Initiation and Formation of Constellations in Systems of Systems

Pontus Svenson and Jakob Axelsson
RISE Research Institutes of Sweden
Stockholm, Sweden
pontus.svenson,jakob.axelsson@ri.se

***Abstract*—Systems of systems engineering is aimed at developing solutions that provide the capability to fulfil a need that cannot be met by an integrated system. In many SoS, there is a need to solve several tasks at the same time. Each set of constituent systems that collaborate to solve such a task is a constellation. In this paper, we discuss the process of forming such constellations. We focus on collaborative systems of systems and describe conceptual models for the formation of constellations.**

Keywords—systems of systems, capability, constellation, mission, transport.

I. Introduction

A *system of systems* (SoS) consists of a set of *constituent systems* (CS) that are independently developed, owned, and operated while cooperating in order to attain synergistic effects [1]. SoS are ubiquitous and will become even more so as digitalization continues and affects more and more business areas. SoS can be organized in different ways, ranging from centrally directed, where each CS is sub-ordinated to a coordinator, to free and open collaborations between equal entities.

Within the SoS, there can also be other entities, such as *mediators* that help the CS to collaborate. Both CS and mediators are *elements* of the SoS. The mediators can be seen as a special kind of CS. Since the CS are independently developed and operated, their developers and operators can also be independent. The same goes for mediators, that also have developers and operators. In principle, a SoS element also has an owner – for simplicity we will however in this paper mostly ignore the owners. In Figure 1 we show a conceptual view of a SoS with multiple CS and mediators.

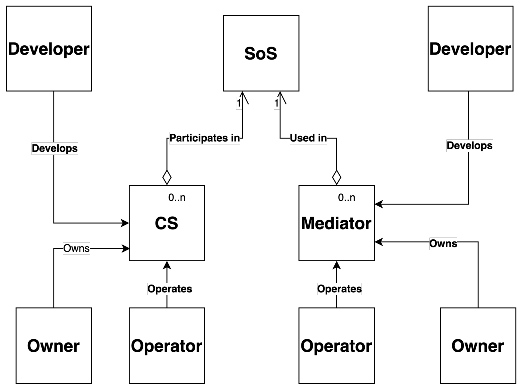

Figure 1. Conceptual view of a SoS, containing multiple CS and mediators, each with owner, developer, and operator.

This research was funded by Vinnova (Sweden's innovation agency), grant no. 2019-05100.

While there is a large literature on SoS, for the most part it focuses on SoS where the CS are fully cooperating and share the same goal more or less completely. We argue that this is often not the case: in many interesting SoS, the CS have multiple goals, only some of which are aligned with the goals of the other CS and the SoS as a whole. There are also many interesting SoS where the only shared goal is to make money: in such SoS, the CS collaborate since this enables them to get more business opportunities than working completely by themselves.

Of course, for such a collection of systems to be properly referred to as a SoS, there must be strong incentives for the individual systems to collaborate – the CS cannot have *completely* different and conflicting goals. In this paper and our ongoing research, we are interested in SoS where the CS are simultaneously both collaborating and competing. In this paper we discuss some aspects of how collaboration in such SoS can be described. We use three examples to illustrate the concepts introduced:

- A mobility SoS, where the CS are vehicles that collaborate to transport people or goods;
- A truck platooning SoS, where the CS are trucks that collaborate by driving closely together in order to save fuel;
- A military task force consisting of units from different coalition partners.

We will flesh out the examples and give them more details as needed in the discussion below. In addition to the presence of both collaboration and competition in these SoS, note also that they all need the capability to solve multiple problems: they are not built to accomplish *one* thing, after which they can be dismantled or re-configured for another task. Instead, they must be able to function for a long time and solve several problems simultaneously. We will return to this aspect in the next section.

The goal of this paper is to contribute to the discussion on SoS terminology, particularly for SoS that contain large numbers of CS that have some degree of competition and where the SoS is intended to solve many problems over an extended period of time. The main contribution of the paper is further elucidation of the concept of *constellation* [2] and a discussion of how such constellations can be formed in SoS.

The remainder of the paper is structured as follows: Section II discusses SoS concepts, in particular constellations, and tasks on SoS, constellation and CS levels. Section III further elaborates on constellations and how competition is an important aspect of some SoS. This is followed by a discussion of conflicting goals and a description of some of the requirements on the constellation formation mediator in Section IV, whereafter we summarize the conclusions in Section V.

II. Systems of systems

While we covered basic SoS concepts in the introduction, the type of SoS under study in this paper CS requires some new terminology.

A. SoS Concepts

A SoS is always engineered with some goal in mind. It is intended to solve *problem instances* within some *problem domain*. Following the terminology introduced by [3] for systems, we will say that the SoS is intended to perform *interventions* that lead to the solution of problems. These interventions are the goals of the SoS. How does the SoS accomplish this goal?

In some cases, the SoS is intended to solve a well-defined problem that has only one problem instance. In such cases, the SoS engineers collaborate during development to find the optimal combination of CS that fulfils the needs. While the CS are still managerially and operationally independent, the SoS engineering problem is still conceptually similar to the development of an integrated system. The independence of the CS and the fact that the CS will have different developing and operating organizations introduces complexity that must be handled, but the shared goal of the SoS developers provides an incentive for collaboration.

Another kind of SoS appears when a new actor sees a business possibility and develops a way of connecting already existing systems to form a new set of capabilities that solve a previously unaddressed need. The sharing economy provides many examples of such SoS. In these SoS, the CS developers may not even be aware that their CS are being used in a SoS; they are thus good examples of virtual SoS in Maier's classification [1].

In the types of SoS that are the subject of this paper, not all of the CS will be actively working on the same task at the same time. Different sets of CS will be working on different problem instances. The problem instances are related: they belong to the problem domain which the SoS was constructed to intervene in.

A set of CS that currently operate together is a constellation of the overall SoS [2]. It is the constellations that accomplish the goals of such SoS. We illustrate these concepts in Figure 2.

As mentioned above, a SoS consists of independently managed and operated CS that also have uses outside of the SoS. One consequence of this is that each CS is also engineered to perform an intervention in a problem domain. A CS, like any system, solves a problem instance that belongs to

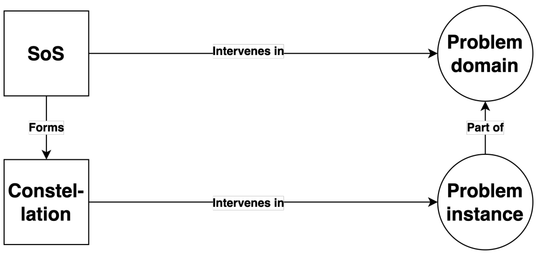

Figure 3. A SoS is built to intervene in a problem domain; it does so by forming constellations that intervene in (solve) problem instances.

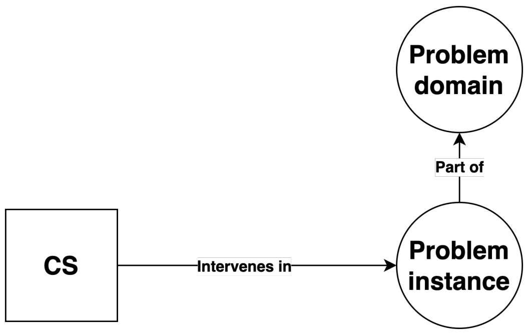

Figure 2. A CS is intended to intervene on a problem instance, that belongs to a problem domain.

a problem domain (see Figure 3). Different CS that belong to the same SoS may be intended to solve problem instances from different domains: one of the advantages of SoS is that they enable the composition of such CS that have different capabilities.

It is important to note here that the problem domains as well as the problem instances are in general different for the SoS and for the CS that form the SoS! To solve a problem instance, the CS will make use of its *capabilities* (see [4]). An SoS is created when someone realises that it is possible to solve other, more complicated, problem instances, by combining the capabilities of individual CS.

To clarify this important difference, we will use the following terminology:

- A CS is intended to intervene in an instance of its problem domain by solving a *CS-level task*;
- A constellation is intended to intervene in an instance of the SoS problem domain by solving a *constellation level task*;
- A SoS is intended to intervene in the SoS problem domain by solving *SoS level tasks*.

These concepts are illustrated in Figure 4.

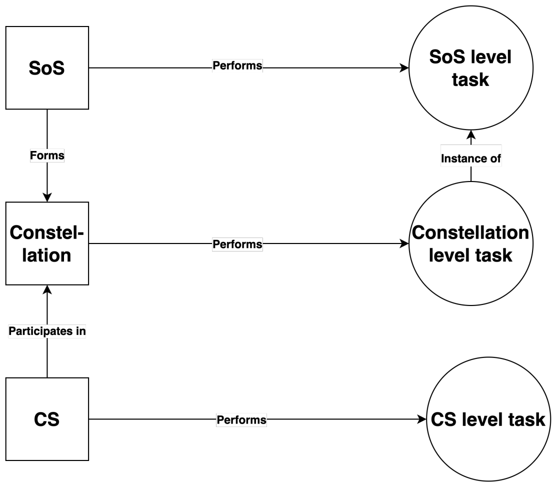

Figure 4. The relations between SoS, constellations, CS and the tasks that they perform.

We now describe the three examples that will be used for illustration in this paper and describe the tasks in them.

B. *Mobility SoS*

In a mobility SoS, the CS are vehicles that can transport people. The CS level task is to transport something from one location to another. The goal of the CS is to get as many transports as possible, i.e., to make as much money as possible.

The vehicles are operated by several different companies, who at the same time cooperate in the mobility SoS and compete to get passengers. The cooperation consists of joint planning and payment of travels. This means that the traveller does not need to know about the different companies involved. Instead, they simply use a mediator to order and pay for a journey that, e.g., first uses a small vehicle from company A to get to a mobility hub, where they board a large vehicle from company B. The vehicles that are involved in such a transport are a constellation. The constellation level task is thus the same as the CS level task: to transport something from one location to another. While the task is the same on the CS and constellation levels, note that there is competition in the SoS: the different CS operators each want to get as much business as possible. To get them to cooperate in the SoS it is necessary to show that this collaboration enables them to get more business.

The SoS level task is to solve all users' transport needs simultaneously. This is the same as forming constellations that solve all the problem instances.

There are several mediators present, for example a payment mediator that ensures that passengers only need to pay once and that each involved CS gets its fair share of the price, and a travel planning mediator that uses the travel needs of travellers and the locations and capacities of the CS to match travellers to a set of CS that can fulfil their needs.

C. *Platooning SoS*

In a platooning SoS [5], the CS are trucks belonging to different haulers and manufactured by different companies. The trucks in a platoon drive very closely together, enabling a reduction in aerodynamical drag which in turn leads to fuel reductions, saving money for the haulers. At the same time, the haulers as well as the truck manufacturers are competing: each of the haulers want to get as many transport assignments as possible, and each of the truck manufacturers want to sell to as many haulers as possible.

The CS level task is to transport some goods from one location to another, i.e., perform the truck's normal transport mission. The goal of the CS is to get as many such transports as possible, i.e., to make money.

The constellation level task is to enable the CS to reduce their fuel consumption. This is *not* the same as the CS level task. To get the CS operators to participate in the SoS, it is necessary to show that the benefit from the constellation level task outweighs any possible drawbacks the collaboration gives the CS level task [6].

The SoS level task is to enable the formation of platoons/constellations.

D. *Military task force*

Our final example is a military task force composed of units from coalition partners that is assumed to use mission command type tactics (i.e., commanders give sub-ordinates goals and the means to achieve them but allow for a high degree of independence in *how* the goals are achieved).

Here, the CS are military operational units that are large enough or contain specialized enough capabilities to be able to operate independently. Units belonging to different countries are naturally independent, but note that also units from the same country's armed forces can be independent to some degree.

While all the units in the task force can be assumed to share common goals (e.g., defeat the enemy; avoid friendly fire) they also have their own individual goals (e.g., ensure survival of own unit, keep vessel afloat, keep aircraft flying). Furthermore, each unit must balance between cooperating with others to reach common objectives and acting selfishly to reach their own objectives.

III. Constellations and Competition

We have thus seen that the SoS level task boils down to forming constellations. We have several times alluded to the importance of competition in the kind of SoS under study and will now delve deeper into this subject.

A. *Constellations*

There are in general several different ways in which a problem instance can be solved. Hence, the SoS must make a choice of how to accomplish the goal. This choice is one major reason for the competitive aspects of the SoS: in making the choice of how to solve a problem instance, the SoS must select some CS that are allowed to participate in the constellation that solves it. The CS that participate in the constellation will get some value from this, and those that do not participate will not get that value. In order to have a well-functioning SoS, it is necessary to consider both the need to solve the problem instances as efficiently as possible and the need to provide enough value to all important CS operators so that they continue to cooperate in the SoS.

We note that the value given need not always be money. It can also be business opportunities or the possibility to reduce costs (as in platooning). In some cases, a CS can get value from *not* being involved in a constellation, while still benefiting from the solution of the problem instance that the constellation solves.

Note that at any given time, there will be multiple active constellations, working on different problem instances. A given CS can participate in several constellations, provided it has the capabilities needed to do so.

Figure 5 illustrates how constellations solve problem instances, and CS can participate in multiple constellations.

B. *Competition*

Recall that CS are independently operated. This means that they have the option to decline participation in a constellation (see [7] for a discussion of the decision options of a CS). The SoS must also take this into account when forming constellation. We now turn to the different goals and objectives that influence the forming of a constellation.

In a collaborative SoS each CS has its own objectives and agenda. It can choose to collaborate with others in order to achieve mutual benefits. The benefit that a CS receives could be directly helping it to achieve its own objectives – in this case, it is very natural to collaborate. However, the benefit

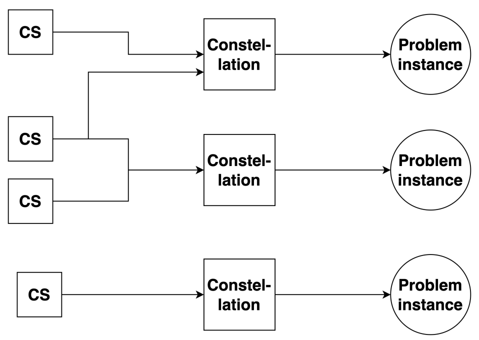

Figure 5. Problem instances are solved by constellations. A CS can participate in multiple constellations.

could also be in the form of compensation (money, points, …) – in such cases the CS might choose to collaborate in the constellation even if the collaboration hinders or reduces its own objectives.

Looking at an individual CS, it can be in several different states with respect to its goals. Sometimes, it can be the case that collaborating with other CS in a constellation would enable it to reach its objectives faster, cheaper, or more efficiently – in this case, it should be actively looking for a constellation to join, and even take initiatives to form a constellation.

In other cases, the CS might determine that the gains of joining a constellation would increase its costs or slow it down – in this case, it should say no to offers to participate in a constellation. Note that this does not mean that it leaves the SoS – it still wants to have the opportunity to join a constellation when this gives it advantages. An example of this from vehicle platooning is when a truck is close to its destination. In this case, it might be better for it to quickly reach its destination, deliver the goods, and only then start looking for platoons/constellations to join.

A perhaps slightly more common case is that the CS is indifferent as to whether it should join a constellation or not. This could be the case when the perceived gain or cost is very small in magnitude. It could be that such CS should select to participate if they are asked to join a constellation, in order to collect goodwill from the other CS.

We will now return to the three examples and discuss constellations and competition in those cases.

C. Mobility SoS

The problem domain of the mobility SoS is transportation of people in a certain area. The problem instances are specific travel needs from specific people. The SoS puts together a constellation of vehicles that can provide a person with their requested travel. The SoS will attempt to use the same CS (vehicle) for multiple travels – the mobility gets more efficient by allowing shared rides. The CS will participate in several constellations (travels). The SoS will need to form constellations that solve the travel needs as efficiently as possible – this is a hard optimization problem. In addition to solving the travel needs, the optimization also needs to ensure that vehicles are used as efficiently as possible: a vehicle with 10 seats should not be used to transport 1 person.

A CS in the mobility SoS has the goal to make money. It gets paid for transport passengers. It thus wants to have as many passengers as possible, while avoiding running empty. It is beneficial for the CS to share data with the travel optimizer, so that it gets more tasks.

D. Platooning SoS

The platooning SoS exists to reduce fuel in general; this is the problem domain. The SoS puts together a constellation that can save fuel; the constellation solves the fuel reduction problem for its CS. The SoS will need to solve optimization problems in order to form constellations. Some of the trucks might need to wait or take a longer route to reach the platoon/constellation – this will need to be taken into account in the optimization.

A CS in a platooning SoS wants to join platoons to reduce fuel. However, it doesn't get any money from being in platoons – so it chooses whether to be in a platoon or not based on the balance between the possible fuel reduction and the costs of joining the platon.

E. Coalition task force SoS

The task force exists to fight an enemy – this is the problem domain. Problem instances are specific tasks to carry out, for example: providing air cover for an area; attacking hostile ground units; seizing and holding ground. The SoS needs to put together constellations for solving each of these problems. A military unit (CS) can participate in several constellations simultaneously, e.g., an aircraft can contribute both to giving air cover and to providing surveillance of hostile units. The SoS needs to solve hard optimization problems.

A military unit is sub-ordinated to its higher commanders. However, the unit still can have goals of its own, and needs to balance the benefits and drawbacks of joining a constellation. For a simple example of this, consider an aircraft with a radar sensor and a vessel with a missile. To launch the missile, the vessel needs to have sensor coverage of the intended target. Assume that the vessel cannot achieve this using own sensors but can get this if the aircraft enables its sensor. A constellation consisting of the aircraft and the vessel can successfully eliminate the target. However, enabling the sensor will instantly reveal the aircraft's position and subject it to hostile fire. The vessel (or the SoS) can suggest the formation of the constellation, but the aircraft might refuse if its own objectives are jeopardized. Of course, depending on command structure, such a refusal could be disallowed. But in many cases the command structures are *not* sufficiently clear to allow this – consider for instance a coalition operation.

F. CS Always Compete

We note in passing that the fact that the CS in a SoS are independently operated actually ensure that their goals could be conflicting. If it is never the case that two CS have conflicting goals, can they really be said to be independent in any meaningful way?

In the next section we turn to how a SoS can form constellations.

IV. Forming constellations

We now turn to the question of how a constellation is created. In this, we will consider who initiates the constellation, what information about CS capabilities and user needs is required, and principles for finding better constellations.

A. *Constellation initiator*

In directed SoS, this is straight-forward – the coordinating agent instructs some CS that they should cooperate. For instance, an emergency dispatcher could instruct an ambulance, a fire truck, and a police patrol to handle the same accident, i.e., to form a constellation that is assigned to the accident. (Of course, a difficult optimization problem still needs to be solved.) An interesting approach to service-based SoS engineering where the SoS developer matches user needs to CS capabilities is presented in [8].

In collaborative SoS, each CS must decide for itself whether to join a constellation or not. As discussed above, a passive CS in a collaborative SoS can be actively looking for a constellation, opposed to joining a constellation, or indifferent. Once an actor in the SoS suggest a possible constellation, all the proposed members of it must analyse the situation and determine whether it is beneficial for them to join it.

Who can suggest the formation of a constellation? One solution is to have each CS determine in each time-step whether to suggest a constellation or not. For platooning, this could for instance be done by having trucks that by chance happen to drive closely suggest forming a platoon to each other. However, a better approach is to use mediator services for this.

A constellation forming mediating service must be able to see opportunities where forming a constellation would be beneficial for all agents that are needed in the constellation. Note that this is not the same as requiring that the *direct* benefit for a CS must be positive. It will often be the case that the benefit for some of the members of the constellation consists in getting paid by the others.

It is of vital importance that all actors involved trust the mediator. If users believe that the mediator will not find them the most cost-effective solution, they will stop using it.

B. *Capability ontology*

To be able to see opportunities for constellations, the mediator needs to have information about the capabilities of the CS as well as information about the needs of the users of the SoS. Note that the mediator should, in general, be allowed to also suggest constellations that would require some of the CS to leave another constellation that they are currently in.

To get information about the CS, the mediator needs to have data sharing agreements with the operating organizations of the CS. There also needs to be a formal language that allows these to express their capabilities in such a way that the mediator can match capabilities to needs. For this, an ontology than can express capabilities is needed. In all but the very simplest of SoS this ontology needs to be machine readable.

C. *Needs ontology*

Getting information about the needs requires the mediator to be connected to the users of the SoS. Sometimes, the users are the CS themselves – this is the case for instance in platooning. Here, it is up to the mediator to suggest possible platoons that enable the trucks to achieve benefits.

Note that to form the constellation, it might be necessary for some CS to incur costs before they can reap the benefits of the constellation. For platooning, this happens when some trucks may have to wait for others or take a detour in order to participate in the platoon. It is important to keep track of these costs, and then to share them amongst the members. For many SoS, it will be necessary to also have a mediator that provides cost sharing.

In other cases, the users are not themselves part of the capabilities that provide the solution. Consider for example a mobility SoS. Here, the CS are vehicles providing different means of transportations, e.g., taxis, public transport, e-scooters, car pool cars, privately owned cars, pods. The users are the people who require transportation. The task of the constellation forming mediator is to first collect the travel needs of people and the locations, plans, and other capabilities of transport units. Second, the mediator must find matchings that solve as many of the transportation needs as possible, with as small cost as possible.

As mentioned above the mediator must know the needs of the users. How can this be accomplished? Perhaps the easiest solution for this is for the mediator to also have a front-end that is the user interface of the SoS. However, this will not work in all cases – it could be the case that there are several services providing users with the ability to express their needs, just as there could be many different constellation forming mediators that match needs to sets of capabilities. Examples of this are provided by the sharing economy. A homeowner could list their home as available for renting in several different services, each of which performs a matching of the home with the needs of their users.

Hence, it is necessary to have an ontology also for expressing the needs of the users. Note that this ontology does not have to be the same as the capabilities ontology – it is enough that the constellation forming mediator understands both.

In Figure 6 we show an overview of the entities and relations needed for constellation forming as applied to an urban mobility scenario. In future work, we intend to expand this into a full description of the constellation level of a SoS ontology.

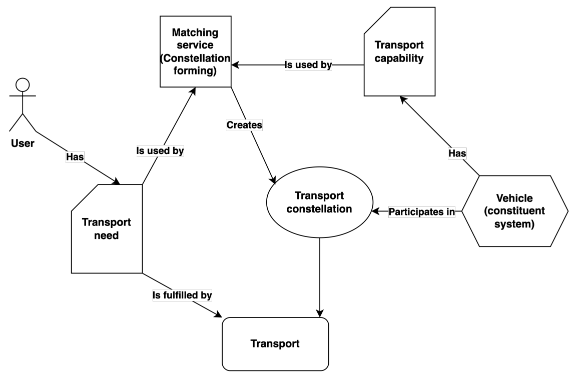

Figure 6. Entities and relations that are needed to accurately describe the constellation forming process, here shown for a mobility SoS. The users of the SoS need to express their needs. The needs are matched to the capabilities of the constituent systems by the matching service, which creates constellations that fulfills the transport needs. Just finding possible constellations is not enough: the mediator should also do some optimization to

D. Optimization

Just finding possible constellations is not enough: the mediator should also do some optimization to ensure that the suggested constellations are good. This optimization problem can in general not be solved exactly, instead an approximate optimization must be used. For many SoS, the constellation forming problem can be naturally divided into two aspects:

- Determining what set of capabilities are required to solve the problem instance;
- Finding a set of CS that together have these capabilities and are also available to form a constellation.

For many SoS where there is a geographical distribution of CS, the distance between the appropriate CS will be the most important factor. In order to provide good enough service to the users of the SoS, it might then be necessary to have a surplus of CS. Consider for instance the mobility SoS. The time needed for a user to get a transport will depend on the number of available CS as well as their geographical distribution. If the SoS designers want to impose a requirement on the maximum time needed before a user can get a transportation, they will need to compute the number of surplus CS and their geographical distribution. This, too, is a difficult optimization problem. In many cases, it might also be necessary to compensate the CS operators who provide the surplus CS in some way. While this is particularly evident for SoS designed for crisis management and response, as we have seen it can also be necessary in mobility SoS.

In addition to approximate solutions of the optimization problems, it could also be possible to use AI techniques such as case-based reasoning or learning as a component in the constellation formation mediator.

V. Conclusions and Future Work

In this paper we discussed the formation of constellations within a SoS, where each constellation is intended to solve a problem instance from the problem domain of the SoS. We discussed the conflicting goals of CS and described how a constellation forming mediator must use ontologies to match a set of CS to the capabilities needed to solve a problem instance. We used examples from three different SoS to illustrate the concepts. Along the way we introduced the concepts of SoS, constellation and CS level tasks, and discussed their relations and discussed the need for additional mediators to ensure that users' needs are taken account of in the constellation forming.

In future work, we intend to continue the discussion of SoS concepts, formalize the ontology needed to match capabilitis to needs, as well as dwell deeper into the optimization aspects of constellation formation.

References

[1] M. W. Maier, "Architecting Principles for Systems-of-Systems," INCOSE Int. Symp., vol. 6, no. 1, pp. 565–573, Jul. 1996.

[2] J. Axelsson, "A Refined Terminology on System-of-Systems Substructure and Constituent System States," in *IEEE Systems of Systems Conference*, pp. 31–36, 2019.

[3] Martin, J. 2004. "The Seven Samurai of Systems Engineering: Dealing with the Complexity of 7 Interrelated Systems." Proceedings of the International Council on Systems Engineering (INCOSE) International Symposium, 2004, Toulouse, France.

[4] J. Axelsson and P. Svenson, "On the Concepts of Capability and Constituent System Independence in Systems-of-Systems", in IEEE Systems of Systems Engineering Conference 2022

[5] J. Axelsson, "An initial analysis of operational emergent properties in a platooning system-of-systems," in 12th Annual IEEE International Systems Conference, SysCon 2018 - Proceedings, 2018, pp. 1–8, doi: 10.1109/SYSCON.2018.8369506.

[6] J. Axelsson, "Business Models and Roles for Mediating Services in a Truck Platooning System-of-Systems," in IEEE Systems of Systems Conference, 2019, pp. 113–118, doi: doi:10.1109/SYSOSE.2019.8753887.

[7] P. Svenson and J. Axelsson, "Should I Stay or Should I Go? How Constituent Systems Decide to Join or Leave Constellations in Collaborative SoS", in IEEE Systems of Systems Engineering Conference 2021, doi: 10.1109/SOSE52739.2021.9497474

[8] G. Lewis, E. Morris, S. Simanta, and D. Smith, "Service orientation and systems of systems", IEEE Software 28, 58–63 (2011).