic are show in chapter3.section.3.5.4 in the conception part. The variable that we put are based on fast google search's, common knowledge, experience or completely random variable to test some outcomes. For the inside venues probability, peds going to seats probability start at

0.1, and each 30m of run time it will increase by 0.3, while each of the other probability decrease by 0.1, since the closest the match is about to start the probability of heading straight to seats increase.

Initial variables for each ped			
Characteristic	Variable	Characteristic	Variable
Pedestrian, movement-speed	Random(0.8, 1.3) m/s	car spawn rate	6 per minute
main-gate peds, line pickup probability	two line 50% to pick for each ped	peds car, spawn rate	Random(1, 5)
cars peds, line pick up probability	park1->gate1 and park2->gate2	Bthroom-prob	0.4 (40%)
seats-prob	0.1	food-prob	0.3
main-gate peds, spawn rate	60 per minute	coffee-prob	0.2
gate1, queue lines	4	gate2, queue lines	4
gate1,2 service point duration	Random(5,10) s	bathroom service duration	Random(3,7)m
fast-food service duration	Random(3,6)m	coffee service duration	Random(2,4)m

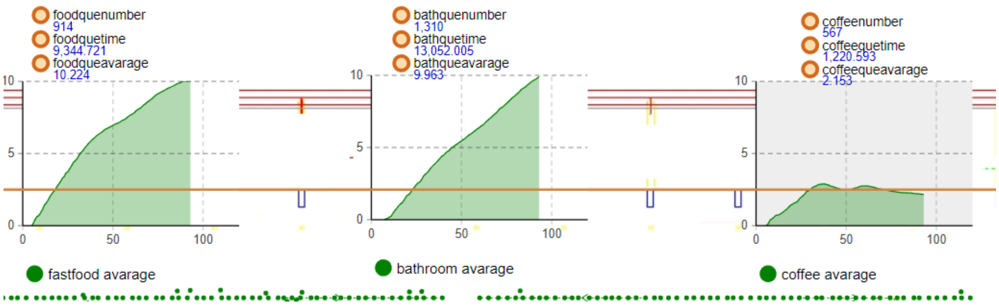
Table 4.1: Crowd entrance initial variables.

This setup serviced all the pedestrian in 105m, that 15m less than our targeted time, but the average queue time is beyond 10m at around 12m, the average time exceded the 10m mark way before the 105m mark as shown in figure 4.15. As for the inside venues even though the results will be different since we are dealing with random variables for each ped. we can notice from figure 4.16 that at this pedestrian rate, the average wait times inside the venues is around the 10m mark, at the values 10m for fast-food service, 11m for the bathroom, and 2m for the coffee.

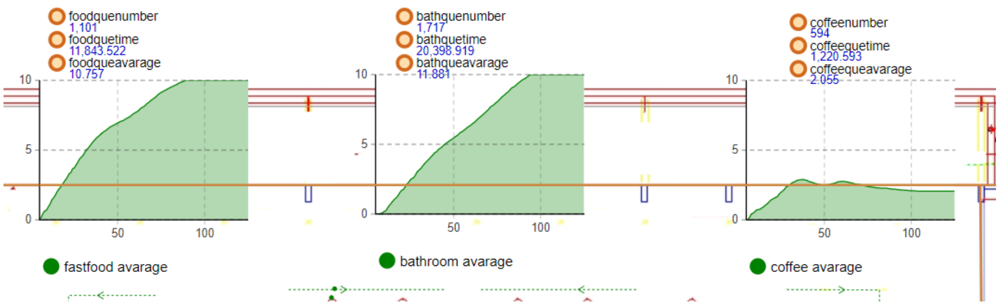


(a) result at 94m outside (b) result at 105 ending

Figure 4.15: outside first scenario results



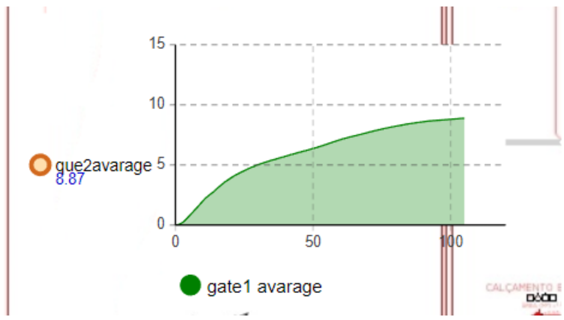
(a) Inside result at 94m



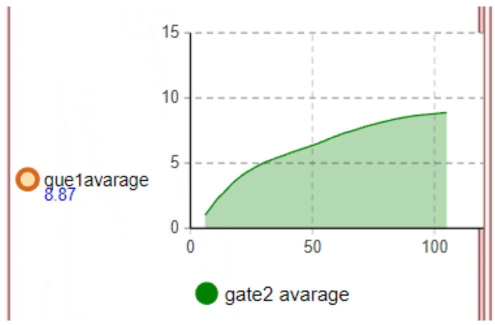
(b) Inside result at the end

Figure 4.16: result at the end

Deciding where to go now is based on the priority at hand. if we want lower average queue time at this pedestrian spawn rate, we need to add more service point or lower the pedestrian spawn rate, since its the main cause for the queue time. And if we want to lower the entrance to the arena time, the only way is to add more service point, increasing the pedestrian spawn rate have no effect on the time required to service the crowd, as long as there is 5248 peds spawn in the course of 100m. see figure4.17 for more detail, the spawn rate for this is 50 ped a minute, the time it took to service this rate is 104m.



(a) result at the end gate1



(b) result at the ending gate2

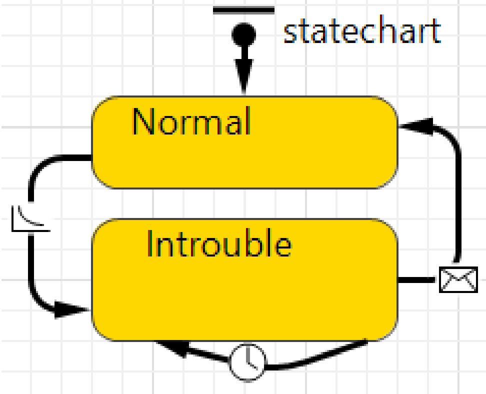
Figure 4.17: average queue time at 50 ped/m

4.4.2 Crowd behaviour during-after the entrance

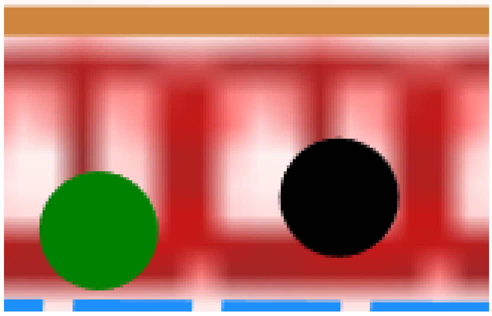
The approach we used to simulate the crowd behaviour, is the same one shown in figure 1.6. By a behaviour in our model we mean, when a pedestrian normal behaviour change, so he starts seeking another security agent attention to him, whether its a big problem or a small one like seeking guidance. Our crowd behaviour is based on **two states** as show in figure 4.18, where the state change occur during the normal behaviour of the fan (following the activity diagram). Our pedestrian behaviour state model work as follow:

- **Normal state**, the normal state is the initial state, that our pedestrian take when spawn shown in figure4.18a. and with a specific rate (e.g.1% of the crowd population), we can start transforming our peds into In-trouble state.

- **In-trouble state**, in the in-trouble state, the transforming agent turns into a black circle shown in figure 4.18b. at this state the agent start seeking the attention from police officers if the ped outside, or the stewards if he is inside. after that if the required agent is free, he can move to the troubled ped and intervene by turning the targeted ped back to the normal state. for more details refer to.section.4.3.2.3.



(a) fan behaviour



(b) Normal- in trouble states

Figure 4.18: Fan behaviour

In most events the reactive agents (e.g police, stewards), are found at specific places at lunch, thus at the start of our simulation each reactive agent is given a spawn location. The movement of these agents are done using **Dijkstra's algorithm**, Dijkstra's algorithm is one of the best short path finding algorithm, to implement this algorithm first we need a network. A Network can be seen as the map that contains detailed paths and crossroads that our agent will move along to. This network is drawn using the Anylogic API on top of the arena drawing (same concept as obstacle drawing), with a distance between each path as weight. Drawing the network over the arena give the illusion of agent moving along side the architectural drawing. Since our agents are moving on these paths, the network need to touch every point of the arena, specially the seating area since all of our peds are heading there as described in the entrance scenario.section.4.4.1. The network implementation is shown below with blue lines in figure 4.19, figure 4.20

Since our world is multi threaded, the possibility of a reaction agent being accessed by multiple peds at the same time is bound to happen. To solve that a simple **semaphore** was put at place. semaphore is a concept used to control access to common resources, in a multi threaded environment. To achieve that the reactive agents are put into a resource pool (java linked list), and when a pedestrian turns into a in-trouble state, the nearest reactive agent from this pool will react to him with him being marked, so other peds don't get access to that agent, until his activity is completed. Each peds that find the resource pool empty is gonna reenter the in-trouble stat after 1m. The agent activity the same as the activity diagram, its consist of him going to the ped location, then turning him back to normal state. for more detail see chapter2.section.3.5.3.

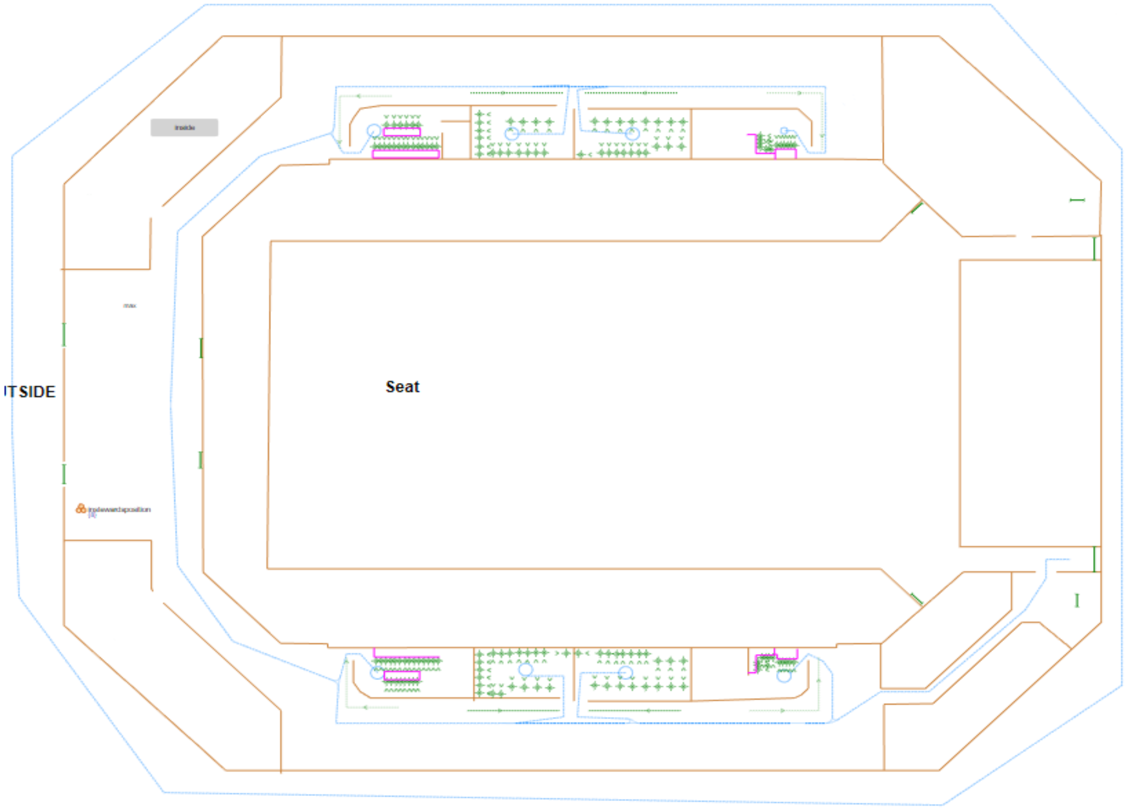


Figure 4.19: Network representation inside and outside the arena.

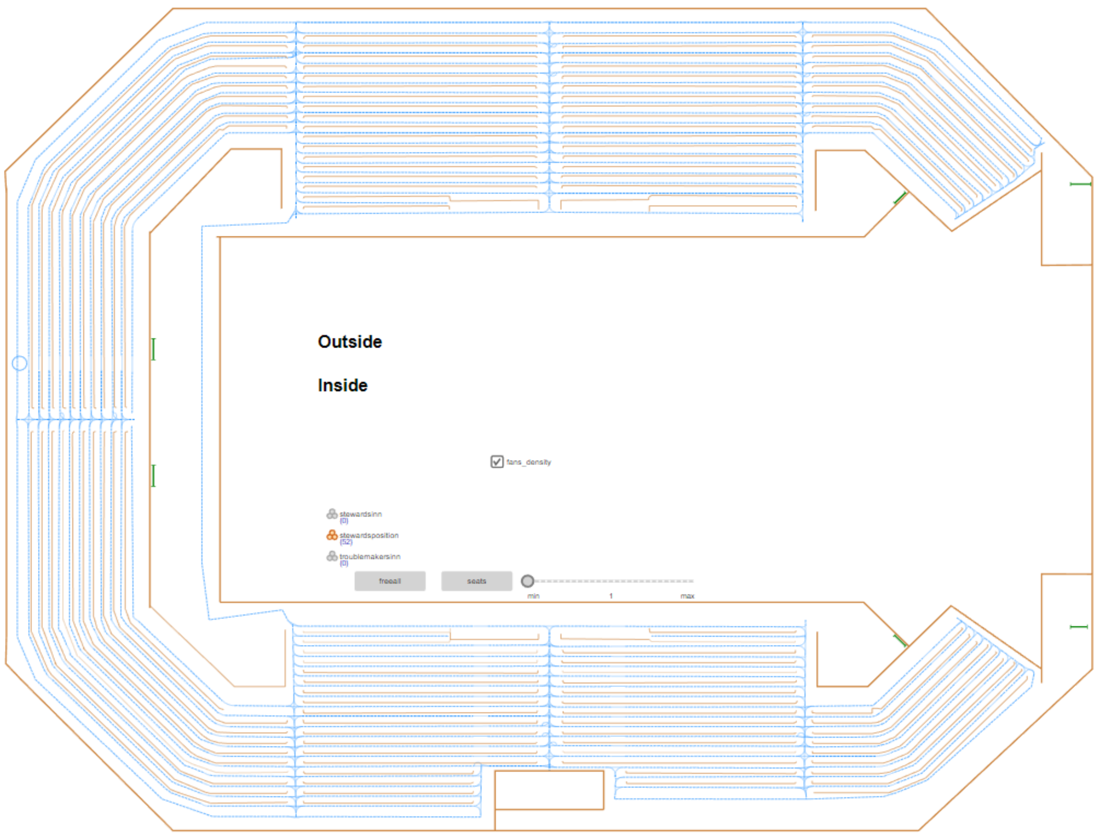


Figure 4.20: Network representation in seating area.

4.4.2.1 Crowd behaviour during-after the entrance mission

For this scenario we are going to focus on, the capability and the performance of the reactive agents in clearing up the in-trouble peds, the data we are going to capture in this scenario, is the average time the peds are going to stay in-trouble state, for this we are using the same average formula we used on the entrance scenario.

$$A = \frac{1}{n} * \sum_{i=1}^{n} p_i \tag{4.2}$$

where:

- A: is the average time for all the pedestrian in the state in-trouble.
- n: is the number of pedestrian that entered the state in-trouble.
- i: i=(1,2,3,4.......n).
- p_i: is the pedestrian number i, in-trouble state time.

Like the entrance scenario we are going to need, a startup data then test its limits, the variable we are starting with are 28 security stewards in seating area, 5 inside, and 3 police officers outside the arena. These agent are spread over the arena, to cover more ground space for the best results. For to state transformation rate we are going to start at 20% in the course of 1 hour. And for the transformation time required to turn the fan back to its normal state will be a random variable from 10 to 20s. we will let the simulation run for 150m, since all of the pedestrian get serviced at 105m mark, thus the first hour is not a good representative of 20% rate, but the second one is, since all the crowd is spawn by then. figure 4.21, figure 4.22 represent the placement of our agents within the arena.

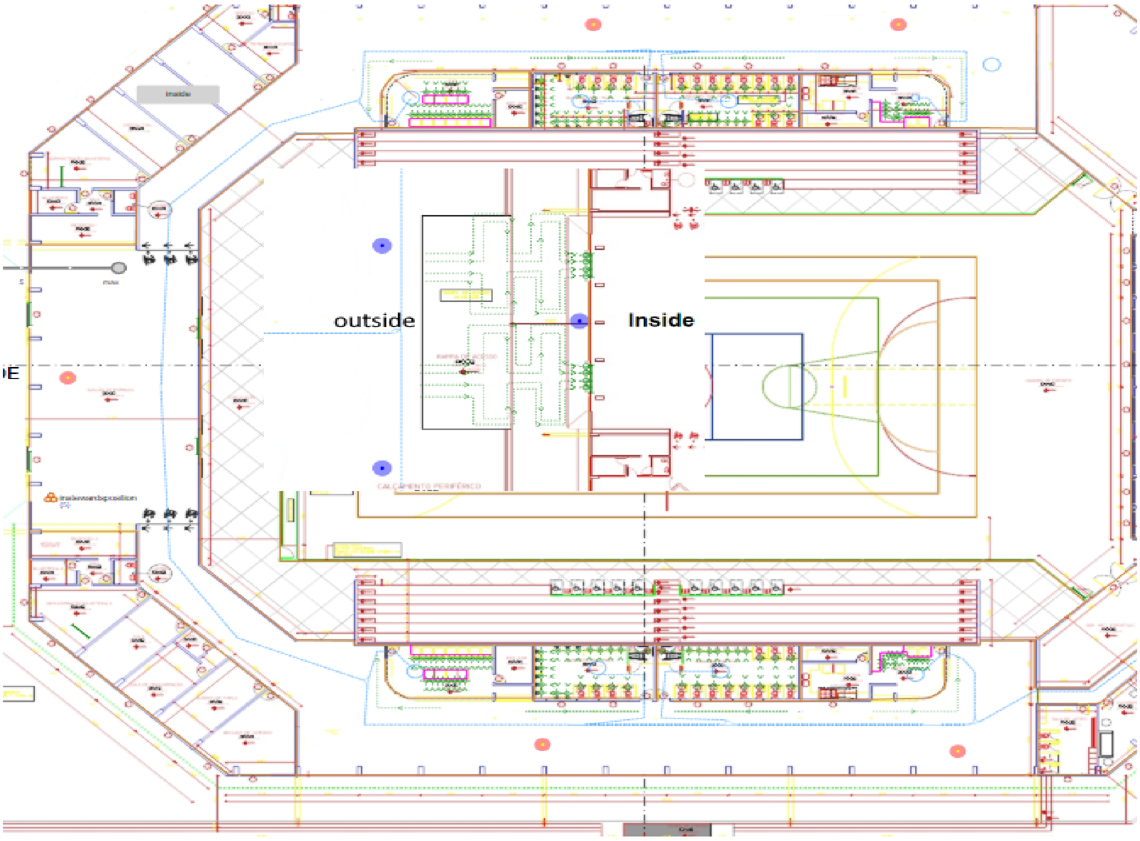


Figure 4.21: agents placement inside and outside the arena.

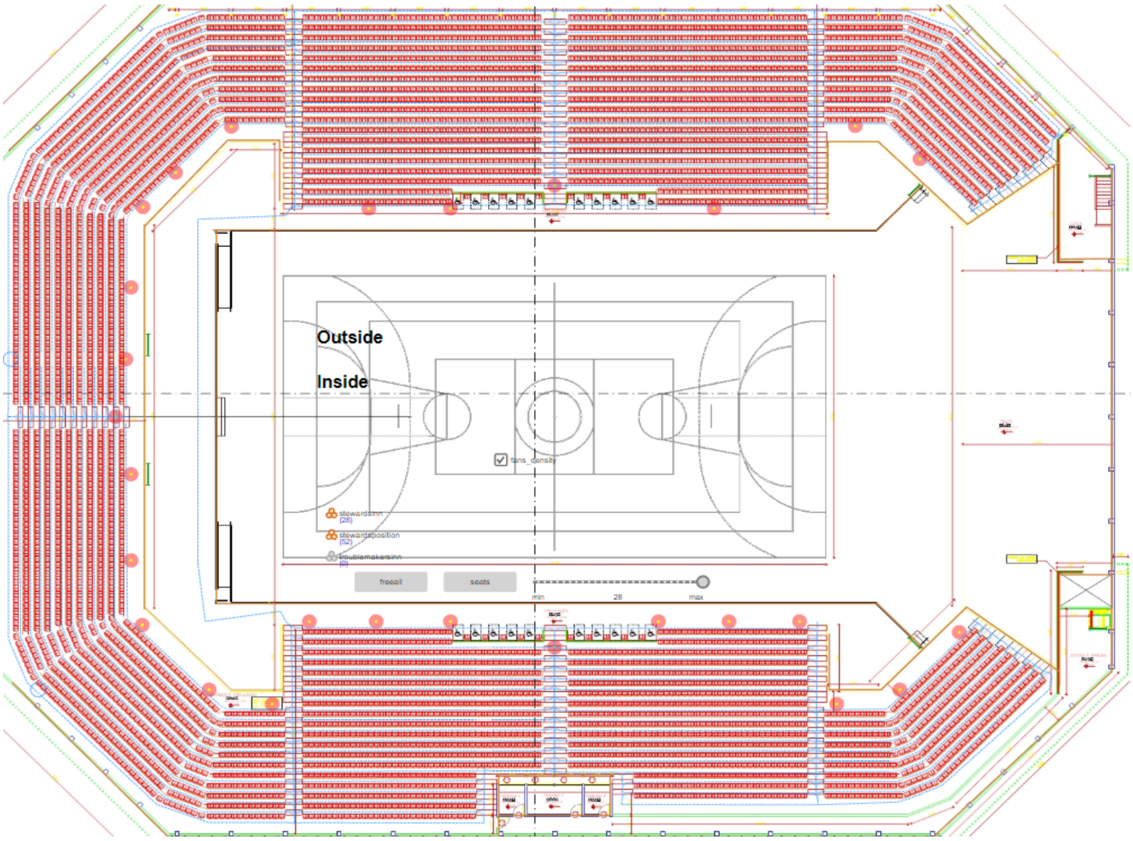


Figure 4.22: agents placement at the seating area.

4.4.2.2 Crowd behaviour during-after the entrance results

The simulation is going to continue where we left off from the first setup, with the addition of some new variable, the table below summarise this scenario setup. and the figure 4.23 show and example from the simulation run, of the security agent at work.

Initial variables for this scenario			
Characteristic	Variable	Characteristic	Variable
agents,movement-speed	Random(0.8, 1.3) m/s	number of agent, at seating area	28
number of agent, inside the arena area	5	number of agent, outside the arena	3
transformation rate	0.2/h	Time to change fan state	Random(10,20)s

Table 4.2: Crowd behaviour during-after the entrance scenario initial variables.

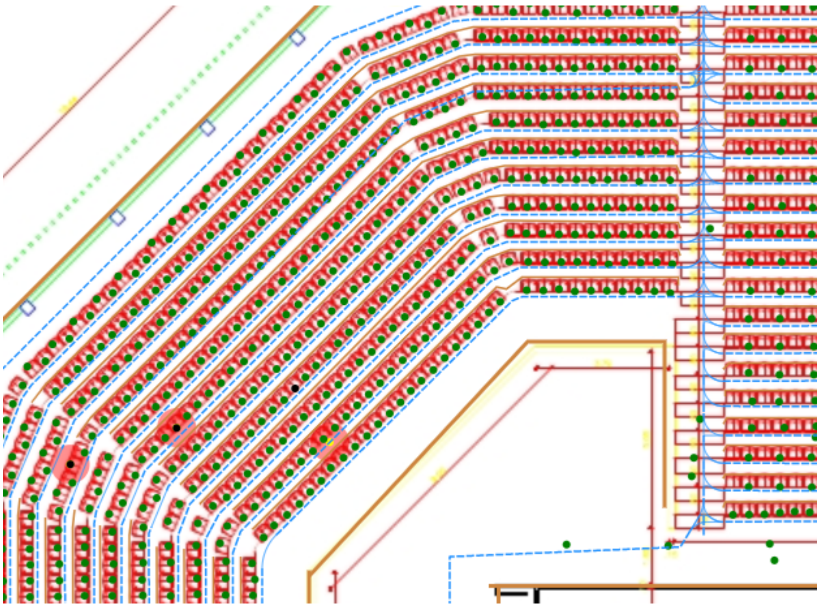


Figure 4.23: security steward at work, at the seating area.

This setup, is able to keep the average time of in-trouble state at 5.4m/ped, the simulation ended with 1722 changed state and 110 fan in in-trouble state. This is the ideal result if we want 5m/ped as our mission target. and if we continue the simulation there is no indication of the average time increasing, we can see this clearly in our chart with the red circle, that show the start of a stabilisation after the full amount of peds are already spawned. the figure 4.24 below show the pedestrians average time in-trouble state during 150m.

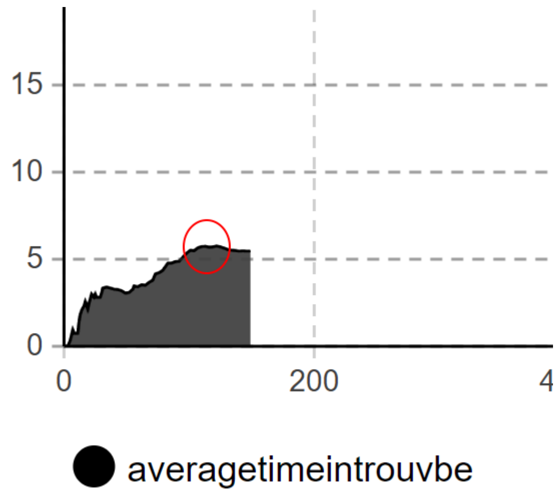


Figure 4.24: around 20% state change rate per hour result.

Increasing the rate to 30% an hour, show a clear result of what a hopeless situation look like in real life, with 9m+/ped average in-trouble state. the setup simply cant keep up, at the end of simulation there is around 734 pedestrian in-trouble state, and the number keep increasing as the simulation progress, and the chart show no sign of stabilising compared to the scenario above. The main reason for this out of control situation, is the rate which control the the crowd behaviour first, and the security agent numbers second. That why in real life scenarios knowing the crowd is important factor in deciding the optimal security approach. Th figure 4.25 below show the result of 30% rate in an hour interval.

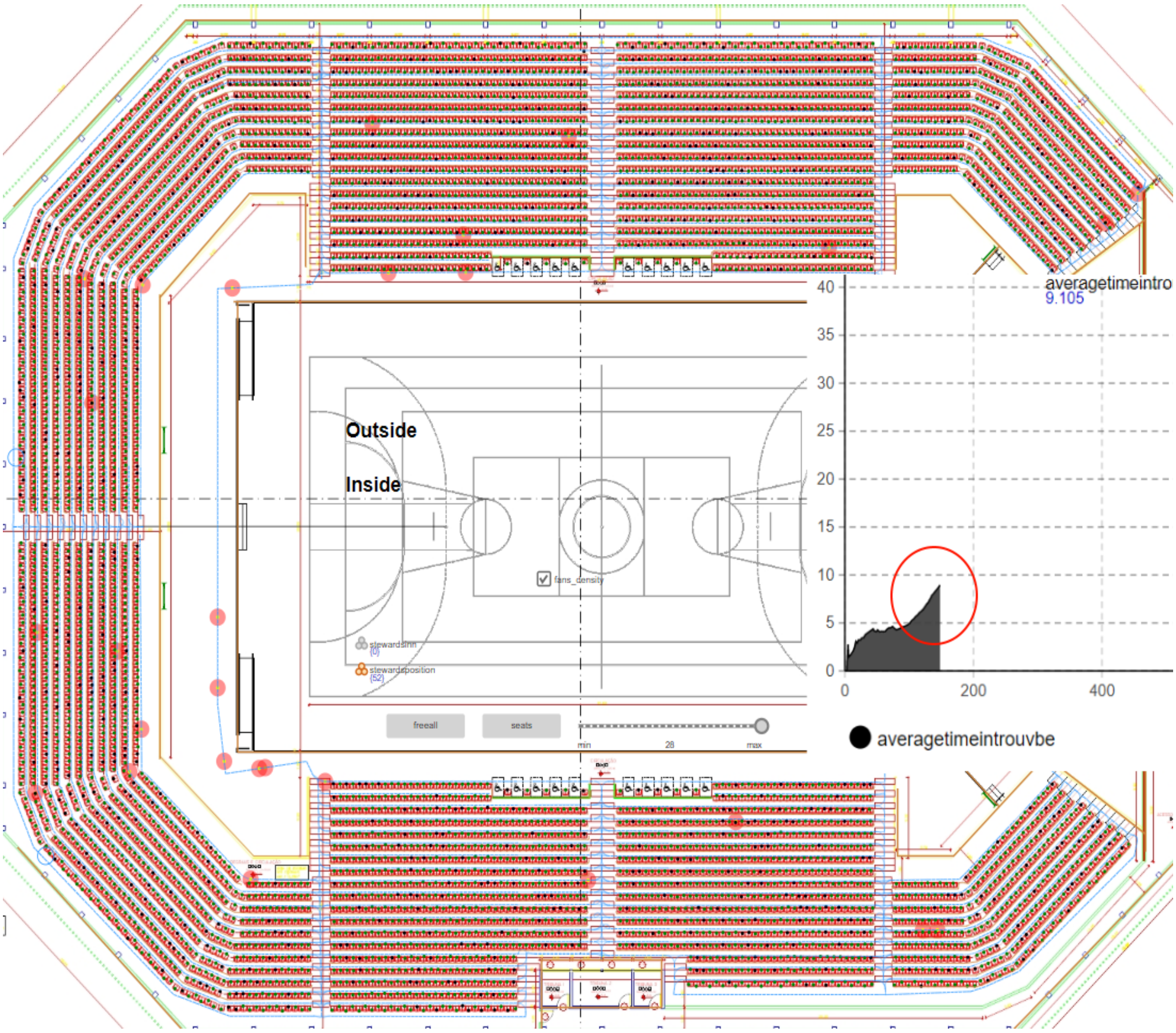


Figure 4.25: around 30% state change rate per hour result.

4.4.2.3 Emergency services

Emergency services, are present in almost all sport event, when you enter a stadium you usually find some firefighter and ambulances parking inside the stadium for fast emergency response. To achieve that they use predefined paths that take the ES from the parking-lot to the desired point. Modeling this is almost already done because the network that we used on the behaviour agents follow the same concept. for reference see figure 4.19, figure 4.20, figure 4.2. The main mission for our es model, can be done by simply analyzing the arena and see if there is any point in the arena that have pedestrian in it, but there is no rout for the ES to get there.

The arena at hand with its pathing routs, are shown in figure 4.2, figure 4.3, figure 4.4, while designing the network, we found one problem in the arena architectural drawing, the problem is inside the arena in the upper half, there is a wall with no way to enter that side of the arena. for more details see figure 4.26 below, and figure 4.19 for reference.

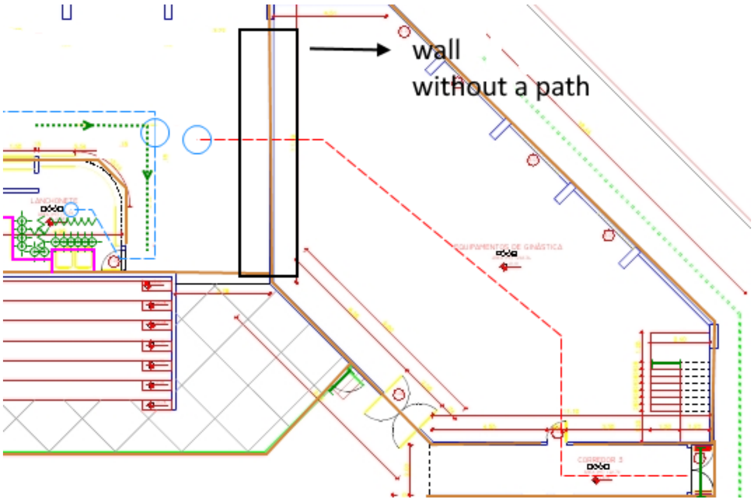


Figure 4.26: upper half of the inside arena.

to see the time we can save using the red path versus the blue one shown in figure 4.19, we spawn a police officer shape at the es parking-lot, and run it on the both networks, we found that creating a path there can save up to 3.2m, since using the long path take 4.4m, and the short path 1.2m.

4.4.3 Crowd exit/Evacuation

Getting the crowd out of the arena safely, is one of the most important scenarios in crowd management, that if its done incorrectly disastrous consequences can happen. The main cause of a safe crowd exit, is a well though arena architecture. The arena layout with its obstacle and flow can be seen in figure 4.5, figure 4.3, figure 4.2.

4.4.3.1 Crowd evacuation mission

The main mission of this scenario in our simulation, is to see the approximate time required for our crowd to exit the arena, and to detect critical areas. **Critical areas** are spaces within the arena, that bottleneck the pedestrians flow, and forces them to stop moving until there is enough room for movement. These areas can be calculated within the simulation by checking if a certain space have a pedestrian density of 4+/pedestrian per m^2 [68]. figure 4.27 below, show an example of a critical area.

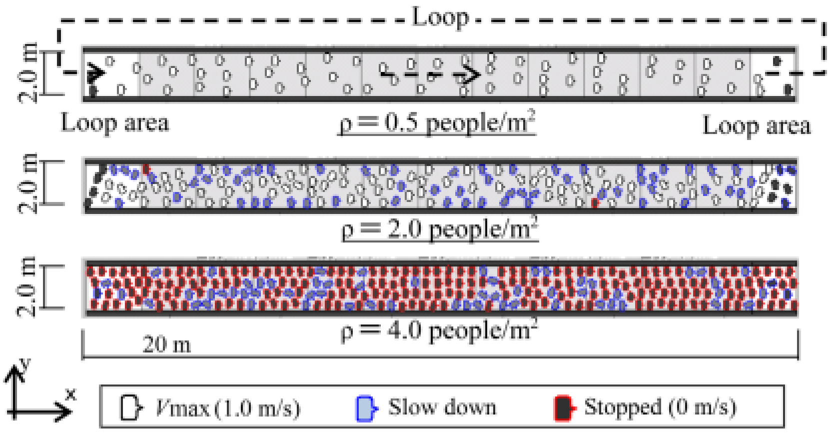


Figure 4.27: Critical area example [68].

The initial setup in this scenario will be the default arena layout at first, the default layout have 4 gate to enter and exit the arena, and 2 stairs at the arena right side. see figure 4.2, figure 4.3 for reference. Each pedestrian is going to head to the nearest exit gate to him. The time of evacuation is calculated at the moment the full amount of pedestrian exit the 4 gates. and the critical areas are represented in red color filter within the simulation. figure 4.28 below show the filter, we are using to display the color of each pedestrian density interval.

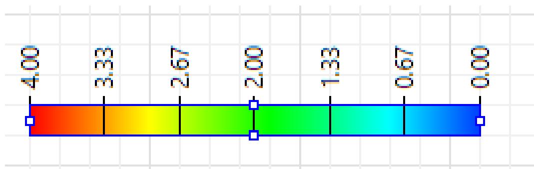


Figure 4.28: colors of the filter used to display pedestrian density.

4.4.3.2 Crowd evacuation results

After 18.9m of run time the evacuation ended, whether this is a desired result or not, is totally dependant on the target timing performance. but seeing the amount of critical area shown in the result figure 4.29 there is a lot of things to improve.

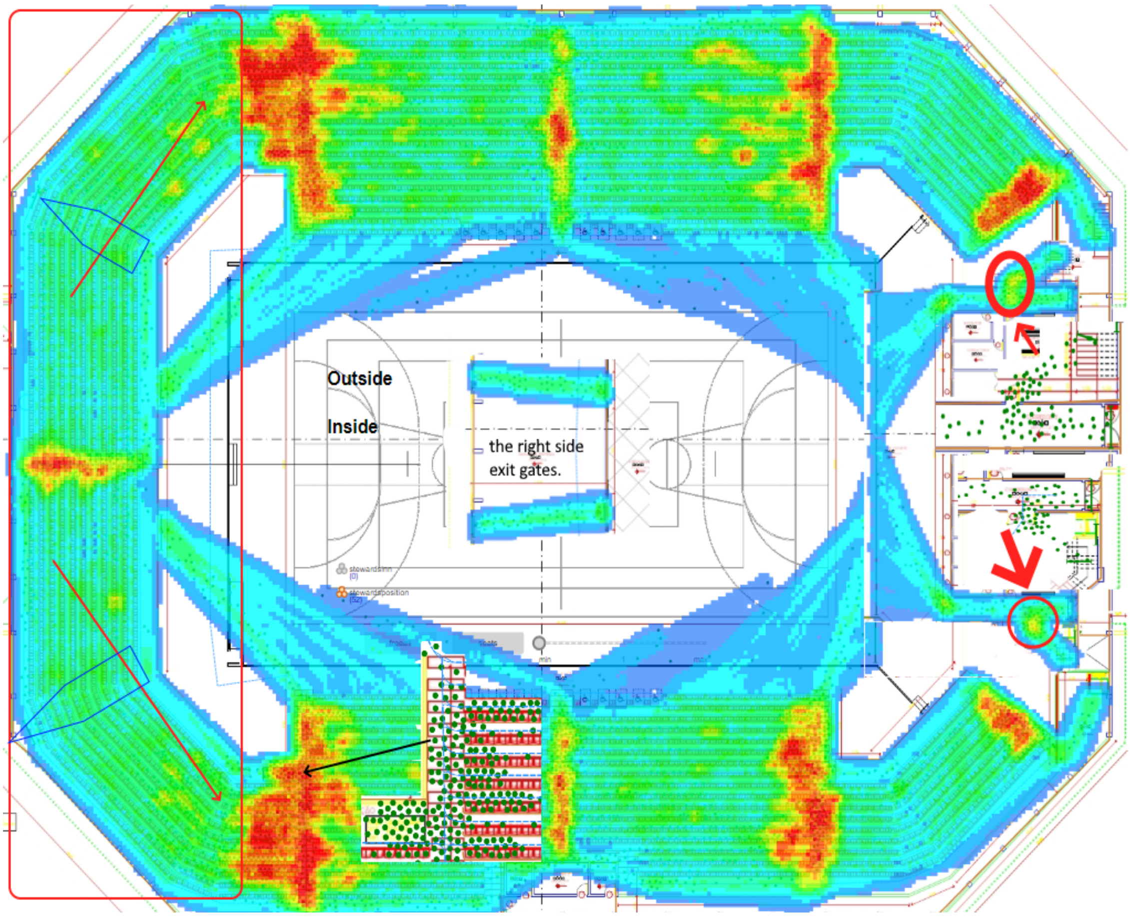


Figure 4.29: default arena setup evacuation run.

from analysing the picture, and it critical areas we can deduct:

- First, from the **center** of the image we can see that inside the arena, the path leading out, is fine its doesn't need a change.

- Second, the **right** side of the image represent the two other gates that lead outside the arena, we can see the first problem there, the problem is coming from the door path that intersect with the path leading to the exits. From looking at the intersection density we can deduct from figure 4.28, that its around three, three is not the worst scenario, but its close enough that we should look at it, to prevent bad outcomes in the future.

- Third the **seating area**, the first thing we noticed, is that almost all stairs have density problems, that shouldn't be a surprise, since these stairs are the main source of navigating in the area. The second thing we noticed, is there is two stairs (red arrows pointing at them), with the biggest density problem. that's because all the pedestrian in the red rectangle at the left of the image, is going to these stairs for an exit opportunity.

Improving this arena architecture for a better evacuation result, is what evacuation scenarios do best. For a better evacuation time, the best approach is analysing the critical density within the arena, and redirect some of the flow that's causing that problem with more exits. to make this evacuation run better, we need to fix the first problem by creating a separate door on each stair location for exit emergency only, since entering the arena have no problems. The second problem, we can create 2 more stairs with exits, in the location of the two blue shapes at the red rectangle. The reason for that is to redirect some of the flow that stressing the stairs with the biggest density.

by Implementing these changes, we can clearly see reduced density of the questioned stairs, by directing some of the pedestrian flow to the new stairs, and the evacuation time is lowered to **11.3m** from 18.9. the figure below.4.30 show the second run that implement these changes.

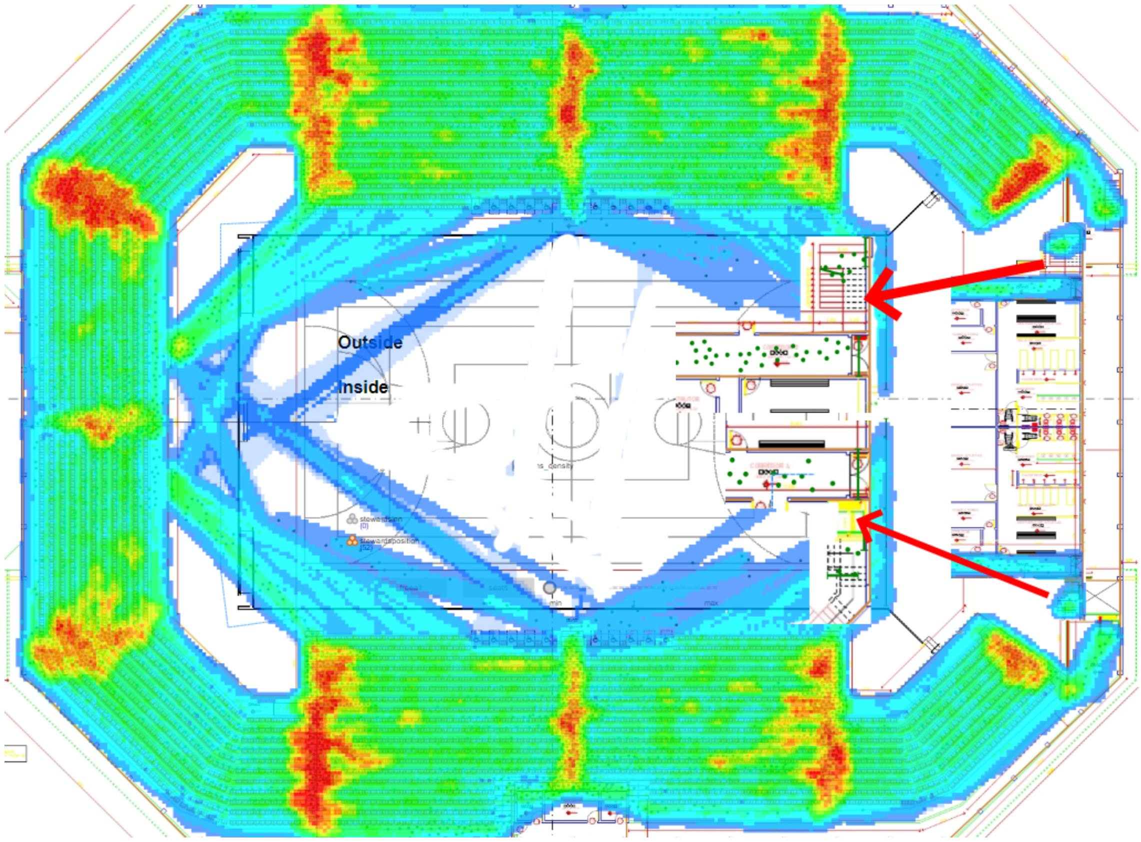


Figure 4.30: modified arena architecture evacuation run.

4.5 Conclusion

In this chapter, we discussed the simulation implementation of our SoS models. We started by picking an architectural arena drawing, then began testing its limits by capturing the emergent behaviour that multiple system brings when trying to achieve a common goal. These emergent behaviour was seen in our three main scenarios crowd entering, crowd behaviour during-after entrance, crowd exit. Each of these scenarios, represent an important part in a successful crowd management mission, and simulation was used to bring the scenarios to life. That allow us to capture key data like the average queue time, the performance of security agent, the performance of the arena in certain setup. which empower us to detect unwanted scenarios and ways to fix them.

General Conclusion

The main goal of this thesis is to propose a simulation approach In context of crowd management in sporting events SoS. The simulation solution proposed is based on the **ABS** presented in section.1.9.1.3. This approach reflect perfectly with the microscopic crowd method presented in section 2.3.2, where each system have his own characteristic within the simulation. In doing so we made a model based on [7] approach, that carries out a basic sport event, crowd management systems behaviour. Then we picked a sport venue architectural drawing. Using anylogic simulation platform we implemented the different agent behaviours, and carried out multiple simulation scenarios based on our performance targets, in order to stress out the arena architect and the systems. These results can empower the experts to further their understanding for the SoS at hand, in order to create a better SoS model following the SoS SE life cycle. The major contributions of our work are:

- Study on the state of System of systems, and crowd simulation concepts (definition, characteristic, types, modeling and simulation approaches). To the point where a basic understanding of the concepts is set, to be able to chose the best methods to simulate crowd management case of sport event.
- Design a basic SysML model in context of crowd management in sporting events SoS.
- Proposition of a solution able to simulate the SoS at hand based on the ABS approach.
- Implementation of the solution proposed, and carrying out multiple scenarios to test out the SoS systems, and the arena venue performance.

Limits and future perspectives

Working on SoS modeling in an iterative process as shown in the SoS SE life-cycle, where SoS's are never in their final forms [16]. Since simulation is a direct model analysis, this leave the room for a large amount of future improvement.

We can improve this simulation by doing it on a real life architecture drawing, with real life data (e.g. the average pedestrian rate entering the arena, average search time upon entering etc), with more details on the crowd and systems architectures and behaviour, to benefit the most of ABS approach.

Another interesting approach for the future improvement is integrating the advancement that Artificial intelligent brings in human behaviour to human based simulation like this, the other way of AI integration into simulation is using it as a way of result analysis, the amount of data we can gather from these simple scenarios if we test every possibility, is big enough to take a huge amount of time to consider a kind of automation result analysis, if we can train Artificial intelligent to understand the mission at hand the amount of time we can save is uncountable, without mentioning the ability to find new insight to the SoS behaviour that a human being may overlook.

During this project, we have encountered many struggles. Running a simulation of this scale with more than 5000 agents was a tough experience, simulation scenarios runs so slow, it hindered the gathering of results due to a lack of Strong CPU performance with a lot of cores. In addition trying to model real life scenarios in no existent arena, was also a tough mission due to the lack of familiarity with the systems and the arena.

Bibliography

[1] Mo Jamshidi. *Systems of systems engineering: principles and applications*. CRC press, 2008.

[2] Claus Ballegaard Nielsen, Peter Gorm Larsen, John Fitzgerald, Jim Woodcock, and Jan Peleska. "Systems of systems engineering: basic concepts, model-based techniques, and research directions". In: *ACM Computing Surveys (CSUR)* 48.2 (2015), pp. 9–13.

[3] Alex Gorod, Brian E White, Vernon Ireland, S Jimmy Gandhi, and Brian Sauser. *Case studies in system of systems, enterprise systems, and complex systems engineering*. CRC Press, 2014.

[4] Jim Woodcock, Peter Gorm Larsen, Juan Bicarregui, and John Fitzgerald. "Formal methods: Practice and experience". In: *ACM computing surveys (CSUR)* 41.4 (2009).

[5] William Ross, Mihaela Ulieru, and Alex Gorod. "A multi-paradigm modelling & simulation approach for system of systems engineering: A case study". In: *2014 9th International Conference on System of Systems Engineering (SOSE)*. IEEE. 2014, pp. 183–188.

[6] George Muller and Cihan Dagli. "Simulation for a coevolved system-of-systems meta-architecture". In: *2016 11th System of Systems Engineering Conference (SoSE)*. IEEE. 2016, pp. 1–6.

[7] Imane Cherfa, Nicolas Belloir, Salah Sadou, Régis Fleurquin, and Djamal Bennouar. "Systems of systems: From mission definition to architecture description". In: *Systems Engineering* 22.6 (2019).

[8] C. B. Keating. "Research foundations for system of systems engineering". In: *2005 IEEE International Conference on Systems, Man and Cybernetics*. Vol. 3. 2005, 2720–2725 Vol. 3. DOI: `10.1109/ICSMC.2005.1571561`.

[9] M. Jamshidi. "System-of-systems engineering-a definition". In: *IEEE SMC* 32 (2005).

[10] A. Sharawi, S. N. Sala-diakanda, A. Dalton, S. Quijada, N. Yousef, L. Rabelo, and J. Sepulveda. “A Distributed Simulation Approach for Modeling and Analyzing Systems of Systems”. In: *Proceedings of the 2006 Winter Simulation Conference*. 2006, pp. 1028–1035. DOI: 10.1109/WSC.2006.323191.

[11] A. Gorod, B. Sauser, and J. Boardman. “System-of-Systems Engineering Management: A Review of Modern History and a Path Forward”. In: *IEEE Systems Journal* 2.4 (2008), pp. 484–499. DOI: 10.1109/JSYST.2008.2007163.

[12] Ludvig Knöös Franzén, Sofia Schön, Athanasios Papageorgiou, Ingo Staack, Johan Ölvander, Petter Krus, Kristian Amadori, and Christopher Jouannet. “A System of Systems Approach for Search and Rescue Missions”. In: *AIAA Scitech 2020 Forum*. 2020, pp. 1–6.

[13] E. Kenneth Boulding. “General systems theory—the skeleton of science”. In: *Management science* 2.3 (1956), pp. 197–208.

[14] B. J. Berry. “Cities as systems within systems of cities”. In: *Papers of the Regional Science Association*. Vol. 13. 1. Springer. 1964, pp. 149–163.

[15] R. L. Ackoff. “Towards a system of systems concepts”. In: *Management science* 17.11 (1971), pp. 661–671.

[16] François Jacob. “The logic of living systems: a history of heredity”. In: Lane. 1974.

[17] Senate US Congress and COMMITTEE ON ARMED SERVICES. “Restructuring of the Strategic Defense Initiative (SDI) Program:US Senate”. In: University of Michigan Library. 1988.

[18] Charles Keating, Ralph Rogers, Resit Unal, David Dryer, Andres Sousa-Poza, Robert Safford, William Peterson, and Ghaith Rabadi. “System of systems engineering”. In: *Engineering Management Journal* 15.3 (2003), pp. 36–45.

[19] Mark W. Maier. “Architecting Principles for Systems-of-Systems”. In: *INCOSE International Symposium* 6.1 (July 1996). DOI: 10.1002/j.2334-5837.1996.tb02054.x. URL: http://dx.doi.org/10.1002/j.2334-5837.1996.tb02054.x.

[20] John Boardman and Brian Sauser. “System of Systems-the meaning of of”. In: *2006 IEEE/SMC International Conference on System of Systems Engineering*. IEEE. 2006.

[21] Russ Abbott. “Open at the top; open at the bottom; and continually (but slowly) evolving”. In: *2006 IEEE/SMC International Conference on System of Systems Engineering*. IEEE. 2006.

[22] Stephen C Cook. “2.3. 1 on the acquisition of systems of systems”. In: *INCOSE International Symposium*. Vol. 11. 1. Wiley Online Library. 2001.

[23] Annette J Krygiel. *Behind the wizard's curtain. an integration environment for a system of systems*. Tech. rep. Office of the Assistant Secretary of Defense Washington DC Command and …, 1999.

[24] MO Jamshidi. “System of systems engineering-New challenges for the 21st century”. In: *IEEE Aerospace and Electronic Systems Magazine* 23.5 (2008), pp. 4–19.

[25] Aaron Shenhar. “A new systems engineering taxonomy”. In: *Proceedings of the 4th International Symposium of the National Council on System Engineering, National Council on System Engineering*. Vol. 2. 1994, pp. 261–276.

[26] Vadim Kotov. *Systems of systems as communicating structures*. Vol. 119. Hewlett Packard Laboratories, 1997.

[27] William HJ Manthorpe. “The emerging joint system of systems: A systems engineering challenge and opportunity for APL”. In: *Johns Hopkins APL Technical Digest*. Vol. 17. 3. 1996, pp. 305–310.

[28] Paul G Carlock and Robert E Fenton. “System of Systems (SoS) enterprise systems engineering for information-intensive organizations”. In: *Systems engineering* 4.4 (2001), pp. 242–261.

[29] David Carney, David Fisher, and Patrick Place. *Topics in Interoperability: System-of-Systems Evolution*. Tech. rep. CMU/SEI-2005-TN-002. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2005. URL: `http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=7369`.

[30] William A Crossley. “System of systems: An introduction of Purdue University schools of engineering's signature area”. In: *Engineering Systems Symposium*. Citeseer. 2004.

[31] Georgious Despotou, Robert Alexander, and Martin Hall-May. “Key Concepts and Characteristics of Systems of Systems”. In: *DARP-HIRTS Strand* 2 (2003).

[32] Daniel Schneider and Mario Trapp. "Runtime safety models in open systems of systems". In: *2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing*. IEEE. 2009, pp. 455–460.

[33] Andrew P Sage and Christopher D Cuppan. "On the systems engineering and management of systems of systems and federations of systems". In: *Information knowledge systems management* 2.4 (2001), pp. 325–345.

[34] Yaneer Bar-Yam, Mary Ann Allison, Ron Batdorf, Hao Chen, Hòa Generazio, Harcharanjit Singh, and Steve Tucker. "The characteristics and emerging behaviors of system of systems". In: *NECSI: Complex Physical, Biological and Social Systems Project* (2004), pp. 1–16.

[35] US Department of Defense. "Systems and Software Engineering. Systems Engineering Guide for Systems of Systems". In: 2008.

[36] Judith S Dahmann and Kristen J Baldwin. "Understanding the current state of US defense systems of systems and the implications for systems engineering". In: *2008 2nd Annual IEEE Systems Conference*. IEEE. 2008.

[37] W Clifton Baldwin, Brian Sauser, and Robert Cloutier. "Simulation approaches for system of systems: Events-based versus agent based modeling". In: *Procedia Computer Science* 44.1 (2015), pp. 2–4.

[38] Jo Ann Lane. "System of systems capability to requirements engineering". In: *2014 9th International Conference on System of Systems Engineering (SOSE)*. IEEE. 2014, pp. 91–96.

[39] Huseyin Dogan, Michael J de C Henshaw, and Julian Johnson. "7.5. 1 An incremental hybridisation of heterogeneous case studies to develop an ontology for capability engineering". In: *INCOSE International Symposium*. Vol. 22. 1. Wiley Online Library. 2012, pp. 956–971.

[40] N. Noy and Deborah Mcguinness. "Ontology Development 101: A Guide to Creating Your First Ontology". In: *Knowledge Systems Laboratory* 32 (Jan. 2001).

[41] Ingo Staack, Kristian Amadori, and Christopher Jouannet. "A holistic engineering approach to aeronautical product development". In: *The Aeronautical Journal* 123.1268 (2019), pp. 1545–1560.

[42] Russell Lock and Ian Sommerville. "Modelling and analysis of socio-technical system of systems". In: *2010 15th IEEE International Conference on Engineering of Complex Computer Systems*. IEEE. 2010, pp. 224–232.

[43] U S Department of Defense. *DoDAF Architecture Framework Version 2.02*. URL: `https://dodcio.defense.gov/Library/DoD-Architecture-Framework/`.

[44] UK Ministry of Defence. *MOD Architecture Framework (MODAF)*. URL: `https://www.gov.uk/guidance/modarchitecture-framework/`.

[45] Matthew Hause, Graham Bleakley, and Aurelijus Morkevicius. "Technology update on the unified architecture framework (uaf)". In: *INSIGHT* 20.2 (2017), pp. 1145–1160.

[46] Judith Dahmann, George Rebovich, JoAnn Lane, Ralph Lowry, and Kristen Baldwin. "An implementers' view of systems engineering for systems of systems". In: *2011 IEEE International Systems Conference*. IEEE. 2011, pp. 212–217.

[47] Paul Bratley, Bennett L Fox, and Linus E Schrage. In: *A Guide to Simulation*. Springer, 1987.

[48] Robert E. Shannon. "Introduction to Simulation". In: *Proceedings of the 24th Conference on Winter Simulation*. WSC '92. Arlington, Virginia, USA: Association for Computing Machinery, 1992. ISBN: 0780307984. DOI: `10.1145/167293.167302`. URL: `https://doi.org/10.1145/167293.167302`.

[49] Ilya Grigoryev. "AnyLogic 7 in three days". In: *A quick course in simulation modeling* 2 (2015), pp. 101–140.

[50] Eric F Wolstenholme. "A current overview of system dynamics". In: *Transactions of the Institute of Measurement and Control* 11.4 (1989), pp. 171–179.

[51] R. Rodrigues. *Bass Diffusion*. accessed October 10, 2020. URL: `https://cloud.anylogic.com/model/84c616af-2575-4aca-9ace-75255d3ad12`.

[52] Peer-Olaf Siebers, Charles M Macal, Jeremy Garnett, David Buxton, and Michael Pidd. "Discrete-event simulation is dead, long live agent-based simulation!" In: *Journal of Simulation* 4.3 (2010), pp. 204–210.

[53] Lissa F Pollacia. "A survey of discrete event simulation and state-of-the-art discrete event languages". In: *ACM SIGSIM Simulation Digest* 20.3 (1989), pp. 8–25.

[54] C. Macal and Michael North. “Tutorial on agent-based modelling and simulation”. In: *J. Simulation* 4 (Sept. 2010), pp. 151–162. DOI: `10.1057/jos.2010.3`.

[55] Jonathan Ozik, Nicholson Collier, Todd Combs, Charles M Macal, and Michael North. “Repast simphony statecharts”. In: *Journal of Artificial Societies and Social Simulation* 18.3 (2015), pp. 41–42.

[56] Stuart Russell and Peter Norvig. *Artificial intelligence: Pearson new international edition: A modern approach*. Pearson Higher Ed, 2013.

[57] Larry B Rainey, Andreas Tolk, et al. *Modeling and simulation support for system of systems engineering applications*. Wiley Online Library, 2015.

[58] Leonie Hallo, Ben Payne, and Alex Gorod. “Model-Based Approach to System of Systems Engineering: Reevaluating the Role of Simulation”. In: May 2019, pp. 5–6. DOI: `10.1109/SYSOSE.2019.8753868`.

[59] Suiping Zhou, Dan Chen, Wentong Cai, Linbo Luo, Malcolm Y. H. Low, Feng Tian, Victor Tay, Darren Ong, and Benjamin Hamilton. “Crowd modeling and simulation technologies”. In: *ACM Transactions on Modeling and Computer Simulation* 20 (Oct. 2010), pp. 7–10. DOI: `10.1145/1842722.1842725`.

[60] Kiran Ijaz, Shaleeza Sohail, and Sonia Hashish. “A Survey of Latest Approaches for Crowd Simulation and Modeling Using Hybrid Techniques”. In: UKSIM ’15. USA: IEEE Computer Society, 2015, pp. 111–113. ISBN: 9781479987139.

[61] Wonho Kang and Youngnam Han. “A Simple and Realistic Pedestrian Model for Crowd Simulation and Application”. In: (Aug. 2017).

[62] Linbo Luo, Cheng Chai, Jianfeng Ma, Suiping Zhou, and Wentong Cai. “ProactiveCrowd: Modelling Proactive Steering Behaviours for Agent-Based Crowd Simulation: Modelling Proactive Steering for Agent-Based Crowd Simulation”. In: *Computer Graphics Forum* 37 (Sept. 2017). DOI: `10.1111/cgf.13303`.

[63] Craig Reynolds. “Steering Behaviors For Autonomous Characters”. In: (June 2002).

[64] Jarosław Wąs, Bartłomiej Gudowski, and Pawel Matuszyk. “Social Distances Model of Pedestrian Dynamics”. In: vol. 4173. Sept. 2006, pp. 492–501. ISBN: 978-3-540-40929-8. DOI: `10.1007/11861201_57`.

[65] Dirk Helbing, Illés Farkas, and Tamás Vicsek. "Simulating Dynamic Features of Escape Panic". In: *Nature* 407 (Sept. 2000). DOI: 10.1038/35035023.

[66] E Kirik, A Malyshev, T Vitova, E Popel, and E Kharlamov. "Pedestrian movement simulation for stadiums design". In: *IOP Conf Ser Mater Sci Eng*. Vol. 456. 2018, p. 012074.

[67] Qianwen Chao, Jingjing Shen, and Xiaogang Jin. "Video-based personalized traffic learning". In: *Graphical Models* 75.6 (2013), pp. 305–317.

[68] Yoshikazu Minegishi and Naohiro Takeichi. "Design guidelines for crowd evacuation in a stadium for controlling evacuee accumulation and sequencing". In: *Japan Architectural Review* 1.4 (2018), pp. 1–5.

[69] *Protest in front of Police Station*. accessed October 25, 2020. URL: https://www.gta5-mods.com/maps/protest-in-front-of-police-station.

[70] M.-L Xu, Hao Jiang, and Xiaogang Jin. "Crowd Simulation and Its Applications: Recent Advances". In: *Journal of Computer Science and Technology* 29 (Sept. 2014), pp. 800–807. DOI: 10.1007/s11390-014-1469-y.

[71] Franziska Klügl and Ana Bazzan. "Agent-Based Modeling and Simulation". In: *AI Magazine* 33 (Sept. 2012), pp. 29–40. DOI: 10.1609/aimag.v33i3.2425.

[72] Sanford Friedenthal, Alan Moore, and Rick Steiner. "OMG systems modeling language (OMG SysML) tutorial". In: *INCOSE Intl. Symp*. Vol. 9. 2006, pp. 3–67.

[73] Lamartine Camilo. *Project gym*. [Online; accessed Oct 10, 2020]. URL: https://www.bibliocad.com/en/library/project-gym_91553/.