

We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

5,800

Open access books available

142,000

International authors and editors

180M

Downloads

Our authors are among the

154

Countries delivered to

TOP 1%

most cited scientists

12.2%

Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com



Chapter

The Foundation for Open Component Analysis: A System of Systems Hyper Framework Model

Ana Perišić and Branko Perišić

Abstract

The interoperability and integration of heterogeneous systems, with a high degree of autonomy and time-dependent dynamic configuration over multilevel and multidimensional feature space, raise the problem configurations complexity. Due to the emergent nature of a large collection of locally interacting components, the properties and the behavior of a collection may not be fully understood or predicted even the full knowledge of its constituents is available. The simplification is contemporary addressed through either dimensional reduction methods, like Principal Component Analysis (PCA), or overall ontology managing through Physics of Open Systems (POS) paradigm. The question is: Is it possible to cope with the complexity by integrating dimension reduction steps with basic POS concepts on the Large Data Objects (LDOs) holding the structure and behavior of the complex system. The intended mission of this chapter is to formulate a starting System of Systems (SoS) based configurable hyper framework model that may be dynamically improved to better suit the static structure and dynamic behavior of complex SoS configurations. That is the reason why the reflexive integration of POS and different dimensional reduction methods, through an interoperability framework, have been proposed as the main contribution of this research chapter.

Keywords: collections complexity, framework modeling, large data objects, principal component analysis, physics of open systems, heterogeneous systems interoperability, system of systems analysis

1. Introduction

The globally accepted definitions of digitalization and digital transformation do not still exist, although the terms are in the field for quite a long time. In Gartner Glossary, digitalization is defined as *the use of digital technologies to change a business model and provide new revenue and value-producing opportunities; it is the process of moving to a digital business* [1]. This definition accents the higher granularity mission concerning global system aspects. Considering the particular enterprise systems, that have decided to move their business into the digital form, there is a challenging activity of developing a completely new set of processes and procedures in compliance

with the formulated digital business model. This process is usually considered as enterprise digital transformation. The proliferation of Information and Communication Technologies (ICT), especially in the last decade, puts digitalization and digital transformation into the focus of arbitrary systems development and sustainable collaboration/cooperation in the context of a huge number of dynamic configurations that may emerge throughout its entire life cycle.

The interoperability of heterogeneous systems and integration processes favor System of Systems (SoS) analysis and synthesis approaches to dominate through the 21st century, a century that has become the era of complexity. From the architectural point of view, every real-life system is an SoS, where its architecture is usually seen as a mechanism that creates the illusion of simplicity. To avoid the traditional hierarchical approach to the representation of systems internals, the SoS metaphor has to be deeply understood due to its inherent meshed topology. SoS may range from the technical-systems counterparts, dominantly addressed in systems engineering, to the social systems like, for example, the education, teaching, learning, and performing ecosystem that favor personal competency profile, that suits the industry 4.0 competencies compliance. An integrated systems approach needs to provide a framework and language that allow different systems, with highly divergent characteristics, to interoperate in favor of the commonly agreed mission. With the natural multidimensionality, embedded in the generic SoS paradigm, the underlining complexity directly affects the management and control of the resulting SoS. As a consequence, the dynamic multilayered architecture emerges as a promising approach to the internals complexity hiding.

The SoS concept assumes that participating systems are characterized by the: autonomous mission; independence of encapsulated operations; the difference in fundamental life cycle model aspects. The SoS life cycle stages generally include *creation* (that usually strictly separates the creation process from operational usage); *sustainable operation* (leaving the variety of SoS artifacts persistent instances); *migration* (SoS evolution that retains functionality and structure while replacing the supportive technologies); *replacement* (complete or partial component replacement while preserving the interoperability over specified interfaces); and *termination* (the retirement of a component(s) system(s) while preserving the structural and functional consistency of the remaining SoS). The SoS level of integration is dominantly differentiated by the way the participating systems are orchestrated. According to ref. [2] SoS may be divided into four main categories: *centrally* (directed); *cooperatively* (acknowledged); *collaboratively* (with common enforcing mechanisms); *emergency control* (virtual control of uncontrolled large-scale behavior) *orchestrated*. To cope with the complexity, the architectural design tends to hide the internals of complex systems by wrapping them with an adaptation layer that, to the outer layer stakeholders, exhibits only high-level granularity concepts through well-defined interfaces.

In general, complex systems are analyzed from two complementary aspects: the structure (static view) and the behavior (dynamic view). The structural complexity emerges from observing the system as a composition of arbitrary, interrelated components (subsystems). These components may be considered or sometimes really are systems with a high degree of autonomy. From the behavioral aspect, complexity is defined as the degree of difficulty in predicting the future dynamic properties and behavior of an overall system assuming that current dynamic properties and behavior of the participating components (subsystems) are known. Due to the emergent nature of a large collection of locally interacting components, the properties and the behavior of a collection may not be fully understood or predicted even the full knowledge of its

constituents is available. Because of that, Complex Systems Science (CSS) requires new mathematical and heuristic frameworks and scientific methodologies to cope with SoS organized structural and behavioral complexity handling.

The complementary approach is to view an arbitrary system through its inherent dimensions, which do not strictly structure or behavior-oriented but add another direction of complexity, the multidimensionality. Combined with the multilevel architecture it creates multidimensional multilevel (hyper) composites. If there is a need for different contexts or configurations management in timed framed scale, the hyper composites may generate an arbitrarily large number of instances that are context, configuration, and time-dependent. These instances form a generic repository that is usually seen as an association of Large Data Objects (LDOs). These objects need to be stored, loaded, searched, modified, and visualized in wide variety of ways. Traversing, for example, the hyper-structure instances, with any rational reason in mind, creates a challenging NP (nondeterministic polynomial time) complex problem that opens the fundamental question of arbitrary dimensionality reduction approach. The dimensionality reduction facilitates the transformation of the initial system/problem into a less complex one (with a lower degree of freedom) that may be efficiently/effectively solved with acceptably lower accuracy (precision).

Modeling is considered a promising approach to cope with complexity. Model-driven SoS development is a challenging paradigm that may generate the initial multidimensional and multilayered meta-SoS model (MSoSM). The originating version of any particular MSoSM may serve as a starting orchestration pattern that, by dynamical inclusion of individual dimensions in particular configuration(s), forms meta structures and facilitates their instantiation. According to that, one of the main challenging approaches, being the main goal of this chapter, is the specification of SoS based open framework model that may serve as the meta-SoS pattern for describing and monitoring arbitrary complex composites that instantiate multilayered multidimensional data objects that may serve as a resources pool for arbitrary component analysis support.

The openness as a global system property is elaborated in remarkable work on Physics of Open Systems (POS) where the complexity of systems is perceived from systems dynamics (a complexity of movement) [3]. The described POS scientific methods and technologies enable the formation of scientifically proven systems ontological knowledge from its empirical descriptions embedded in a huge amount of semi-structured, multimodal, multidimensional, and heterogeneous data.

The main question arises: *Is it possible to upraise the SoS architecture as a reflection of successive and self-improve dimension reduction processes concerning basic POS principles applied on the persistent information resources base that preserves the SoS dynamics descriptions in the form of LDOs?*

Gaining a definitive answer to this question is far beyond the scope of this chapter. Its intended mission is to formulate a starting SoS-based framework model that may be further dynamically improved to better suit the static structure and dynamic behavior of complex SoS configurations. That is the reason why the reflexive integration of POS and different dimensional reduction methods, through an interoperability framework, have been proposed as the main contribution of this research chapter.

The rest of the chapter is organized as follows: Section 2—The background and motivation elaborate problem domain aspects of SoS multilevel multidimensional and representation, interoperability and integration aspects, architectural challenges, and framework-based approach with selected related work analysis and discussion. Section 3—System of Systems Hyper Framework Model (SoS-HFM)—states and

elaborates the proposed model as a foundation for arbitrary component analysis method application. It is intended to serve as a reliable LDOs pool for incremental dimension reduction activities that converge to the SoS architecture upraise. Section 4—Conclusion is devoted to the concluding remarks, challenges, and the directions of future research activities. In the Appendices section, there are several Java code templates of SoS-HFM concepts presented. References—contains a list of references that have been analyzed and used as the problem domain origins and comparative research results evaluation basis.

2. Background and motivation aspects

Scientific research is a robust and dynamic practice that employs multiple methods toward investigating systems or phenomena including experimentation, description, comparison, and modeling. According to ref. [4], these methods, although often used in combination, appear more effective if used alone. Experimental methods are used to investigate the relationship(s) between two or more phenomena in a strictly controlled environment. Description methods utilize the observations and measurements of natural phenomena and their relationships, to collect the relevant data set that describes their behavior. Comparison is used to determine and quantify relationships between two or more phenomena by observing different groups that are, either by choice or circumstance, exposed to different treatments. Scientific knowledge cannot be obtained from empirical data by purely logical means because the ontologies of scientific and empirical knowledge differ significantly. Physics of Open Systems (POS), briefly introduced in the previous section [3], facilitate the generation of scientifically proven knowledge about the ontology of open systems via data mining techniques, applied on a huge amount of semi-structured, multimodal, and heterogeneous data that is dynamically generated throughout the SoS lifecycle. The identification of characteristic symmetries in an ontology model is used to simplify the structure and behavior of open systems over the state space defined by the ontology model.

The modeling is a well-established mechanism for struggle with the complexity that is the main obstacle of contemporary systems and solutions. The results of the modeling process are a single or a combination of physical and/or computer-based models of natural systems and/or phenomena that are afterward used as a framework for experiments and/or observations. Scientific development (progress) addresses the scientific approach to overall and sustainable development concerning the wide variety of contemporary problems that are either global or domain specific [5].

Concerning the SoS discipline, despite the inherent complexity, the large picture approach is usually the most promising one. The SoS exhibits an organized form of complexity and therefore cannot be accurately described by the traditional analysis techniques. The key concept of complexity science is universality, which is the idea that many systems in different domains exhibit phenomena with common underlying features that can be described using the same scientific models. Complexity science can provide a comprehensive, cross-disciplinary analytical approach that complements traditional scientific approaches that are focused on a specific observed subject in each domain. Complex systems are often characterized by many components that interact in multiple ways among each other and, potentially, with their environment too. These components form then dynamic networks of interactions, with wide variety of network topologies. They generally range from configurations with a small number

of components that are involved in a large number of interactions, to configurations that involve the enormous number of components involved in a small number of interactions (**Figure 1**).

Interactions may generate novel information that makes it difficult to study components in isolation or to completely predict their future structure and/or behavior. The main challenge of complexity science is not only to have a sense of the parts and their connections but also to understand how these connections give rise to the whole. Advanced mathematical and computational modeling, analysis, and simulations are almost always required to investigate how these configurations are structured and change with time.

The growth of stakeholder-driven content has fueled a rapid increase in the volume and type of data that is generated, manipulated, analyzed, and archived. In addition, varied newer sets of sources, including sensors, Global Positioning Systems (GPS), automated trackers, and monitoring systems, are generating a huge amount of data multidimensional. These larger volumes of data sets, often termed *big data*, are imposing newer challenges and opportunities considering: storage, retrieval, analysis, visualization, and long-term archival. Computer-based analysis of massive data, emerging from complex systems structure and behavior, enables the recognition of embedded data/information/knowledge/wisdom (DIKW) patterns that contribute the further understanding of structure and behavior either of the wholes and/or its parts, thereby fostering the more accurate prediction of forthcoming structure and/or behavior. The second challenge raises directly from the multilevel and multidimensional nature of the artifacts that are consciously or unconsciously reflected through the complex systems' time and configuration-dependent state transitions. In **Figure 2**, a cognitive DIKW pyramid is presented that relates LDOs that may be generated with the inherent semantics in mind.

Different forms of data representation are well established and experienced in engineering practice. It is generally not the case with Information, Knowledge, and Wisdom, because they are dominantly context-dependent. The SoS persistency layer is a crucial component of SoS based analytics and possible drawbacks of LDO concepts need careful attention while specifying a supportive framework model. Although there is a huge amount of intellectual value embedded in arbitrary real-life systems

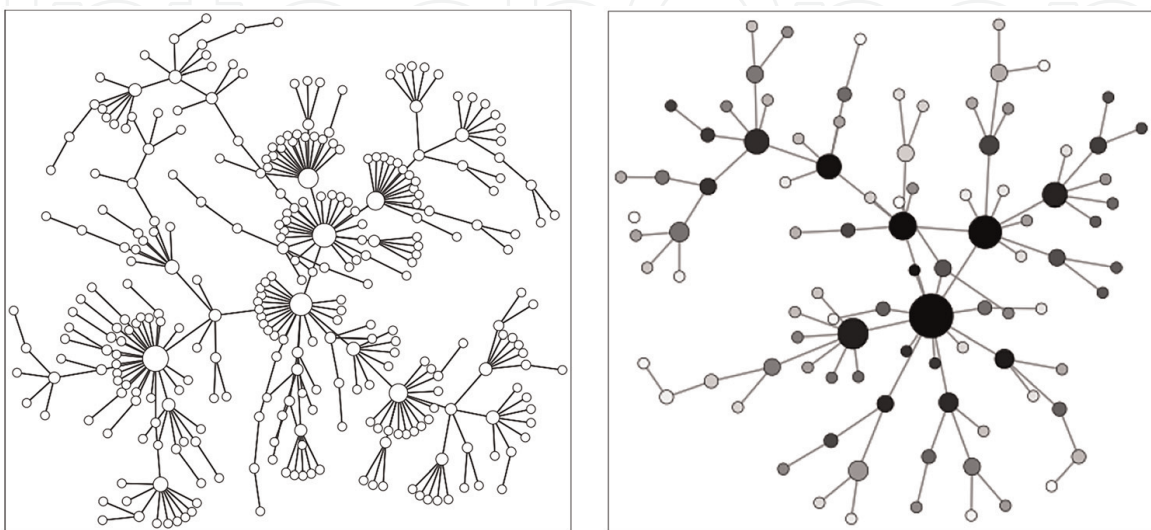


Figure 1.
Network representations of composite components [6, 7].

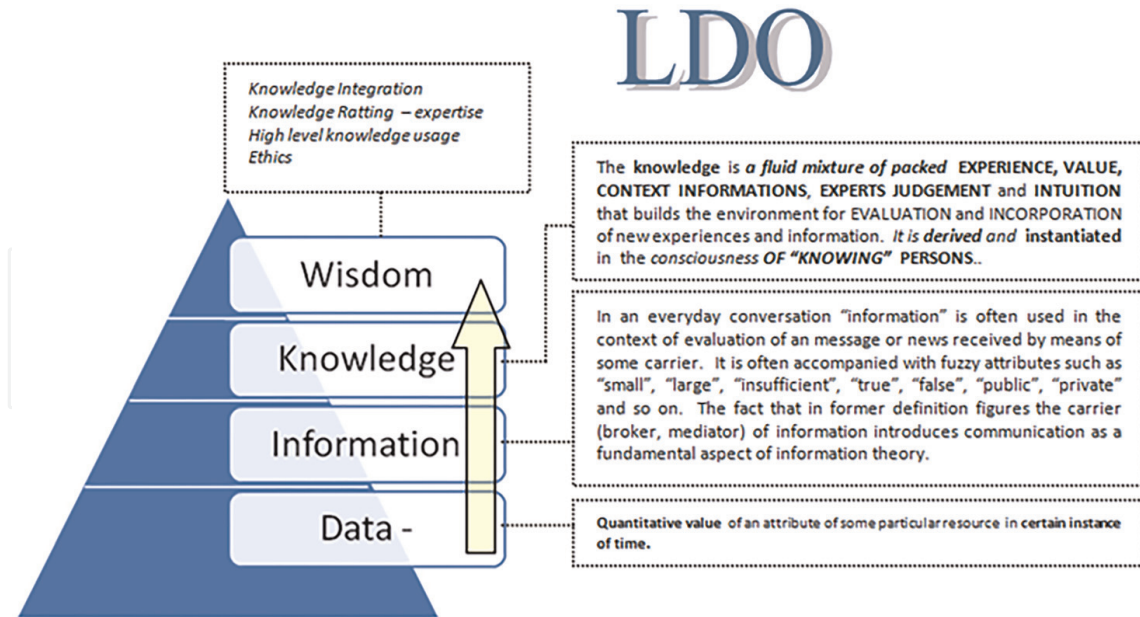


Figure 2.
The representation of LDOs genesis.

that are naturally represented in semi-structured or unstructured form, the contemporary legacy Enterprise Systems are still dominantly operated over structured repositories. Concerning the growing shift from SQL to NoSQL persistency the hybrid repository model seems an appropriate solution to start with.

In ref. [8], the authors elaborate Big Data management in the context of three inherent supportive dimensions: *technology* (dominantly related to storage, analytics, and visualization); *people* (addressing the human aspects); and *processes* (addressing technological and business approaches to management aspects). The semantic value, quality of data, and data security are stated as dominant challenging issues concerning the Big Data foundation of arbitrary SoS artifacts. The Framework-based approach to Big Data analytics application in high-level education environments is presented in ref. [9]. Although strictly conceptual the proposed framework model may be applied beyond the scope of the education domain. In ref. [10], there is the application of Linear Mixed Modeling (LMM) promoted as a flexible approach for scientific experimental data analysis. The nature of experimental data opens a challenging question of dataset quality metrics that may be proliferated to the SoS dynamic configurations instances. The multilevel ontological generation of semantic relations extracted from the significant amount of heterogeneous linguistic data, persisting in Big Data repositories, has been proposed by Popova et al. [11]. The solution is based on a specific XML format that enforces the interoperability of information across individual levels of generated multilevel ontology, for a particular problem domain. The software engineering perspectives of the Big Data foundation are surveyed in ref. [12]. The refinements of software development activities through the challenging aspects of corresponding Big Data concepts are discussed with the particular accent on architecture design, software quality insurance, and data quality assessment. In ref. [13] the intelligent systems design processes are discussed through multidimensional modeling of knowledge and knowledge transfer between internal components and external counterparts of arbitrary intelligent systems viewed as a four-dimensional (*grade, atomization, abstractness, timing*) cellular architecture. The engineering aspects of spatial data is an challenging domain for engineering disciplines that are based on

different natural phenomena analysis and simulation like daylight illumination of residential buildings in ref. [14] where the representation of large data objects and its potential dimensionality reduction has an dramatic impact on urban planning.

Although computer-based analysis of massive data sets is in the field for a quite long time it is far from being a routine activity. From the architecture aspect, it is essential to separate the external repository from the internal dynamic storing and presentation layers and thereby hide the particular characteristics of persistent LDO form from its operational counterparts. This is the first pillar of the proposed SoS framework model.

The computational complexity of high-dimensional LDOs processing is the main obstacle for real-time or near real-time applications. The LDO's complexity reduction appears as a promising approach to the system/problem simplification process. Because less complex LDOs are easier to navigate, explore, visualize and analyze in different contexts they are more suitable for effective machine learning.

That is the main reason why several complexity reduction methods, based on different dimensionality reduction algorithms, have been proposed, formalized, applied, and verified. Among them, there are two main unsupervised methods worth mentioning: hierarchical clustering (HC) and principal component analysis (PCA). HC tries to build a tree-like structure with leaves representing the individual objects and nodes (pseudo objects) representing the clustering points of leaves with the highest degree of similarity. In further iterations, the individual clusters (as surrogates of clustered objects) replace the whole group and appears as individual objects with a certain accuracy payoff. PCA, on the other hand, creates a lower-dimensional representation of the initial data set on top of principal components as patterns encoding the highest variance in the data set, trying to preserve as much as possible of the original data set variance in process of dimensionality reduction.

Reducing the complexity of a particular LDO has its payoff in the accuracy of the reduced counterpart. The quality of a dimensionality reduction method is measured by its ability to gain the lowest possible complexity with the highest possible accuracy. Being unsupervised, HC and PCA methods are better suited for the generation of sustainable simpler LDOs, and consequently simpler SoSs configurations, rather than their verification.

Due to the fundamental focus of this chapter, the rest of the section is devoted to a more detailed elaboration of solely the PCA methods-related publications analysis. The mathematical elaboration is completely avoided due to the huge amount of references that have excessively addressed the foundation. A remarkable complete, simplified, step-by-step analysis of the original PCA method is presented in ref. [15], through five consecutive steps that lead to the data set dimensionality reduction. It starts with the standardization of the initial variable (dimension) range to comparable scale, to eliminate the possible supremacy of dominant instances, followed by the calculation of the covariance matrix of all possible pairs of scaled variables (dimensions) to uncover the correlation nature of each possible variable pairs. In the third step, the principal components isolation is performed by computing the eigenvectors of the covariance matrix and ordering them by their eigenvalues in descending order. This process isolates the principal components, which are the surrogates of correlated dimensions, in order of their significance. In the fourth step, the Feature vector is created by selecting the representative subset of principal components that leads to the desired dimensional reduction with preferable accuracy. The last step is the generation of reduced dimensional data set by data recasting over selected principal components. The mathematical foundation of linear PCA is gradually presented in ref. [16], and joined with the context

of the previously referenced article completes its light-weighted approach. In ref. [17] the original row-based two-dimensional principal component analysis (2DPCA) and its extensions in the 2D image processing domain, have been discussed. The overview of several extension frameworks have been presented (bilateral projection nonlinear and iterative, kernel-based, supervised (with four variations), alignment-based, and random (with three variations)). The robustness of all of the elaborated extensions has exhibited a performance increase in comparison to the original 2DPCA in image recognition, which has been a traditional application domain of PCA methods. PCA is discussed in ref. [18] through the extensive survey of five RPCA published models, with their comparative analysis in the context of video stream background management. In ref. [19] the modification of classical PCA (MPCA) with subspace learning framework based on multiple similarity measurements. In ref. [20] a comprehensive review and future PCA development have been presented. Although the reference is almost six years old, it has been used for the sake of the overall problem domain clarification where PCA is addressed as the linear exploratory tool for data analysis. Due to its application in different fields, the initial PCA has been modified in several ways to better suit the domain-specific characteristics. The authors enlist and discuss: static and dynamic functional PCA (FPCA), simplified PCA (SPCA), robust PCA (RPCA), and symbolic data PCA (SDPCA). In ref. [21], authors present the modification of nonlinear Kernel PCA (KPCA), with adaptive feature—Adaptive Kernel PCA (AKPCA) that is integrated with the gray relation analysis (GRA) for fault detection in complex nonlinear chemical processes.

The comprehensive analysis of different domain-specific PCA applications is far beyond the reasonable research effort. In ref. [22] the application of state-space functional PCA (SS-FPCA), as a 3-level hierarchical model built under the state-space model framework, for identification of spatiotemporal patterns based on satellite remote sensing of lake water quality in the form of time series of spatial images with missing observations. The authors of ref. [23] elaborate the Principal Component-based support vector machine (PC-SVM) as a hybrid machine learning technique that combines PCA and SVM to cope with the potential software defects especially concerning the mission-critical software systems. In ref. [24] the contemporary challenging study of the implementation of machine learning methods for identification of patients affected by COVID-19 based on X-ray images. Two commonly used classifiers were selected: logistic regression (LR) and convolution neural networks (CNN) joined with PCA for complexity reduction and shorting the elapsed time to gain the quality diagnostic answer. The complex boundary generation method, presented in ref. [25], illustrates an practical application of dimensionality variation through the recursive search of the optimal residential building outer shape form, based on variable set of parameters.

This short survey and the much broader repertoire of similar research articles fully qualify the research motivation of this chapter, the formulation of supportive SoS Hyper Framework Model.

Physics of Open Systems is the additional paradigm for SoS Hyper Framework Model development where the system is considered as a tool where the knowledge and sense of its complexity are harvested. It is necessary to reference [3] for the rest of the relevant influencers. The directly influencing POS intellectual machine analytical core technologies that support systems: reconstruction through ontology mode variations; examination based on communication model variations; design based on state model variation; empirical context formation—the generation of LDOs in this chapter context; solutions behavior generation—the dynamic representation of varied model;

visualization. Its formulation is transformed in the formulation of POS dimensions of the system model in the context of this chapters' SoS Hyper Framework Model proposal.

Software-supported frameworks of the arbitrary kind are usually targeted in model-driven software development (MDSD) and model-based system engineering (MBSE) approaches. They are closely related to the general architecture modeling paradigms and constitute the core of different contemporary enterprise architecture (EA) frameworks. There are several EA Frameworks proposed and specified, among which the most advocated are:

- *Zachman Framework* (ZF), the framework for enterprise architecture (EA). According to ref. [26], it is an EA ontology-based on the visualization of an enterprise and its inherent information system components and their relations from different perspectives. It is a two-dimensional classification scheme structured as a matrix containing 36 cells, each of them focusing on one dimension or perspective of the enterprise. Although popular in EA academic education and learning environments, due to its inexplicable practical utility, it has not reached substantial practical achievements and is expected to fade out from the EA community in near future [27].
- *The Open Group Architecture Framework* (TOGAF), is declared as the most commonly used enterprise architecture framework [28]. Due to its architecture development method (ADM) orientation, it is considered dominantly process-oriented. The overall TOGAF process is organized into four groups: *Business architecture*; *Application architecture*; *Data architecture*; and *Technical architecture*. Due to its OO community origins, it has been announced as a promising EA framework. On the contrary, the extensive search through the contemporary available resources on Global Network shows that there is still a lack of usable TOGAF software support tools which makes its practical usability questionable.
- *Federal Enterprise Architecture Framework* (FEAF), according to ref. [29], has offered a shared approach for the consolidation of strategic, business, and technology management as a component of organization design and performance management. It is composed of 6 interconnected Reference Models (Strategy; Business; Data; Applications; Infrastructure; and Security), each relating to a sub-architectural domain of the framework that is linked together through the Consolidated Reference Model (CRM). The main problem FEAF faces today is the lack of real verifiable success stories even by the enterprise architects working for the U.S. Government [27].
- *CMMI Framework*—Capability Maturity Model Integrated, Software Engineering Institute (SEI) Framework. Although originally tailored toward software, the latest CMMA Framework version is more general and applies to hardware, software, and service development across all industries [30]. It is a process-oriented framework, whose main purpose is to assess the maturity of an organization's processes and to provide guidance on their improvement to deliver high-quality products. In the CMMI model, version 1.3, there are 22 process areas defined, together with the process-related goals and the set of activities that are often used to meet them. Because the CMMI Framework is based on well-defined ontology, classification system, experience, training, and appraisal infrastructure

(governed by SEI), it is expected to evolve into a de-facto standard for the maturity classification of at least Software Development Companies.

- *International Council On Systems Engineering* (INCOSE)—System of Systems Framework [31]- is closely related to the Model-Based Systems Engineering (MBSE) approach that emerged from the INCOSE projects in the SoS engineering domain. It is clustered over three main concepts: **model** (*a formally simplified version of an entity of interest*), **systems thinking** (*a holistic approach to interacting entities and their components*) [32], and **systems engineering** (*transdisciplinary and integrative approach to the engineering of interacting entities, based on systems principles and concepts in the context of scientific, technological and management methods application, through the entire life cycle*). MBSE does not strictly prescribe any process framework the arbitrarily selected process model has to address the four essential systems engineering domains: requirements/capabilities; the static structure (systems architecture or topology) [33]; the dynamic structure (systems behavior); verification and validation aspects [34] (is it the right system? and if is, is the system right?). In ref. [35] there is a remarkable well-illustrated approach to SoS mission needs break down to capabilities and functions through the architecture framework and related ontology that has inspired several concepts of the SoS Hyper Framework Model specification.

The related work analysis shows that there is a tremendously large number of documents, studies, standards, procedures, and scientific articles that dominantly address particular aspects of the Principal Component Analysis approach to handle the dimensionality reduction problem, but far fewer references concerning POS paradigm and its implementation aspects in the context of SoS.

On the other hand, there is also a lack of research concerning the interoperability framework approach with the integration mission. These facts favor the large-picture-based approach facilitating the Generic System of Systems Framework that sustainably orchestrates: Domain-Specific and Generic concepts and dimensions of complex SoS configurations that are opened for arbitrary POS and PCA methods extension. The System of Systems Hyper Framework Model, presented in the next section of this chapter, is considered as a first step toward the established goal. The collaborative frameworks, like one elaborated in ref. [36], has served as an initial framework specification of the proposed model presented in this article.

3. The foundation for open component analysis: SoS hyper framework model (SoS-HFM)

3.1 Why framework-based approach?

In general, a framework may be defined as a real or conceptual foundation, with a specified level of complexity that serves as a support or a guide for the building of a particular artifact or performing a particular activity by expanding and specializing the generic structure that specifies the family of interrelated products and/or procedures. Framework favors reusability by managing the overall control flow and orchestration of dynamically configured components in an inversion of control way. There are two major categories of contemporary framework: non-software empowered (usually represented as a set of structured and/or semi-structured

documents) and software empowered (software supported collaborative/cooperative environments supporting the digital transformation of problem domain). The development of software empowered interoperability frameworks have usually been preceded by the intensive and time-consuming: specification; modeling; and meta-modeling activities performed and managed within the scope of related projects and/or portfolios.

3.2 Why interoperability-based approach?

In the context of this chapter, cooperation and collaboration of independently developed or specified systems may be generally achieved through total homogenization; harmonization; adaptation; and orchestration interoperability principles. The total homogenization principle states that all of the related systems have to be designed or redesigned to achieve absolute compliance with the globally accepted and standardized template. Homogenization is the most radical, heavy-weight, approach. A harmonization principle is a lightweight approach that assumes the definition and standardization of interfaces that hide the internal characteristic of participating systems and enable their cooperation/collaboration over the unique communication protocol and through the standardized interfaces only. The participating systems need to be functionally complete while retaining the structural diversity. The adaptation interoperability principle assumes cooperation/collaboration of functionally and structurally incomplete systems where participating systems are homogenized up to the functional and structural completeness, in a virtual or real way (where homogenization payoff may substantially differ from system to system) and harmonized afterward. Adaptation interoperability may be seen as a middle-weight approach. It is more complex than the harmonization but, compared to the total homogenization, significantly more acceptable and achievable faster. The orchestration interoperability is the ability of the heterogeneous systems, with arbitrary functionality and topology, to interact toward the mutually beneficial dynamically configured mission, build through the functional and/or structural orchestration of participating systems features and/or resources. Each system retains and shares everything it can perform and/or deliver and delegates and/or acquires everything that is beyond its scope, but available as a mutual benefit of a current configuration. The orchestration interoperability is an example of broker-based service-oriented dynamic architecture that may be formally described by the swarm intelligence concepts.

3.3 Why the interoperability framework approach?

If carefully combined the best characteristics of two, previously discussed promising concepts, may result in an empowered solution capable of handling the growing complexity of SoS dynamic configurations associated with the multidimensional and multilevel LDOs. The development of interoperability framework generally requires a multi-stakeholder process and the long-term vision of a highly reusable generic solution that, in the context of SoS-HFM, may impact the overall ontology, configurable topology, state-driven behavior, and time and context-dependent LDOs: creation, processing, storing, retrieval and visualizing.

The starting point of SoS and POS paradigms integration is the formulation of a hybrid system meta-model that combines multidimensionality and context-based multilevel features. The meta-concept (MetaConcept) of SoS-HFM is modeled as a typed composite MetaElement presented in **Figure 3**.

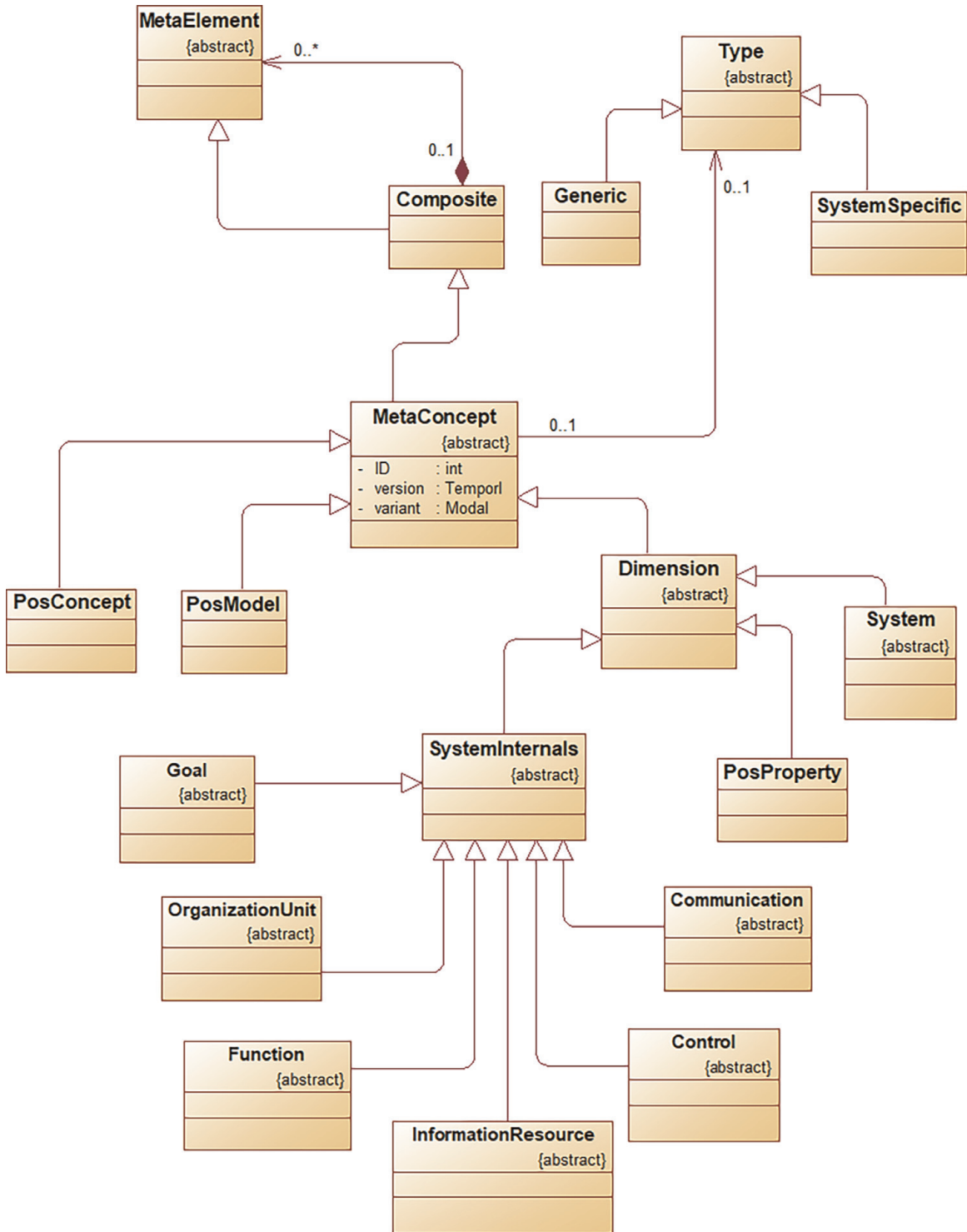


Figure 3. SoS-HFM—meta concept model (a part of).

The composition property enables arbitrary topology creation in different contexts (determined by variant property as *Modal* object, and version property as *Temporal* object), and type associated property that enables the classification over an open set of classifiers with currently two specified: Generic and system specific.

The open set of MetaConcept specializations is currently composed of three elements: PosConcept (relates the framework ontology with POS System ontology); PosModel (relates the framework ontology with POS Model ontology segment); and

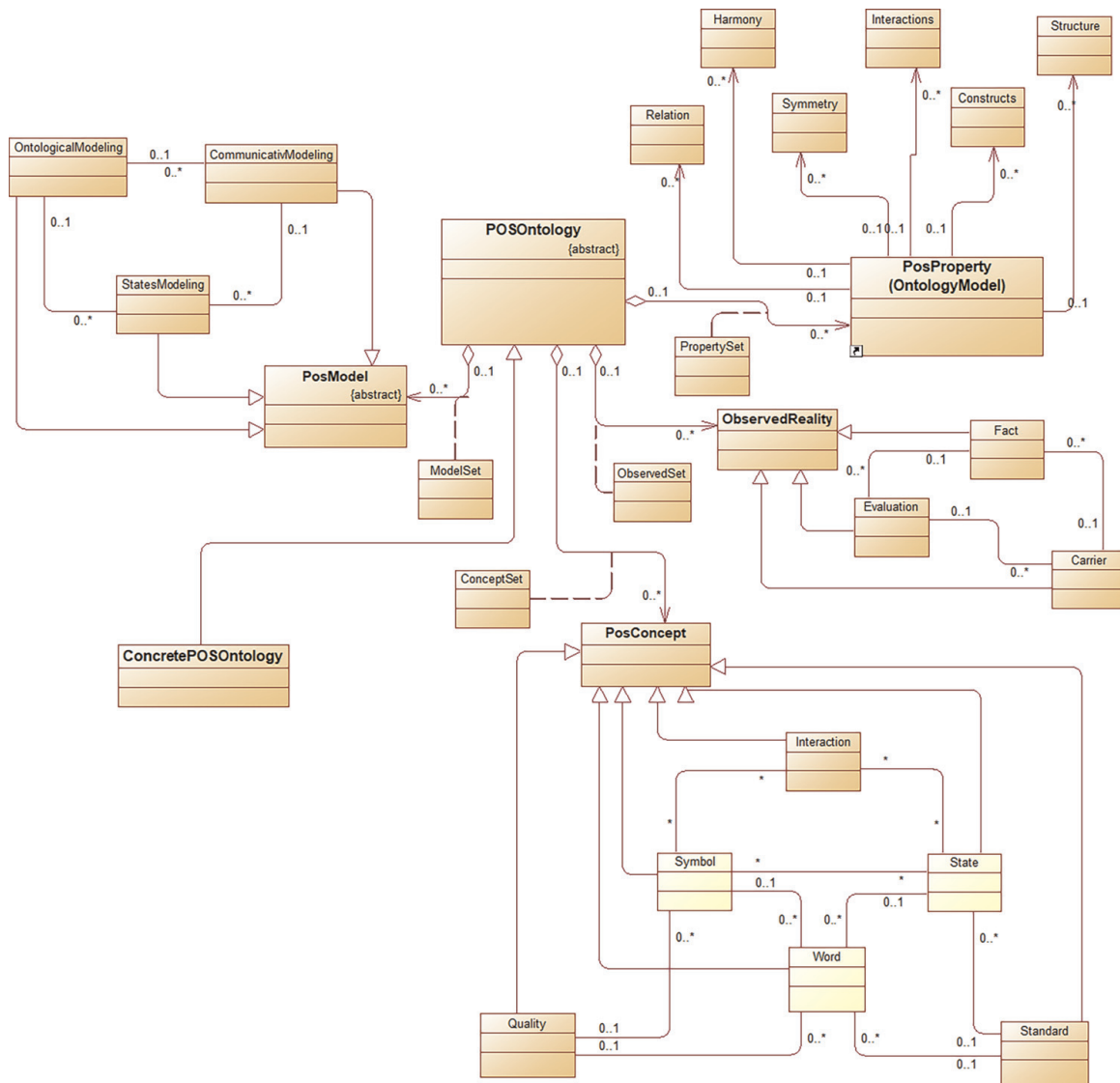


Figure 4.
 The POSOntology model.

Dimension (abstract MetaConcept that clusters the fundamental SoS-HFM object of interest). The Dimension specialization is further specialized by an open set currently 3 concepts: System and POSProperty that will be defined later on and System Internals with fundamental internal dimensions of an arbitrary general system (Goal; OrganizationUnit; Function; InformationResource; Control; and Communication) that are elaborated in following segments of this Section.

The POSOntology model, developed in compliance with [3], is presented in **Figure 4**. POSOntology is an LDO that is composed of four collections:

1. PosProperty collection, organized by the PropertySet associative class, is an LDO that is configured from six predefined POS contextual dimensions (Relation; Harmony; Symmetry; Interactions; Constructs; and Structure);
2. PosModel collection, organized by the ModelSet associative class, represents the POS model triangle composed of contextual modeling dimensions (OntologicalModeling; CommunicativModeling; and StatesModeling);

3. ObservedReality collection, organized by the ObservedSet associative class, represents the POS model triangle composed of contextual observed dimensions (Fact, Evaluation, and Carrier);
4. PosConcept collection, organized by the ConceptSet associative class, represents two interrelated POS triangles composed of contextual concept dimensions (Symbol; State; Word) and (Interaction; Quality; Standard)

ConcretePOSOntology is a domain or system-specific specialization of POSOntology that forms the ontology segment of SoS-HFM and is the targeted ontology of arbitrary orchestrated POS methods.

The detailed System Dimension of the SoS-HFM ontology meta-model is presented in **Figure 5**. The system is an LDO, defined as the SoS-HFM dimension that represents the organized set of interrelated components that are orchestrated (configured) with the specific mission in mind.

It is composed of five embedded associative collections and one inherited associative collection. The five embedded associative collections are:

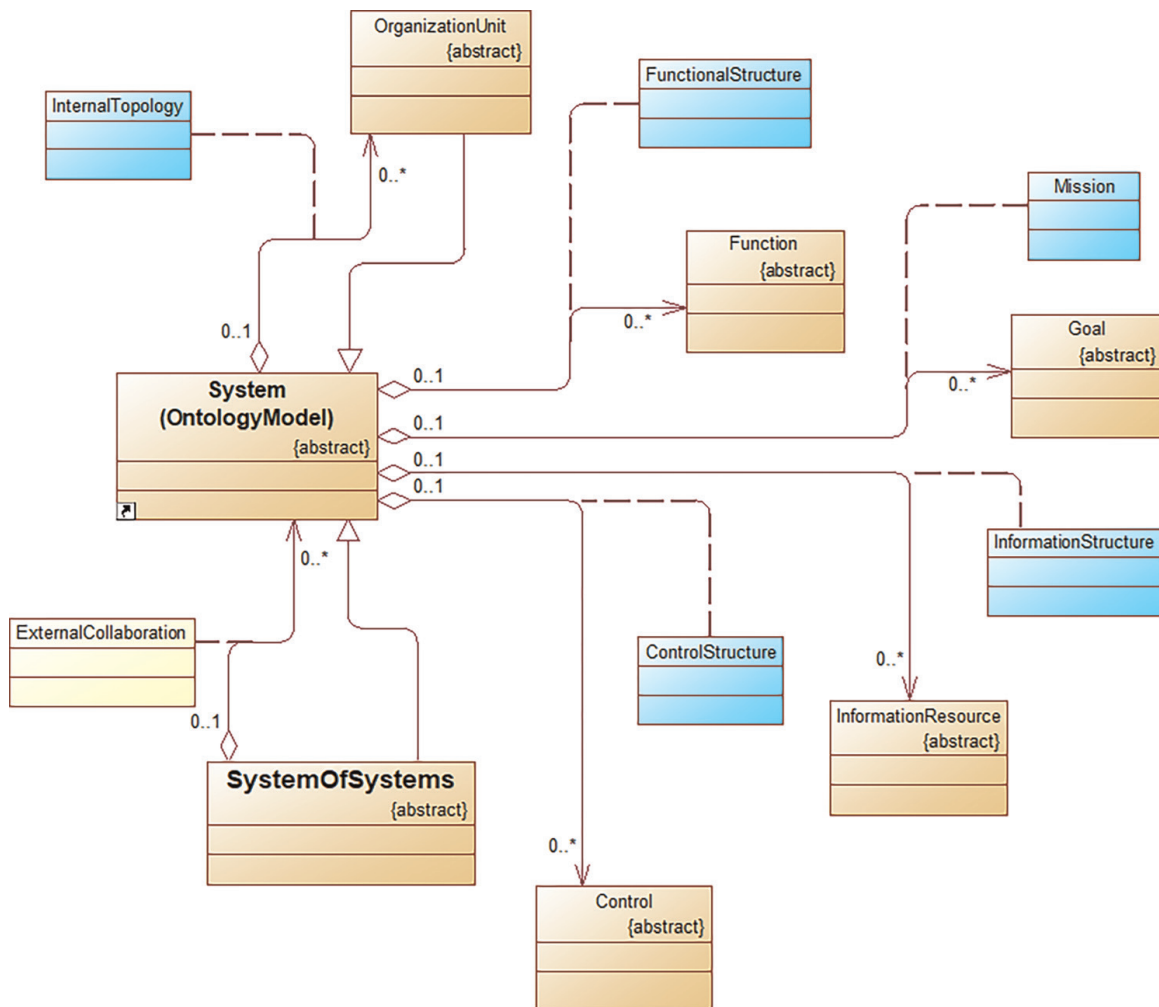


Figure 5.
The system dimension of SoS-HFM meta model.

1. Goal set collection, organized by the Mission associative class, that enables the modeling of domain-specific systems mission (the goal or set of goals that justifies the existence of a system);
2. OrganizationUnit set collection, organized by the InternalTopology associative class, that forms the instances of internal systems architecture;
3. Function set collection, organized by the FunctionalStructure associative class, that forms thy configurations of internal activities that are spread over the OrganizationUnit topology in favor of the overall Mission;
4. InformationResource collection, organized by the InformationStructure associative class, that forms the supporting information infrastructure of a Functional topology;
5. Control set collection, organized by the ControlStructure associative class, forms the monitoring, control, and management of the configuration instances in favor of gaining the overall systems Mission.

The inherited associative collection, organized by the ExternalCollaboration associative class, represents the SoS dimension of the SoS-HFM meta-model, with SoS defined as a specialization of system meta concept.

The SoS-HFM dimensionality reduction meta-model is presented in **Figure 6**. The key meta concepts defined are the configuration and the LargeDataObject.

1. Configuration is the specialization of SoS meta-class with three associative collections: Orchestrated method represents the aggregation of orchestrated dimensionality reduction methods for the particular configuration instance, managed by the dimension meta-class; ConcretePOSOntology relates configuration with the set of associated POS ontologies; and LargeDataObject composite collection that is either original or dimensionally reduced counterpart.
2. LargeDataObject (LDO) encapsulates the dimensionality information and the status of each dimension that is either reducible (Reducible) or not, for an unreduced instance, and reduced (Reduced) if reducible in a reduced instance. LDO is associated with a particular ontology (ConcretePOSOntology). Being a MetaConcept LDO inherits temporal and modal characteristics and marks the particular instances with.

In **Figure 7**, the conceptual model of SoS-Hyper Framework software architecture is presented. It is modeled as MVC architectural pattern where: SoSHFModel—handles the framework dynamic data structure; SoSHFView—encapsulates the collection of SoSHFModel visualization methods, and SoSHFControler—supports framework dynamics. ServiceControler handles an open set of framework services with currently specified: DataSetBuilder; SesionManager; ActivityTracker; and SecurityManager. RepositoryManager encapsulates an open set of repository handlers with currently specified: RelationalRepository; and NoSQL open set of Non-relational repository handlers.

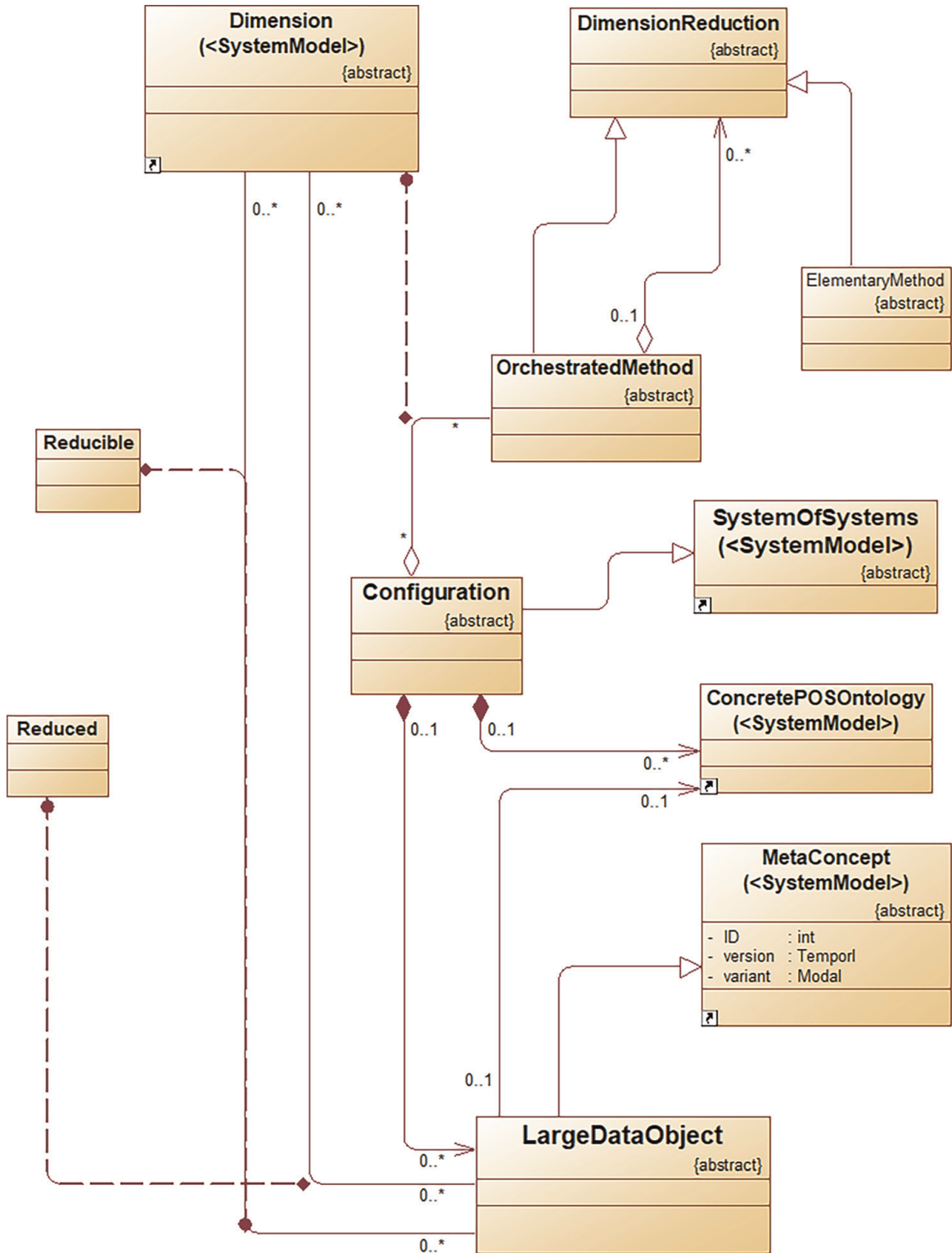


Figure 6.
The SoS-HFM dimensionality reduction meta model.

In **Figure 8**, there is a more detailed conceptual model of the SoSHF Controller segment that encapsulates POS and dimensionality reduction methods (PCA) presented. It utilizes a two-component interface: *AccessIR* (manipulation of arbitrary information resources designated as LDOs in this chapter); and *DimensionAccess* (manipulation of the collection of dimensions embedded in LDOs).

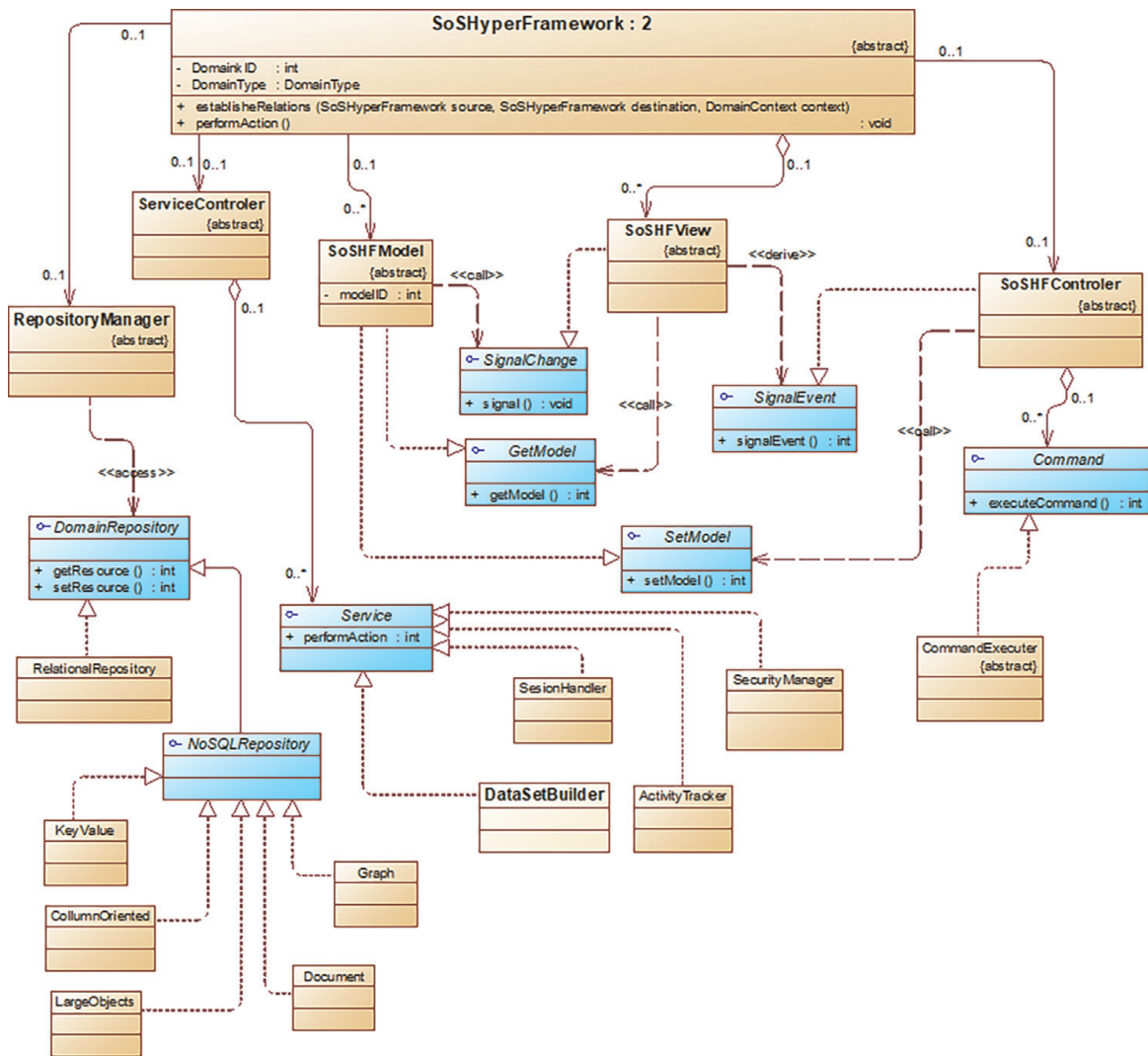


Figure 7. SoS-hyper framework conceptual model (MVC).

3.4 The discussion

As previously mentioned, the main challenge of complexity science is not only to understand and manage the individual components and their connections that make the configuration (static structure) but also to understand how these connections affect the whole. The Foundation for Open Component Analysis is specified on top of three synergic contexts, The Physics of Open Systems ontology, System specification as an SoS composition with multidimensional and multilevel instantiation capabilities, and the simplification mechanisms based on the extendible set of the Principal Component Analysis methods. The orchestration is specified through the System of Systems Hyper-Framework Model where advanced mathematical and computational modeling, analysis, and simulations may be applied to investigate how these orchestrated configurations are structured and change with time.

4. Conclusion

Considering the inherent complexity of System of Systems, the mission of creating the foundation of Opened Component Analysis emerged with an SoS- Hyper

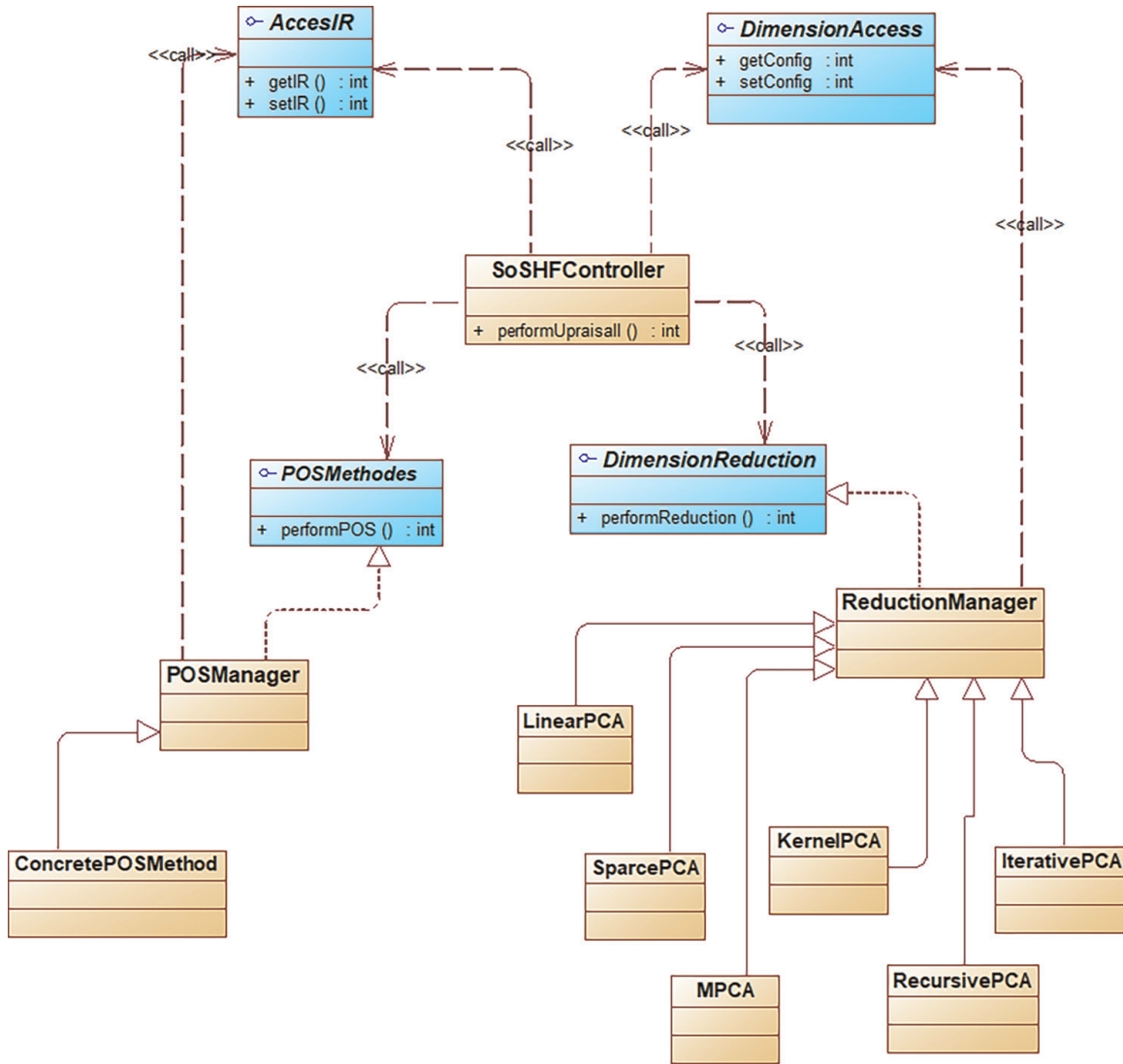


Figure 8. SoSHFController—POC and PCA dimensionality reduction.

Framework Model that has inherited the challenging aspects of Physical Open System fundamental concepts and methods and dimensionality reduction methods by the orchestration of an opened set of Principle Component Analysis methods. The main characteristics of the proposed framework model are elaborated in the Introductory section, founded through the Background and motivation section with related work analysis, and explicated in Section 3 with the presentation of the key aspects of SoS-HFM. In Appendices A.1, there is a sample of Java code generated from the corresponding Configuration meta class presented.

The proposed model is attended to serve as a starting specification for the development of the future: open, heterogeneous, cooperative/collaborative, a service-oriented software framework that may be tailored according to the SoS configurations. These are also the main directions of future research and work.

Conflict of interest

The authors declare no conflict of interest.

Appendices

A.1 SoSHyperFramework: model-based generated Java code skeleton

```
/*
 * Module: SoSHyperFramework.java
 * Author: Branko
 * Purpose: Defines the Class SoSHyperFramework
 */
import java.util.*;
/** @pdOid 5199ec80-82c8-4224-adf4-6f79e037a32c */
public abstract class SoSHyperFramework {
    /** @pdOid 29f37534-7bc3-46d9-9fa5-f8996a25eb8d */
    private int domainkID;
    /** @pdOid 3f9f6a74-0c4c-448d-878e-1e4d6b216519 */
    private DomainType domainType;
    public java.util.Collection frameworkRelationsB;
    /** @pdRoleInfo migr=no name=DomainContext assc=association3 coll=java.util.Collection impl=java.util.HashSet mult=* */
    public java.util.Collection<DomainContext> domainContext;
    /** @pdRoleInfo migr=no name=SoSHFControler assc=association6 mult=0..1 */
    public SoSHFControler soSHFControler;
    /** @pdRoleInfo migr=no name=SoSHFView assc=association7 coll=java.util.Collection impl=java.util.HashSet mult=0..* type=Aggregation */
    public java.util.Collection<SoSHFView> soSHFView;
    /** @pdRoleInfo migr=no name=SoSHFModel assc=association8 coll=java.util.Collection impl=java.util.HashSet mult=0..* */
    public java.util.Collection<SoSHFModel> soSHFModel;
    /** @pdRoleInfo migr=no name=RepositoryManager assc=association9 mult=0..1 */
    public RepositoryManager repositoryManager;
    /** @pdRoleInfo migr=no name=ServiceControler assc=association10 mult=0..1 */
    public ServiceControler serviceControler;
    public CooperatingDomains[] frameworkRelationsA;
    /** @param source
     * @param destination
     * @param context
     * @pdOid 744debe3-5548-4295-a6ea-328715b03caa */
    public SoSHyperFramework establisheRelations(SoSHyperFramework source,
    SoSHyperFramework destination, DomainContext context) {
        // TODO: implement
    }
    /** @pdOid 75f8bfff-a41b-47be-a366-c6b44131030d */
    public void performAction() {
        // TODO: implement
    }
    /** @pdGenerated default getter */
    public java.util.Collection<DomainContext> getDomainContext() {
        if (domainContext == null)
            domainContext = new java.util.HashSet<DomainContext>();
    }
}
```

```
return domainContext;
}
/** @pdGenerated default iterator getter */
public java.util.Iterator getIteratorDomainContext() {
    if (domainContext == null)
        domainContext = new java.util.HashSet<DomainContext>();
    return domainContext.iterator();
}
/** @pdGenerated default setter
 * @param newDomainContext */
public void setDomainContext(java.util.Collection<DomainContext>
newDomainContext) {
    removeAllDomainContext();
    for (java.util.Iterator iter = newDomainContext.iterator(); iter.hasNext();)
        addDomainContext((DomainContext)iter.next());
}
/** @pdGenerated default add
 * @param newDomainContext */
public void addDomainContext(DomainContext newDomainContext) {
    if (newDomainContext == null)
        return;
    if (this.domainContext == null)
        this.domainContext = new java.util.HashSet<DomainContext>();
    if (!this.domainContext.contains(newDomainContext))
        this.domainContext.add(newDomainContext);
}
/** @pdGenerated default remove
 * @param oldDomainContext */
public void removeDomainContext(DomainContext oldDomainContext) {
    if (oldDomainContext == null)
        return;
    if (this.domainContext != null)
        if (this.domainContext.contains(oldDomainContext))
            this.domainContext.remove(oldDomainContext);
}
/** @pdGenerated default removeAll */
public void removeAllDomainContext() {
    if (domainContext != null)
        domainContext.clear();
}
/** @pdGenerated default getter */
public java.util.Collection<SoSHFView> getSoSHFView() {
    if (soSHFView == null)
        soSHFView = new java.util.HashSet<SoSHFView>();
    return soSHFView;
}
/** @pdGenerated default iterator getter */
public java.util.Iterator getIteratorSoSHFView() {
    if (soSHFView == null)
        soSHFView = new java.util.HashSet<SoSHFView>();
```

```
        return soSHFView.iterator();
    }
    /** @pdGenerated default setter
     * @param newSoSHFView */
    public void setSoSHFView(java.util.Collection<SoSHFView> newSoSHFView) {
        removeAllSoSHFView();
        for (java.util.Iterator iter = newSoSHFView.iterator(); iter.hasNext(); )
            addSoSHFView((SoSHFView)iter.next());
    }
    /** @pdGenerated default add
     * @param newSoSHFView */
    public void addSoSHFView(SoSHFView newSoSHFView) {
        if (newSoSHFView == null)
            return;
        if (this.soSHFView == null)
            this.soSHFView = new java.util.HashSet<SoSHFView>();
        if (!this.soSHFView.contains(newSoSHFView))
            this.soSHFView.add(newSoSHFView);
    }
    /** @pdGenerated default remove
     * @param oldSoSHFView */
    public void removeSoSHFView(SoSHFView oldSoSHFView) {
        if (oldSoSHFView == null)
            return;
        if (this.soSHFView != null)
            if (this.soSHFView.contains(oldSoSHFView))
                this.soSHFView.remove(oldSoSHFView);
    }
    /** @pdGenerated default removeAll */
    public void removeAllSoSHFView() {
        if (soSHFView != null)
            soSHFView.clear();
    }
    /** @pdGenerated default getter */
    public java.util.Collection<SoSHFModel> getSoSHFModel() {
        if (soSHFModel == null)
            soSHFModel = new java.util.HashSet<SoSHFModel>();
        return soSHFModel;
    }
    /** @pdGenerated default iterator getter */
    public java.util.Iterator getIteratorSoSHFModel() {
        if (soSHFModel == null)
            soSHFModel = new java.util.HashSet<SoSHFModel>();
        return soSHFModel.iterator();
    }
    /** @pdGenerated default setter
     * @param newSoSHFModel */
    public void setSoSHFModel(java.util.Collection<SoSHFModel>
newSoSHFModel) {
        removeAllSoSHFModel();
    }
}
```

```
        for (java.util.Iterator iter = newSoSHFModel.iterator(); iter.hasNext();)
            addSoSHFModel((SoSHFModel)iter.next());
    }
    /** @pdGenerated default add
     * @param newSoSHFModel */
    public void addSoSHFModel(SoSHFModel newSoSHFModel) {
        if (newSoSHFModel == null)
            return;
        if (this.soSHFModel == null)
            this.soSHFModel = new java.util.HashSet<SoSHFModel>();
        if (!this.soSHFModel.contains(newSoSHFModel))
            this.soSHFModel.add(newSoSHFModel);
    }
    /** @pdGenerated default remove
     * @param oldSoSHFModel */
    public void removeSoSHFModel(SoSHFModel oldSoSHFModel) {
        if (oldSoSHFModel == null)
            return;
        if (this.soSHFModel != null)
            if (this.soSHFModel.contains(oldSoSHFModel))
                this.soSHFModel.remove(oldSoSHFModel);
    }
    /** @pdGenerated default removeAll */
    public void removeAllSoSHFModel() {
        if (soSHFModel != null)
            soSHFModel.clear();
    }
}
```

A.2 A nomenclature list

ADM	Architecture Development Method
DIKW	Data/Information/Knowledge/Wisdom pattern
HFM	Hyper-Framework Model
LDO	Large Data Object
LMM	Linear Mixed Modeling
MBSE	Model Based Systems Engineering
MDSD	Model-Driven Software Development
NoSQL	Not only Structured Query Language
PCA	The Principal Component Analysis
POS	The Physics of \open \systems
SoS	The System of Systems
SQL	Structured Query Language
XML	Extensible Markup Language

IntechOpen

Author details


Ana Perišić¹ and Branko Perišić^{2*}

1 Faculty of Technical Sciences, University of Novi Sad, Novi Sad, Serbia

2 University Singidunum Belgrade, Belgrade, Serbia

*Address all correspondence to: bperisic@singidunum.ac.rs

IntechOpen

© 2022 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/3.0>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. 

References

- [1] Gartner Glossary. Available from: <https://www.gartner.com> [Accessed: January 30, 2022]
- [2] Guide to the Systems Engineering Body of Knowledge (SEBoK), version 2.5. Available from: <https://www.sebokwiki.org/> [Accessed January 30, 2022]
- [3] Kachanova TL, Fomin BF, Fomin OB. Generating scientifically proven knowledge about ontology of open systems: Multidimensional knowledge-centric system analytics. In: *Ontology in Information Science*. London: IntechOpen; 2017. DOI: 10.5772/intechopen.72046
- [4] Carpi A, Anne EE. *The Practice of Science: An Introduction to Research Methods*. available from: <https://www.visionlearning.com/en/library/Process-of-Science/49/The-Practice-of-Science/148> [Accessed: January 30, 2022]
- [5] Schneegans S, Straza T, Lewis J, editors. *UNESCO Science Report: The Race Against Time for Smarter Development*. Paris: UNESCO Publishing; 2021
- [6] Complexity. Available from: <https://www.complexity-explorables.org/explorables/clustershuck/> [Accessed January 10, 2022]
- [7] Complexity. <https://www.complexity-explorables.org/explorables/knitworks/> [Accessed January 1, 2022]
- [8] Rossi R, Hiram K. Characterizing big data management. *Issues in Informing Science and Information Technology*. 2015;12:165-180
- [9] Klačnja-Milićević A, Ivanović M, Budimac Z. Data science in education: Big data and learning analytics. *Computer Applications in Engineering Education*. 2017;25(6):1066-1078. DOI: 10.1002/cae.21844
- [10] Ćwiek-Kupczyńska H, Filipiak K, Markiewicz A, et al. Semantic concept schema of the linear mixed model of experimental observations. *Science Data*. 2020;7:70. DOI: 10.1038/s41597-020-0409-7
- [11] Maryna Popova, Larysa Globa and Rina Novogrudska. Multilevel ontologies for big data analysis and processing, *Proceedings of the 9th International Conference on Applied Innovations in IT, (ICAIIT)*, 2021. Koethen (Germany), 28 April 2021
- [12] Davoudian A, Liu M. Big data systems: A software engineering perspective. *ACM Computing Surveys*. 2020;53:5
- [13] Konstantin Kostenko .2020. Knowledge flows processes at multidimensional intelligent systems, *Russian Conference on Artificial intelligence (RCAI 2020)*, October, 10–16, 2020, Moscow, Russia
- [14] Perišić A, Lazić M, Obradović R, et al. Daylight and urban morphology: A model for analysing the average annual illumination of residential housing. *Tehnički vjesnik*. 2016;23(5):1343-1350. DOI: 10.17559/TV-20150526191843
- [15] Jaadi Z. A Step-by-Step Explanation of Principal Component Analysis (PCA). Available from: <https://builtin.com/data-science/step-step-explanation-principal-component-analysis> [Accessed: January 20, 2022]
- [16] Keho Y. The basics of linear principal components analysis. In: *Sanguansat P*,

editor. *Principal Component Analysis*. London: InTech; 2012. Available from: <http://www.intechopen.com/books/principal-component-analysis/the-basics-of-principal-component-analysis>

[17] Sanguansat P. Two-dimensional principal component analysis and its extensions. In: Sanguansat P, editor. *Principal Component Analysis*. London: InTech; 2012. Available from: <http://www.intechopen.com/books/principal-component-analysis/2dpca-and-its-extensions>

[18] Guyon C, Bouwmans T, Zahzah E-h. Robust principal component analysis for background subtraction: Systematic evaluation and comparative analysis. In: Sanguansat P, editor. *Principal Component Analysis*. London: InTech; 2014. Available from: <http://www.intechopen.com/books/principal-component-analysis/robust-principal-component-analysis-for-background>

[19] Fan Z, Yong X, Zuo W, et al. Modified principal component analysis: An integration of multiple similarity subspace models. *IEEE Transactions on Neural Networks and Learning Systems*. 2014;**26**(8)

[20] Jolliffe IT, Cadima J. Principal component analysis: A review and recent developments. *Philosophical Transactions of Royal Society A*. 2016; **374**:20150202

[21] Han Y, Song G, Liu F, Geng Z, Ma B, Xu W. Fault monitoring using novel adaptive kernel principal component analysis integrating grey relational analysis. *Process Safety and Environmental Protection*, Volume. 2022;**157**:397-410

[22] Gong M, Miller C, Scott M, et al. State-space functional principal component analysis to identify

spatiotemporal patterns in remote sensing lake water quality. *Stoch Environment Res Risk Assessment*. 2021; **35**:2521-2536. DOI: 10.1007/s00477-021-02017-w

[23] Mustaqem M, Saqib M. Principal component based support vector machine (PC-SVM): A hybrid technique for software defect detection. *Cluster Computing*. 2021;**24**:2581-2595

[24] Rasheed J, Hameed AA, Djeddi C, et al. A machine learning-based framework for the diagnosis of COVID-19 from chest X-ray images, *Interdisciplinary Sciences: Computational. Life Sciences*. 2021;**13**:103-117

[25] Lazić M, Perišić A, Perišić B. Residential Buildings Complex Boundaries Generation Based on Spatial Grid System. *Appl. Sci*. 2022;**12**:165. DOI: 10.3390/app12010165

[26] Available from: <https://www.zachman.com/resources/ea-articles-reference/327-the-framework-for-enterprise-architecture-background-description-and-utility-by-john-a-zachman> [Accessed January 10, 2022]

[27] Svyatoslav Kotusev. A comparison of top four enterprise architecture frameworks. Available from: <https://www.bcs.org/articles-opinion-and-research/a-comparison-of-the-top-four-enterprise-architecture-frameworks/#16> [Accessed January 10, 2022]

[28] The Open Group. The Open Group Architecture Framework, (TOGAF). Available from: <https://www.opengroup.org/togaf>, <https://www.opengroup.org/architecture-forum> [Accessed January 1, 2022]

[29] FEA. Available from: <https://www.feainstitute.org/> [Accessed January 1, 2022]

[30] Carnegie Mellon University, Software Engineering Institute, Capability Maturity Model Integrated. Available from: <https://cmmiinstitute.com/> [Accessed January 1, 2022]

[31] Cook SC, Pratt JM. Advances in systems of systems engineering foundations and methodologies. *Australian Journal of Multi-Disciplinary Engineering*. 2021;**17**(1):9-22. DOI: 10.1080/14488388.2020.1809845

[32] Yearworth, M., Terry, AJ, Godfrey, PS., & Edwards, G. (2010). Systems thinking research: Principles, and methodologies to grapple with complex real-world problems. In *INCOSE UK Annual Systems Engineering Conference (ASEC 2010)*, Chipping Norton, Oxfordshire

[33] Potts MW, Sartor P, Johnson A, Bullock S. A network perspective on assessing system architectures: Foundations and challenges. *Systems Engineering*. 2019;**22**:485-501

[34] Salado A, Kannan H. Elemental patterns of verification strategies. *Systems Engineering*. 2019;**22**:370-388

[35] Knöös Franzén L, Staack I, Krus P, Jouannet C, Amadori KA. Breakdown of system of systems needs using architecture frameworks, Ontologies, and Description Logic Reasoning, *Aerospace*. 2021;**8**:118. DOI: 10.3390/aerospace8040118

[36] Perisic A, Lazic M, Perisic B. The extensible orchestration framework approach to collaborative design in architectural, urban and construction engineering. *Automation in Construction*. 2016;**71**:210-225. DOI: 10.1016/j.autcon.2016.08.005