

# Agent-Based Modeling of Systems of Systems

Jean-Baptiste Soyez, Gildas Morvan, Rochdi Merzouki and Daniel Dupont

**Abstract**—This paper deals with the generic modeling of system of systems (SoS) using agent based modeling (ABM). SoS are large scale systems including numerous—possibly heterogenous—interacting component systems (CS) evolving a dynamic environment. The goal of this article is to provide a generic formalism which allows to represent and control the whole complexity of a SoS using agent-based simulations. In particular organizational aspects of SoS are managed with Agent-Group-Role (AGR) concepts. Functional aspects, guiding SoS to accomplish their global goals, are handled via a functional specification. Multi-level aspects are modelled with the Influence Reaction Model for Multi-Level Simulation (IRM4MLS) agent-based meta-model. Models generated using this formalism encompass static and dynamic aspects of SoS. They consider reorganization of SoS caused by changes of goals or sub-system capacity. All these elements are illustrated in this article using a SoS case study of Intelligent Automated Vehicles (IAV) initiated by the InTrade european project to automate port container logistic.

**Keywords**—Agent-based modeling, Systems of Systems, complex system engineering

## I. INTRODUCTION

THIS article is concerned with the design and control of artificial complex systems, i.e., large scale systems composed of numerous communicating components [1]. Complex system engineering methods generally take into account the hierarchical organization of such systems. This is particularly true for systems that are themselves composed of complex systems like federations of systems (FoS). FoS can be differentiated from conventional systems by their high degree of autonomy, heterogeneity and spatial dispersion of their components [2], [3]. Systems of systems (SoS) can be considered as particular type of FoS: component systems (CS) operate independently and are driven by their local goals, nonetheless they have to cooperate in order to fulfil global goals potentially contradicting local goals [4]. SoS is a subject of growing interest in many research domains: such as biology, social sciences or military simulations [5]. Indeed, it is a promising concept that should be able to catch the whole complexity of such real systems.

Our work takes place in the frame of the european project InTrade [6]. It deals with automated loading and unloading container operations in large-scale ports of northwest Europe (Dublin, Oostend, Le Havre and Rouen), using Intelligent Autonomous Vehicles (IAV) controlled by a central operator. IAVs are semi autonomous vehicles, in charge of container transport tasks. IAVs can react to changes in environment or goal affectation, they can also reason about their internal dynamic and communicate with human operators or other IAVs. The container terminal system consists of several IAVs, cranes, operators and boats. It can be seen as a SoS with

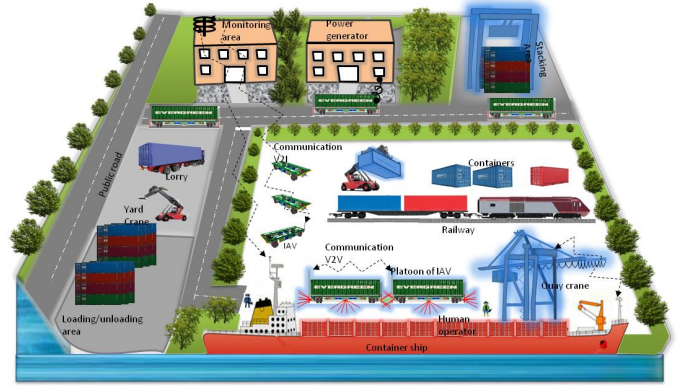


Fig. 1. Illustration of InTrade SoS complexity.

numerous heterogeneous elements and several organizations present at different levels and communicating with each other. To illustrate the proposed SoS modeling and simulating approach, a single practical case, taken from the InTrade project will be considered throughout this article.



Fig. 2. Photo of a IAV in a port container.

To some extent, IAV technology is very similar to Autonomous Guided Vehicle (AGV) technology. It exists a rich litterature interested on issues generated by the control of numerous AGVs [7]. But, approaches considering IAVs like AGVs with just more intelligence and independance fail to capture the real complexity of such systems. A SoS model seems adapted to consider organizational and functional

aspects unfolded on several levels of an IAVs fleet. That is why this article example modeling concerns a IAVs fleet.

Our goal is to create a modeling tool for SoS in order to engineer and operate such systems. To do so, we need a formalism able to define a SoS in terms of behaviour of its CS. However, it does not exist any generic, rigorous and complete enough modeling method for SoS. In this article we introduce a formalism able to capture the dynamic and static aspects of SoS. This formalism allows to divide the complexity of a system by splitting it into a set of levels and organizations that represent the system at different scales and structured by groups of CS.

#### A. Related Works

Because SoS have been introduced in various domains, dedicated tools have been proposed to model and operate them. Nevertheless, until recently, SoS remained a theoretical concept with no generic simulation formalism. Several modeling approaches of SoS are non generic and focus on domain-eated issues: [8] proposed a classical modeling framework, defining interfaces between existing CS. [4] used the agent-based modeling (ABM) paradigm to represent situated CS constrained by their place in a discretized environment (locality). [9] introduced a model representing the effects of climate policy using also an ABM, nested in a Putman two level game. [10] focus on the CS connections represented by N square matrix.

[11] also developed a formalism to represent SoS models, using metrics and attributes. But this representation focuses on data more than on SoS entities. The proposed tool is designed to evaluate and predict the performances of a SoS rather than controlling one.

[12] proposed a formalism based on hypergraphs to represent directed SoS but he did not provide generic mechanisms to ensure that the constitutive SoS characteristics are preserved during their evolution (adding or deleting CS, change of general goal).

[13] proposed a generic modeling method for SoS using ABM. Thus, SoS are endowed with a structure that constraint interactions between agents (communication network or topological environment).

[14] proposed an ABM approach where CS propose services that define roles into communities. It focuses on SoS reconfiguration, based on evaluations of the chance to reach a given goal with a good performance. The reconfiguration process is based on rules adapted to graph grammar. Models only have two levels: the whole SoS and its CS. One weakness of this approach is the lack of functional specification for SoS goals which forces the model to try several configuration before finding the optimal one. This process works by enhancing the current SoS configuration but there is no guarantee that it will converge.

It seems that the most generic attempts to model SoS rely on ABM. Indeed, as it is shown later, there are many similarities (and some differences though) between ABM and

SoS, such as the autonomy property of agents and CS.

#### B. Contributions

The contributions of this article, are of two orders: 1) select and enhance a SoS definition generic and precise ; 2) develop a generic SoS formalism. The SoS definition is generic enough to be accepted by the SoS community and concrete and precise enough to be operational in a modeling and simulation context. The proposed formalism allows to easily apprehend the system structure and organizations. Models generated using this formalism fit to represent IAVs fleet of the InTrade project which are SoS directed by a global goal.

Resulting models should contain simple independent CS forming organizations able to accomplish some tasks that single CS cannot accomplish which is possible in MAS. To make the model easy to apprehend and to divide its computational complexity we will cut the SoS by scales or independent aspects using the multi-level agent-based modeling metamodel IRM4MLS [15].

During SoS reconfiguration, caused by global goal or environment change, in [13] and [14] models, new SoS configuration are established applying rules contained in CS or in obligations out the system. There is no independent representation of the global goal to decompose it and establish various plans to accomplish it. Also there is no independent organizational representation of groups of CS to choose the best SoS structure to complete previous plans. Organizational and functional definitions allow to produce such representations. So, we take inspiration from the *Moise*<sup>+</sup> model which contains these descriptions and a deontic one which links the previous two in scope to let the maximum liberty to agents while constraining them to accomplish a global goal [16] [17]. *Moise*<sup>+</sup> is based on the AGR model and reuse its main concepts.

This formalism allows the modeling and simulation of SoS in order to control, diagnose and assess performances of real systems with the same characteristics. Some approaches highlight the evolutionary aspect of SoS, but only few consider SoS in a dynamic way including SoS reconfiguration. Thus, this article proposes to integrate control mechanisms in SoS to preserve their characteristics during their evolution like changes in their environment or in the global goal definition.

[11] highlights a characteristic resulting from the 5 SoS fundamental characteristics. It is the fact that all CS are not needed to fulfil the global goal. That does not mean that SoS have to possess redundant CS to overcome the failure of any CS during its mission accomplishment. It means that in case of failure, SoS have to be able to reallocate the faulty mission to an available CS or one created for this purpose. It induces less the obligation of the feasibility of such a reconfiguration than the presence of reconfiguration mechanisms and the insurance that if the reconfiguration is possible it will be completed.

Therefore, according to evolutionary aspect of SoS, a formalism to represent SoS should possess two mechanisms: any CS can detect that it is not able to fulfil its missions and, if it serves the global goal, transmits this information to the CS which has authority on it and then can do the same. The other mechanism concern these failing CS by the reconfiguration of the SoS with the reallocation of missions and/or the organizations change. These mechanisms and concepts are exposed later in this article.

### C. Plan

This article is organized as follows. Section II defines what is a System of Systems SoS, what are the different types of SoS and gives a graphical representation for SoS. Section III introduces the concept of Multi-Agent Systems (MAS) and presents ABM tools, adapted to represent complex systems like SoS. Then the Multi-level Agent-based Model (MAM) IRM4MLS is introduced. In section IV, we give static and dynamic elements necessary to describe Multi-Agent SoS (MASoS) from an organizational point of view and represent its multi-level structure. After that, we show how to link the previous elements to achieve system-level goals to allow the evolving execution of a MASoS. Finally, we give some conclusion in section V.

## II. SYSTEM OF SYSTEMS (SoS)

In this section we present the organizational concept of SoS and show some applications. Then, we explain the need of a solid formalism to represent SoS. After that, we expose the different families of formalisms apt to represent SoS. Finally, a graphical notation is introduced to represent SoS in the rest of this article.

### A. SoS Definition

Many definitions of SoS have been proposed in the literature. This article leans on Sage and Coppan's definition [2], [18], inspired by Maier's one [19], that can be summarised as follows:

A SoS is a set of autonomous Component Systems (CS) endowed with a global goal. CS can be heterogenous. They manage their own resources and under CS in an independent way and can coexist and cooperate to accomplish a mission that a single CS cannot realize. CS are geographically distributed without any physical link. A SoS has to be robust and adaptive: its environment, goal or structure (by adding/deleting CS) can evolve without modifying its capacity to fulfil its global goal.

An **under CS** of a given CS is one of its CS, in a lower level, forming it. For a given CS, its **super CS** is the CS, in a higher level, to which it belongs.

This definition respects the five following characteristics illustrated with an example from the InTrade project.

- 1) **Operational independence:** Each CS possesses its own resources necessary to accomplish its mission. For example, an IAV has its own stock of energy and its personal status diagnosed online.
- 2) **Managerial independence:** Once a mission is attributed to a CS, it manages itself and its under CS, on its own, to accomplish it. For example, a fleet of IAVs attached to a quay decides on its own how to organize its IAVs to accomplish its allocated missions.
- 3) **Geographic distribution:** CS can exchange information but there is no physical exchange of energy. For example, when two IAVs are attached to form a platoon to be able to carry 40 feet containers, they lost their geographic distribution and cease to be considered as CS.
- 4) **Emergent behaviour:** The global goal of a SoS can only be reached by the joint action of more than one CS. For example, if one IAV is sufficient to reach the SoS goal to perform all transport tasks in the port, the SoS has no reason to exist.
- 5) **Evolutionary development:** A SoS can adapt itself to the addition or deletion of CS or evolution of the environment or change of the global goal to stay able to reach it by the actions of CS. For example, when an IAV endowed with a mission lost its capacity to fulfil it, it informs its super CS. This super CS has means to overtake this loss of capacity by allocating the mission to another CS agent with intact capacities.

Similar characterizations like Bordman and Sausser's one [20] exist. This characterization provides the following characteristics differentiating an SoS from monolithic systems: 1) *autonomy* - CS exercise autonomy in order to fulfil the SoS global goal; 2) *belonging* - CS choose to belong to the SoS based on cost/benefit basis; 3) *connectivity* - CS provide dynamic connectivity to enhance overall SoS capability; 4) *diversity* - a product characteristic of an SoS not available from single systems; 5) *emergence* - capability is provided that was not originally foreseen during development, leading to early detection and elimination of undesirable behaviors. [21] establishes a review of SoS engineering (SoSE) methods and uses previous SoS characterization to create a SoSE management framework.

[22] gave a SoS definition recognized in the SoS community:

"Systems of systems are large-scale integrated systems that are heterogenous and independently operable on their own, but are networked together for a common goal".

Other definitions seem to be too vague to be exploited here. In many cases [19], [23] they only describe information systems where interactions between systems are limited to information exchanges. This is too restrictive to represent systems in strong interaction [24], e.g., two robots joining each other to carry an object too heavy for only one robot.

A more comprehensive definition, in relation with the MAS paradigm, is given in section IV.

### B. Types of SoS

The SoS concept has many applications in different domains such as military and maritime simulation, biology, climatology and sociology [5], [25], [26], [27]. In sociology, the gregarious behavior of humans can lead to naturally emergent organizations or societies, considered as SoS. In a military context, SoS are sometimes the only available approach to represent influences between organizations at different levels, with their own area of intervention (land, air, sea) in large-scale operations.

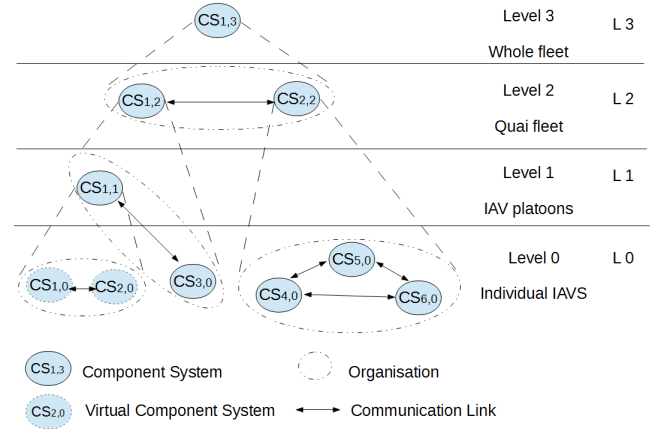
In its guide to engineer SoS, the US Department of Defense (DoD) describes four types of SoS, depending on their degree of centralization [28]. They can be: 1) **Virtual**. The SoS lacks a central management and a centrally agreed-upon purpose. 2) **Collaborative**. CS within the SoS interact more or less voluntarily to fulfil agreed-upon central purposes. 3) **Acknowledged**. The SoS has acknowledged objectives, a designated manager, and resources, while CS retain their independent ownership, objectives, funding, development, and sustainment approaches. 4) **Directed**. The SoS is built to fulfil specific purposes. CS operate independently, but their operational mode is subordinated to central management purposes.

Virtual SoS are generally used to study the evolution of natural complex systems such as social or biological systems [5]. And they put forth the hypothesis that purely collaborative SoS emerge only if the environment influences the CS, posing a new problem which requires the SoS existence. These cases do not match the subject of this article, which focuses on engineering systems endowed with a hierarchy and nested level structure. Therefore, this article deals mainly with directed or acknowledged SoS.

### C. Graphical Representation and Particular CS

In this section a graphical representation of the real entities of a SoS is introduced. For more details on CS notation see section IV. It permits to easily see how a CS is constituted by its lower-level CS and distinguishes between inclusion and communication links. Let consider an example from the Intrade project. A whole fleet of IAVs,  $CS_{1,3}$ , is composed of quay fleet from quay 1,  $CS_{1,2}$ , and quay 2,  $CS_{2,2}$ . The  $CS_{1,2}$  fleet is composed of a  $CS_{1,0}$  platoon (including  $CS_{1,0}$  and  $CS_{2,0}$  IAVs) and an IAV  $CS_{3,0}$ .  $CS_{2,2}$  fleet is composed of three IAVs:  $CS_{4,0}$ ,  $CS_{5,0}$  and  $CS_{6,0}$ .

Some CS or set of CS can temporarily violate some of the SoS characteristics explained further. That does not mean the end of the SoS, but that CS may have to be deleted, at least temporarily, to make the SoS coherent with these characteristics. We call this entities **virtual component system**. For example  $CS_{1,0}$  and  $CS_{2,0}$  are IAVs physically



- 3) which is driven by a set of tendencies (under the form of individual objectives or a satisfaction or survival function that it tries to optimize),
- 4) which possesses its own resources,
- 5) which is able to perceive (in a limited way) its environment,
- 6) which disposes only of a partial representation (eventually none) of this environment,
- 7) which possesses competencies and offers services,
- 8) which can eventually reproduce itself,
- 9) whose behaviour tends to satisfy its objectives with its resources and its competencies and according to its perceptions, representations and communications it receives).

We call Multi-agent System (or MAS), a system composed by following elements:

- 1) An environment  $E$ , i.e, a topological space, that generally embodies a metric,
- 2) A set of objects  $O$ . These objects are situated: at a given time, it is possible to associate to each object a position in  $E$ . Objects can be perceived, created and modified by agents,
- 3) A set of agents  $A$ , which are particular objects ( $A \subseteq O$ ), that represent the active entities of the system,
- 4) A set of relations  $R$  that link objects (and then agents) between them,
- 5) A set of operators  $Op$  allowing agents of  $A$  to perceive, produce, consume, transform and manipulate objects of  $O$ ,
- 6) Some operators dealing with the representation of the application of previous operators and the world reactions of modifications, which can be called universe laws.

### B. Agent-Based Modeling of Complex Systems

MAS are widely used to simulate interactions between complex, autonomous entities acting in parallel. This kind of entities are easily found in complex systems like SoS.

[30] presents a MAS model to represent systems with the High Level Architecture (HLA). HLA is a metamodel that allows to make coexist and coordinate different simulations in the same platform. Even if HLA is more concerned with implementation than our approach, the proposed model deals with coexisting independent entities forming different levels of organization.

[31] also presents a MAS model, PADAWAN, whose specificity is to include agents and their nested environment in other agents. This way of seeing a system with aggregate organization at a level creating individuals at a higher level is common with SoS.

[32] proposes a bibliography to list the existing multi-level agent-based modeling approaches. This article identifies the different theoretical issues, meta-models, platforms and applications related to that domain.

First of all a SoS should be considered as an organizational concept, and therefore it can be compared to MAS organizational models such as AGR (Agent, Group, Role) [33], [34]. This formalism organizes a system into groups depending in which agents play roles.

[35] uses a hierarchical multi-agent metamodel (CRIO) based on the concept of holon [36]. A holon represents both an individual entity and a set of organizations composed with entities with the structure. CRIO contains also an organizational decomposition aspect. In his thesis he presents a complete methodology to model and implement a complex system. He used ASPECS, an engineering software process which describes all necessary steps to conceive a software. It is based on CRIO.

[16], [17] studied organization oriented MAS and to insure that the system will complete its global goal while guaranteeing agents autonomy in terms of behaviour and creation of organizations, they use deontic specification to link groups and allocation of missions in their model: *Moise*<sup>+</sup>. In a way this is similar to our approach to SoS. But in our approach, the existence of groups is strongly constrained by the necessity to fulfil the global goal of the system.

[37] created a metamodel mixing holonic organizational aspects of ASPECS and deontic specification (renamed norms) issued from *Moise*<sup>+</sup> to instantiate holonic MAS (HMAS). The created HMAS are designed to reach a global goal with coordinating independent entities respecting the system integrity during its changes. We can inspire ourselves of this approach if we consider the common points and the difference between SoS and holonic systems.

### C. Influence Reaction Model for Multi-Level Simulation (IRM4MLS)

A SoS is a specific case of complex system whose elements are interacting at different levels. Such multi-level aspects of SoS can be represented using IRM4MLS. Another interest to represent a SoS with a multi-level model is to divide the complexity of the system and make it easier to apprehend and compute by separating scales of organizations or independent aspects of a SoS.

IRM4MLS is a MAM. This meta-model allows to represent multiple entities in interaction situated in different levels. Here a level is considered in a general way and not obligatory as a scale. Moreover interactions between entities in different levels and their results are represented without any bias thanks to the influence reaction model used in IRM4MLS [38], [39].

A IRM4MLS model is characterized by a set of levels,  $L$ , and relations between levels. Two types of relations are considered in IRM4MLS: influence (agents in a level  $l$  are able to produce influences in a level  $l' \neq l$ ) and perception (agents in a level  $l$  are able to perceive the state of a level  $l' \neq l$ ). These relations are respectively formalized by two digraphs,  $\langle L, E_I \rangle$  and  $\langle L, E_P \rangle$  where  $E_I$  and  $E_P$  are sets of edges, i.e., ordered pairs of elements of  $L$ . The dynamic set

of agents at time  $t$  is denoted  $A(t)$ .  $\forall l \in L$ , the set of agents in  $l$  at  $t$  is  $A_l(t) \in A(t)$ .

Let  $\delta(t) \in \Delta$  be the dynamic state of the system at time  $t$ :  $\delta(t) = \langle \sigma(t), \gamma(t) \rangle$ , where  $\sigma(t) \in \Sigma$  is the set of environmental characteristics and  $\gamma(t) \in \Gamma$  the set of influences representing system dynamics. The state of a level  $l$  at time  $t$  is noted:  $\delta_l(t) = \langle \sigma_l(t), \gamma_l(t) \rangle$ , where  $\sigma_l(t) \in \Sigma_l$  and  $\gamma_l(t) \in \Gamma_l$  are respectively set of environmental characteristics and influences representing dynamics, both in  $l$ .

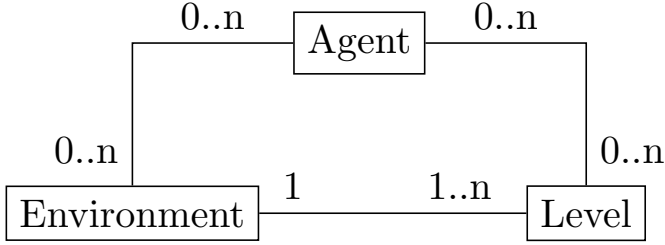


Fig. 4. Central concepts of IRM4MLS (cardinalities are specified with the UML way)

The state of an agent  $a \in A$  is characterized by :

- its physical state  $\phi_a \in \Phi_a$  with  $\Phi_a \subset \Sigma_a$  (e.g, its position)
- its internal state  $s_a \in S_a$  (e.g., its beliefs)

An agent acts in a level if a subset of its physical state belongs the state of this level. An agent can act in multiple levels at the same time. Environment is also a top-class abstraction. It can be viewed as an agent with no internal state that produces influences, representing the natural dynamic of environmental properties, in the level. The scheduling of each level is independent: models with different temporalities can be simulated without temporal bias. On the another hand, it permits to execute only the relevant processes during a time-step. A major application of IRM4MLS is to allow microscopic agents (members) to aggregate and form-up lower granularity agents (organizations). It can be useful to create multiple levels at a same scale to represent different domain parts of the same phenomenon. In the rest of this article we consider that two levels are at the same scale if they have the same spatial and temporal extents.

If levels form a hierarchy of nested scales, they can be figured by a hierarchical graph presented in [40], which indicates the level imbrication. A simple edge represents an inclusion link between two levels. For example, an  $(l_1, l_2)$  edge signifies that  $l_2$  has higher spatial or temporal extents than  $l_1$ . Then the agents situated in  $l_1$  can be aggregated and the resulting aggregate can be instantiated in  $l_2$ . A pair of symmetric edges means there is a complementarity link between two levels. For example, the  $(l_1, l_3)$  and  $(l_3, l_1)$  pair of edges mean that  $l_1$  and  $l_3$  are at the same scale. Thus agents simultaneously present and activated in  $l_1$  and  $l_3$  can figure different aspects of the same entities.

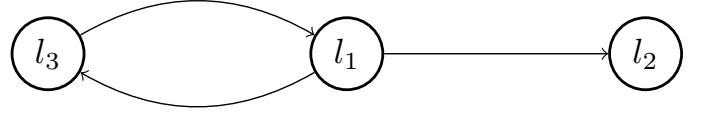


Fig. 5. Example of hierarchical graph.

#### IV. AGENT FORMALISM TO REPRESENT SoS

MAS definition lacks concepts such as an organizational and a hierarchical definition to define which CS integrates and/or allocates missions to another CS, to be able to represent a SoS. Also, the notion of global goal and missions are expressed without taking in account SoS multi-level structure. This makes harder the SoS to reason about them to reorganize itself in order to fulfil them. These concepts are crucial because they are needed to express the characteristics of managerial independence, evolutionary development, cooperation and coexistence.

In this section, we give all static (1) and dynamic (2) elements necessary to represent a SoS and that are not basically included in all MAS: 1) an organizational description of entities and a multi-level environment and 2) two mechanisms to conduct SoS initialization and capacity evolution. Then, we give some elements to prove that multi-agent models, created using our formalism, respect SoS fundamental characteristics.

##### A. Static Aspects

SoS is an organizational concept in which CS are entities that can be individuals or aggregations of CS in lower level(s). A good way to represent these groups and provide the super CS a way to reason about the mission distribution to reach its allocated goal is to use the Agent-Group-Role model (AGR) [41], [42].

From the AGR point of view, an agent is an entity playing a set of roles in several groups. An agent is characterized by its roles, knowledge and capacities. A *capacity* is the abstract description of a knowledge. It regroups the means that allow to accomplish a task. This represents a logical functionality for providers (owners) or applicant (users) entities. In a more prosaic way, an agent or group capacity is the possibility for this entity to accomplish a goal, i.e., reaching a target level state, by the means of its actions, in a precise given level state.

An *agent role* is a concrete instance of a role. It describes the behaviour in context defined by a group. It confers a status and means in this group to interact with other agents playing roles in that group. Before the initialisation of the SoS, groups are just definitions and a same set of agents can instantiate several groups sequentially or instantaneously. These groups possess capacities determined by the roles played by agents and their individual capacities.

in function of roles played by its agents and their capacities.

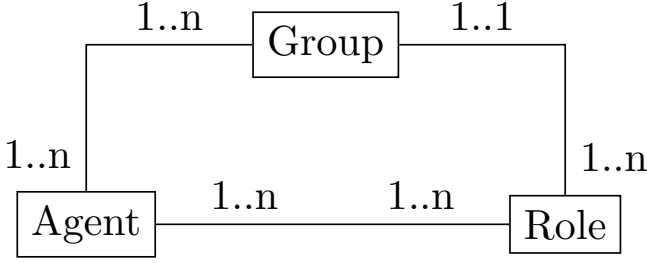


Fig. 6. Central concepts of AGR (cardinalities are specified with the UML way)

1) *CS Modeling*: To designate in a unique way and without any ambiguity a CS in a SoS we can use the following formalism:  $CS_{n,l}$  designates the  $n^{th}$  CS and its representing agent in the level  $l$ . Here is a definition inspired by Gaud. A CS,  $CS_n$  of a level  $l$ , can be noted as follows:

$CS_{n,l} = \langle CS_{n',l+1}, R_{n,l}, CS_{n'',l-1}, OP, \Psi, \Lambda \rangle$ , with :

- $CS_{n',l+1}$ : a super CS which is directly composed of  $CS_{n,l}$  and possibly other under CS. This CS has to be in a level higher than  $l$ .
- $R_{n,l}$ : the set of roles played by  $CS_{n,l}$ ,  $R_{n,l} = 2^{Roles(O_{n,l})}$  with  $O_{n,l}$  the set of groups in which  $CS_{n,l}$  plays at least one role.
- $CS_{n'',l-1}$ : the set of under CS constituting the super CS:  $CS_{n,l}$ . These CS have to be in level(s) lower than  $l$ .
- $OP$ : the set of groups taking part in life and functioning of  $CS_{n,l}$  CS and contributing to accomplish the objectives linked to the  $R_{n,l}$  roles.
- $\Psi : CS_{n'',l-1} \rightarrow 2^{Roles(OP)}$ : A function associating an under CS member to the set of roles that it plays in groups defined in  $CS_{n,l}$ , such as  $\forall csi \in CS_{n'',l-1}, \Psi(csi) \neq \emptyset$ . The role function provides the set of all roles defined in  $OP$  groups.
- $\Lambda : \Sigma_l * \Sigma_l \rightarrow \{0,1\}$ : A capacity function indicating if a goal state in  $l$  is reachable, by the action of  $CS_{n,l}$ , knowing the actual state of  $l$ . This function depends on the evolving capacities of  $CS_{n,l}$  and also on the capacities offered by  $OP$  to  $CS_{n,l}$  when  $CS_{n,l}$  is not an elementary CS.

In the case of SoS engineering there is a hierarchy of levels of increasing abstraction, approximation and scale. These levels can be represented by a hierarchical graph, as shown above, which indicates the level imbrication. When a level  $l_i$  is directly included in a level  $l_j$ , CS in  $l_i$  compose higher scale CS in  $l_j$ . A CS can directly belong to only one CS of a higher level, therefore, here, a level can be directly included in at most one level. When a CS  $CS_{i',j'}$  is an under CS of  $CS_{i,j}$  it is noted  $CS_{i',j'} \in CS_{i,j}$ .

2) *Group of CS Modeling*: To decide how to form super CS from organization of under CS and how CS are organized to reach goals it is necessary to endows our model with set

of group specification defined as follows:

$gs =_{def} \langle R, L_R, \mathcal{L}, C^{intra}, C^{inter}, nr, nc \rangle$ , with

- $R$ : the set of non abstract roles playable by agents in groups created from  $gs$ .
- $L_R : R_{gs} \rightarrow L$ : a function indicating the level of an agent which can play a given role in  $gs$ .
- $\mathcal{L}$ : indicates the level(s) of the group created from  $gs$ . if  $\mathcal{L}$  contains only one level  $l$  and  $l$  is higher than any level of  $L_R(R)$  then the group defined by  $gs$  has to be instantiated by a CS agent situated in  $l$ . CS agents playing role(s) in  $R$  are the under CS of this new CS agent.
- $C^{intra}$ : a compatibility function which indicates if two roles are compatible in the same group. By default, two roles are not compatible.  $\rho_a \Delta \rho_b$  says that agents playing role  $\rho_{ho_a}$  are authorized to play role  $\rho_{ho_b}$ . This relation is reflexive and transitive.
- $C^{inter}$ : a compatibility function which indicates if two roles are compatible in two different groups. It possesses the same formalism than  $C^{intra}$ .
- $nr : R_{gs} \rightarrow N * N$ : a function which specifies the number (minimum, maximum) of roles which has to be played in a group, ex.,  $nr_{gs}(carrier) = (1, 3)$  means that groups emanating from  $gs$  have to possess at least one and at most 3 agents playing the *carrier* role.
- $nc : R_{gs} * C \rightarrow N * N$ : a function which specifies for a given role and capacity the quantity (minimum, maximum) of this capacity which has to be available for each agent playing this role and for the whole population of agents playing this role.

Here a capacity should be expressed in a functional way and has to be quantified, i.e., not expressed in terms of goal states but in terms of concrete results of agent actions.

$nc_{gs}(carrier, carryOneContainer/hour) = (1, \infty, 3, \infty)$  means that an agent playing the role carrier in a group emanating from  $gs$  has to possess at least one unit of the *\CarryOneContainer/hour* capacity and the whole population of agents playing this role in the group has to possess at least three cumulative units of the *\CarryOneContainer/hour* capacity with in both cases no maximum limit for this quantified capacity.

3) *Multi-level environment and goal decomposition*: IRM4MLS provides a support to define global goals. A global goal is a world state to be reached by the system, which can be expressed as a description of the system and its environment variables. A global goal can be divided into a set of missions. This cut out can be functional, geographic or other. Because the modeling of SoS include multi-level representation, this cut out can be done according to levels. Thus missions of a level can be expressed as states of this level, i.e., the states of CS and environment of this level. Moreover the global goal of a SoS can be expressed as a goal state of the highest level  $L_H$ . It is used to define the global goal of the SoS :  $Gg = L_H(t') = \langle L_H(t'), L_H(t') \rangle$ .  $t'$  can

be considered as a time limit for the SoS to reach  $Gg$ .

Because it is the most abstract and approximate, the wished state  $L_H(t')$  of the highest level  $L_H$  can be divided into a set of states in every lower level corresponding to local missions which describe more in details the accomplishment of the global goal. For every inclusion link,  $(l_i, l_j)$  with  $l_i, l_j \in L$ , in the SoS hierarchical graph, there is a function  $F_{i,j}$ , which indicates how a goal state in  $l_j$  can be translated into several goal states in  $l_i$ . Each of these translated goals can correspond to a given mission allocated to CS in  $l_i$  that allows to reach the goal of  $l_j$ . Because for CS in a given level, there are several ways to fulfil a goal expressed in a higher level.

### B. Illustration of Static Aspects

To illustrate the static aspects we model the SoS presented in section II with our formalism.

$L = \{l_o, l_1, l_2, l_3\}$  with

- $l_3 = l_H = \text{levelOfWholeFleet}$ ,
- $l_2 = \text{levelOfQuayFleet}$ ,
- $l_1 = \text{levelOfIAVPlatoon}$  and
- $l_0 = \text{levelOfIndividualsIAVs}$ .

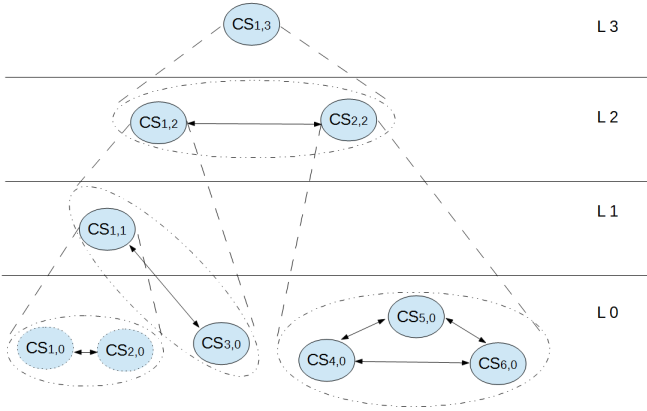


Fig. 7. Example of a SoS.

All CS are represented by CS agents.  $\mathcal{A} = \{CS_{1,3}, CS_{1,2}, CS_{2,2}, CS_{1,1}, CS_{1,0}, CS_{2,0}, CS_{3,0}, CS_{4,0}, CS_{5,0}, CS_{6,0}\}$

Here we only represent in detail  $CS_{2,2}$  and its under CS  $CS_{4,0}$ ,  $CS_{5,0}$  and  $CS_{6,0}$ .

$$CS_{2,2} = \langle CS_{1,3}, \{quayTransporter, quayTransmitter\}, \{CS_{4,0}, CS_{5,0}, CS_{6,0}\}, \{g_1\}, \psi_{2,2}, \lambda_{2,2} \rangle$$

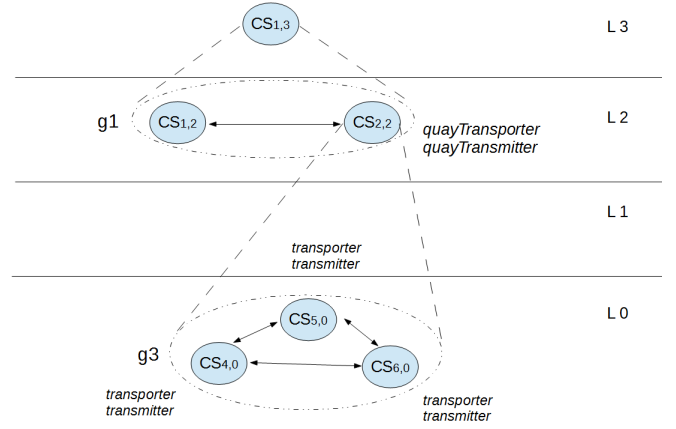


Fig. 8.  $CS_{2,2}$ , its under CS and their roles in their group.

$CS_{2,2}$  is an under CS of  $CS_{1,3}$ , the whole fleet of IAVs in the port. It plays the roles *quayTransporter* and *quayTransmitter* which consist in managing communication and container transport of a fleet of individual IAVs attached to a quay. Its under CS are individual IAVs,  $CS_{4,0}$ ,  $CS_{5,0}$  and  $CS_{6,0}$ . Here  $CS_{1,3}$  is constituted of only one group  $g_1$  built of  $CS_{2,2}$  and  $CS_{2,2}$ .  $\psi_{2,2}$  indicates the roles played by  $CS_{4,0}$ ,  $CS_{5,0}$  and  $CS_{6,0}$  in  $CS_{2,2}$ .  $\lambda_{2,2}$  is the function indicating if  $CS_{2,2}$  can reach its goals in its  $l_2$  level. This function takes in account  $CS_{2,2}$  capacities and the individual capacities of its IAVs,  $CS_{4,0}$ ,  $CS_{5,0}$  and  $CS_{6,0}$ .

The group specification of  $g_1$  can be :

$$g_1 = \langle \{quayTransporter, quayTransmitter\}, \{l_{quayTransporter} = \{l_2\}, l_{quayTransmitter} = \{l_2\}\}, \{l_3\}, \emptyset, \{quayTransporter \Delta quayTransmitter\}, \{nr_{quayTransporter} = (1, \infty), nr_{quayTransmitter} = (1, \infty)\}, \{nc_{quayTransporter} = (1cont/hour, \infty, 1cont/hour, \infty), nc_{quayTransmitter} = (2mess/sec, \infty, 2mess/sec, \infty)\}, \rangle$$

CS agents can play the roles *quayTransporter* and *quayTransmitter* in groups defined by  $g_1$ . These CS agents are in level  $l_2$ . These groups produce CS agents in level  $l_3$ . When a CS agent plays a role in a group defined by  $g_1$ , it cannot play a role in any other group. A CS agent can play simultaneously both *quayTransporter* and *quayTransmitter* roles in a group defined by  $g_1$ . A group defined by  $g_1$  must contain—at least—one CS agent playing the role *quayTransporter* and, at least, one CS agent playing the role *quayTransmitter*. A CS agent playing *quayTransporter* must possess the capacity to transport, at least, one container per hour. A CS agent playing *quayTransmitter* must possess the capacity to transmit, at least, two messages per second. And the whole population of agents playing *quayTransmitter* must possess the cumulative capacity to transmit, at least, two

messages per second.

$$\begin{aligned} CS_{4,0} &= \langle CS_{2,2}, \{transporter, transmitter\}, \emptyset, \{g_3\}, \emptyset, \lambda_{4,0} \rangle \\ CS_{5,0} &= \langle CS_{2,2}, \{transporter, transmitter\}, \emptyset, \{g_3\}, \emptyset, \lambda_{5,0} \rangle \\ CS_{6,0} &= \langle CS_{2,2}, \{transporter, transmitter\}, \emptyset, \{g_3\}, \emptyset, \lambda_{6,0} \rangle \end{aligned}$$

$CS_{4,0}$  is an under CS of  $CS_{2,2}$ , the quay fleet of quay 2. It plays the roles *transporter* and *transmitter* which consist in transporting a container and communicate with distant IAVs or the central authority of the port.  $CS_{4,0}$  has no under CS. Here  $CS_{1,3}$  is constituted of only one group  $g_3$  built the three individual IAVs,  $CS_{4,0}$ ,  $CS_{5,0}$  and  $CS_{6,0}$ .  $CS_{4,0}$  has no under CS roles to define.  $\lambda_{4,0}$  is the function indicating if  $CS_{4,0}$  can reach its goals. Because  $CS_{4,0}$  is a elementary CS, this function takes only into account capacities of the individual IAVs instanciated by  $CS_{4,0}$ .  $CS_{5,0}$  and  $CS_{6,0}$  are defined in the same way than  $CS_{4,0}$ .

The group specification of  $g_3$  can be :

$$\begin{aligned} gs_3 &= \langle \{transporter, transmitter\}, \\ &\quad \{l_{transporter} = \{l_0\}, l_{transmitter} = \{l_0\}\}, \{l_2\}, \\ &\quad \emptyset, \{transporter \Delta transmitter\}, \\ &\quad \{nr_{transporter} = (1, \infty), nr_{transmitter} = (1, \infty)\}, \\ &\quad \{nc_{transporter} = (1cont/hour, \infty, 1cont/hour, \infty), \\ &\quad nc_{transmitter} = (2mess/sec, \infty, 2mess/sec, \infty)\}, \rangle \end{aligned}$$

$gs_3$  is a group specification very similar to  $gs_1$  with a change of level and scale.

### C. Dynamic Aspects

In this section we present the mechanisms that insure the feasibility of the global goal during SoS evolution. The first one concerns the creation of groups with capacity matching accomplishment of the global goal and the second concerns the change of capacities for CS.

There are several articles deal with SoS construction or reconfiguration [43], [44], some including the management of CS capacity [45], [46], but without addressing the control of such systems.

1) *SoS Creation*: In our approach we try to incite CS agents to adopt a organizational structure appropriated to cooperate and fulfil a global goal.

At the SoS initialisation, outside the top CS and all CS with a physical existence, like most of the elementary CS, all intermediary CS have to be instanciated. The situation is quite similar when the global goal or even a non elementary CS goal is modified: all its under CS except those with no under CS (virtual or not) have to be created again. This is done in order to design organizations with capacities adapted to reach the new goal and its sub-goals. To form organizations of CS

matching the SoS global goals needs our model has to lean on a functional specification.

Missions can be seen as sequences of CS agent actions required to meet a goal. Missions in that model express the fact that a CS agent can engage simultaneously to fulfil several goal with the same mission. The fact that a mission  $\mathcal{M}x$  is allocated to the  $CS_{n,l}$  CS is noted  $\mathcal{M}x_{n,l}$ .

The global goal is decomposed as a graph tree. Each time a goal is divided into subgoals with a change of scale, it shows that these goals will be accomplished by a CS in one level and its under CS in at least another level. A goal  $g_x$  can be noted with its corresponding level  $l$ :  $g_{x,l}$ .

Here follows a graph tree presenting the possible global goal allocated to the SoS presented in fig. 1. CS agent  $CS_{1,3}$  would be engaged on  $\mathcal{M}1$ ,  $CS_{1,2}$  on  $\mathcal{M}2$ ,  $CS_{2,2}$  on  $\mathcal{M}3$  and so on. So the set of missions for this goal decomposition should be equal to  $\mathcal{M} = \{\mathcal{M}1_{1,3}, \mathcal{M}2_{1,2}, \mathcal{M}3_{2,2}, \dots\}$

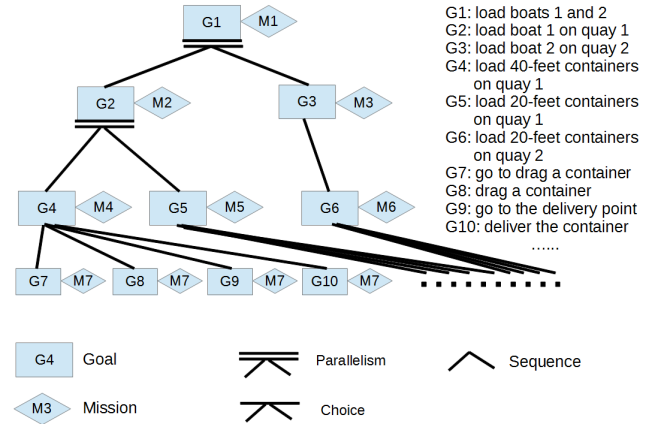


Fig. 9. The functional decomposition of SoS global goal.

It has to be noted that  $CS_{1,0}$  and  $CS_{2,0}$  cannot have any allocated mission because they are not considered as CS but as virtual CS.

A functional specification adapted to our formalism can be defined as follows:

$fs = \langle \mathcal{G}, \mathcal{M}, \mathcal{P}, mo \rangle$ , with

- $\mathcal{G}$ : the set of goals of the SoS.
- $\mathcal{M}$ : the set of labels for all missions. The missions can be allocated ( $\mathcal{M}x_{n,l}$ ) or not ( $\mathcal{M}x$ ).
- $\mathcal{P}$ : the set of global plans resulting from the goal decomposition tree.
- $mo : \mathcal{M} \rightarrow 2^{\mathcal{G}}$ : a function specifying for each mission the set of attached goals.

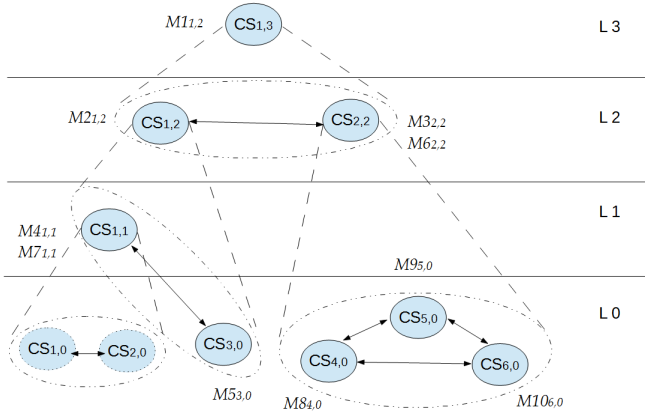


Fig. 10. Partial allocation of missions over the CS SoS.

Once the functional specification is established the SoS instantiates groups corresponding to goal needs in terms of capacity and available agents, using available group specifications. When all group are formed and the elementary CS agent matches the lower level groups, all groups can be instantiated by a CS agent. Then the SoS is fully formed and can accomplish its global goal.

#### Algorithm 1 SoS reconfiguration

**Require:** the set of levels:  $L$ , the set of CS agent definition:  $CS_{def}$ , the set of group specifications:  $Gr_{spe}$ , the functional definition of the SoS:  $fs_{SoS}$ , the hierarchical graph:  $Gr_L$ , the set of all existing CS agent:  $CS$  regrouping the set of elementary CS:  $CS_0$  and the top CS:  $CS_H$ .

**Ensure:** the new SoS configuration respects SoS fundamental characteristics or there is no available configuration.

**while**  $\exists CS_{n,l} \in CS \wedge CS_{n,l} \notin CS_0 \wedge CS_{n,l}(CS_{n'',l-1}) = \emptyset$   
**do**

Take a  $CS_{n,l}$  of the higher possible level  $l$ .

Take the goal  $g$  of  $CS_{n,l}$  using  $\mathcal{M}$  and  $mo$  of  $fs_{SoS}$ .

Decompose  $g$  into a set of subgoals  $\}$  present in level(s)  $\mathcal{L}$  and create the associated missions  $mg$ .

Place  $\}$  in  $\mathcal{G}$  of  $fs_{SoS}$ , place  $mg$  in  $\mathcal{M}$  and  $mo$  of  $fs_{SoS}$ .

Update  $\mathcal{P}$  of  $fs_{SoS}$  with the new goal decomposition.

Select a group specification  $gs \in Gr_{spe}$  which can represent  $CS_{n,l}$ .

Select CS agent definitions in  $CS_{def}$  which can play the necessary roles in  $gs$  and can fulfil  $\}$ .

Instantiate previous CS agent definitions and allocate  $mg$  to them.

**end while**

**return**  $fs_{SoS}$

2) *Capacity Evolution:* As said previously CS agent capacities evolve through time. Sometimes CS can lost capacities which compromise their mission and possibly the

mission of their super CS. When the capacities of a CS are modified, its super CS capacities are calculated thanks to CS and group specification. The capacity evolution is spread through levels thanks to influences. That means all CS agents with under CS can calculate its own capacity from its under CS capacities and group specification. So when a CS loses its capacities, it first tries to reorganize itself to stay able to fulfil its allocated mission and if it is impossible, it informs its super CS which do as exactly the same until the top CS agent is informed.

If the top CS agent is not able to reorganize itself anymore, by adopting a new organization structure, it is able to inform the modeler that the SoS has failed and there is no available mean to reach the global goal.

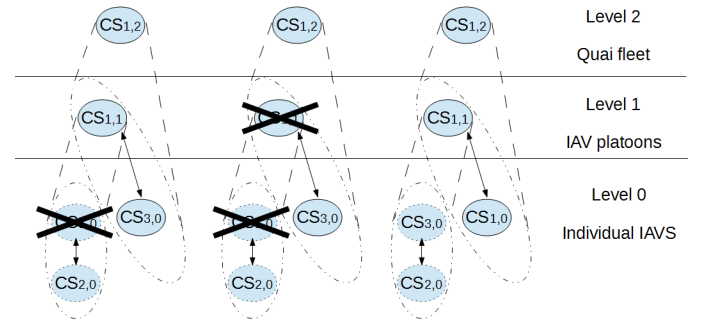


Fig. 11. Reorganization of a SoS guided by the CS capacity evolution.

This mechanism can be illustrated by the following example. The IAV  $CS_{1,0}$  has a failure and its capacity to carry a 40-feet container is compromised. So the capacity of the IAV platoon  $CS_{1,1}$  is also modified and it cannot reorganize itself only with its under CS  $CS_{1,0}$  and  $CS_{2,0}$ . It can no more accomplish its mission to deliver a series of 40-feet containers to the boat. So the failure is spread to  $CS_{1,2}$ , the IAV fleet in quay 1.  $CS_{1,2}$  can reorganize itself by switching IAVs  $CS_{3,0}$  and  $CS_{1,0}$ . That what is done, maintaining the structure of  $CS_{1,2}$ .

3) *Particular case of virtual CS instantiation:* In this section we consider the particular case of the creation of a non elementary CS whose under CS lost their CS status.

In a MAS model representing SoS, every CS has to be instantiated by a CS agent which respects the above CS definition and its characteristics. In particular, the identified elementary CS and the top CS have to possess their own instantiating CS agent during all the modelisation. When a CS stops to respect the SoS characteristics, it ceases to be a CS but it and its under CS interact with the SoS, even if, they can be no more considered as CS. That is why agents representing these CS should be deleted in the model but replaced with virtual CS agents which indicates their status of potential CS.

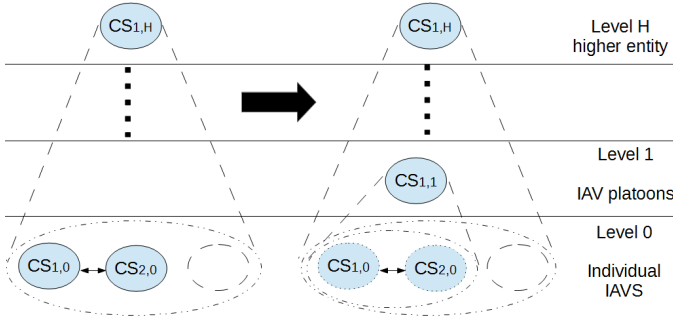


Fig. 12. Creation of super CS by aggregating under CS in group(s) and updating their status.

To illustrate this, let take a simple example. Two CS agents,  $CS_{1,0}$  and  $CS_{2,0}$ , representing individual IAVs as elementary CS in a level  $l_0$  and another agent,  $CS_{1,1}$ , representing the platoon formed by these two vehicles as a super CS in level  $l_1$ , higher than  $l_0$ . When the two AIVs are linked together to carry a heavy container, and one of them became the leader of this tasks,  $CS_{1,0}$  and  $CS_{2,0}$  lost their status of CS because they are violating operational and managerial independences and geographical dispersion characteristics.  $CS_{1,1}$ , after the aggregation of  $CS_{1,0}$  and  $CS_{2,0}$ , represents an elementary CS because its under CS have lost their CS status. Thus CS agents  $CS_{1,0}$  and  $CS_{2,0}$  are replaced by virtual CS agents because they are still used to determine  $CS_{1,1}$  evolution and capacities and can represent again non virtual CS in the future, when the aggregation link between  $CS_{1,0}$  and  $CS_{2,0}$  will be deleted.

#### D. Illustration of Dynamic Aspects

To illustrate the dynamic aspects we consider once again the SoS presented in section II and whose mission distribution was exposed in fig. 10 with our formalism.

Before the SoS is built its model should only contain the top CS at the highest level,  $CS_{1,3}$  and all its elementary CS at the lowest level,  $CS_{1,0}$ ,  $CS_{2,0}$ ,  $CS_{3,0}$ ,  $CS_{4,0}$ ,  $CS_{5,0}$  and  $CS_{6,0}$ . Other elements of the model should be a set of levels, with hierarchical, influence and perception graphs to apprehend links between levels, a set of CS agent definitions for every levels, a set of group specifications corresponding to these agents definition and a nearly empty functional specification. Only the global goal of the SoS should be known and allocated to the top CS. And its functional specification should be:

$$f_{SoS} = \langle \{g_{13}\}, \{\mathcal{M}_{1,3}\}, \emptyset, \{\mathcal{M}_1 \rightarrow \{g_{13}\}\} \rangle$$

According to fig. 9 and 10, once the SoS model is fully built, its functional speciation should be:

$$f_{SoS} = \langle \{g_{13}, g_{22}, g_{32}, g_{41}, g_{50}, g_{62}, g_{71}, g_{80}, \dots\}, \{\mathcal{M}_{1,3}, \mathcal{M}_{2,2}, \mathcal{M}_{3,2}, \mathcal{M}_{4,1}, \mathcal{M}_{5,0}, \mathcal{M}_{6,2}, \mathcal{M}_{7,1}, \mathcal{M}_{8,0}, \mathcal{M}_{9,0}, \mathcal{M}_{10,0}, \dots\}, \{ "g_{13} = g_{22}, g_{32} ", "g_{22} = g_{41}, g_{50} ", "g_{32} = g_{62} ", \dots\}, \{\mathcal{M}_1 \rightarrow \{g_{13}\}, \mathcal{M}_{2,2} \rightarrow \{g_{22}\}, \mathcal{M}_{3,2} \rightarrow \{g_{32}\}, \mathcal{M}_{4,1} \rightarrow \{g_{41}\}, \mathcal{M}_{5,0} \rightarrow \{g_{50}\}, \dots, \mathcal{M}_{7,1} \rightarrow \{g_{71}, g_{81}, g_{91}, g_{101}\}, \dots\} \rangle$$

#### E. Respecting Fundamental characteristics

In this section we show that directed SoS characteristics are respected in agent-based models generated using our formalism.

It is important to consider that, in the first three characteristics, the CS agents  $CS_{i',j'}$  and  $CS_{i'',j''}$  of a system  $CS_{i,j}$  are not related by a direct hierarchical relation:

$$\forall CS_{i',j'}, CS_{i'',j''} \in CS_{i,j}, \\ CS_{i',j'} \notin CS_{i'',j''} \wedge CS_{i'',j''} \notin CS_{i',j'}$$

Here are the five characteristics that any SoS should respect (see section I) and their translation in a MAS model representing a SoS:

- 1) **Operational independence:** CS agents  $CS_{i',j'}$  and  $CS_{i'',j''}$  of a system  $CS_{i,j}$  are independent in an operational way iff they are instantiated by CS agents,  $\mathcal{A}$ , and possess their own private variables,  $s_a$ , representing their resources.

$$\forall CS_{i',j'}, CS_{i'',j''} \in CS_{i,j}, \\ CS_{i',j'}, CS_{i'',j''} \in \mathcal{A} \wedge \\ s_{CS_{i',j'}} \neq \emptyset \wedge s_{CS_{i'',j''}} \neq \emptyset \wedge \\ s_{CS_{i',j'}} \cap s_{CS_{i'',j''}} = \emptyset$$

Fig. 13 shows that  $CS_{4,0}$  is operationally independent, but  $CS_{1,0}$  is not, because it shares a part or its resources with  $CS_{2,0}$ .

- 2) **Managerial independence:** CS agents  $CS_{i',j'}$  and  $CS_{i'',j''}$  of a system  $CS_{i,j}$  are independent in a managerial way iff they do not share any part of their own missions,  $\mathcal{M}_{i',j'}$  and  $\mathcal{M}_{i'',j''}$ .

$$\forall CS_{i',j'}, CS_{i'',j''} \in CS_{i,j}, \\ \mathcal{M}_{i',j'} \cap \mathcal{M}_{i'',j''} = \emptyset$$

In non specific SoS, this characteristic says that for each CS agent, no other CS agent gives it direct orders and only its super CS agent can allocate missions to it. But because we consider directed SoS endowed with a

strict hierarchy, the above notation is sufficient. Each CS agent manages its own missions independently of other CS agents.

$CS_{4,0}$  of fig. 13 is independent in a managerial way, but  $CS_{1,0}$  is not, because it shares its missions with  $CS_{2,0}$ .

- 3) **Geographic distribution:** CS agents  $CS_{i',j'}$  and  $CS_{i'',j''}$  of a system  $CS_{i,j}$  are geographically distributed iff their physical state,  $\phi_a$ , are totally distinct and not directly dependent. This last property can be formulated by the fact that it is not possible to compute directly all or any part of the physical state of a CS agent (using a function  $f$ ) from all or any part of the physical state of another CS agent.

$$\begin{aligned} & \forall CS_{i',j'}, CS_{i'',j''} \in CS_{i,j}, \\ & \phi_{CS_{i',j'}} \cap \phi_{CS_{i'',j''}} = \emptyset \wedge \\ & (\forall \phi'_{CS_{i',j'}} \subset \phi_{CS_{i',j'}}, \forall \phi'_{CS_{i'',j''}} \subset \phi_{CS_{i'',j''}}, \\ & \nexists f : \Phi \rightarrow \Phi, f(\phi'_{CS_{i',j'}}) = \phi'_{CS_{i'',j''}}, \\ & \forall f(\phi'_{CS_{i'',j''}}) = \phi'_{CS_{i',j'}}) \end{aligned}$$

In this case, no physical link between CS agents is authorized to accomplish missions. Only information exchanges between CS agents are authorized.

Practically, in a agent-based model it also can be noticed that if each CS agent is situated in an environment endowed with a metric and is not physically linked to other CS agents, according to this metric.

Fig. 13 shows that  $CS_{4,0}$  is geographically dispersed, but  $CS_{1,0}$  and  $CS_{2,0}$  are not. Because, in the case of  $CS_{1,0}$  is the leader of the platoon  $CS_{1,1}$  and  $CS_{2,0}$  is a follower, then it exists a function  $f$  and  $f(pos_{CS_{1,0}}) = pos_{CS_{2,0}}$ , where  $pos_{CS_{1,0}}$  and  $pos_{CS_{2,0}}$  designate the part of the physical state of these two CS agents which represents their position.

- 4) **Emergent Behaviour:** A set of CS agents  $CS_{i',j'}$  of a system  $CS_{i,j}$  cooperates iff CS agents cooperate to realize a global mission  $\mathcal{M}_{i,j}$  of the system formed of a set of missions  $\mathcal{M}_{i',j'}$ , that each CS agent of  $CS_{i,j}$  cannot realize alone.

$$\begin{aligned} & \forall CS_{i',j'} \in CS_{i,j}, \\ & \bigcup \mathcal{M}_{i',j'} = \mathcal{M}_{i,j} \end{aligned}$$

Said in another way, an agent based model represents a SoS iff joint actions of several CS agents is obligatory necessary to reach the global goal.

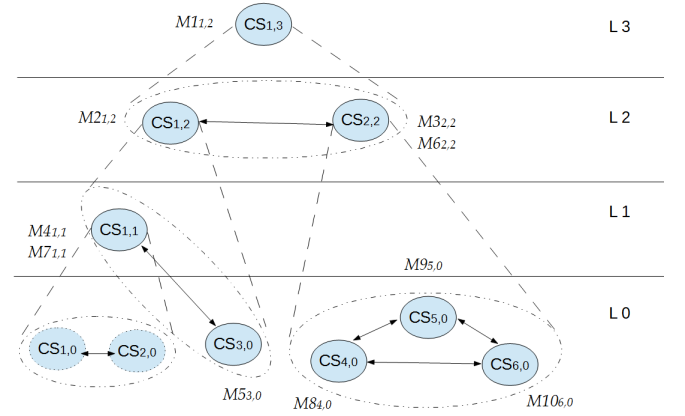


Fig. 13. Characteristics of a SoS.

CS agent  $CS_{2,2}$  of fig. 13 cannot fulfil its missions  $M_{32,2}$  and  $M_{62,2}$  only through the cooperation of its under CS  $CS_{4,0}$ ,  $CS_{5,0}$  and  $CS_{6,0}$  which fulfil missions  $M_{84,0}$ ,  $M_{95,0}$  and  $M_{106,0}$ .

- 5) **Evolutionary development:** This characteristic is different from the others because it can be noticed only during the SoS execution. In particular, this characteristic is expressed in three situations: a) a CS agent is added to the SoS, b) a CS agent of the SoS is removed or c) a CS agent notices that one of its missions is impossible to fulfil according to its capacity function.

To allow a SoS to accept new CS agents like in situation a) it is necessary that the new CS agents are placed in groups whose specification contains roles which do not affect the capacities of the CS agent formed by that group. In this case, the addition of new CS agents does not affect the capacity of the SoS to fulfil its global goal. To take advantage of this addition, to enhance the capacity of the SoS, an algorithm is necessary to optimally reorganize the SoS, including the new CS agent. This kind of algorithm is very specific to the design of the SoS problem and this not the topic of this article. Therefore, it is not developed here.

Situations b) and c) are quite similar. Because when the deletion of a CS agent doesn't affect its super CS capacity, it doesn't affect the capacity of the whole SoS. And when it affects its super CS capacity this super CS is in situation c). In this case the loss of capacity has to be communicated via influences, impossible missions have to be set free and the SoS reorganization has to be performed like in previous algorithm.

A system  $CS_{i,j}$ , formed by several CS agents  $CS_{i',j'}$  follows an evolutionary development iff when one of its CS agents allocated mission  $Mx_{i',j'}$  is detected as impossible to fulfil according to its capacity function  $\delta$  and the actual state of the level  $j'$   $\lambda_{i,j}$  and  $\delta_{j'}(t)$ , this mission is no more allocated to any CS agent, but is set free in the set of missions  $\mathcal{M}$  in the functional specification of the SoS.

$$\begin{aligned} &\forall CS_{i',j'} \in CS_{i,j}, \\ &\lambda_{i,j}(\delta_{j'}(t), \delta(Mx_{i',j'})) = 0 \\ &\rightarrow Mx_{i',j'} \notin \mathcal{M} \cup Mx \in \mathcal{M} \end{aligned}$$

Once the mission is set free an algorithm has to allocate it to another CS agent or reorganize the SoS to fulfil it, in a feasible way. [12] presents such SoS reorganisations without evoking a formalized algorithm.

Fig. 14 shows three steps in the life of a SoS. The second step is obtained by the suppression of  $CS_{3,0}$ . Step three represents the same SoS after the addition of  $CS_{3,0}$  and  $CS_{4,0}$  to the SoS. It should be noted that  $CS_{3,0}$  from step one and three is not obligatory the same CS agent.

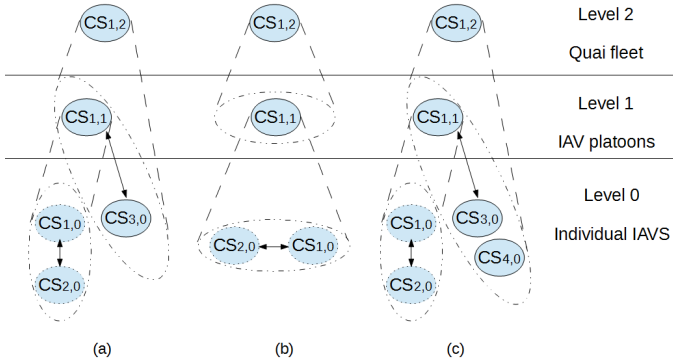


Fig. 14. Three steps in the life of a SoS: a) initial state, b) deletion of  $CS_{3,0}$  and c) addition of  $CS_{3,0}$  and  $CS_{4,0}$ .

## V. CONCLUSION AND FUTURE WORK

In this article we presented an operational SoS definition based on the fundamental characteristics for CS: operational independence, managerial independence, geographic distribution, evolutionary development and emergent behaviour. We proposed a generic multi-level multi-agent formalism to represent SoS managed by a central authority. The proposed model can be applied to any SoS controlling it in a changing environment affecting CS capacities. Only repeated execution of multi-agent based simulations (MABS) seems suited to test and validate algorithms to control, diagnose systems with numerous heterogeneous interacting sub-systems endowed of a changing hierarchical structure.

Moreover the system is cut into groups and levels. We also give elements to prove the validity of our approach.



Fig. 15. A platoon of IAVs in SCANerstudio.

These concepts are applied by modeling SoS whose elementary CS are IAVs in the InTrade project [6]. The resulting simulation is being implemented on MaDKit multi-agent platform endowed with AGR model [47]. SCANerstudio [48] is a simulation software which allows to pilot AIVs in realistic conditions. The resulting simulation should be used to control direct system by interfacing a SoS model of IAVs fleet in MaDKit and a realistic representation of IAVs in a container port in SCANerstudio. In particular, it permits to observe the influence of low level events managed by SCANerstudio at the same level or in higher levels. It could be an IAV failure spreading to its platoon or the complete loss of a vehicle which forces a fleet of IAV or the whole SoS to recalculate their scheduling.

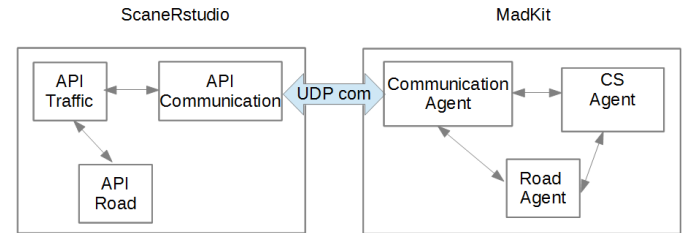


Fig. 16. Relations between SCANerstudio and MadKit via UDP communications. SCANerstudio is a network of communicating APIs and MadKit is a multi-agent platform.

A development of the applied part of this work is to couple the multi-level multi-agent based model in MaDKit with other simulation tools than SCANerstudio. Flexsim and its Container Terminal Library (Flexsim CT) is used to model

traffic flow in container terminal environments [49]. It could figure the same SoS but represented at a higher level and providing semi-realistic inputs for that level. For the moment, levels of the IRM4MLS are only considered for simulating different scales of granularities. But IRM4MLS offers the possibility to model levels in parallel, representing relatively independant aspects of a phenomenon or SoS and simultaneously activated. This can be experimented using InTrade examples, creating levels reserved for diagnostic and other driving of IAVs. A SoS of IAVs considered through these two functions will lead to two different structures. In the future, our work could be applied to other forms of SoS but not only directed ones with addition of authority and organizational aspects. Another track is to improve the building group mechanism. It can be done by specifying strategies to optimize the group specification choice and/or the choice of CS agents to constitute these groups according to the robustness of the resulting SoS or the quality of the accomplished mission.

## REFERENCES

- [1] P. Anioté, E. Cariou, and E. Gouardères, "Modélisation de systèmes complexes distribués : l'ingénierie des modèles pour l'intégration des paradigmes "agent" et "composant"," in *Proceedings of the 2<sup>nd</sup> Journée Multi-Agent et Composant (JMAC)*, Nîmes, 21 Mars 2006.
- [2] A. P. Sage and C. D. Cuppan, "On the systems engineering and management of systems of systems and federations of systems," *Inf. Knowl. Syst. Manag.*, vol. 2, no. 4, pp. 325–345, Dec. 2001. [Online]. Available: <http://dl.acm.org/citation.cfm?id=1234195.1234200>
- [3] A. Sage and S. Biemer, "Processes for System Family Architecting, Design, and Integration," *Systems Journal, IEEE*, vol. 1, no. 1, pp. 5–16, sept. 2007.
- [4] M. Huhn, J. Muller, J. Gormer, G. Homoceanu, N.-T. Le, L. Martin, C. Mumme, C. Schulz, N. Pinkwart, and C. Muller-Schloer, "Autonomous Agents in Organized Localities Regulated by Institutions," in *Digital Ecosystems and Technologies Conference (DEST), 2011 Proceedings of the 5th IEEE International Conference on*, 31 2011–june 3 2011, pp. 54–61.
- [5] Y. Bar-Yam, M. A. Allison, R. Bartdorf, H. Chen, H. Generazio, H. Singh, and S. Tucker, "The Characteristics and Emerging Behaviors of Systems of Systems," *NECSI : Complex Physical, Biological and Social Systems Project*, 2004. [Online]. Available: <http://necsi.edu/education/onexee/winter05/NCESISoS.pdf>
- [6] [Online]. Available: <http://www.intrade-nwe.eu/fr>
- [7] T. Le-Anh and M. De Koster, "A review of design and control of automated guided vehicle systems," *European Journal of Operational Research*, vol. 171, no. 1, pp. 1–23, May 2006. [Online]. Available: <http://ideas.repec.org/a/eee/ejores/v171y2006i1p1-23.html>
- [8] J. M. Parker, "Applying a System of Systems Approach for Improved Transportation," *S.A.P.I.E.N.S.*, vol. 3, no. 2, 2010.
- [9] E. Sloane, T. Way, V. Gehlot, R. Beck, J. Solderitch, and E. Dziembowski, "A Hybrid Approach to Modeling SOA Systems of Systems Using CPN and MESA/Extend," in *Systems Conference, 2007 1st Annual IEEE*, april 2007, pp. 1–7.
- [10] J. Simpson and M. Simpson, "System of Systems Complexity Identification and Control," in *System of Systems Engineering, 2009. SoSE 2009. IEEE International Conference on*, 30 2009–june 3 2009, pp. 1–6.
- [11] J. M. Held, "Systems of Systems: Principles, Performances, and Modelling," Ph.D. dissertation, The University of Sydney, 2008.
- [12] W. Khalil, R. Merzouki, B. Ould-Bouamama, and H. Haffaf, "Hypergraph Models for System of Systems Supervision Design," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 42, no. 4, pp. 1005–1012, july 2012.
- [13] B. Zhou, A. Dvoryanchikova, A. Lobov, J. Minor, and J. Martinez Las-tra, "Application of the Generic Modelling Method for System of Systems to Manufacturing Domain," in *IECON 2011 - 37th Annual Conference on IEEE Industrial Electronics Society*, nov. 2011, pp. 352–358.
- [14] T. Gezgin, C. Etzien, S. Henkler, and A. Rettberg, "Towards a Rigorous Modeling Formalism for Systems of Systems," in *Object/Component/Service-Oriented Real-Time Distributed Computing Workshops (ISORCW), 2012 15th IEEE International Symposium on*, april 2012, pp. 204–211.
- [15] G. Morvan, A. Veremme, and D. Dupont, "IRM4MLS: The influence Reaction Model for Multi-Level Simulation," in *11th International Workshop on Multi-Agent-Based Simulation (MABS'10)*, may 2010.
- [16] J. F. Hübner, J. Simão, and O. Boissier, "Spécification structurelle, fonctionnelle et déontique d'organisations dans les SMA," in *Journées Francophones pour l'Intelligence Artificielle Distribuée et les Systèmes Multi-Agents (JFIADSMA'02)*, 2002.
- [17] —, "Moise+: Towards structural, functional and deontic model for MAS organization," in *Proceeding of the first International joint conference on Autonomous Agents and MultiAgent Systems (AAMAS'02)*, Bologna, Italy, July 15–19 2002, p. 502.
- [18] B. Sauser, J. Boardman, and D. Verma, "Systemics: Toward a biology of system of systems," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 40, no. 4, pp. 803–814, july 2010.
- [19] M. Maier, "Architecting Principles for Systems-of-Systems," in *proceeding of the Sixth Annual International Symposium INCOSE*, INCOSE, Boston, 1996, pp. 567–574.
- [20] J. Boardman and B. Sauser, "System of Systems the meaning of of," in *Proceeding of the 2006 IEEE/SMC International Conference on SoS Engineering*, Los Angeles, CA, USA, April 2006.
- [21] A. Gorod, B. Sauser, and J. Boardman, "System-of-systems engineering management: A review of modern history and a path forward," *Systems Journal, IEEE*, vol. 2, no. 4, pp. 484–499, dec. 2008.
- [22] M. Jamshidi, *System of Systems Engineering Innovations for the 21st Century*. Hoboken: Wiley, 2008, ch. Introduction to System of Systems, pp. 1–20.
- [23] J. P. Dauby and S. Upholzer, "Exploring Behavioral Dynamics in Systems of Systems," *Procedia Computer Science*, vol. 6, no. 0, pp. 34–39, 2011, complex adaptive systems. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1877050911004741>
- [24] F. Michel, A. Gouaich, and J. Ferber, "Weak Interaction and Strong Interaction in Agent Based Simulations," *Lecture Notes in Computer Science*, vol. 2927, pp. 43–56, 2003.
- [25] M. D. Gerst, P. Wang, A. Roventini, G. Fagiolo, G. Dosi, R. B. Howorth, and M. E. Borsuk, "Agent-based modeling of climate policy: An introduction to the ENGAGE multi-level model framework," *Environment Modelling & Software*, vol. 1–14, 2012.
- [26] (2004, accessed March 4) "SoS" and "FoS" FAQ, Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics, <http://www.acq.osd.mil/dpap/Docs/FAQs%20-%20SoS%20&%20FoS.doc>. Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics. [Online]. Available: <http://www.acq.osd.mil/dpap/Docs/FAQs%20-%20SoS%20&%20FoS.doc>
- [27] V. Mahulkar, S. McKay, D. Adams, and A. Chaturvedi, "System-of-Systems Modeling and Simulation of a Ship Environment With Wireless and Intelligent Maintenance Technologies," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 39, no. 6, pp. 1255–1270, nov. 2009.
- [28] "Systems Engineering Guide for Systems of Systems: Summary," Washington, D.C.: Office of the Director, Defense Research and Engineering, Director of Systems Engineering, 2010.
- [29] J. Ferber, *Les Systèmes Multi-Agents: Vers une Intelligence Collective*. InterEditions, 1995.
- [30] D. Scerri, S. Hickmott, A. Drogoul, and L. Padgham, "An Architecture

- for Distributed Simulation with Agent-Based Models,” in *Proc. of 9th Int. Conf on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Lésperance, Luck, and Sen, Eds., Toronto, Canada, May, 10-14, 2010, pp. 541–548.
- [31] S. Picault and P. Mathieu, “An interaction-Oriented Model for Multi-Scale Simulation,” in *the 22nd International Joint Conference on Artificial Intelligence (IJCAI’11)*, 2011.
  - [32] G. Morvan, “Multi-level agent-based modeling - Bibliography,” *CoRR*, vol. abs/1205.0561, 2012. [Online]. Available: <http://arxiv.org/abs/1205.0561>
  - [33] J. Ferber and O. Gutknecht, “A meta-model for the analysis and design of organizations in multi-agent systems,” in *Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS98)*, 1998, pp. 128–135.
  - [34] J. Ferber, *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*. Addison-Wesley Longman Publishing Co., Inc, 1999.
  - [35] N. Gaud, S. Galland, F. Gechter, V. Hilaire, and A. Koukam, “Holonc Multilevel Simulation of Complex Systems : Application to Real-Time Pedestrians Simulation in Virtual Urban Environment,” *Simulation Modelling Practice and Theory*, vol. 16, pp. 1659–1676, 2008.
  - [36] A. Koestler, *The Ghost in the Machine*. Hutchinson, 1967.
  - [37] P. Ribino, C. Lodato, S. Lopes, V. Seidita, V. Hilaire, and M. Cossentino, “A Norm-Governed Holonic Multi-Agent System Metamodel,” in *Proc. 13th International Workshop on Agent-Oriented Software Engineering*, Valencia, Spain, June, 4 2012.
  - [38] J. Ferber and J.-P. Müller, “Influences and Reaction: a Model of Situated Multiagent Systems,” in *2nd International Conference on Multi-agent systems (ICMAS-96)*, 1996, pp. 72–79.
  - [39] F. Michel, “The IRM4S Model: The Influence/Reaction Principle for Multi-Agent Based Simulation,” in *AAMAS ’07: Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*. New York, NY, USA: ACM, 2007, pp. 1–3.
  - [40] J.-B. Soyez, G. Morvan, D. Dupont, and R. Merzouki, “A Methodology to Engineer and Validate Dynamic Multi-level Multi-agent Based Simulations,” in *MABS’12 - 13th International Workshop on Multi-Agent Based Simulation*, Valencia, Spain, 4th-5th June 2012.
  - [41] O. Gutknecht, J. Ferber, and F. Michel, “Integrating tools and infrastructures for generic multi-agents systems,” LIRMM, 161, rue Ada - Montpellier - France, Tech. Rep., 2001.
  - [42] J. Ferber, O. Gutknecht, and F. Michel, “From Agents to Organizations: an Organizational View of Multi-Agent Systems,” in *Agent-Oriented Software Engineering (AOSE)*, vol. 4, 2004, pp. 214–230.
  - [43] P. Acheson, L. Pape, C. Dagli, N. Kilicay-Ergin, J. Columbi, and K. Haris, “Understanding System of Systems Development Using an Agent-Based Wave Model,” *Procedia Computer Science*, vol. 12, no. 0, pp. 21 – 30, 2012, complex Adaptive Systems 2012. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1877050912006151>
  - [44] F. Yang, C. Dagli, and W. Wanga, “Cognition Evolutionary Computation for System-of-systems Architecture Development,” *Procedia Computer Science*, vol. 6, no. 0, pp. 40 – 45, 2011, complex adaptive systems 2011. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1877050911004753>
  - [45] C. O. Adler and C. H. Dagli, “Enabling Systems and the Adaptability of Complex Systems-of- Systems,” *Procedia Computer Science*, vol. 12, no. 0, pp. 31 – 36, 2012, complex Adaptive Systems 2012. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1877050912006163>
  - [46] D. Flanagan and P. Brouse, “System of Systems Requirements Capacity Allocation,” *Procedia Computer Science*, vol. 8, no. 0, pp. 112 – 117, 2012, *Conference on Systems Engineering Research*. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1877050912000233>
  - [47] [Online]. Available: <http://www.madkit.org>
  - [48] [Online]. Available: <http://www.scanersimulation.com/>
  - [49] [Online]. Available: <http://www.flexsim.com/>