software

Article

Defining and Researching "Dynamic Systems of Systems"

Rasmus Adler *, Frank Elberzhager, Rodrigo Falcão and Julien Siebert

Fraunhofer Institute for Experimental Software Engineering IESE, Fraunhofer-Platz 1, 67663 Kaiserslautern, Germany; frank.elberzhager@iese.fraunhofer.de (F.E.); rodrigo.falcao@iese.fraunhofer.de (R.F.); julien.siebert@iese.fraunhofer.de (J.S.)
* Correspondence: rasmus.adler@iese.fraunhofer.de

Abstract: Digital transformation is advancing across industries, enabling products, processes, and business models that change the way we communicate, interact, and live. It radically influences the evolution of existing systems of systems (SoSs), such as mobility systems, production systems, energy systems, or cities, that have grown over a long time. In this article, we discuss what this means for the future of software engineering based on the results of a research project called DynaSoS. We present the data collection methods we applied, including interviews, a literature review, and workshops. As one contribution, we propose a classification scheme for deriving and structuring research challenges and directions. The scheme comprises two dimensions: scope and characteristics. The scope motivates and structures the trend toward an increasingly connected world. The characteristics enhance and adapt established SoS characteristics in order to include novel aspects and to better align them with the structuring of research into different research areas or communities. As a second contribution, we present research challenges using the classification scheme. We have observed that a scheme puts research challenges into context, which is needed for interpreting them. Accordingly, we conclude that our proposals contribute to a common understanding and vision for engineering dynamic SoS.

Keywords: dynamic systems of systems; research challenges; classification

check for updates

Citation: Adler, R.; Elberzhager, F.; Falcão, R.; Siebert, J. Defining and Researching "Dynamic Systems of Systems". *Software* **2024**, *3*, 183–205. https://doi.org/10.3390/software3020009

Academic Editor: Sanjay Misra

Received: 12 March 2024
Revised: 12 April 2024
Accepted: 18 April 2024
Published: 1 May 2024

1. Introduction

Digital transformation is advancing across industries, enabling products, processes, and business models that radically change the way we communicate, interact, and live together. Systems, players, and markets that were once isolated are now being connected through digital platforms and form novel digital ecosystems. Software-based systems are at the heart of this evolution, adding value and delivering the desired system characteristics.

Existing digital ecosystems already consist of complex networks of interconnected and globally distributed applications. Rapid advances in miniaturization, storage capacity, intelligent information exchange, and processing enable the digitization of more and more diverse objects. Vision papers in many domains—be it production-as-a-service in Industry 4.0, autonomous transport systems in intelligent mobility, or cyber–physical systems in digital health—indicate that the future of software engineering will have to deal with socio-technical systems that are increasingly large and diverse (i.e., systems of systems), complex in scale and dynamics (i.e., complex systems), involve a growing number of actors from different organizations, and have a large-scale impact on society (i.e., socio-technical systems). In this paper, we refer to such types of systems as dynamic systems of systems (dynaSoS).

Today, systems with characteristics similar to dynaSoS already pose technical challenges, particularly in terms of managing their quality and complexity, and these challenges will intensify with the technical advances of and the growing demands on these systems. Moreover, as dynaSoS can be large-scale (national, transnational, or even global) socio-technical systems, they can directly affect societies (think, for example, of the propagation of information—true or false—on social networks, or the management of a continental energy network) and the environment (such as the impact cloud computing is having on greenhouse gas emissions [1]). While there are various challenges, there are also opportunities: dynaSoS harbor the potential to resolve tensions between pressing ecological, social, and economic challenges, not least because of their ability to integrate information from various sources at large scales.

DynaSoS imply an evolution of the existing software and systems engineering approaches to ensure reliable and secure operation despite their high complexity. These approaches must not only be able to handle the complex interactions of technical subsystems but also the interactions between technology, people, and the environment. In this context, the publicly funded project "DynaSoS" (BMBF, grant no. 01IS211, https://dynasos.de/, last accessed 1 April 2024) examined the evolution of the current state of software engineering to support digital transformation in the area of systems of systems (SoSs) development. The project objective is expressed as follows:

> *To identify and classify research challenges with respect to dynaSoS from the point of view of the researchers in the context of the project DynaSoS.*

The researchers had to deal with the issue that the term "dynaSoS" was not defined in the beginning of the project and that the investigation of an undefined or abstract topic tends to deliver results that are subject to significant threats to validity. A major threat is that the results depend a lot on the involved people and their background and related subjective opinions. We minimized subjective interpretation of the term "dynaSoS" by involving many diverse stakeholders from different sectors and deriving their consensus about the required (digital) transformation of existing systems of systems. We used this intersubjective purpose-oriented perspective to sharpen the notion of "dynaSoS"; to relate this novel notion to the popular notion of SoSs presented in 1998 in [2]; and to identify relevant research fields, challenges, and directions.

In this article, we provide two contributions:

C1 The main contribution (C1) is a classification scheme for reasoning about dynaSoS and for deriving and structuring research challenges and directions. The scheme comprises two dimensions: scope and characteristics. The scope refers to trends like that toward an increasingly connected world and sketches a vision explaining why these trends make sense. The characteristics enhance and adapt established SoS characteristics in order to include novel aspects and to better align them with the structuring of research into different research areas and research communities.

C2 A further contribution (C2) is the presentation of the identified research challenges along with the dynaSoS characteristics and proposals for the research directions.

This article presents the major results of the DynaSoS project. We organized this paper as follows: Section 2 presents the research method we used to collect the research challenges and shows that the results are based on comprehensive and sound analyses. Section 3 introduces the dynaSoS classification scheme (C1). Section 4 presents the research challenges by means of the scheme (C2). In Section 5, we discuss our findings and give some indications on where we see promising and probable future research for dynaSoS. Section 6 presents the related work. Finally, Section 7 concludes this article by discussing the limitations and provides an outlook on the future of dynaSoS.

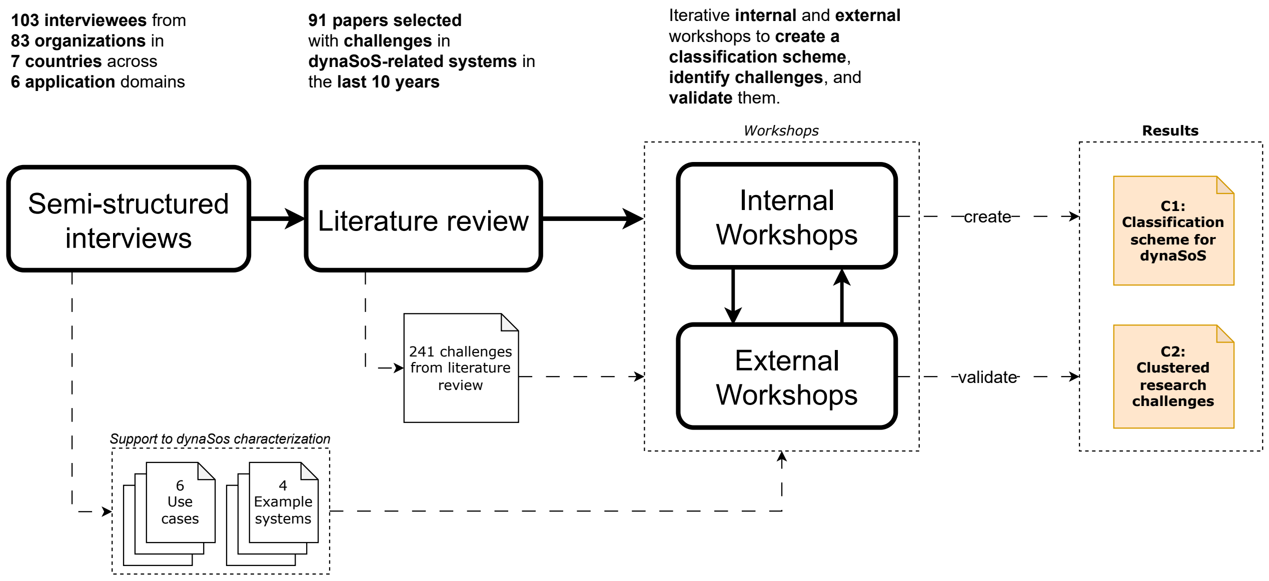
2. Method

We performed successive surveys to identify and classify dynaSoS challenges. Our data collection methods comprised interviews, a literature review, and workshops:

- The interviews worked as a preliminary phase to help us characterize dynaSoS through use cases and example systems.
- The literature review supported the collection of the challenges in dynaSoS.
- Benefiting from the results of the interviews and the literature review, we performed several workshops iteratively to create a classification scheme for dynaSoS and use it to classify the identified research challenges. The internal workshops predominantly

created the classification scheme for dynaSoS and clustered the identified challenges, whereas the external workshops helped improve the results by adding to, reviewing, and validating them.

Figure 1 provides an overview of the methods used. Each part will be described in the following subsections.



103 interviewees from 83 organizations in 7 countries across 6 application domains
91 papers selected with challenges in dynaSoS-related systems in the last 10 years
Iterative internal and external workshops to create a classification scheme, identify challenges, and validate them.
Workshops
Results
Semi-structured interviews
Literature review
Internal Workshops
External Workshops
create
validate
C1: Classification scheme for dynaSoS
C2: Clustered research challenges
241 challenges from literature review
Support to dynaSos characterization
6 Use cases
4 Example systems

Figure 1. Research method.

2.1. Interviews

To characterize dynaSoS, we interviewed experts in industry and academia. The purpose was twofold: On the one hand, we wanted to describe concrete use cases for dynaSoS in different domains in order to find domain-independent characteristics of these systems; on the other hand, we aimed to enable the description of example dynaSoS in order to make their properties more tangible and differentiate them more clearly from the traditional properties of SoSs [2,3]. We organized ourselves into seven teams of researchers, with six of them focusing on specific application domains and one focusing on cross-cutting aspects of dynaSoS. We prepared an interview questionnaire and each team conducted semi-structured interviews lasting approximately 60 min with the participants. All the interviews were conducted online via video conferencing. The interviewees were invited via e-mail through our network of partners in industry and academia. Participation was voluntary.

In total, we interviewed 103 people from 83 organizations in seven countries. Table 1 shows the frequency of organizations per type and the frequency of the roles of the individuals of these organizations. Most of the organizations are from Germany (72), followed by Austria (3), Sweden (3), Switzerland (2), Estonia (1), Singapore (1), and The Netherlands (1). Participants came from different application domains: Smart City and Region (23), Smart Health (21), Smart Mobility (17), Smart Farming (10), Smart Energy (9), and Smart Manufacturing (8). Fifteen participants were not bound to any particular domain, so they were interviewed by the cross-cutting analysis team.

The results of each interview were consolidated within the respective teams and served as a basis for the description of example systems in each application domain. The teams working in each application domain were free to choose the appropriate format of these artifacts, which differed due to differences in the level of abstraction: While the Smart Mobility and Smart City and Region scenarios take a bird's eye view of a metropolitan setting, Smart Farming and Smart Manufacturing consider a specific application or production site with selected machinery. As the description of each individual example system would go beyond the scope of this article, they can be found on the project website [4].

Table 1. Frequency of organization type (N = 83 organizations) and frequency of roles per organization type (N = 103 individuals).

Organization (N = 83)			Role (N = 103)					
Type	**Detail**	**%**	**Manager**	**Project/ Team Leader**	**Employee**	**Director**	**Professor**	**Other**
Industry	Companies	43.4%	13	11	13	5	0	4
	Startups	3.6%						
	Infrastructure	2.4%						
Research	Institutes	21.7%	10	12	4	12	8	0
	Universities	16.9%						
Public administration	Public authorities and government	6.0%	6	0	0	0	0	0
Associations	Associations, unions	6.0%	1	3	0	1	0	0

2.2. Literature Review

In order to identify concrete research challenges, we conducted a literature review covering the research challenges for dynaSoS-related systems, such as cyber–physical systems, IoT-based systems, adaptive systems, complex systems, and autonomous systems. We used Scopus [5] as the search engine, as it indexes publications from more than 7000 publishers [6]. We designed the search string to embrace three aspects in the results: SoSs, challenges, and dynamism. Furthermore, we restricted the search to results from the last ten years. Table 2 shows the terms we used to parameterize the search engine.

Table 2. Terms used to parameterize the search engine.

Aspect	**Terms**
Systems of Systems	TITLE (“systems of systems” OR “multi agent” OR “agent oriented” OR “agent based” OR “IoT” OR “internet of things” OR “Cyber Physical Systems” OR “CPS” OR “complex adaptive systems” OR “adaptive systems” OR “complex systems” OR “digital twin”)
Challenges	TITLE ({research agenda} OR “opportunities” OR “roadmap” OR {research roadmap} OR “issues” OR “challenges” OR “vision” OR “trends”)
Dynamism	TITLE-ABS-KEY(dynamic)
Last 10 years	PUBYEAR >2012

The search engine returned 160 documents, on which two researchers performed a series of selection steps independently, according to the following criteria:

- Inclusion criteria: Documents containing concrete challenges or research questions concerning SoSs.
- Exclusion criteria: Proceedings, documents not written in English, and primary work that presents solutions and evaluations.

The first selection step was to analyze the title of the document. The documents that were not excluded in the first step went to the second step, an analysis of the abstract; the remaining selected documents were then submitted to the third step: reading of the introduction and conclusion and document screening. After the third step, the two researchers compared their results. All documents that were accepted by at least one researcher through the three steps were used in the extraction phase. In total, 91 documents remained after the selection procedure.

In the extraction phase, the 91 selected papers were divided among three researchers, each of whom extracted the following information from the documents: research challenges, research questions, and research agendas. In total, 241 challenges were identified from the 91 papers. These challenges were organized into 58 groups of challenges based on similarity and redundancy.

These groups of challenges identified in the literature review were one of two inputs we utilized to identify and classify research challenges for dynaSoS; the second input came from several workshops we performed (see Section 2.3).

2.3. Workshops

We performed several workshops with two main purposes: On the one hand, we wanted to create and refine a classification scheme for dynaSoS; on the other hand, we aimed to identify more challenges that might not have been captured in the literature review and organize them all into the classification. Therefore, the definition of the classification scheme was an iterative process that took place throughout the workshops.

There were internal and external workshops. External refers to workshops where individuals who were external to our institution participated. Although all the activities were performed in both internal and external types of workshop, the former put more emphasis on the creation of the classification scheme and identification of the challenges, whereas the latter contributed more to the validation of the classification and challenges. In total, we performed six external workshops. Table 3 shows the demographics of the seventeen external participants by organization type and role.

Table 3. Participants of the external workshops by organization type and role.

Type	% (N = 17)	Manager	Project/Team Leader	Employee	Director	Professor	Other
Industry	41.2%	1	2	3	1	0	0
Research	47.0%	1	0	3	0	4	0
Public adm.	11.8%	0	1	1	0	0	0

3. A Classification Scheme for DynaSoS

The interviews provided us with many results showing how dynaSoS can contribute to urgent real-world challenges such as climate change and that their continuous engineering raises many software-related research challenges. Many of these challenges were found in the literature review. However, both the interviews and the literature review did not provide a big picture that illuminates novelties from different perspectives. What is actually novel considering that research addressing SoSs has a long history? Are these novel trends actually required to handle urgent challenges such as climate change or are they nonsensical digitalization madness? Why are these novelties challenging from a software engineering perspective? How do we structure these software-related challenges? How do we relate them to challenges of other research disciplines?

With the aim of providing a comprehensive understanding of dynaSoS, we have structured the results and derived a classification scheme. This scheme not only offers a big picture of dynaSoS but also supports discussions about it. It defines the characteristics of dynaSoS, helps to reflect on the evolution of current software systems, and organizes research challenges related to dynaSoS in a systematic manner. The classification distinguishes between scope and characteristics, which are detailed in the following sections.

A visual overview of the scheme is provided in Figure 2. The arrows show how the dynaSoS characteristics map to traditional SoS characteristics [2,3].

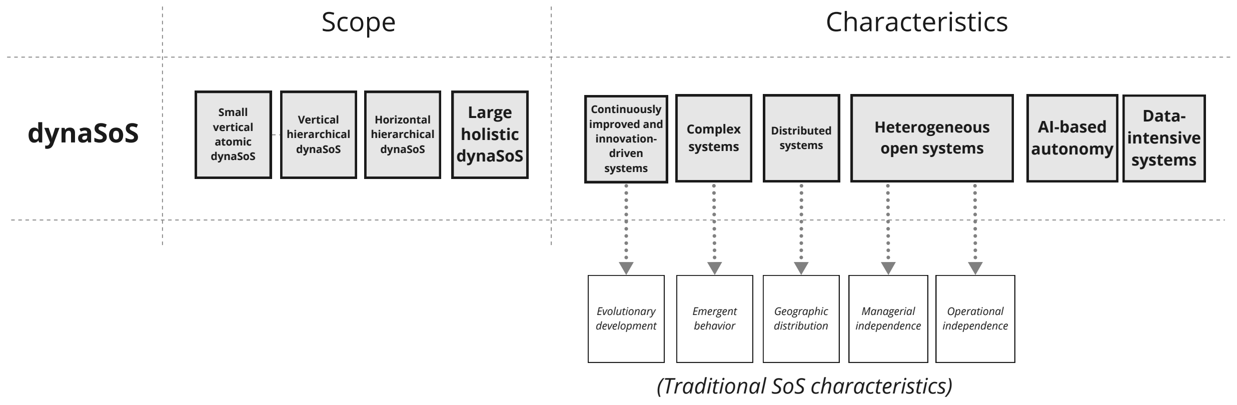


Figure 2. Scope and characteristics of dynaSoS, and their relationship with classical SoS characteristics.

3.1. Scopes of DynaSoS

The scope is characterized by three things: the geographic scale of the dynaSoS, the structure of the dynaSoS, and the applicability of the dynaSoS. The scale can vary from local to global systems; the structure can be atomic or hierarchical; its applicability vertical or horizontal.

Hierarchy level, applicability, and geographic scale are loosely related to each other. Atomic dynaSoS tend to be local and vertical. Hierarchical dynaSoS can be local and vertical, but the higher the hierarchy level, the more likely the dynaSoS is horizontal and the more likely it has a wider geographic distribution. According to these loose dependencies, we roughly distinguish between four kinds of dynaSoS, which are illustrated in Figure 3 and described in the following. On the left of the figure, we can consider, for example, collaborating robots and drones that together form a dynaSoS. The robots and the drones are the constituent systems (CSs) of the dynaSoS. If none of the CSs in a dynaSoS is a dynaSoS itself, we call the dynaSoS *atomic*. Otherwise, we call it *hierarchical* and refer to *hierarchy levels* for the depth of its nested structure. If a dynaSoS belongs to a single application domain such as mobility or energy, we call it *vertical*. Otherwise, we call it *horizontal*. Finally, we use terms such as local, regional, national, supranational, and global to describe the geographic distribution of a dynaSoS.

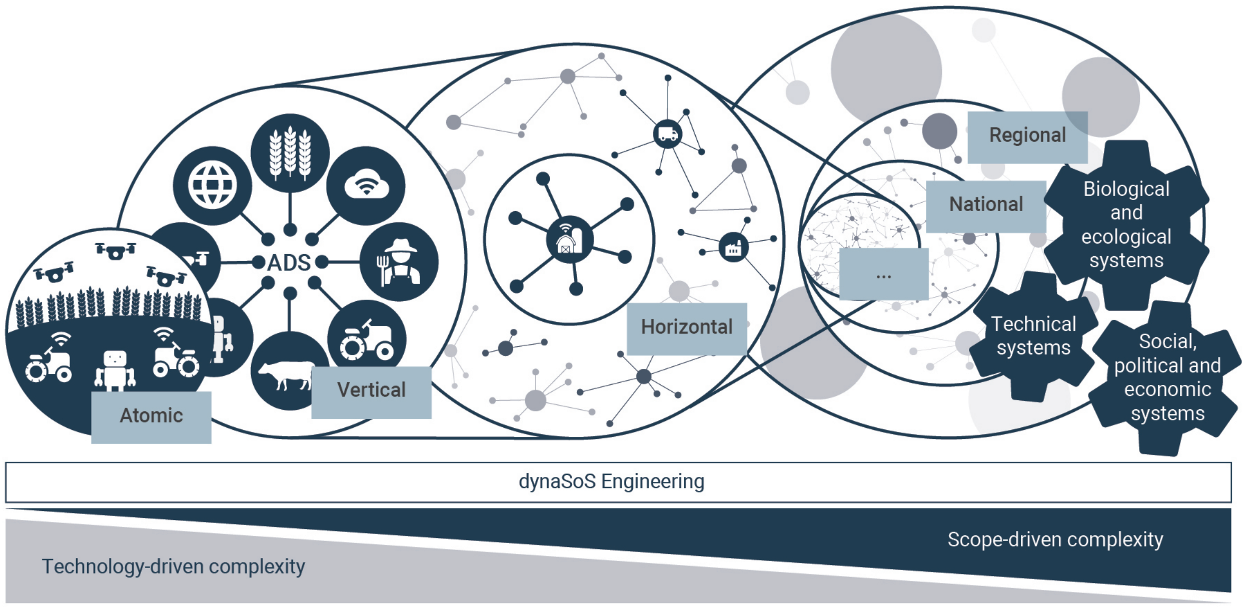


Figure 3. Illustration of different scopes of dynaSoS.

3.1.1. Small Vertical Atomic DynaSoS

Small atomic vertical dynaSoS are composed of some (autonomous) devices such as robots, drones, or other machines that collaborate to implement a domain-specific task. In doing so, they can dynamically adapt to various context conditions. This dynamism is often enabled by AI and not considered in conventional SoSs. It can be used for various purposes, but the general purpose is to maximize efficiency by following the rules of economy and the constraints of regulation. As sustainability is increasingly reflected in regulatory constraints and economic drivers, many of these dynaSoS aim at sustainable development. These dynaSoS might not evolve from existing SoSs and thus not fulfill the SoS characteristic "evolutionary development". Furthermore, they might not fulfill the SoS characteristic "managerial independence", meaning that the CSs are generally individually acquired and integrated and maintain a continuing operational existence that is independent of the SoSs.

Consider, for example, a swarm of field robots that can collaborate when weeding. The CS of these SoSs might be developed from scratch because they did not previously exist. For instance, a farmer may have only conventional agricultural machinery but no field robots that are able to collaborate. A manufacturer of agricultural machinery will likely develop collaborative field robots independent of existing conventional agricultural machinery because the latter have no collaboration capabilities. This is thus rather a development from scratch than an evolution of a SoS. Furthermore, the manufacturer might do this without considering collaboration with field robots from other manufacturers. This means that the SoS characteristic "managerial independence" might not be fulfilled.

3.1.2. Vertical Hierarchical DynaSoS

A vertical hierarchical dynaSoS involves various systems (including SoSs) from the same verticals. It collects information from its CS and dynamically influences their behavior. The novelty is the dynamism of the dynaSoS, which is enabled by the dynamism of the constituent dynaSoS, data spaces, AI, and information and communication technology (ICT). ICT enables the collection of more and/or better information. Existing SoSs in various verticals are transformed by introducing ICT and by using this information to dynamically control and adapt the processes in these verticals. Often, a lot of information needs to be processed within a very short time. This rapid processing typically goes beyond human skills and motivates the application of technology, including AI-based control or AI-based recommendations. Thus, ICT and AI are enablers for the transformation from SoSs to dynaSoS. ICT and AI were already known when Maier introduced the SoS characteristics in 1998 [2], but the technological possibilities have grown tremendously since then. Moreover, there is increasing awareness and political pressure for existing SoSs in domains such as energy, mobility, and agriculture to be transformed in order to achieve sustainability objectives [7]. We consider political systems responsible for implementing the sustainability goals for a sector as part of the vertical dynaSoS.

For instance, an Agricultural Data Space (ADS) (https://agridataspace-csa.eu/, last visited 1 April 2024) may collect information to support the decision about when it is time for weeding or irrigation. Based on this information, a service tells weeding robots or irrigation robots how to do their job. Similar data spaces or platforms are envisioned in other verticals. For instance, a smart grid requires a platform that organizes the matching between energy demands and offers or that trades the flexibilities of energy demands and offers. The vision of shared multimodal mobility requires various transportation means to be connected to a platform that organizes the matching between mobility demands and offers. These platforms and their related smart devices provide opportunities to achieve a sector's sustainability goals, such as the reduction in CO_2 emissions defined for the sector. Whether and how these technological opportunities will be used depends on political systems that generate constraints, rules, and incentives affecting technology transitions and user behavior.

3.1.3. Horizontal Hierarchical DynaSoS

A horizontal hierarchical dynaSoS involves various systems (including hierarchical dynaSoS) from different verticals. This can happen at different geographical scales. Increasing the scope in these directions requires dealing with more regulatory regimes because the regimes are organized along verticals and geographic areas (states/countries). We refer to this issue as scope complexity.

For instance, let us consider a city. A city integrates many systems that provide public services, such as mobility, energy, and water. The digital transformation may transform each of these systems into a hierarchical vertical dynaSoS, but the local optimization within these dynaSoS provides limited means to provide the best services for citizens with given resources. For this reason, a smart city breaks up the silos of its vertical dynaSoS and becomes itself a horizontal dynaSoS. As an example on a larger geographical scale, let us consider a global food supply chain. The supply chain integrates systems, such as farming systems, production and packaging systems, mobility systems, and so on. This integration refers to goals that all of these systems shall fulfill together. Apart from the goal to reliably deliver high-quality food, this includes sustainability aspects as defined in the German Supply Chain Act. Fulfilling these goals is the main function of the food supply dynaSoS. It dynamically chooses its CS with respect to this main function.

3.1.4. Large Holistic DynaSoS

The attributes large and holistic refer to two important aspects of popular notions of sustainability. One notion is that sustainability involves not only ecological aspects but also social and economic aspects. This notion provides a *holistic* perspective on the challenge of humans living on Earth for a very long time. It is closely related to the notion that sustainability focuses on the complete planet and living within planetary boundaries. These boundaries define the safe operating space for humans with respect to the Earth system and are associated with the planet's biophysical subsystems or processes [8]. Our vision is for dynaSoS to be engineered with respect to this safe operating space. A top–down engineering approach following this vision would be to define goals at the global level and break them down so that they can be allocated to some large and holistic dynaSoS. A prominent example of engineering sustainability at a global scale is given by the 17 UN Sustainability Development Goals (SDGs) [9]. These global objectives can be broken down into supranational objectives. For instance, the European Commission addresses the SDGs with the European Green Deal and other concepts [10]. These objectives are further broken down into nations and verticals, such as energy, transportation, production, and agriculture. Establishing meaningful global goals and breaking them down in a reasonable manner requires analyzing the interaction between biological/ecological systems, social/economic/political systems, and technical systems. These systems are loosely coupled, geographically distributed, evolve permanently, and generate emergent behavior. These properties relate to the SoS characteristics presented in [2,3]. Furthermore, the interaction between the systems is dynamic in nature, as described in [11]: "'Dynamics' refers to the patterns of complexity, interaction (and associated pathways) observed in the behavior over time of social, technological and environmental systems". The right part of Figure 3 illustrates the interaction between biological/ecological systems, social/economic/political systems, and the "mediating role of technology in altering and being altered by natural and social-political systems" [11]. Large and holistic dynaSoS refer to such ultra-large-scale systems.

3.2. DynaSoS Characteristics

An engineering dynaSoS requires coping with different kinds of complexity. An engineering small atomic dynaSoS requires coping with the complexity of the technical CS and the related emergent SoS behavior. Engineering large and holistic dynaSoS with respect to sustainability may require some technological innovations, but the main issues relate to ecology, politics, economy, and so on. With our classification, we want to provide a means to discuss the role of systems and software engineering in dealing with these different kinds of complexity. Systems and software engineering are obviously essential for coping with technology-driven complexity. Software drives innovations and systems engineering has the transdisciplinary role of combining software engineering with other engineering disciplines. The transdisciplinary role of systems engineering has a strong focus on technical disciplines but may involve non-technical disciplines. In this way, it can contribute to handling scope-driven complexity and to supporting reasoning about the interdependencies between technical systems, ecological systems, and socio-political systems. It utilizes systems thinking principles and systems theory to organize the body of knowledge from different stakeholders, disciplines, and domains. This systems-oriented approach motivated us to come up with dynaSoS characteristics in order to address the following questions: What characterizes dynaSoS? What do dynaSoS with different scopes have in common? How do these commonalities relate to research fields that are related to systems and software engineering? How do these commonalities relate to the five SoS characteristics that are well established in the literature: managerial independence, operational independence, evolutionary development, geographic distribution, and emergent behavior [2,3]? How do these commonalities relate to the increased usage and application of AI, autonomous systems, Big Data, and data science?

With these questions in mind, we came up with six characteristic aspects:

- Heterogeneous Open Systems (Managerial Independence and Operational Independence).
- Continuously Improved and Innovation-Driven (Evolutionary Development).
- Complex Systems (Emergent Behavior).
- Distributed Systems (Geographic Distribution).
- Data-Intensive Systems.
- AI-based Autonomy of Constituent Systems.

The first four dynaSoS characteristics relate to the five SoS characteristics mentioned in brackets. The last two dynaSoS characteristics highlight the role that Big Data, AI, and autonomy have in dynaSoS. The categories were chosen because they give rise to challenges that do not necessarily overlap and relate to research fields or research communities. We did not introduce the dynamic aspect as an isolated characteristic because dynamism comes with different notions stemming from the SoS characteristics or the situation-specific behavior of autonomous systems.

In the following, we will discuss each characteristic and our thoughts for introducing it.

3.2.1. Heterogeneous Open Systems (Managerial Independence and Operational Independence)

Two SoS characteristics are operational independence and managerial independence. This means that an SoS is composed of systems that operate independently and are managed by different organizations. This leads to heterogeneous open systems and the challenge to achieve interoperability for these systems. On the technological level, various hardware and software technologies may need to interact. On the organizational level, different parties may have to agree on common goals and to synchronize their processes while pursuing their particular goals, which may not always converge on the overarching goal of the SoS to which they belong. We call the cluster of these challenges "Heterogeneous Open Systems" because research communities rather refer to heterogeneity and openness than to managerial and operational independence. Operational independence contributes dynamics because a constituent system is not bound to an SoS. It is loosely coupled and can also contribute to other SoSs with its capabilities. This means that a CS can dynamically enter and leave an SoS. This notion of dynamism holds for all SoSs, not only for dynaSoS.

3.2.2. Continuously Improved and Innovation-Driven Systems (Evolutionary Development)

A third SoS characteristic is its evolutionary development. This means that the development is based on developments in the CS. These developments may take place asynchronously based on the independent development processes of the CS. Consequently, the SoS evolves incrementally instead of being "delivered" as normally envisioned in single system development or acquisition. We refer to this as "continuously improved" and "innovation-driven" because these terms are a better fit for the research communities. There is a research community around continuous software engineering, but their challenges and solutions are not limited to SoSs.

The continuous improvement of the CS is another notion of dynamism. All SoSs provide this kind of dynamism, but it plays a special role in dynaSoS. Continuous software engineering was not such a big topic when the SoS characteristics were introduced. Now, advances in continuous software engineering can shorten the time between incremental improvement steps.

3.2.3. Complex Systems (Emergent Behavior)

A fourth SoS characteristic is emergence. Emergence occurs when a system is observed to have properties that its constituent parts do not have on their own. Such emergent properties or behaviors arise from interactions between constituent systems. In addition, an emergent phenomenon affects its constituents: There is a feedback loop between the whole and its parts [12,13]. At least two levels of abstraction are needed to see, understand, and control emergent phenomena: the micro level, which describes the components, and the macro level, which describes the system as a whole. A traffic jam (e.g., on a motorway) is a simple example of an emergent phenomenon. The cars are the micro-level components. The traffic jam is the emergent phenomenon that happens at the macro level. A traffic jam can occur when cars interact and slow down, e.g., because of merging lanes or an accident. A traffic jam is a phenomenon with its own dynamics. It flows in the opposite direction of the cars and can spread in space (some can grow to several kilometers in length) and in time (sometimes even when the source of the slowdown is no longer present). Incoming cars are also affected by the traffic jam: They also have to slow down. Some emergent phenomena are desirable, while others are unwanted. For instance, an intelligent traffic management system may generate emergent behavior that optimizes traffic flow, but other phenomena may lead to a traffic jam. We refer to complexity and not to emergence, because this term fits better to the research community that studies emergent phenomena.

3.2.4. Distributed Systems (Geographic Distribution)

A fifth SoS characteristic is their geographic distribution. We already captured this aspect in one dimension of the dynaSoS scope. Considering relevant software-related research fields, geographic distribution hints at research around distributed systems. The CSs are physically located in different places and interconnected through communication networks. A major research question here is as follows: How do we use the execution platforms of the CS most efficiently but with assured quality? This question comes along with many notions of dynamism. The dynamic allocation of software functions to different CSs. The dynamic allocation of software functions within a CS. Static solutions have the advantage that they are easier to analyze at design-time. This supports quality assurance but limits efficiency, which could be better with more dynamism.

3.2.5. AI-Based Autonomy

There is no SoS characteristic that hints at increasingly autonomous CSs or autonomous hierarchical control of CSs. We consider these kinds of autonomy because many challenges arise from engineering and assuring autonomy. There is currently no clear research community addressing autonomy. Existing research communities rather try to address the challenges that autonomy brings for them. In the future, autonomy might become a discipline and research field of its own [14]. There are many definitions (e.g., [15–17]) involving different notions of autonomy [18]. One notion is that autonomous systems are not controlled or operated by a human operator. Therefore, they have to sense and understand the environment. Based on this context awareness, which can also include anticipation of future scenarios, they make decisions and act accordingly. This context awareness supports situation-specific behavior, which is another notion of autonomous behavior and directly related to dynamism. An autonomous system adapts dynamically to the current situation. A further notion is the "self" of an autonomous system. An autonomous system does not follow a few comprehensible simple rules but is based on learning or other techniques that make the behavior hard to predict. This leads not only to challenges with respect to quality assurance but also to descriptions such as "self-determined".

3.2.6. Data-Intensive Systems

There is no SoS characteristic that hints at data-intensive systems, but for many dynaSoS use cases, data are an essential aspect. Kleppmann defines an application as data-intensive "if data is its primary challenge—the quantity of data, the complexity of data, or the speed at which it is changing—as opposed to compute-intensive, where CPU cycles are the bottleneck" [19]. This is equivalent to the concept of Big Data. In simple terms, Big Data is a situation that occurs when conventional approaches to data processing (i.e., moving data from storage to the main computer memory for processing and then moving the results to storage) become virtually unfeasible or too expensive. Big Data can occur because the amount of data is simply too large to be processed by the computer at hand. It can also occur because the speed at which the data must be processed is too demanding [20,21].

4. Research Challenges

We use the dimensions scope and characteristics to classify the challenges of dynaSoS. In this section, we focus on the challenges clustered according to the dimension "characteristic" of dynaSoS.

4.1. Heterogeneity and Openness

The technical systems in a dynaSoS are potentially heterogeneous in any technical dimension, including, for example, hardware, software, networks, and protocols [22–24]. Finding common languages and ways to interoperate is part of the digitalization journey, and we already see interoperability artifacts, such as taxonomies, norms, standards, or reference architectures (like AUTOSAR, https://www.autosar.org/, last visited 1 April 2024; RAMI 4.0, https://www.plattform-i40.de/IP/Redaktion/EN/Downloads/Publikation/rami40-an-introduction.html, last visited 1 April 2024; or the Smart Grids Architecture Model (SGAM), https://energy.ec.europa.eu/smart-grid-reference-architecture_en, last visited 1 April 2024) that help overcome heterogeneity issues. Such standardization activities help to find a common language for communication between engineers, communication between technical systems, and communication between humans and technical systems.

There are many interoperability challenges, even if we only consider the simplest case where a small dynaSoS is developed by a single company. First, it is challenging to assure that contextual data are interoperable so that all systems have a sufficient understanding of the scenario in order to achieve the objective of the dynaSoS. Proper decision-making requires correct context information, but confidence in correctness is always limited. For this reason, the sender of information has to communicate the level of confidence in the correctness of the information so that the receivers can take this into account when making decisions or deriving information and determining the confidence in the correctness of this derived information. This issue is addressed by Conditional Safety Certificates (ConSerts) and Digital Dependability Identities (DDIs) (https://real-time-conserts.feuerberg.iese.fraunhofer.de/opus/overture.html, last visited 1

April 2024). ConSerts and DDIs have reached a high technology readiness level, but their adoption by industry is pending. A major reason why such interoperability solutions with a high technology readiness level are still the state of the practice is that the collaboration features of a product are only valuable for a customer if enough other products in the field have related collaboration features. For instance, a tractor that provides an open interface for its control only makes sense if there are some tractor implements in the field that use this interface.

In addition to direct collaboration between increasingly autonomous machinery, larger vertical dynaSoS also require indirect collaboration via (cloud-based) infrastructure and data spaces and the integration of the knowledge provided by sector-specific data spaces. This is also reflected in the roadmap in [25], which presents an evolution of shared context awareness in the context of the cloud and infrastructure (cf. dimension "Cloud and Infrastructure" in Figure 1 of [25] and related sections). One challenge in the context of knowledge integration are possible mechanisms that can query data across different formats and structures [26]. Ideas toward modeling and meta-modeling raise questions about the adequate level of the abstraction of these models—in order to prevent both oversimplification and overengineering—and bring with them computational challenges related to efficiency when we talk about large models [27]. As the details of each system are not or cannot be known in advance by the other systems, black-box integration via interface specifications is needed—which, in turn, poses a challenge for achieving interoperability [28].

For horizontal dynaSoS, cross-domain interoperability has to be established. This requires collaboration between sector-specific interoperability working groups, like those in the German Plattform Industrie 4.0, or the agrirouter (https://agrirouter.com/en/, last visited 1 April 2024) working groups.

For large and holistic dynaSoS, the limited scope of political systems is an issue. A political system can introduce strict (regulatory) interoperability rules to avoid that the local business-driven perspective from companies hinders interoperability solutions that are better from a more holistic perspective. However, the more holistic perspective of the political system might not be large enough, as it is limited to a certain region. Even though regulation is a powerful means to influence the evolution of SoSs, established SoS engineering guidelines like ISO/IEC/IEEE 15288 [29] consider laws and regulations as input for SoS engineering but not as something that can be adjusted.

4.2. Continuously Improved and Innovation-Driven

Continuously improving the systems in a small dynaSoS requires handling the asynchrony of changes [30,31]. Changes to one system can affect the behavior of the entire dynaSoS. The development teams responsible for different systems require means to ensure that changes are integrated smoothly and that there are no conflicts between different systems. This requires a clear understanding of the overall architecture and an effective communication strategy that enables teams to work together seamlessly [32,33]. Particularly in safety-critical domains, it is essential to have an effective change management strategy that enables changes to be integrated into the systems without compromising safety. Safety engineering artifacts like safety assurance cases [34,35] need to be updated when the system or its environment changes. Field feedback from experiments with demonstrators can be used for bootstrapping confidence in future safety based on past safe operation [36]. Furthermore, continuously improving the safety-related parts of systems is challenging from the perspective of regulation and certification. Regulations and safety standards demand all essential safety functions to be realized. If essential safety functions are realized by a dynamic composition of (autonomous) systems, then it is challenging to assure that the essential safety functions are always provided in sufficient quality. Research addressing this issue is referred to as runtime certification [37] or dynamic risk management [38].

Larger vertical dynaSoS involve cloud-based approaches and data spaces to enable collaboration between small dynaSoS. Continuous and innovation-driven development is common in this area [39] to improve the quality of software systems, but their adoption by an organization is challenging and needs time [39,40]. Moreover, cloud-based software is typically developed faster than embedded software. The different speed in development makes it more challenging to handle the previously mentioned asynchrony of changes.

Horizontal dynaSoS and holistic dynaSoS do not lead to additional software engineering challenges but may put into question whether an innovation is reasonable. From the limited scope of a sector or a region, a technology transition may be reasonable as side-effects on other sectors or other regions in the world are not taken into account.

4.3. Complexity

Small dynaSoS are complex systems even though they might not have many constituent systems. One reason for this is that the constituent systems can already be very complex. For instance, humans and autonomous robots are complex. It is hard to predict and explain how they behave in a particular situation. Another reason is that the number of possible interactions grows exponentially with the number of interconnected systems. It is therefore hardly possible to monitor all possible interactions and predict the resulting behavior of the dynaSoS. In addition, the effect of interactions can be non-linear. This means that small changes can have large consequences (the butterfly effect). For instance, small disturbances like a lane change of a single (autonomous) vehicle can cause a traffic jam several kilometers away [41]. Emergent phenomena and butterfly effects are unsolved issues when it comes to assuring dependability attributes, like safety. Leveson argues in [42] that for "the complex, software-intensive systems being created and operated today, non-linear accident causality models and analysis techniques are needed". In [43], the author proposes considering safety as a dynamic control problem. This approach and the related methods are the state of the art and the state of the practice for coping with the complexity of collaborative autonomous vehicles, robots, drones, and other kinds of machinery.

Larger vertical or horizontal dynaSoS are composed of complex systems, including, for instance, small dynaSoS, social systems, or ecological systems. When engineering larger vertical or horizontal dynaSoS, one has to cope with the behavior that emerges from the interaction of the (emergent) behaviors of these complex systems. For instance, the inadequate behavior of a smart grid may cause a power outage with severe consequences. Dealing with such harmful scenarios in large dynaSoS differs from handling harmful scenarios in small dynaSoS. In [44], we elaborated on this difference using a flood management system as an example. Similar modeling and analysis techniques might be applicable at every hierarchy level because non-linearity, feedback loops, and other features that contribute to emergent phenomena occur at all hierarchy levels. However, the skills and incentives of the stakeholders involved regarding the tailoring and application of such techniques differ. In his seminal book *The Logic of Failure* [45], Dietrich Dörner presents six modes of faulty behavior in coping with complex systems. One of them is insufficient coordination of different measures. The larger the scope of a dynaSoS, the more challenging this issue and the more this is a pure political and social issue. Systems and software engineering are rather a means to implement some (complex) measures, like the transition from a conventional energy system to a smart grid. In the domain of software engineering, complexity and emergent phenomena have been acknowledged, and shifts in the way systems and software are engineered and operated are already taking place [46]. Even if novel processes or development approaches are in place, failure management for complex systems remains a major challenge [47]. A major issue is that factors influencing the behavior of components might not be measurable directly. Assessing causal effects and finding root causes of problems in software systems is a long-standing problem, and many techniques and processes such as immutability, using a staging environment to test systems in conditions as close as possible to the real ones, A/B testing, and others have been developed to assess and improve software quality. In parallel, identifying causes and assessing causal effects in complex systems is a problem that research fields such as causal inference and causal discovery seek to address. These fields provide methods for identifying and evaluating causal effects even when the underlying data are not from randomized experiments or when confounding factors cannot be measured [48,49]. While some causal inference methods have found their way into software engineering [50], determining the root cause of quality in complex software systems remains a challenging aspect. Apart from such technical aspects, organizational aspects also need to be considered [46]. For instance, Spotify [51], Github [52], or Valve [53] have adopted a less hierarchical type of structure to manage complexity. These have been shown theoretically [54] to improve information flow and are thought to be one of the reasons why such organizations are better at engineering complex software. However, the question of how the complexity of the system being built affects organizations, their processes, and their missions is far from being resolved [55].

4.4. Distributed Systems

Small dynaSoS may involve constituent systems that are already distributed systems. For instance, Jo et al. [56] propose a distributed system architecture for an autonomous vehicle. Such a distributed architecture for a single (not geographically distributed) system addresses the internal feedback loop of an autonomous system but not the collaboration between autonomous constituent systems [14]. This collaboration will require new methods to manage interference, for example, in terms of timing, memory usage, or power consumption [57].

Larger vertical and horizontal dynaSoS involve cloud-based solutions and data spaces. Decreasing latency by means of new generations in mobile technology (5G and 6G) makes the computational power and memory of clouds attractive for many use cases. The application software of devices can be shifted to be executed in the edge or the cloud instead of running from devices to the edge or the cloud. This edge-to-cloud trend is complemented by a cloud-to-edge trend. Processing needs to be closer to where the data are produced in order to address latency, security, privacy, and environmental needs. One objective in this context is to achieve the cloud–edge–IoT computing continuum [58]. This provides additional means for distributed computing and increases the flexibility to execute software on available hardware resources in the cloud–edge continuum. On the one hand, the dynamic allocation of software to hardware resources can minimize the required resources and energy. On the other hand, it creates challenges for assuring critical system properties, like safety. Safety standards for single systems such as road vehicles or agricultural machinery provide guidance for dealing with failures of execution platforms for application software. The safety requirements for the execution platform depend on the criticality of the application software. Moreover, it needs to be assured that application software of lower criticality does not affect application software of higher criticality via shared resources for their execution. Fulfilling such requirements is challenging if approaches are applied that are common in distributed computing and cloud computing. For instance, containerization enables the deployment of multiple applications using the same operating system on a single virtual machine or server. If the applications have different levels of criticality, then it is challenging to assure freedom from interference requirements. On the other hand, these approaches enable the implementation of dynamic safety mechanisms. For instance, containerization can be used to run several copies of a safety-critical application and implement fault tolerance based on voting strategies. The number of replicas can be changed at runtime depending on the current criticality of the application. This is a promising approach, but common-cause failures need to be considered. The general research questions are as follows: Which conventional safety measures can be applied in the context of distributed computing and cloud computing? Which additional safety measures can be applied? How can one demonstrate that all applied safety measures are at least equally effective as conventional ones?

4.5. Data-Intensive Systems

One way of thinking about how data-related challenges might arise in dynaSoS is to look at the data lifecycle and the different activities taking place: generating, storing, exchanging, transforming, using, archiving, and finally deleting data.

In the case of external systems generating data, questions arise as to whether this information can be trusted, what the context and application for which these data were generated are, and whether it is transferable to another system operating in a different context and for a different purpose. It is also possible that the data sources provide their information in different modalities (e.g., text, images, videos, audio, etc.). Merging different data sources is known as data fusion [59]. Recent advances in deep learning have improved multimodal data fusion techniques, but this research and its applications are still in a preliminary stage [60].

Exchanging and sharing data leads to challenging aspects of data protection, data usage control, and data sovereignty: Who has what kind of data, who has access and for what purpose, and how are the data being used and exploited? The issue about who owns the data that are collected and exchanged has not been solved, and there is a legal vacuum regarding this topic [22]. Beyond the legal implications, there are also ethical concerns, especially when personal data are involved.

Finally, data end-of-life can be problematic. With the volume of data generated and stored increasing, a growing number of organizations are faced with the problem of deciding what data to archive and what to delete.

A key challenge across the data lifecycle phases is data quality. Data preparation is claimed to consume much of the time of any data-driven project. Data quality is a broad topic, and several standards have been devised to define it (e.g., ISO 8000 [61], ISO/IEC 25012 [62], or ISO/IEC 5259 [63]). There are basically two main sources of problems that can cause poor data quality. The first one is related to the way data are acquired, transformed, and stored. For example, the resolution of the initial measurements might be insufficient; raw data might be aggregated and transformed, losing some information on the way; and storage capacity may impose limitations on how much data can be kept and how good its resolution can be. This first set of challenges is common to every software system, whether data-intensive or not, but increases with the scale and complexity of the software system.

The second aspect relates to the intended use of the data and is particularly relevant to data-driven decision systems. Data capture only part of reality. This means that some relevant information may be missing. More importantly, a part of reality that is irrelevant to the problem at hand may be captured by the data. Some of the latest AI applications have recently been accused of not being fair and of perpetuating stereotypes. One example is the Word2Vec method, which transforms words into vectors. When trained on available data from news articles, it has been shown to reproduce gender stereotypes [64]. Research on algorithmic fairness, accountability, and transparency has grown in recent years (see, for example, work conducted and published at these conferences: https://facctconference.org/, last visited 1 April 2024), providing initial tools to avoid this kind of undesirable behavior in data-driven systems and also helping to identify and address related data quality issues, which have in turn been implemented in major cloud providers (see, for instance, https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-fairness-and-explainability.html, last visited 1 April 2024, or https://learn.microsoft.com/en-us/azure/machine-learning/concept-fairness-ml, last visited 1 April 2024). However, these problems are far from being solved and will be relevant for dynaSoS.

4.6. AI-Based Autonomy

The autonomy of the constituent systems in a small dynaSoS is typically achieved by applying approaches from the discipline of AI. For instance, sub-symbolic AI approaches like deep learning can be used for realizing the sensing part of a sense–plan–act architecture. A challenge in this regard is that the output of data-driven models is subject to uncertainty [65]. One approach to dealing with this uncertainty is to estimate it during operation and consider the estimated uncertainty in the subsequent planning of the behavior [66]. Symbolic AI approaches can represent knowledge that is required to realize the planning part. As shown in Figure 1 in the German AI standardization roadmap [67], the combination of learning methods and knowledge-based methods is a promising research direction. Last but not least, AI also deals with agent-based systems or the collaboration of autonomous systems. Consequently, AI addresses many functional aspects for developing small dynaSoS, but it does not consider safety and other dependability attributes. Accordingly, a series of scientific workshops have been launched in recent years to address this gap, e.g., the workshops WAISE, SafeAI, and AIsafety.

Larger vertical and horizontal dynaSoS may also involve autonomous control. For instance, a smart grid may autonomously control the flow of energy, an intelligent traffic management system may autonomously control the traffic flow, and a smart city may autonomously control the interaction between its smart grid and its intelligent traffic management system. Autonomous control is based on context awareness, including an understanding of how the current situation may evolve. The larger the scope of a dynaSoS, the more additional sources for context information are available. However, access to context information with sufficient quality is often the bottleneck when it comes to realizing autonomous control. For instance, it is even challenging to obtain high-precision indoor positioning [68], due to the accumulation of adequate data, context factor discovery, and filtering of contextual information [69]. Another challenge is the design of autonomous control because it is difficult to identify which of all the available contextual information is relevant for a certain task of interest, or how the individual contextual factors can be combined to describe a context-aware behavior [70]. A special aspect of context-aware behavior is the consideration of risks or possible harmful scenarios [71]. It is challenging to formalize risks and measure them because of their subjective nature. Accordingly, automated risk reasoning raises questions related to the ethics of risk. These questions have been studied in the context of automated driving [72] but not with respect to risks due to the emergent behavior of large horizontal dynaSoS, like a smart city.

5. Discussion

Based on the identified research challenges, we discussed in the project what might be concrete ideas for future topics regarding the engineering of dynaSoS. The discussion is still ongoing, but we would like to present some ideas where we currently see research directions. This list is neither complete nor on a very detailed level (a comprehensive organization of directions, impact, and time frame is available in [73] together with a detailed mapping of the research challenges and research directions), but it can provide some impressions and might serve as a baseline for future ideas and refinements. We organized our ideas for research directions into six clusters, as shown in Figure 4.

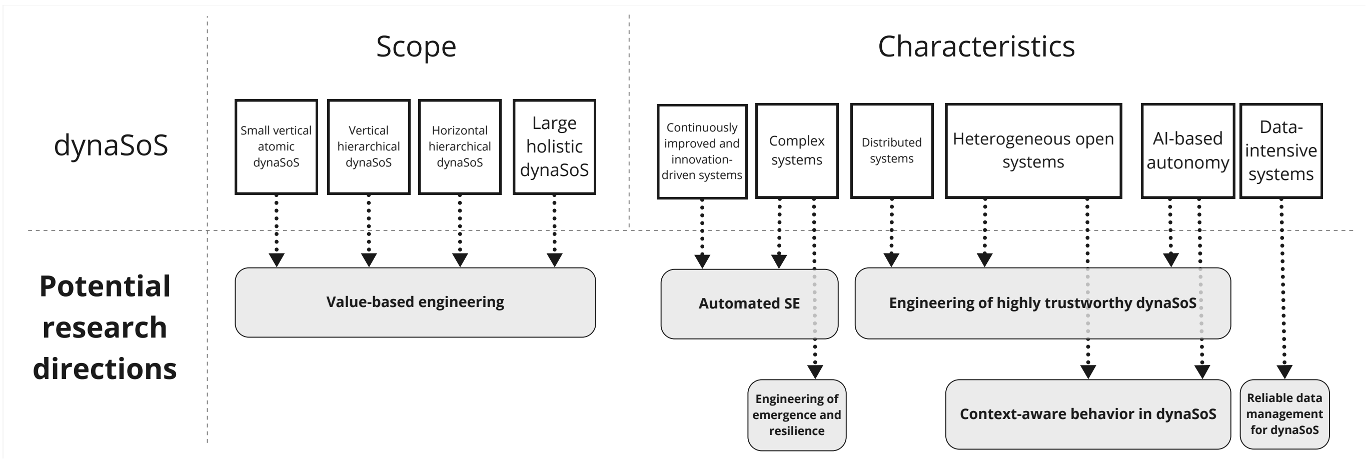

Scope
Characteristics
dynaSoS
Small vertical atomic dynaSoS
Vertical hierarchical dynaSoS
Horizontal hierarchical dynaSoS
Large holistic dynaSoS
Continuously improved and innovation-driven systems
Complex systems
Distributed systems
Heterogeneous open systems
AI-based autonomy
Data-intensive systems
Potential research directions
Value-based engineering
Automated SE
Engineering of highly trustworthy dynaSoS
Engineering of emergence and resilience
Context-aware behavior in dynaSoS
Reliable data management for dynaSoS

Figure 4. Scope and characteristics of dynaSoS and potential research directions.

5.1. Automated Software Engineering for dynaSoS

The automation of software engineering tasks has long been a goal of research and industry. The development of technologies (e.g., cloud, fog, edge, and serverless), the scope of systems to be developed, and advances in analysis methods influence each other and increase the need for research in this area. One direction is the research efforts in the area of automation of software engineering tasks (including, but not limited to, AI-based approaches) while ensuring that these new methods can be empirically validated in realistic configurations. Other important points include ensuring that the underlying analytics methods do not lead to a lock-in effect (e.g., only a few organizations can train very large deep learning text models, such as Microsoft, Nvidia, OpenAI, Facebook, etc.), considering and mitigating the underlying costs in terms of power consumption, and continuing research in terms of model privacy and explainability.

5.2. Reliable Data Management for dynaSoS

DynaSoS are data-intensive systems in which different organizations share information and in which added value is created from data from different sources. As such, dynaSoS require special attention to data management. The term “data management” encompasses several data-related aspects, such as data architecture, data acquisition, data storage, data quality, data integration, and data governance. In dynaSoS, large volumes of data are generated in different formats and qualities. This poses both technical and organizational challenges for the design, implementation, and operation of dynaSoS: On the one hand, composite systems are fundamentally dependent on data to implement their autonomous behavior and intelligence; on the other hand, the reliability of the data is impaired by the great heterogeneity and openness that characterize dynaSoS. Ways need to be found to design, implement, operate, and ensure reliable data for dynaSoS, thus making these systems more resilient to the open nature of the environments in which they operate. As possible research directions, we see investments in data acquisition methods optimized for dynaSoS, in the design and development of data architectures for dynaSoS, and in the exploration of appropriate data governance capabilities.

5.3. Engineering of Safe and Highly Trustworthy dynaSoS

The digital transformation of existing SoSs into dynaSoS is essential to counter ecological risks, such as climate change. However, it also brings with it complexity and other characteristics that contradict the well-known principles for avoiding and mitigating risks, such as “Keep it simple, stupid!” (KISS principle). Security research already offers some approaches for dealing with this conflict. We recommend supporting the further development of these approaches, their integration, and their application to other aspects of trustworthiness. This is particularly relevant for small atomic dynaSoS. To address larger and holistic dynaSoS, we recommend in [74] broadening the scope of safety research and integrating it with sustainability and complexity research.

5.4. Context-Aware Behavior in dynaSoS

Usage scenarios for dynamic systems of systems are often presented as smart scenarios, e.g., smart agriculture, smart health, smart mobility, and smart city. The “smartness” of these scenarios is not perceived by anyone but humans, and context awareness is often what is behind such smart behaviors. Therefore, it is realizing that the systems of the future require investments to address the current technical challenges associated with context-aware systems, which are amplified given the characteristics of dynaSoS. We see further research in the field of software engineering of context-aware systems, covering both design-time and runtime aspects.

5.5. Value-Based Engineering of dynaSoS

Economic rules and regulatory constraints determine which atomic dynaSoS are developed from scratch and how existing SoSs are transformed into dynaSoS. They must be consistent with the values of the societies or individuals concerned and create the right incentives for sensible change. Even though this open topic belongs to the realm of sustainability science, ethics, and regulatory science rather than computer science, systems and software engineers are increasingly confronted with profound societal and environmental issues, and software engineering has a role to play. Although ethics and sustainability have long been part of the engineering curriculum, they are still electives, and the training for practitioners is inadequate. The IEEE 7000 series of standards already addresses this topic, but there is a lack of concrete methods and tools for implementing its requirements. In particular, methods and tools that allow the potential sustainability effects of software systems to be investigated are still in their infancy. We therefore recommend accelerating research and application at the interface of systems and software engineering, ethics, and sustainability.

5.6. Engineering of Emergence and Resilience

DynaSoS are inherently complex, and their behavior depends on complex non-linear interactions at multiple levels. The development of such complex systems requires an appropriate mindset, methods, and tools. Recent advances in complexity science and the science of complex networks are generally applied to software development. For example, understanding the flow of information in complex networks such as Small Worlds has led some companies to shift their organizations to flatter structures to improve communication and value flow. However, it is unclear how ideas from complexity science are being incorporated into software development. Apart from the fact that they speak different languages, the challenge is that complexity science examines emergent phenomena through the lens of models that describe complex systems as a whole and looks at the problem at a macro level. Software engineering, on the other hand, is concerned with the design of components that are part of complex systems and must deal with the situation that the overall view of the whole is only partially available. We recommend continuing to support transdisciplinary projects in order to transfer methods from basic research on complex systems (e.g., from non-linear physics, network science, or the social sciences) to the field of software engineering.

6. Related Work

Several roadmaps and research agendas are related to our work. The roadmaps in [75–77] were developed in the context of EU projects. A similar methodology based on interviews and literature research was applied for deriving the roadmaps. The focus was on SoS engineering and [78] specifically focused on SoS requirements engineering. In contrast to that, we focused on additional aspects like the dynamism that dynaSoS have compared to conventional SoSs and the related role of data and AI.

In [79], the International Council on Systems Engineering (INCOSE) published a vision for the evolution of systems engineering until 2035. It presents challenges and recommendations related to SoS engineering. In contrast to this work, it focuses less on research and addresses the systems engineering community. We adopted the differentiation between technology complexity and scope complexity from INCOSE's vision 2035. We presented a more detailed discussion on scope complexity in [74]. The advanced systems engineering strategy presented in [80] outlines recommendations for government, industry, and science to advance systems engineering. However, the scope is limited to engineering within regulatory constraints, and regulatory science for developing reasonable regulation is not considered. In [33], a research roadmap addressing AI for systems engineering and systems engineering for AI is presented. It abstracts from the scope of engineered systems, like small vertical dynaSoS or large dynaSoS. Papers specifically focusing on architectures are presented in [81–87]. Our focus is broader, as we consider research fields related to properties of dynaSoS. The research agenda in [88] focuses on the modeling and simulation of dynaSoS.

A roadmap for ultra-large-scale systems is presented in [89]. The scope is limited to ultra-large-scale systems, which relates to horizontal dynaSoS. Small atomic dynaSoS and the relation between different scopes of dynaSoS are not considered.

The position paper from SafeTRANS on Safety, Security, and Certifiability of Future Man-Machine Systems [16] describes a hierarchy of cyber–physical systems (CPSs) by distinguishing between individual cyber–physical systems (CPSs), groups of CPSs, homogeneous collectives of CPSs, and heterogeneous collectives of CPSs. This hierarchy relates to the scopes of the dynaSoS that we presented in Section 3.1. One difference is that we focus more on the overall SoS with its properties and not only the technical CPS parts of the SoS. Furthermore, we have a broader scope and consider not only safety and security but also aspects like sustainability. The report in [16] describes the evolution of autonomous systems in different domains and how they grow together across domains. It focuses on opportunities and risks regarding the different application domains and, unlike our work, not on a literature review for describing the research landscape. The strategic research report in [90] considers SoSs as one out of four foundational technology layers and presents SoS challenges as well as a long-term vision for SoSs. The report was developed in parallel to our dynaSoS classification scheme. After its publication, we incorporated relevant results into our dynaSoS scheme.

7. Conclusions

Research addressing SoSs has a long history. Nevertheless, SoSs are also a core aspect of many current research agendas. Why is this the case? One possible answer is that there are many new research questions because the answer to one question typically generates several new questions. Advances in distributed ledger technology, AI, and other fields open up new opportunities for solving unsolved research challenges. Furthermore, there is an increasing need to apply SoS thinking in order to address sustainability issues.

We see climate change and other sustainability issues as major drivers for the (digital) transformation of existing SoSs. We proposed distinguishing between different scopes of SoSs in order to reflect on the evolution of different SoSs and the dependencies between them. Technology-oriented research roadmaps tend to focus on smaller SoSs where ecological and political systems are largely out of the engineering scope. This limits the opportunities to engineer sustainability, as the economic objectives of a small SoS might be in conflict with the ecological or social objectives of the larger SoS in which the small SoS is embedded. Research in the field of complexity science is not limited to this narrow technology-focused scope. As shown in the complexity map [91], it increasingly involves the application of complexity theory for managing challenging policy issues. We conclude that different scopes of dynaSoS demand the application of different research areas in order to understand and control the evolution of SoSs. With our hierarchy, we want to take a first step toward structuring the problem domain of SoS evolution and fostering collaboration of the different solution domains in order to deal with the dependencies of SoS evolution.

We focused on the solution domain of software and systems engineering. In order to structure this field with respect to SoS engineering, we related SoS properties like geographic distribution to research areas like distributed systems engineering. In doing so, we took into account that Big Data, AI, and autonomy strongly influence SoS engineering and relate to the dynamics of dynaSoS. Furthermore, we discussed challenges for each dynaSoS characteristic considering the different scopes that a dynaSoS can have. This structuring approach was the best of all the approaches we tried. We conclude that this approach might also be valuable for others who want to structure dynaSoS research. Moreover, we believe that the content that we put into the structure may help other researchers position themselves with their work and find related work.

Author Contributions: Conceptualization, R.A., F.E., R.F. and J.S.; methodology, R.A., F.E., R.F. and J.S.; validation, R.A. and R.F.; formal analysis, R.A. and R.F.; investigation, R.A., R.F. and J.S.; writing—original draft preparation, R.A., F.E., R.F. and J.S.; writing—review and editing, R.A., F.E., R.F. and J.S.; supervision, R.A. and F.E.; project administration, R.A. and F.E.; funding acquisition, R.A. All authors have read and agreed to the published version of this manuscript.

Funding: This research was funded by the German Federal Ministry of Education and Research (BMBF), grant no. 01IS21104.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Dataset available on request from the authors.

Acknowledgments: We thank all the people who participated in the interviews, the workshops, and the discussions leading to these results. We also thank Sonnhild Namingha for proofreading.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Gröger, J.; Liu, R.; Stobbe, L.; Druschke, J.; Richter, N. Green Cloud Computing: Lebenszyklusbasierte Datenerhebung zu Umweltwirkungen des Cloud Computing. Berlin, Germany, 2021. Available online: https://www.umweltbundesamt.de/sites/default/files/medien/5750/publikationen/2021-06-17_texte_94-2021_green-cloud-computing.pdf (accessed on 1 April 2024).
2. Maier, M.W. Architecting principles for systems-of-systems. *Syst. Eng. J. Int. Counc. Syst. Eng.* **1998**, *1*, 267–284. [CrossRef]
3. Gorod, A.; Sauser, B.; Boardman, J. System-of-systems engineering management: A review of modern history and a path forward. *IEEE Syst. J.* **2008**, *2*, 484–499. [CrossRef]
4. DynaSoS. Example Systems. 2022. Available online: https://dynasos.de/tag/example-systems/ (accessed on 3 April 2023).
5. Elsevier. Scopus—Document Search. 2023. Available online: https://www.scopus.com (accessed on 3 April 2023).
6. Elsevier. Scopus Content. 2024. Available online: https://www.elsevier.com/products/scopus/content (accessed on 1 April 2024).
7. European Commission, Directorate-General for Structural Reform Support, Niestroy, I. Managing the Implementation of the SDGs. Technical Report. Brussels, Belgium, 2021. Available online: https://data.europa.eu/doi/10.2887/949364 (accessed on 1 April 2024).
8. Rockström, J.; Steffen, W.; Noone, K.; Persson, Å.; Chapin III, F.S.; Lambin, E.; Lenton, T.M.; Scheffer, M.; Folke, C.; Schellnhuber, H.J.; et al. Planetary boundaries: Exploring the safe operating space for humanity. *Ecol. Soc.* **2009**, *14*, 2. [CrossRef]
9. UN. *Transforming Our World: The 2030 Agenda for Sustainable Development*; United Nations: New York, NY, USA, 2015.
10. European Commission. Delivering on the UN's Sustainable Development Goals—A Comprehensive Approach. Technical Report. Brussels, Belgium, 2020. Available online: https://commission.europa.eu/system/files/2020-11/delivering_on_uns_sustainable_development_goals_staff_working_document_en.pdf (accessed on 1 April 2024).
11. Scoones, I.; Leach, M.; Smith, A.; Stagl, S.; Stirling, A.; Thompson, J. *Dynamic Systems and the Challenge of Sustainability*; Technical Report; STEPS Centre: Brighton, UK, 2007. Available online: https://steps-centre.org/wp-content/uploads/final_steps_dynamics.pdf (accessed on 1 April 2024).
12. Siegenfeld, A.F.; Bar-Yam, Y. An introduction to complex systems science and its applications. *Complexity* **2020**, *2020*, 1–16. [CrossRef]
13. Parrend, P.; Collet, P. A review on complex system engineering. *J. Syst. Sci. Complex.* **2020**, *33*, 1755–1784. [CrossRef]
14. Saidi, S.; Ziegenbein, D.; Deshmukh, J.V.; Ernst, R. Autonomous systems design: Charting a new discipline. *IEEE Des. Test* **2021**, *39*, 8–23. [CrossRef]
15. *ISO/IEC 22989:2022*; Information Technology—Artificial Intelligence—Artificial Intelligence Concepts and Terminology. Technical Report. International Organization for Standardization: Geneva, Switzerland, 2022. Available online: https://www.iso.org/standard/74296.html (accessed on 1 April 2024).
16. Kagermann, H.; Gaus, N.; Euler, K.; Hauck, J.; Beyerer, J.; Wahlster, W.; Brackemann, H. Fachforum Autonome Systeme im Hightech-Forum: Autonome Systeme–Chancen und Risiken Für wirtschaft, Wissenschaft und Gesellschaft. Technical Report, Berlin 2017. Available online: https://www.acatech.de/publikation/fachforum-autonome-systeme-chancen-und-risiken-fuer-wirtschaft-wissenschaft-und-gesellschaft-abschlussbericht/ (accessed on 1 April 2024).
17. Huang, H.E. *Autonomy Levels for Unmanned Systems Framework, Volume I: Terminology*; Version 2.0; NIST Special Publication 1011; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2008.
18. Adler, R.; Reich, J.; Hawkins, R. Structuring Research Related to Dynamic Risk Management for Autonomous Systems. In *Proceedings of the International Conference on Computer Safety, Reliability, and Security, 13 September 2023*; Springer: Cham, Switzerland, 2023; pp. 362–368.
19. Kleppmann, M. *Designing Data-Intensive Applications: The Big Ideas behind Reliable, Scalable, and Maintainable Systems*; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2017.
20. Laney, D. 3D data management: Controlling data volume, velocity and variety. *META Group Res. Note* **2001**, *6*, 1.

21. Das, B. An overview on big data: Characteristics, security and applications. *J. Netw. Commun. Emerg. Technol. (JNCET)* **2020**, *10*, 1–5.
22. Singh, S.; Shehab, E.; Higgins, N.; Fowler, K.; Tomiyama, T.; Fowler, C. Challenges of Digital Twin in High Value Manufacturing; Technical Report, SAE Technical Paper. 2018. Available online: https://saemobilus.sae.org/papers/challenges-digital-twin-high-value-manufacturing-2018-01-1928 (accessed on 1 April 2024).
23. Younan, M.; Houssein, E.H.; Elhoseny, M.; Ali, A.A. Challenges and recommended technologies for the industrial internet of things: A comprehensive review. *Measurement* **2020**, *151*, 107198. [CrossRef]
24. Li, F.; Lam, K.Y.; Li, X.; Sheng, Z.; Hua, J.; Wang, L. Advances and emerging challenges in cognitive internet-of-things. *IEEE Trans. Ind. Inform.* **2019**, *16*, 5489–5496. [CrossRef]
25. Damm, W.; Heidl, P. *SafeTRANS Roadmap on Safety, Security, and Certifiability of Future Man-Machine Systems*; SafeTRANS e.V: Oldenburg, Germany, 2021.
26. Diène, B.; Diallo, O.; Rodrigues, J.J.; Ndoye, E.H.M.; Teodorov, C. Data management mechanisms for IoT: Architecture, challenges and solutions. In Proceedings of the 2020 5th International Conference on Smart and Sustainable Technologies (SpliTech), Split, Croatia, 23–26 September 2020; pp. 1–6.
27. Uday, P.; Marais, K. Designing resilient systems-of-systems: A survey of metrics, methods, and challenges. *Syst. Eng.* **2015**, *18*, 491–510. [CrossRef]
28. Liu, Z.; Wang, J. Human-cyber-physical systems: Concepts, challenges, and research opportunities. *Front. Inf. Technol. Electron. Eng.* **2020**, *21*, 1535–1553. [CrossRef]
29. *ISO/IEC/IEEE 15288:2023*; Systems and Software Engineering–System Life Cycle Processes. Technical Report. International Organization for Standardization: Geneva, Switzerland, 2023. Available online: https://www.iso.org/standard/81702.html (accessed on 1 April 2024).
30. Bauer, T.; Antonino, P.O.; Kuhn, T. Towards architecting digital twin-pervaded systems. In Proceedings of the 2019 IEEE/ACM 7th International Workshop on Software Engineering for Systems-of-Systems (SESoS) and 13th Workshop on Distributed Software Development, Software Ecosystems and Systems-of-Systems (WDES), Montreal, QC, Canada, 28 May 2019; pp. 66–69.
31. Theobald, S.; Diebold, P. Interface problems of agile in a non-agile environment. In *Agile Processes in Software Engineering and Extreme Programming: Proceedings of the 19th International Conference, XP 2018, Porto, Portugal, 21–25 May 2018*; Proceedings 19; Springer: Cham, Switzerland, 2018; pp. 123–130.
32. Tisi, M.; Bruneliere, H.; de Lara, J.; Di Ruscio, D.; Kolovos, D. Towards Twin-Driven Engineering: Overview of the State-of-the-Art and Research Directions. In *Advances in Production Management Systems. Artificial Intelligence for Sustainable and Resilient Production Systems: Proceedings of the IFIP WG 5.7 International Conference, APMS 2021, Nantes, France, 5–9 September 2021*; Proceedings, Part I; Springer: Cham, Switzerland, 2021; pp. 351–359.
33. McDermott, T.; DeLaurentis, D.; Beling, P.; Blackburn, M.; Bone, M. AI4SE and SE4AI: A research roadmap. *Insight* **2020**, *23*, 8–14. [CrossRef]
34. Rushby, J. *The Interpretation and Evaluation of Assurance Cases*; Technical Report SRI-CSL-15-01; Computer Science Laboratory, SRI International: Menlo Park, CA, USA, 2015. Available online: https://www.csl.sri.com/~rushby/papers/sri-csl-15-1-assurance-cases.pdf (accessed on 1 April 2024).
35. Alves, E.E.; Bhatt, D.; Hall, B.; Driscoll, K.; Murugesan, A.; Rushby, J. *Considerations in Assuring Safety of Increasingly Autonomous Systems*; Technical Report NASA/CR–2018-220080; NASA, SRI International: Hampton, VA, USA, 2018. Available online: https://ntrs.nasa.gov/citations/20180006312 (accessed on 1 April 2024).
36. Bishop, P.; Povyakalo, A.; Strigini, L. Bootstrapping confidence in future safety from past safe operation. In Proceedings of the 2022 IEEE 33rd International Symposium on Software Reliability Engineering (ISSRE), Charlotte, NC, USA, 31 October–3 November 2022; pp. 97–108.
37. Rushby, J. Runtime certification. In *Proceedings of the Runtime Verification: 8th International Workshop, RV 2008, Budapest, Hungary, 30 March 2008*; Selected Papers 8; Springer: Berlin/Heidelberg, Germany; New York, NY, USA, 2008; pp. 21–35.
38. Schneider, D.; Trapp, M. B-space: Dynamic management and assurance of open systems of systems. *J. Internet Serv. Appl.* **2018**, *9*, 1–16. [CrossRef]
39. Ebert, C.; Hochstein, L. DevOps in Practice. *IEEE Softw.* **2022**, *40*, 29–36. [CrossRef]
40. Fitzgerald, B.; Stol, K.J. Continuous software engineering: A roadmap and agenda. *J. Syst. Softw.* **2017**, *123*, 176–189. [CrossRef]
41. Randelhoff, M. Die drei Haupttheoreme der Stauforschung: Der Schmetterlingseffekt, unsichtbare Wellen (=Phantomstau) und die Tragik des Zufalls. *Zukunft Mobilität*, 2011. Last Updated: 22 December 2017. Available online: https://www.zukunft-mobilitaet.net/3344/analyse/wie-entstehen-staus-phantomstau/ (accessed on 1 April 2024).
42. Leveson, N.G. *Shortcomings of the Bow Tie and Other Safety Tools Based on Linear Causality*; Technical Report; MIT: Cambridge, MA, USA, 2019. Available online: http://sunnyday.mit.edu/Bow-tie-final.pdf (accessed on 1 April 2024).
43. Leveson, N.G. *Engineering a Safer World: Systems Thinking Applied to Safety*; The MIT Press: Cambridge, MA, USA, 2012. [CrossRef]
44. Patel, A.R.; Haupt, N.B.; Adler, R.; Elberzhager, F.; Liggesmeyer, P. Exploring Safety Challenges in Dynamic Systems-of-Systems for Flood Management. In Proceedings of the 2023 18th Annual System of Systems Engineering Conference (SoSe), Lille, France, 14–16 June 2023; pp. 1–8.
45. Dörner, D. *Die Logik des Mißlingens: Strategisches Denken in Komplexen Situationen*; Rowohlt Verlag GmbH: Hamburg, Germany, 2011.

46. Jamshidi, P.; Pahl, C.; Mendonça, N.C.; Lewis, J.; Tilkov, S. Microservices: The journey so far and challenges ahead. *IEEE Softw.* **2018**, *35*, 24–35. [CrossRef]
47. Woods, D. STELLA Report from the SNAFU Catchers Workshop on Coping with Complexity. SNAFU Catchers Consortium, Downloaded Stella. Report. 2017. Available online: https://snafucatchers.github.io/ (accessed on 1 April 2024).
48. Glymour, C.; Zhang, K.; Spirtes, P. Review of causal discovery methods based on graphical models. *Front. Genet.* **2019**, *10*, 524. [CrossRef] [PubMed]
49. Pearl, J.; Mackenzie, D. *The Book of Why: The New Science of Cause and Effect*; Basic Books: New York, NY, USA, 2018.
50. Siebert, J. Applications of statistical causal inference in software engineering. *Inf. Softw. Technol.* **2023**, *159*, 107198. [CrossRef]
51. Smite, D.; Moe, N.B.; Levinta, G.; Floryan, M. Spotify guilds: How to succeed with knowledge sharing in large-scale agile organizations. *IEEE Softw.* **2019**, *36*, 51–57. [CrossRef]
52. Burton, R.M.; Håkonsson, D.D.; Nickerson, J.; Puranam, P.; Workiewicz, M.; Zenger, T. GitHub: Exploring the space between boss-less and hierarchical forms of organizing. *J. Organ. Des.* **2017**, *6*, 1–19. [CrossRef]
53. Möller, U.; McCaffrey, M. Levels without bosses? Entrepreneurship and valve's organizational design. In *The Invisible Hand in Virtual Worlds: The Economic Order of Video Games*; McCaffrey, M., Ed.; Cambridge University Press: Cambridge, UK, 2021.
54. Barabási, A.L.; Pósfai, M. *Network Science*; Cambridge University Press: Cambridge, UK, 2016.
55. Kuusisto, M. Organizational effects of digitalization: A literature review. *Int. J. Organ. Theory Behav.* **2017**, *20*, 341–362. [CrossRef]
56. Jo, K.; Kim, J.; Kim, D.; Jang, C.; Sunwoo, M. Development of autonomous car—Part I: Distributed system architecture and development process. *IEEE Trans. Ind. Electron.* **2014**, *61*, 7131–7140. [CrossRef]
57. Möstl, M.; Schlatow, J.; Ernst, R.; Dutt, N.; Nassar, A.; Rahmani, A.; Kurdahi, F.J.; Wild, T.; Sadighi, A.; Herkersdorf, A. Platform-centric self-awareness as a key enabler for controlling changes in CPS. *Proc. IEEE* **2018**, *106*, 1543–1567. [CrossRef]
58. European Commission. Investing in Cloud, Edge and the Internet of Things, 2023. Available online: https://digital-strategy.ec.europa.eu/en/policies/iot-investing (accessed on 1 April 2024).
59. Bleiholder, J.; Naumann, F. Data fusion. *ACM Comput. Surv. (CSUR)* **2009**, *41*, 1–41. [CrossRef]
60. Gao, J.; Li, P.; Chen, Z.; Zhang, J. A survey on deep learning for multimodal data fusion. *Neural Comput.* **2020**, *32*, 829–864. [CrossRef] [PubMed]
61. *ISO 8000:2022*; Data Quality. Technical Report. International Organization for Standardization: Geneva, Switzerland, 2022. https://www.iso.org/standard/81745.html (accessed on 1 April 2024).
62. *ISO/IEC 25012:2008*; Software Engineering–Software Product Quality Requirements and Evaluation (SQuaRE)–Data Quality Model. Technical Report. International Organization for Standardization: Geneva, Switzerland, 2008. Available online: https://www.iso.org/standard/35736.html (accessed on 1 April 2024).
63. *ISO/IEC FDIS 5259*; Artificial Intelligence–Data Quality for Analytics and Machine Learning (ML). Technical Report. International Organization for Standardization: Geneva, Switzerland. Under development. Available online: https://www.iso.org/standard/81088.html (accessed on 1 April 2024).
64. Bolukbasi, T.; Chang, K.W.; Zou, J.Y.; Saligrama, V.; Kalai, A.T. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In Proceedings of the NIPS'16: 30th International Conference on Neural Information Processing Systems, Barcelona, Spain, 5–10 December 2016.
65. Kläs, M.; Sembach, L. Uncertainty wrappers for data-driven models: Increase the transparency of AI/ML-based models through enrichment with dependable situation-aware uncertainty estimates. In Proceedings of the Computer Safety, Reliability, and Security: SAFECOMP 2019 Workshops, ASSURE, DECSoS, SASSUR, STRIVE, and WAISE, Turku, Finland, 10 September 2019; Proceedings 38; Springer: Berlin/Heidelberg, Germany, 2019; pp. 358–364.
66. Groß, J.; Adler, R.; Kläs, M.; Reich, J.; Jöckel, L.; Gansch, R. Architectural patterns for handling runtime uncertainty of data-driven models in safety-critical perception. In Proceedings of the International Conference on Computer Safety, Reliability, and Security, Munich, Germany, 6–9 September 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 284–297.
67. Wahlster, W.; Winterhalter, C. (Eds.) *German Standardization Roadmap on Artificial Intelligence*; DIN e.V., DKE: Berlin/Frankfurt am Main, Germany, 2020.
68. Kim, M.S. Research issues and challenges related to Geo-IoT platform. *Spat. Inf. Res.* **2018**, *26*, 113–126. [CrossRef]
69. Ahlawat, P.; Rana, C. An Era of Recommendation Technologies in IoT: Categorisation by techniques, Challenges and Future Scope. *Pertanika J. Sci. Technol.* **2021**, *29*. [CrossRef]
70. Falcão, R.; Villela, K.; Vieira, V.; Trapp, M.; de Faria, I.L. The practical role of context modeling in the elicitation of context-aware functionalities: A survey. In Proceedings of the 2021 IEEE 29th International Requirements Engineering Conference (RE), Notre Dame, IN, USA, 20–24 September 2021; pp. 35–45.
71. Feth, P. *Dynamic Behavior Risk Assessment for Autonomous Systems*; Fraunhofer Verlag: Stuttgart, Germany, 2020.
72. Geisslinger, M.; Poszler, F.; Betz, J.; Lütge, C.; Lienkamp, M. Autonomous driving ethics: From trolley problem to ethics of risk. *Philos. Technol.* **2021**, *34*, 1033–1055. [CrossRef]
73. Adler, R.; Elberzhager, F.; Falcão, R.; Siebert, J.; Groen, E.C.; Heinrich, J.; Balduf, F.; Liggesmeyer, P. A Research Roadmap for Trustworthy Dynamic Systems of Systems-Motivation, Challenges and Research Directions. Technical Report IESE-001.23/E, Fraunhofer Institute for Experimental Software Engineering (IESE), 2023. Available online: https://www.iese.fraunhofer.de/content/dam/iese/publication/dynasos-research-roadmap-fraunhofer-iese.pdf (accessed on 1 April 2024).

74. Adler, R.; Elberzhager, F.; Baldauf, F. Engineering a sustainable world by enhancing the scope of systems of systems engineering and mastering dynamics. *arXiv* **2024**, arXiv:2401.14047.
75. Henshaw, M.; Siemieniuch, C.; Sinclair, M.; Henson, S.; Barot, V.; Jamshidi, M.; DeLaurentis, D.; Ncube, C.; Lim, S.L.; Dogan, H. Systems of Systems Engineering: A research imperative. In Proceedings of the 2013 IEEE International Conference on System Science and Engineering (ICSSE), Budapest, Hungary, 4–6 July 2013; Szakál, A., Ed.; IEEE: Piscataway, NJ, USA, 2013; pp. 389–394. [CrossRef]
76. Henshaw, M. *The Systems of Systems Engineering Strategic Research Agenda: Created by the Trans-Atlantic Research and Education Agenda in Systems of Systems (T-AREA-SoS) Project. Grant Number: 287593*; Technical Report; Loughborough University: Loughborough, UK. Available online: https://www.researchgate.net/profile/Michael-Henshaw-3/publication/316688269_The_Systems_of_Systems_Engineering_Strategic_Research_Agenda/links/590da9beaca2722d185e8c4e/The-Systems-of-Systems-Engineering-Strategic-Research-Agenda.pdf (accessed on 1 April 2024).
77. Dogan, H.; Ncube, C.; Lim, S.L.; Henshaw, M.; Siemieniuch, C.; Sinclair, M.; Barot, V.; Henson, S.; Jamshidi, M.; Delaurentis, D. Economic and societal significance of the systems of systems research agenda. In Proceedings of the 2013 IEEE International Conference on Systems, Man, and Cybernetics, SMC 2013, Manchester, UK, 13–16 October 2013. [CrossRef]
78. Ncube, C.; Lim, S.L.; Amyot D.; Maalej W.; Ruhe G. On systems of systems engineering: A requirements engineering perspective and research agenda. In Proceedings of the 2018 IEEE 26th International Requirements Engineering Conference, RE 2018, Banff, AB, Canada, 20–24 August 2018. [CrossRef]
79. INCOSE. *Systems Engineering Vision 2035*; Technical Report; INCOSE: San Diego, CA, USA, 2022. Available online: https://www.incose.org/docs/default-source/se-vision/incose-se-vision-2035.pdf (accessed on 1 April 2024).
80. Advanced Systems Engineering. Available online: https://www.advanced-systems-engineering.de/ (accessed on 1 April 2024).
81. Axelband, E.; Baehren, T.; Dorenbos, D.; Madni, A.; Robitaille, P.; Valerdi, R.; Boehm, B.; Jackson, S.; Nadler, G.; Settles, S. A research agenda for systems of systems architecting. In Proceedings of the 17th Annual International Symposium of the International Council on Systems Engineering, INCOSE 2007—Key to Intelligent Enterprises , San Diego, CA, USA, 24–28 June 2007.
82. Dridi, C.E.; Benzadri, Z.; Belala, F. System of Systems Modelling: Recent work Review and a Path Forward. In Proceedings of the 2020 International Conference on Advanced Aspects of Software Engineering (ICAASE), Constantine, Algeria, 28–30 November 2020. [CrossRef]
83. Guessi, M.; Neto, V.; Bianchi, T.; Felizardo, K.R.; Oquendo, F.; Nakagawa, E.Y.; Shin, D. A systematic literature review on the description of software architectures for systems of systems. In Proceedings of the ACM Symposium on Applied Computing, Salamanca, Spain, 13–17 April 2015. [CrossRef]
84. Klein, J.; van Vliet, H. A systematic review of system-of-systems architecture research. In Proceedings of the QoSA 2013: 9th International ACM Sigsoft Conference on the Quality of Software Architectures, Columbia, Canada, 17–21 June 2013. [CrossRef]
85. Mohsin, A.; Janjua, N.K. A review and future directions of SOA-based software architecture modeling approaches for System of Systems. *Serv. Oriented Comput. Appl.* **2018**, *12*, 183–200. [CrossRef]
86. Mohsin, A.; Janjua, N.K.; Islam, S.; Graciano Neto, V.V. Modeling approaches for system-of-systems dynamic architecture: Overview, taxonomy and future prospects. In Proceedings of the 2019 14th Annual Conference System of Systems Engineering (SoSE), Anchorage, AK, USA, 19–22 May 2019. [CrossRef]
87. Santos, D.S.; Oliveira, B.R.N.; Kazman, R.; Nakagawa, E.Y. Evaluation of Systems-of-Systems Software Architectures: State of the Art and Future Perspectives. *ACM Comput. Surv.* **2022**, *55*, 67. [CrossRef]
88. Tolk, A.; Rainey, L.B. Toward a Research Agenda for M&S Support of System of Systems Engineering. In *Modeling and Simulation Support for System of Systems Engineering Applications*; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 2015. [CrossRef]
89. Northrop, L.; Feiler, P.; Gabriel, R.P.; Goodenough, J.; Linger, R.; Longstaff, T.; Kazman, R.; Klein, M.; Schmidt, D.; Sullivan, K. *Ultra-Large-Scale Systems: The Software Challenge of the Future*; Technical Report; Carnegie Mellon University, Software Engineering Institute (SEI): Pittsburgh, PA, USA, 2006. Available online: https://apps.dtic.mil/sti/tr/pdf/ADA610356.pdf (accessed on 1 April 2024).
90. Electronic Components and Systems. Strategic Research and Innovation Agenda 2023. Technical Report. Available online: https://ecssria.eu/ECS-SRIA (accessed on 1 April 2024).
91. Castellani, B.; Gerrits, L. Map of the Complexity Sciences; Art and Science Factory, LLC. 2021. Available online: https://www.art-sciencefactory.com/complexity-map_feb09.html (accessed on 1 April 2024).