Article

Not peer-reviewed version

Digital Transformation Hyper-Framework – the Meta-Specification Based Life Cycle Model, Architecture, Configuration, and Information Resources Handling

Ana Perisic and Branko Perisic *

Posted Date: 31 October 2024

doi: 10.20944/preprints202410.2523.v1

Keywords: digital transformation; systems engineering; system-of-systems; user experience; life cycle models; meta-modeling; information resource abstraction; hyper-framework

Article

Digital Transformation Hyper-Framework—The Meta-Specification Based Life Cycle Model, Architecture, Configuration, and Information Resources Handling

Ana Perisic [1] and Branko Perisic [2,*]

[1] University of Novi Sad, Faculty of Technical Sciences, Novi Sad, Serbia; anaperisic@uns.ac.rs
[2] Independent Researcher, Novi Sad, Serbia
* Correspondence: perisic@uns.ac.rs; Tel.: +381-64-134-0-173

Abstract: To cope with the complexity, the digital transformation of cyber-physical and socio-technology systems demands the utilization of heterogeneous tailorable development environments with dynamic configuring ability and transparent integration of independently developed dedicated frameworks. Hyper-frameworks, aka Framework-of-Frameworks, correspond to the System-of-Systems paradigm and designate a software-supported, meta-model-based generator of executable configurations. Armed with a meta-modeling layer, the incremental development of hybrid architecture instances focuses on language-based models and their transformations into functional, interpretable environments. Data-driven reasoning and decision support for the entire life cycle of digitally transformed systems critically depends on the effectiveness of involved generative mechanisms and the trustworthiness of related information resources. The Architecture, Meta-Configuration, and Information Resource Handling are the essential segments of the initial version of the proposed evolution prototype. In this article, we formulate five essential hypotheses founding the proposed digital transformation hyper-framework project that integrates the framework model with a meta-configuration-based generative mechanism. The Life Cycle Modeling component framework creatively integrates System, Software, and Operation Engineering. Specification and development of the additional component frameworks, in compliance with specified generative mechanisms, direct further refinements of the proposed Hyper-framework.

Keywords: digital transformation; systems engineering; system-of-systems; user experience; life cycle models; meta-modeling; information resource abstraction; hyper-framework

1. Introduction

The historical road to digital transformation (DGT) assumes the methodology and technical approaches that tend to remove the virtual line, separating the actual system operation from the operation of its digitally empowered version. Although digital transformation opens the possibilities for innovative approaches to reengineering products and services, it often represents a disruptive mechanism that questions the existing organizational business models, topology, and well-established operation modes. It dynamically associates a concrete real-world organizational system and its virtual equivalent according to the digital sophistication level of involved, coarse-grained stakeholders appearing as subjects (external or internal active participants) or objects (internal, transformation-prone participants) of a digital transformation endeavor. The Digital Engineering (DE) paradigm establishes an enterprise as a socio-cultural-technical system and transforms the Systems Engineering (SE) practices toward technology innovation drivers [1]. It assumes the collaboration/cooperation of all relevant stakeholders over a Digital Engineering Systemic

Framework that integrates related resources and activities throughout the entire life cycle of a transformed system. DE goes beyond using specific tools and models and frames a culture of innovation, collaborative problem-solving, and continuous improvement [2]. The current orientation towards DGT additionally originates from the vision of the so-called Great Reset, which addresses the essential domains of global crises and urgent needs for a systematic and sustainable globally synchronized response [3]. Unfortunately, recent research shows that the main obstacles come from the inherent complexity of global systems and different approaches to economic and political perspectives [4].

On the other hand, DGT research has a transdisciplinary nature and relies on the Transdisciplinary Systems Research Methodologies (TSRM). Instead of directly implicating a solution, TSRMs aim to provide the necessary context to understand, evaluate, and argue for or against many possible solutions, narrowing down the solution space to specify the directions for future implementation actions [5]. The uncertainties related to the DGT success do not highly correlate with the existence of a solution but, more significantly, with an implementation failure.

The most influencing DT drivers are closely related to recent industrial revolution frameworks, starting from the three-dimensional Reference Architectural Model Industry 4.0 (RAMI 4.0) and its virtual description of production or service objects with Architecture (interoperability layers), Life Cycle, and Value Stream (differentiating product type and product instances), and Hierarchy (hosting of the production process) dimensions [6] (Figure 1).

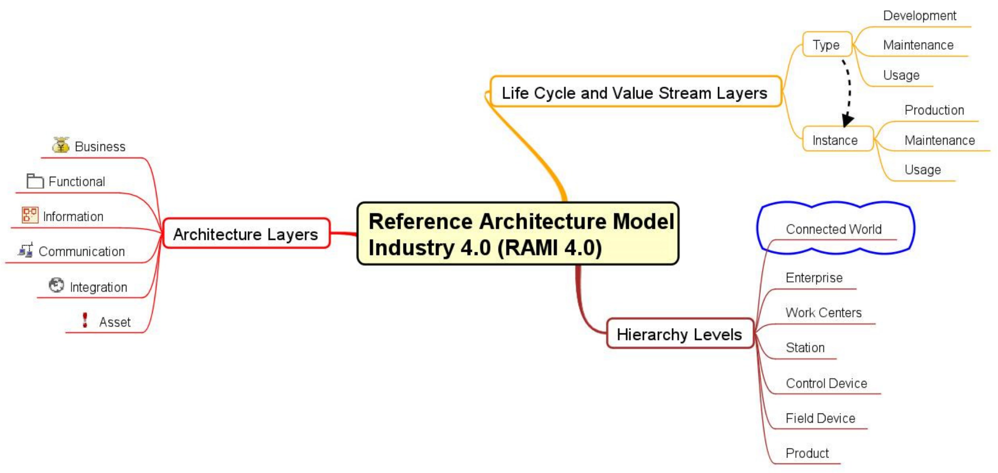


Figure 1. Reference Architectural Model Industry 4.0—Mind map diagram (adapted from [6]).

Industry 4.0 frames a digital transformation from traditional to cyber-physical-systems (CPS) with economic benefits in mind. It utilizes innovative methods and technological means to upgrade the organizational system resources (organizational structure, business models, human resources, processes, and assets). Therefore, the existence of an intelligent enterprise information system aids the continuous rise of the system's maturity level. That is why developing an assessment tool for digital transformation progress tracking, aligned with the Industry 4.0 drivers, appears mandatory [7]. The following essential technology pillars appear as supportive for digital transformation towards Industry 4.0: Big Data and Analytics, Digital Twins and Simulation, Industrial Internet of Things, Augmented Reality, Autonomous Robots, Horizontal and Vertical Software Integration, Cybersecurity, and Cloud-based Additive Manufacturing [8]. While Industry 4.0 assumes intelligent manufacturing with variable degrees of strategic impacts on business model transformation, the importance of intelligent support in the decision-making process, based on the embedded fuzziness of key Industry 4.0 transformation success factors, emerges [9]. In [10], the authors claim that currently, Industry 4.0 concepts are mainly adopted in large companies, and thereby propose a sustainable methodology suitable for Small and Medium Enterprises (SMEs) based on a specified conceptual framework, performance measurement structure synchronized with the continuous transformation achievements, with the particular emphasize on an intelligent system supporting SMEs in digital transformation courses.

On the other hand, Industry 5.0 returns the essential human values and vital social needs on the DGT scene with the recognized duality of virtual and real-world systems. The transition from Industry 4.0 to Industry 5.0 exhibits a four-dimensional paradigm shift: from Information to Intelligence, from Communication to Connectivity, from Cyber-Physical to Cyber-Physical-Social systems, and from Physical Automation to Knowledge Automation [11]. The essential novel technologies associated with the Industry 5.0 paradigm include Blockchain, Autonomous Area Vehicles, 5G&6G Wireless Networks, Exoskeleton, Mixed Reality, Additive Manufacturing, Artificial Internet of Things, Motion Capture, and Digital Twin [12]. Current literature raises a challenging dilemma of whether Industry 5.0 complements, replaces, or transparently integrates with Industry 4.0. The comparative analysis of evolutionary trajectories of Industry 4.0 and Industry 5.0 from the management perspectives argues the need for revolutionary changes in a relatively short period [13], while [14] elaborates on three different stages of Industry 5.0 perspectives. From the assessment point of view, there is a lack of novel enterprise maturity models that augment well-documented Industry 4.0 maturity models with Industry 5.0 novel dimensions [15]. In [16], the authors suggest a hybrid solution that leverages the high automation level of decision-making concepts introduced by Industry 4.0 and the leading role of humans, reactivated by Industry 5.0, based on a collective (hybrid) intelligence as the decision-making driver.

Further propositions concerning novel industry revolutions, like Industry 6.0, represent symbiotic integration of technological transformation, production chain innovation, and social sustainability progression towards the Smart Factory 6.0 paradigm as a sophisticated automated manufacturing facility that integrates emerging technologies and digital systems. Smart Factory 6.0 reflects the upcoming wave of cognitive manufacturing [17] and sublimates IoT, AI, ML, robotics, the Blockchain, big data analytics, quantum computing, and cloud computing [18–20] technologies.

On the other hand, Industries beyond 6.0, announced by arbitrary predictive visions, introduce significant speculative assumptions reflecting turbulences emerging from the contemporary scientific, educational, technology, and social contexts. In Figure 2, a mind map diagram correlates different inclusive technologies related to discussed paradigms that challenge ongoing and future digital transformation projects.

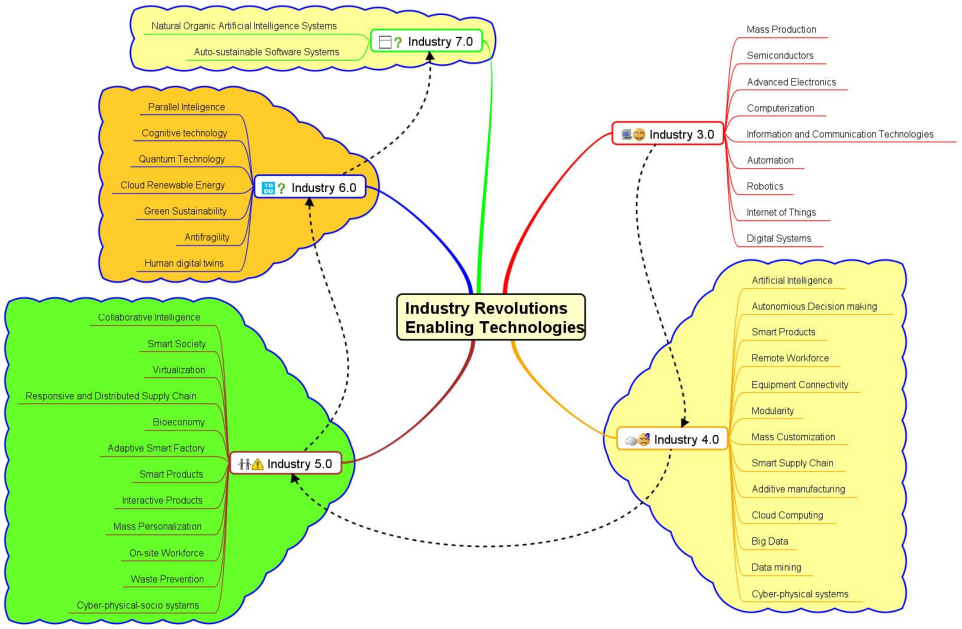


Figure 2. The Industrial Revolutions Enabling Technologies (derived from [5–20]).

Regarding organizational systems' digital sophistication, two main aspects direct their ability to cope with digitalization and digital transformation: the strategy (higher-order capabilities) and the operation (low-order capabilities). They rely on available organizational resources to guide their transformation and achieve the desired outcome of a transformed system [21].

The development of universal classification and the general typology of organizational systems concerning their DGT abilities has been mainstream in recent scientific research publications. Being business transformation-driven, it has promoted the Business Process Modeling (BPM) foundation for strategic and operational activities. The BPM framework approach to digital transformation, elaborated in [22], defines six core elements of successful BPM initiatives: Strategic Alignment, Governance, Method, Information technology, People, and Culture, and proposes a set of corresponding recommendations derived through critical analysis of chosen large companies' DGT projects. The generic and specific practices embedded in the proposed recommendations have strongly influenced the digital transformation hyper-framework requirements model building. Each core element represents a challenging domain-specific digital transformation dimension that deserves full-scale independent support. Configuration-based integration or orchestration of individual domain-specific elements (components or frameworks) enables need-to-have dynamic binding optimized for the particular transformation context.

The existence of the appropriate malty-dimensional strategy that guides the creation and manages the conducting of transformation initiatives represents the foundation for a supportive digital transformation canvas [23]. Figure 3 shows the 11 P's (Purpose, Process, Partner, Platform, People, Project, Product, Performance, Planet, Protection, and Privacy) of the digital transformation strategy foundation, grouped into four pillars (Strategy, Operation, Value, and Pitfalls), adapted and modeled as a mind-map diagram. Besides the component-based definition of DGT processes that may constitute an opened digital transformation life cycle model (ODGTLCM), the embedded, role-profiled stakeholders' support through software-empowered solutions is essential for effective collaborative/cooperative operational framework development.

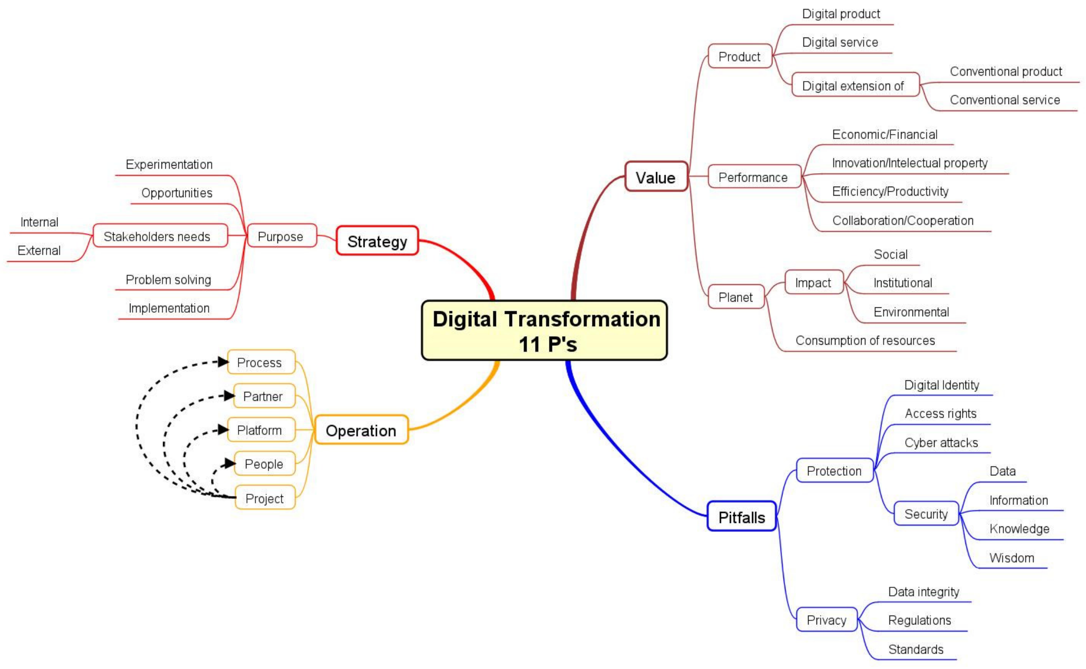


Figure 3. Digital transformation strategy foundation (adapted from [23]).

The dynamic nature of DGT justifies the current shift to an agile methodology that better suits iterative or incremental development, continuous delivery, customer-priority-driven collaboration/cooperation, and continuous adaptation to change [24]. On the other hand, DGT is not possible if all systems-wide required resources (produced, disseminated, stored, retrieved, and consumed in any different sense) are not digitally twinned. A digitally transformed system is not a system with embedded IT support (composite pattern). It is a new system (decorator pattern) with a different identity and radically changed structure and behavior. Digital transformation endeavors have shown that exclusively focusing on information technologies (IT) supporting the transformation is not enough to guarantee effective and evolutionary transformation and sustainable operation through the entire life cycle of a transformed system. Information technologies, combined with the strategic importance of ethics, introduce a broader sense of systems value network and split the uncertainties into two main categories: external and internal. In addition to the organization's decision-making mechanisms, with the digital transformation, the impetus moves to society and industry trends [25].

The importance of organizational culture and people emphasizes the social and organizational aspects that deserve careful analysis before defining the strategic and operational foundations for evolutionary transformation activities guided by the appropriate methodology selection. The categorization of different organizational cultures enables the grouping of potential type-related enablers and disablers that may affect the entire life cycle of a digitally transformed system [26]. The positive synergy between IT and human resource management (HRM) strategy is empirically recognized as a DGT booster [27] influencing future strategy formulation. The digital human resource management (DHRM) strategy combined with the organizational culture typology results in the integrated typology model [28] that better reflects the nondeterministic nature of socio-technology systems.

The existing literature has thoroughly examined the influence of DGT on external stakeholders with a focus on customer relationships, but the crucial importance of internal stakeholders' (workforce) digital proficiency appears less rigorously addressed. By fostering a culture of collaboration/cooperation, empowerment, and innovation, organizational systems can raise the collective intelligence and creativity of its internal stakeholders and force them to seize emerging opportunities of fostering individual digital literacy and digital readiness and aid the certainty of the overall DGT process in organizations [29].

With human-operated/used systems, success depends on the minimal distance between problem-domain concepts and activities and their digital equivalents. This distance is expressed through two quite different but often interchangeably used terms: User Interface (UI) and User Experience (UX). User Interface addresses the observable (visible) representations of a product, application, system, service, or function serving the purpose of interaction. It is traditionally recognized as a human-computer interaction (HCI), or more generally, human-machine interaction (HMI) engineering domain that has experienced significant changes in the conceptual foundation throughout the entire development history. Initially, general human cognition factors solely directed the design. Later on, user-centric contextual specializations augmented them. With the inclusion of social and cultural impact mitigation, current professional ethics, and fundamental moral principles, the UI design has evolved into a multidisciplinary endeavor [30]. User Experience (UX) encapsulates UI and relates to the end-user's subjective perceptions experienced through the entire scenario, directing the use of a product, system, service, or function developed by a particular vendor and disseminated as UI usable objects. The scenario includes pre-use, use, and post-use phases that, joined together, build the subjective feeling of a used object. The proliferation of AI and Machine learning technologies has opened novel challenges to UI design, adding profile-based dynamic configuration abilities [31,32], shifting from tangible to intangible interactions [33], and integrating emotional intelligence and interaction optimization supporting highly diverse and personalized interactions [34]. Digital Experience Platforms [35], Generative Artificial Intelligence [36,37], Virtual and Augmented reality paradigms additionally foster the UX foundation [38,39].

The highest transformation quality of interactive systems emerges when the interaction mechanisms sequencing appear natural in fulfilling the intended user's mission, not solely as the elements of the application's Graphical User Interface (GUI). The proper balancing of human-directed and system-directed activities while operating complex cyber-physical and socio-technology systems is of great importance for their acceptance and sustainable operation through the entire system's lifecycle. On the other hand, with the tight incorporation of contemporary information technologies and the growing tendency to minimize human-related dependencies, total automation appears as the Holy Grail. The level and efficiency of automation depend on the reactive coupling of UX and UI objects. The automatic generation of UI demands the extendible GUI library support for UI building blocks and the directed UX feedback that synchronizes UI-related meta-models with actual experiences gained through operational usage. The reflective synchronization of the actual twin and its virtual counterpart represents one of the main aspects of the digital twining paradigm [40].

With digital transformation, it is essential to rely on life-long integrated support that does not separate systems design, software design, and Information Technology operating infrastructure. The successful DGT never ends. It evolutionarily transforms the actual system and the corresponding digital equivalent according to the built-in ability to adapt and operate in the context of any future transition. That is why the sustainable engineering/reengineering of cyber-physical and socio-technology systems assumes the existence of powerful, tool-based support applied through the entire life cycle at arbitrary abstraction levels or development stages. Consequently, the Systems Engineering (SyEng), Software Engineering (SwEng), and Operations Engineering (OpEng) domains represent three cornerstones in the DGT process of complex cyber-physical and cyber-social systems. The well-balanced approach to process and product synergy enables the continuous assessment of the engineering process maturity level and quality management, flexible product architecting, and the effective collaboration/cooperation of the real-world (Actual) system and its Computer-supported Virtual reflection. Agile development emerged as a promising approach based on the higher flexibility, continuous delivery of small, usable systems increments, and better response to change management [41]. With the emphasis on agile framework development, machine learning (ML), and artificial intelligence (AI) support, the automation of routine tasks appears feasible.

As an additional challenge, digitally transformed systems exhibit a dramatic growth in volumes and diversity of generated, stored, retrieved, and processed data. The proliferation of the Internet of Things (IoT) and the Digital Twin paradigm introduce the real-time dimension of acquired and stored data instances uncommon with conventional enterprise systems. They demand more sophisticated data engineering efforts for storing and retrieving data and the higher involvement of machine learning (ML) and artificial intelligence (AI) powered framework technologies to support valuable data analytics [42]. The Internet of Things (IoT) and the Digital Twin paradigm face standardization problems due to many vendors distributing IoT devices. This diversity causes variations in communication protocols, data models, data exchange formats, and security requirements. Therefore, the development of IoT-based systems ends with a higher effort and, due to the characteristics of individual application domains, results in a lower reusability level and higher extendibility risks [43]. The Web-of-Things (WoT) paradigm enables interoperability over IoT platforms by abstracting WoT technology building blocks through properties, events, and actions [44]. The importance of reliable and trustworthy infrastructure supporting the innovative business ecosystem assumes modern technologies used in fifth-generation (5G) networks through slicing, virtualization, and blockchain [45].

Measuring the success level of DGT faces the potential paradox that the transformation process of a successful system never ends. Everything is transient except change (transformation), which is eternal and eternally drives the system's transformations. The successful DGT involves complex engineering/reengineering activities framed by three synergic dimensions: forward-looking (vision, specification, modeling, design, implementation activities, methods, techniques, and platforms), backward-looking (product verification and validation), and downward-looking (process monitoring and control). Coping with the inherent complexity, it demands the utilization of corresponding Digital Transformation Environments (DGTEs) with dynamic configuring ability and transparent integration of independently developed dedicated frameworks [46–48].

The essential motivation factors for our research directions originate from the Editor's Corner section of [49], where the Editor in Chief estimates the common-values-motivated drivers that will eliminate previously experienced obstacles and support the future challenges towards a total digital transformation. Consequently, the synergic support for DGT drivers assumes a lined, evolutionary process that requires a radical shift in engineering methodology, organizational culture, and expectations.

All engineering activities exhibit the duality of the engineering process and the engineered product. Model-Based Engineering (MBE) represents an engineering approach resulting in a hyper-model instance that digitally connects different types of engineering artifacts created during various engineering process steps, establishes forward and backward artifacts traceability, and minimizes the risks and effort in change management activities. MBE relies on the formal modeling languages that support specification, modeling, simulation, emulation, design, analysis, testing, validation, and verification of a virtual twin representing the structure and behavior of an engineered system. The underpinned formalisms enable the implementation of tailorable workbenches (software tools) providing generic services such as edition, visualization, transformation, comparison, storage, retrieval, export, import, and additional operations on virtual twin instances.

The primary mission of this research article is to specify a minimal subset of requirements that determine the development of an extendible, hyper-framework, all-in-one, software-empowered prototype of DGTE that suites the essential characteristics of a meta-specification-driven digital transformation supporting tool. In the context of this research, we have formulated the following five hypotheses.

The first hypothesis (H1) is that an interoperable multi-staged, Digital Twin-based, hyper-framework that interrelates Systems Engineering (SyEng), Software Engineering (SwEng), and Operation Engineering (OpEng) features into a collaborative/cooperative specification, development, and operational environment based on an extendible library of tailorable life cycle models (TLCMs) is essential.

The second hypothesis (H2) is that the hyper-framework has to mandatory support the stage-based creation of different configurations, ranging from individual services to system-of-systems, that form the executive infrastructure reflecting the events and services supported by the particular stage, independent of their digitalization status.

The third hypothesis (H3) is that each configuration instance has to represent an executable virtual system capable of real-time emulation (imitate with another system, surrogate), simulation (model-based prediction), operation (traceable running), monitoring (noninvasive and secured), modeling, decision-support mechanism, access (User Interface), and evaluation (User Experience) of configured components structure and behavior.

The fourth hypothesis (H4) is that each component needs to rely on abstract information resource conceptualization, hiding the persistency layer typology from the dynamic resource management layer and establishing the foundations for Data Engineering (DatEng) and Data Analytics (DatAna) features.

The fifth hypothesis (H5) is that the explicit support for H1, H2, H3, and H4 has to be a meta-specification driven and integrated into the DGT hyper-framework (DGTHyF). Meta-specifications, interpreted by the strategy pattern with extendible Generative Artificial Intelligence-supported concrete strategies, incline to Industry 6.0 and Beyond.

Concerning the fact that even the longest journey starts with a first step and faces the coexistence of different approaches and solutions, we hope the stated hypotheses suitably frame a promising one.

The rest of the article contains four sections. Section 2, Materials and Methods, elaborates on the research foundation of the stated hypothesis. Section 3, Results, introduces the main conceptual characteristics of the proposed hyper-framework, meta-specifications that enable dynamic handling of the different life cycle models, Architecture Foundation, Dynamic configuration and reconfiguration of participating components, and the generic support to extendible heterogeneous information resources handling. In Section 4, we discuss and cross-relate the analyzed references to justify the appropriateness of the proposed hyper-framework. Section 5 contains the concluding remarks and the future research directions.

2. Materials and Methods

The accurate qualification of the proposed research with the distinction between multidisciplinary, interdisciplinary, and transdisciplinary approaches is essential. Relying on [50], in the context of this article, we have formulated these differences as follows. An approach is multidisciplinary if a research team concurrently and collaboratively studies the same phenomenon from multiple domains but without the integration of individual insights, thus remaining in the particular phenomenon domain. An approach is interdisciplinary if it is multidisciplinary, and a research team, to better understand a researched concept or phenomenon, integrates individual insights coming from the different disciplines. An approach is transdisciplinary if it augments the research team by the stakeholder groups outside the research team, creating a mix of collaborative and cooperative environments and steps beyond the particular phenomenon domain.

According to findings elaborated in the introductory section, DGT correlates different complex issues that naturally belong to multiple scientific or pragmatic domains and thereby inclines to the transdisciplinary category. This classification demands a balanced approach to specification and development of a sustainable hyper-framework leveraging unrestricted cooperative/collaborative set of domain-specific hyper-frameworks.

The proposed set of hypotheses sublimates global dimensions that direct the specification and development of the DGTEs family (a hyper-framework following the generic Quintuple helix model adopted from [48] and presented in Figure 4). The individual family members are either fabricated according to the course-grained abstract factory or incrementally built by the orchestration strategy of a generative-builder creation pattern. Concerning their mission, we find a detailed elaboration of underlying foundations mandatory.

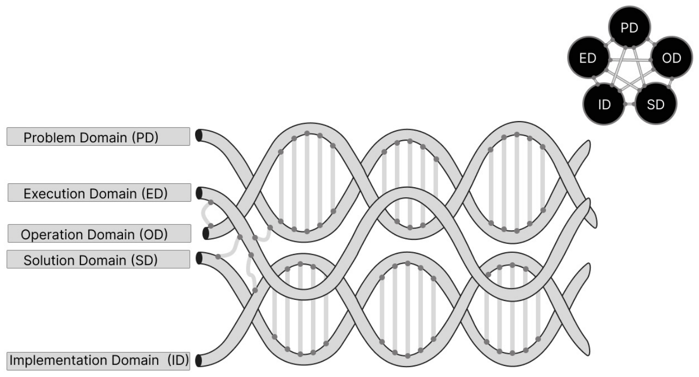


Figure 4. The Quintuple Helix model og generic hyperframework (adopted from [48]).

2.1. The Foundation of Hypothesis 1 (H1)—Integrated Tailorable Life Cycle Models

The first hypothesis addresses the most challenging aspect of the proposed approach and suggests horizontal and vertical integration of traditionally disjoined domains. Although systems engineering and software engineering are sometimes interchangeably used, traditionally, there has been a mutual agreement that it is necessary to distinguish them by the separate life cycle models. Additionally, the complexity of the supportive infrastructure design and operation emerges from the technology-prone engineering disciplines (computer networks and communication) and demands additional expertise to mitigate it.

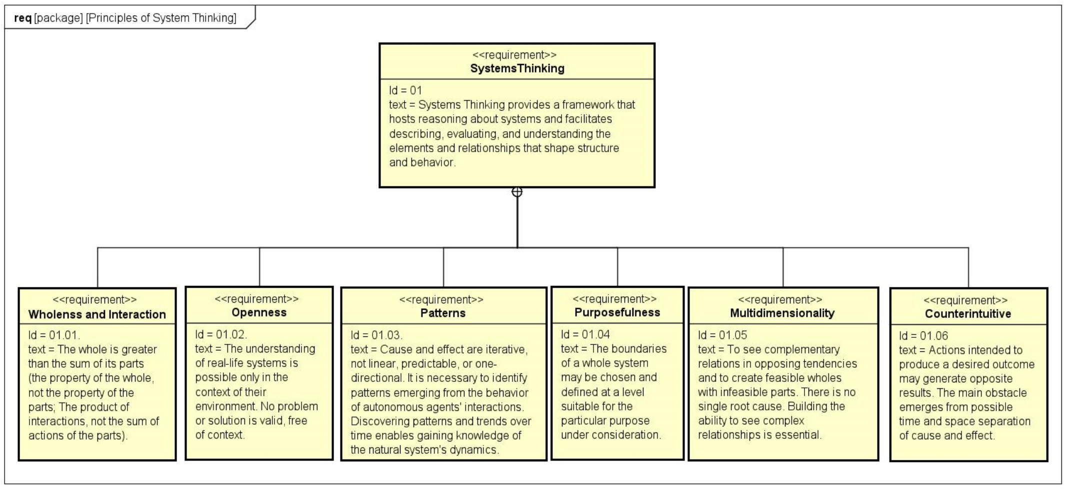
Contemporary systems, software, and operations engineering methodologies rely on the underpinned process-oriented life cycle models (LCMs) specified, developed, and refined throughout the entire history of engineering. They break down engineering activities into manageable groups, streamlining and systematizing their progression from conception to completion and maintenance, operation to migration, and retirement. *Focusing on DGT, a suitable business plan moderating the transformation context apostrophes a lean approach [51].* The incorporation of LCMs in the contemporary context of Model-Based-Systems-Engineering (MBSE) modeling formalisms (SysML) through cross-relating with ISO/IEC/IEEE 15288 standard, results by the proposition of 15288-SysML Grid framework, elaborated in [52], where the authors formulate the guidelines that frame SysML models creation by the 15288 specified processes.

The first hypothesis advocates a need for an interoperable multi-staged, Digital Twin-based, hyper-framework hosting the extendible library of collaborative/cooperative life cycle meta-models that integrate Systems Engineering (SyEng), Software Engineering (SwEng), and Operation Engineering (OpEng) activities. Based on a generic artificial intelligence and the information resources harvested from the persistent data layer, individual meta-models transform into particular DGT context instances.

2.1.1. The Role of Systems Engineering (SyEng)

The multidisciplinary nature of traditional Systems Engineering (SyEng) emerges from the Systems Engineering Body of Knowledge (SEBoK) [53], particularly the SyEng Foundation Knowledge Areas (KAs) (Systems Fundamentals, The Nature of Systems, Systems Science, Systems Thinking, Representing Systems with Models, and Systems Approach Applied to Engineered Systems), Enterprise Systems Engineering KAs (Creating Value, Resource Optimization, enabling Systems Engineering in the Organization, Kinds of Knowledge used by the Enterprise, Projects Programs and Business), Systems of Systems (SoS) KAs, Human Systems Integration KAs, and the variety of associated standards [54]. They specify SyEng as the overall range of knowledge, skills, competencies, and activities appearing, either on the large-scale (socio-technical systems on a national, international, transnational, or global scale) or locally scaled systems with either technology-driven or scope-driven complexity [55]. Systems Engineering focuses on the design, integration, deployment, operation, and management aspects of complex cyber-physical or socio-technology systems from their vision to the retirement stage [56]. In the context of digital transformation, SyEng role is to creatively link the problem and operation domains of the digitally transformed system (real twin) with the corresponding model (virtual twin) that reflects the current DGT context and stage.

The systems thinking principles appear promising in creating a synergy between engineering and management activities [57]. Systems Thinking provides a framework that hosts reasoning about systems and facilitates describing, evaluating, and understanding the elements and relationships that shape their structure and behavior over six principles, adapted from [58] and presented in the Systems Modeling Language (SysML) requirements diagram (Figure 5). The contemporary theory of systems streams to the System-of-systems (SoS) paradigm and inaugurates system-of-systems thinking that, in a way, augments systems thinking [59,60].



Id = 01
text = Systems Thinking provides a framework that hosts reasoning about systems and facilitates describing, evaluating, and understanding the elements and relationships that shape structure and behavior.

Id = 01.01.
text = The whole is greater than the sum of its parts (the property of the whole, not the property of the parts; The product of interactions, not the sum of actions of the parts).

Id = 01.02.
text = The understanding of real-life systems is possible only in the context of their environment. No problem or solution is valid, free of context.

Id = 01.03.
text = Cause and effect are iterative, not linear, predictable, or one-directional. It is necessary to identify patterns emerging from the behavior of autonomous agents' interactions. Discovering patterns and trends over time enables gaining knowledge of the natural system's dynamics.

Id = 01.04
text = The boundaries of a whole system may be chosen and defined at a level suitable for the particular purpose under consideration.

Id = 01.05
text = To see complementary relations in opposing tendencies and to create feasible wholes with infeasible parts. There is no single root cause. Building the ability to see complex relationships is essential.

Id = 01.06
text = Actions intended to produce a desired outcome may generate opposite results. The main obstacle emerges from possible time and space separation of cause and effect.

Figure 5. The six principles of Systems Thinking (adapted from [58] as SysML requirements diagram).

The emerging concept of integrated cyber-socio-technology-system-of-systems (CSTSoS) transforms traditional systems engineering from a multidisciplinary to a transdisciplinary research domain and thereby justifies our approach to DGT support through the hyper-framework. Additionally, with the proliferation of Artificial Intelligence (AI) and Machine Learning (ML), the Autonomous system emerges as an engineered system that leverages AI algorithms to perform specific tasks with or without human involvement. The evolving symbiotic relationship between AI development and systems engineering based on Large Language Models (LLMs) opens a new era of engineering innovation characterized by collaborative partnerships between humans and AI [61]. The specification and modeling of functional requirements for software-supported human activities, besides the system's nonfunctional requirements, significantly depend on human-related constraints that, if not satisfied, risk operational acceptance [62].

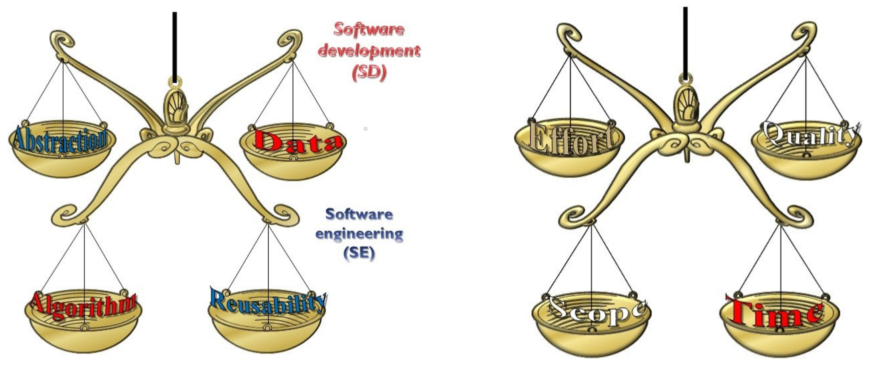
Regarding the Level of Autonomy (LoA), further augmentation of the SoS concept to the systems-of-autonomous-systems (SoAS) appears [63–65]. The novel concepts discussed above apostrophe the challenging issue of SoS dynamic interoperability [66]. Model-Based-Systems-Engineering (MBSE) mathematical foundation review elaborated in [67] aids the maturity and rigor of model-based DGT.

Consequently, the role of systems engineering in the specification of integrated tailorable life cycle models reflects concepts, methods, and tools that enable the creation of problem and operation domains vision of a virtual twin, aka Systems Engineering Vision, that frames the global aspects of contemporary DGTs. According to the conceptual framework model proposed in [48] (Figure 4), Systems Engineering maps to the Problem and Operation helixes.

2.1.2. The Role of Software Engineering (SwEng)

The multidisciplinary nature of Software Engineering (SwEng) emerges from the Software Engineering Body of Knowledge (SWEBOK) [68] Key Knowledge Areas (KAs) (Software Requirements, Software Design, Software Construction, Software Testing, Software Maintenance, Software Configuration Management, Software Engineering Management, Software Engineering Process, Software Engineering Models and Methods, Software Quality), the Related KAs (Software Engineering Professional Practices, Software Engineering Economics, Computing Fundamentals, Mathematical Fundamentals, and Engineering Fundamentals), and the variety of associated Software Engineering Standards defined by the mainstream standardization organizations (the Institute of Electrical and Electronic Engineers (IEEE), and the International Organization for Standardization (ISO)) [69]. The recent modification of SWEBOK V.3.0 preliminary announced as the SWEBOK V.4.0 document is, unfortunately, not publically available at this research preparation time.

Software engineering has been in the field for almost 30 years. Although recognized as a stand-alone professional discipline, distinct from computer science and other engineering disciplines, it is still considered relatively young. The criteria for gaining a maturity profession status, established by [70], with minor adaptation and refinement, form a maturity-assessment class model (Figure 6). There is a mutual agreement that the essential condition of the prospective maturity evolution for an arbitrary profession assumes the long-range stability of a mainstream doctrine that persists independent of digital technology progress (for example, Medicine, Linguistics, and Mathematics). The current research streams show no mutual agreement on the software engineering profession's maturity status.



(**a**) (**b**)

Figure 6. Software product and process relations (**a**) Software product invariants; (**b**) Software process invariants.

In [71], the author questions the possibility of software engineering's current and future maturing abilities. The uncertainties primarily come from the emergent nature of the SWEBoK knowledge areas. Besides, its engineering segment relies on a dynamic set of rules and logical principles suitable for human-less automation based on large language models (LLMs), significantly removing the proclaimed professional ingenuity. The future development of digital technology is highly unpredictable, causing instability in supportive infrastructure. The lack of a comprehensive legal and ethical framework covering all application areas, including the challenging dimensions of AI, is an additional obstacle. Regarding the maturity status metrics, we claim that the lack of globally accepted certification and licensing mechanisms additionally leads to prognostic uncertainty.

Research Software Engineering (RSE) has been announced as an emerging profession that highly depends on the stability of novel computing technologies and their adoption by the other dependent professions [71]. Consequently, the proliferation of quantum software engineering (QSE), quantum software life cycle development, and quantum software reuse methodology add challenging concepts to the future software engineering profession. They are recognized as highly engaged research topics, resulting in novel theories and techniques regularly appearing in contemporary publications [72]. Software architecture and patterns for quantum computing systems [73], software engineering issues [74], and quantum communications [75] represent typical examples of challenging domains that will affect software engineering's role in framing future DGTs. Actual software development for quantum computing exhibits difficulties due to a lack of appropriate methods and insufficiently scalable technology [76]. The novelty of RSE and QSE represents the challenges to current automatic code generation and automated software engineering support through more sophisticated frameworks and tools. Incorporating component-based software engineering (CBSE) principles in quantum algorithm automation is proposed in [77] as a promising approach to easier development, testing, and optimization of quantum algorithms for different problem domains.

A clear distinction between software engineering and software development aids the overall understanding of Model-Driven-Software-Development (MDSD), Model-Driven-Software-Engineering (MDSE), and the associated LCMs that address software process management (Figure 7). Software engineering is generally closer to systems engineering, assuming that software artifact represents a system. It perceives software as a final product of interleaved engineering activities with a broader scope than just particular software artifacts development. Consequently, software development closely relates to program development, often called programming methodologies.

According to our experience, the only invariant through the entire history of Software development (SD) (aka programming) is the eternal balancing between Algorithm and Data complexity, presented as the opposite ends of an exotic scale's arm (Figure 6 a), meaning that to decrease the unmanageable data complexity, reached in a particular programming stage, the only possible solution is to increase the Algorithmic complexity and vice versa. Accordingly, the only invariant throughout the entire history of Software engineering (SE) is the eternal balancing between the Abstraction level and the degree of Reusability, meaning that for the sake of higher Reusability, the abstraction level has to decrease and vice versa. Considering that SE encapsulates SD, the overall complexity emerges from the nonlinear synergy of all four dimensions (Algorithm, Data, Abstraction, and Reusability).

On the other hand, the pares (Effort, Time) and (Quality, Scope) (Figure 6 b) represent the essential invariant dimensions of Software process management. The Effort/Time scale arm interrelates two unavoidable project management assets, Budget(Effort) and Deadline (Time), while the (Quality, Scope) scale arm reflects the individual quality and number of delivered services in a particular project stage. Different assessment contexts may result from the universal obstacles during software project management. Assuming, for example, that the quality is not negotiable (aka mutually agreed), with a fixed deadline and budget, when the project runs into problems, the only possible solution to preserve the fixed ends is to negotiate on decreasing the scope.

Global Software Development (GSD) trends assume the fastest time-to-market, minimal costs, and high-quality software development foundation. GSD relies on component-based software development through the interoperable orchestration of individual instances harvested from commercially available component libraries (Component-of-the-Shelf—COTS and Open-Source-Software components) and the synchronous or asynchronous collaboration/cooperation through a distributed network of globally available experts. The success depends on a commonly understood and agreed software integration model (SIM) that guides the integration process of independently developed components [78].

Consequently, the role of Software Engineering in the specification of integrated tailorable life cycle models reflects the virtual world that emerges from the transformation (refinement) of Systems Engineering Vision into a digital system delegate or digital ecosystem. Existing digital ecosystems consist of complex networks of interconnected and globally distributed software-supported assets (systems), adding new value through the transparent and continuous delivery of the embedded system's services. According to the conceptual framework model proposed in [48] (Figure 4), Software Engineering maps to the Solution and Implementation helixes.

2.1.3. The Role of Operations Engineering (OpEng)

In the context of this research, Operations Engineering (OpEng) is a multidisciplinary field that utilizes operational thinking and skills to establish, support, optimize, and maintain the technical infrastructure of complex cyber-physical or socio-technology systems and uphold standards and internal documents and procedures that regulate structure and behavior of the operational environment. Generally, it represents system-of-infrastructure-systems (SoIS) that can contribute services to one or more logical system-of-systems that run on top of it. The Logical Systems layer corresponds to the digital ecosystem whose configurations represent a dynamic network of logically interrelated software assets implementing concrete component-system's services or their surrogates depending on the DGT stage of a concrete component-system. Digital technology infrastructure (DTI) abstractions enable the dynamic and transparent creation and maintenance of configuration networks that hyperlink software assets, communication system assets, and computer infrastructure and create the OpEng infrastructure instances (Figure 7). The current concretization of the abstract DTI concept is a computer network (CN).

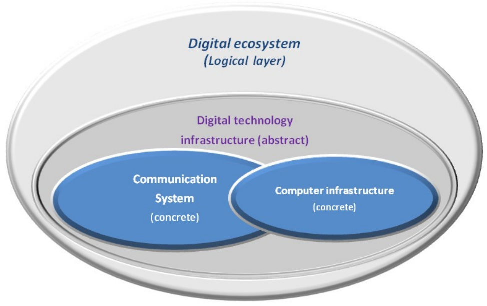


Figure 7. The Context of OpEn—Digital ecosystem and Digital technology infrastructure.

The design, installation, maintenance, monitoring, control, and recovery of complex OpEng infrastructure assumes explicit incorporation of nonfunctional requirements concerning the uninterruptable, trustworthy, responsive, and transparent physical operation.

A communication system (CS) is the SoIS that establishes, operates, and maintains personal, local, and global multipath connectivity regardless of the access point technology, transmitting media type, communication protocols, and the nature of disseminated communication objects. Currently, the term communication is a synonym for digital communication. Two digital communication reference models, specified to mitigate the inherent complexity through layered architecture, ISO's Open Systems Interconnection (ISO-OSI) and TCP/IP, are pillars of classical communication systems. ISO-OSI is a generic protocol-independent standard that guides the communication between the network and the end user. On the other hand, TCP/IP is a concrete communication protocol that facilitates host-to-host connectivity over a communication network and, in a way, represents the implementation of OSI guidelines.

From the DGT framework aspect, it is essential to support the simulation and emulation of the architecture, network protocols, and various communication network parameters [79–81]. We claim that due to the higher number of layers, ISO-OSI meta-specification is more suitable for verification purposes (due to better fidelity). Consequently, a TCP/IP meta-specification is better suited for validation and monitoring purposes (due to the direct compliance with the real-time operating context).

The digital communication domain faces a fast development of methods, principles, techniques, and technologies challenging the sustainability of contemporary DGT approaches. The most challenging is the transition from contemporary (3G/4G) to 5G/6G and beyond technologies [82–86], the broader incorporation of Blockchain [87], the impact of the artificial intelligence [88], and the predicted transition from classical to semantic digital communications [89–92].

Computer infrastructure (CI) is the SoIS that physically hosts the related digital ecosystem and represents the most inner responsibility layer of DGT endeavors. A dominant fast development of technologies and the proliferation of novel computing principles (primarily quantum computing) represent the additional challenging aspects of a continuous DGT operation independent of infrastructure reconfiguration. Similarly to the CS, CI determines a cost/performance relation of particular DGT activities. A CI incorporates a broad scope of networked computer assets ranging from the IoT edge and cloud [93] support, globally distributed ledgers [94], computer clusters [95], gateway-isolated private computer networks [96], public computer networks [97], to personal and pervasive computer devices [98]. Currently, a computer network topology is a synonym for CI topology, making a computer network a generic CI but with the extracted CS domain. The mind-map scope model of CI topology scope is represented in Figure 8 [99] and frames the potential ranges of DGT.

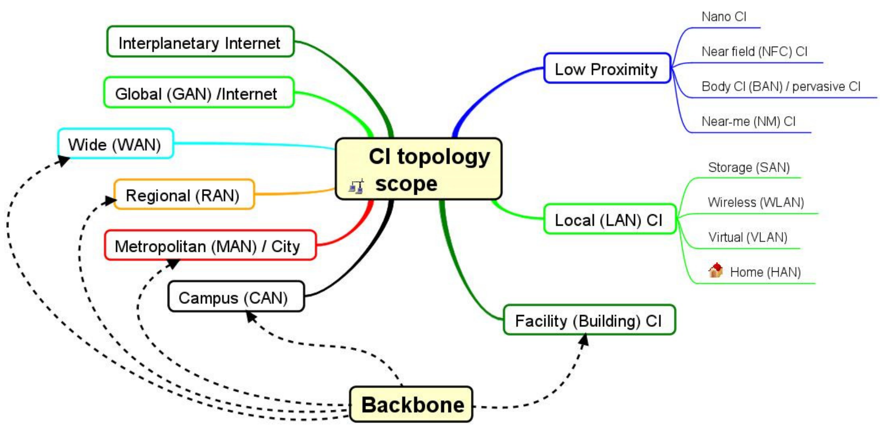


Figure 8. CI Topology scope model (addapted from [99]).

The course of OpEng originates from the technical infrastructure's similarity with a human's vegetative neural system. When it functions, nobody even cares or is conscious of its existence. Proper functioning is intrinsically assumed, while continuous service delivery appears as a spontaneous natural phenomenon from the stakeholder's angle. When the technical infrastructure fails, it appears from the darkness blamed as it has never functioned.

The lean nature of the DGT process and the level of contemporary systems, software, and operations engineering skills and knowledge, when joined with the fast appearance of quantum operational platforms, favor the specification, modeling, and development of frameworks that hide the heterogeneity of supporting infrastructure [100,101] and improve the scaling transparency. The DGT framework's angle simulation, emulation, and connectivity analysis represent the essential services supporting the verification stage, while real-time monitoring mechanisms enable machine learning for infrastructure optimization and dynamic load balancing.

Consequently, the role of Operations Engineering in the specification of integrated tailorable life cycle models reflects the impact of executive infrastructure that transparently supports a virtually unlimited digital ecosystem in the uninterruptable delivery of the embedded system's services. According to the conceptual framework model proposed in [48] (Figure 4), Operations Engineering maps to the Execution helix.

2.2. The Foundation of additional Hypothesis (H2 to H5)

The additional hypotheses (H2, H3, H4, and H5) address the minimal basic set of essential principles that guide the proposed DTG hyper-framework (DGTHyF) specification and development and augment the generic approach to DGT life cycle model management.

2.2.1. The Collaboration/Cooperation of Heterogeneously Staged Components (systems)—(H2)

The trade between heterogeneity and homogeneity principles in a leaned approach to engineering complex cyber-physical-socio-technology systems has been in the field for a long time. It addresses the basic principles determining the relations between the whole and its parts. One strategic approach is to homogenize the candidate system before the successful integration. This approach assumes the existence of mutually agreed, value-aided, standardized templates that guide every candidate-component system in transforming its structure and behavior to achieve absolute compliance before the possible integration. Collaboration through the gradual transformation is not possible while cooperation prevails. Although standards and standardization represent the engineering cornerstones, several suspicious factors exist. The way of their creation, the creator's motivation and interest, the associated costs, and the instant pressure for transformation represent significant obstacles for most stakeholders. The exact global migration management, discussed in [102], is a typical challenging homogenization issue related to the contemporary socio-technology context that administratively shapes people's lives through unique identification mechanisms. In [103], the author discusses homogenization, based on decentralized blockchain oracles distributed ledger technology (DTL), to mitigate the diversities appearing in the industrial data and justify the proposed approach by compatibility, security, and efficiency issues. The Smart City (SC) concept has introduced SC hubs to mitigate the synchronized network flow between the virtual worlds and their physical counterparts [104]. Consequently, the main problems with homogenization are the higher disruption level and the ultimate change pressure, while the main benefits relate to the organizational, structural, procedural, and technological unification that may, in the end, decrease the overall system's complexity.

The other strategic approach assumes the parallel existence of diverse collaborating/cooperating systems that retain internal mechanisms and comply only through well-defined interfaces, favoring the coexistence of heterogeneous component systems. This strategy is less stress-prone for the component systems but demands more complex interoperability support, usually delegated to the engineering segment of DGT endeavors. Instead of the instant transformation pressures, an accustomed operating environment supports a lined approach and *enables uninterrupted operations but may indefinitely preserve the internal status quo. Consequently, the overall system's complexity may significantly arise, causing the transfer of responsibilities almost exclusively to the technology infrastructure. The role and the significance of systems, software, and operational engineering professionals prevail and justify their effort and expertise to encapsulate the DGT risks. Although demanding, we find the heterogeneity more realistic for a long-range leaned DGT of arbitrary complex cyber-socio-technology systems that highly stimulate novel engineering methods, techniques, and tools specification and* development. The possibility of federated learning with preserving component taxonomy, security, and privacy in heterogeneous environments concerning promising technologies appears in [105,106]. The role and novel representation of Distributed Computing Continuum (DCC), beyond the traditional computer architecture, assumes the highest possible heterogeneity level of the participating devices or networks, challenging models, and the lifelong learning framework [107–110]. Distributed task allocation represents another research domain challenged by the collaboration/cooperation of heterogeneous end-point devices [111].

The previous elaboration on DGT fundamentals advocates the lean approach as a foundation of a sustainable DGT life cycle model. Consequently, through the entire lifecycle, the transformed system contains the architectural components (parts) that are in one of the following stages: not transformed (performing in a problem domain without digital support), in transition (digital transformation is in progress), transformed (digitally supported, verified, and validated), deployed (installed and usable), operational with refactoring abilities (used with user experience feedback abilities), and occasionally temporarily or permanently removed. The heterogeneity of this form, complexity, and structure represents one of the most challenging interoperability obstacles to continuous service delivery through the entire DGT life cycle.

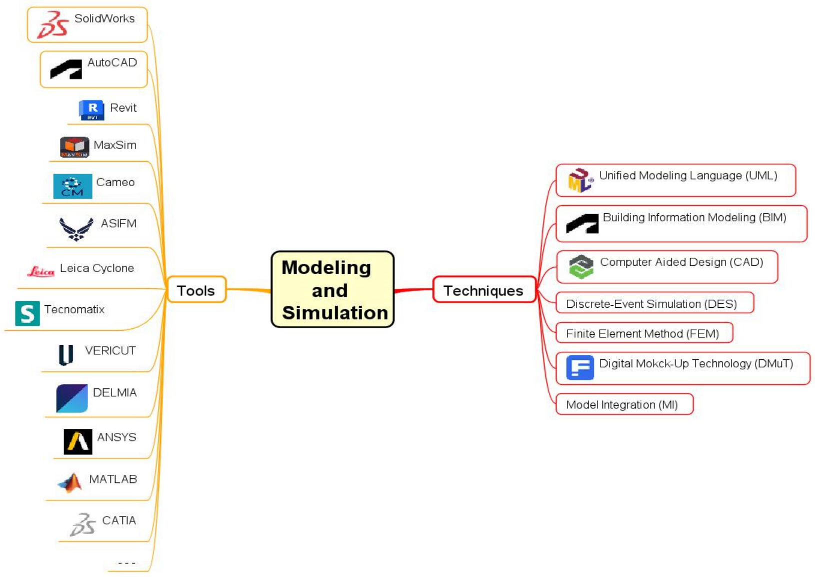
2.2.2. The Executable Virtualization Ability—(H3)

The efficient and effective use of digital technology infrastructure is essential in current enterprise architectures. Two diverse approaches that aid the transparency of software services in a heterogeneous environment are commonly available: containerization and virtualization.

Containers are portable, stand-alone executable units that pack together software asset and all the necessary resources to enable independent running in the arbitrary computer environments. The Docker is currently the most widely utilized containerization tool, while the Kubernetes actually represents an industry standard for containers orchestration. The representative container applications include building the Microservice architectures, continuous integration, continuous deployment, hybrid and multi cloud configuration support, the deployment of artificial intelligence and machine learning applications, and coupling the large language models (LLMs) with the related generative artificial intelligence mechanisms [112].

On the other hand, virtualization assumes replacing the physical computers with software-defined virtual machines (VMs), operated under the control of the VMs manager (hypervisor), enabling the automatic creation of service management workflows by superposing the current configuration-instance portfolio and the current user's profile (service policies) [113]. While VM virtualizes the hosting machine resources, container virtualizes the hosted operating system. Both mechanisms support the deployment of software services over digital technology infrastructure and enable the run-time separation of the logical operation context (virtual twin) and the physical execution context (physical twin).

Hypothesis 3 (H3) states that each configuration instance has to be an executable virtual entity capable of real-time emulation (imitate with another system, surrogate), simulation (model-based prediction), operation (traceable running), monitoring (noninvasive and secured), modeling, decision-support mechanism, access (User Interface), and evaluation (User Experience) of configured components structure and behavior. The DGTHyF needs to support an open set of modeling and simulation technologies and tools that, packed with the software asset, enable the transparent execution over the entire digital technology infrastructure. The contemporary representative repertoire of modeling techniques and simulation tools, adapted from [114], is presented in Figure 9.



Modeling and Simulation
Tools
SolidWorks
AutoCAD
Revit
MaxSim
Cameo
ASIFM
Leica Cyclone
Tecnomatix
VERICUT
DELMIA
ANSYS
MATLAB
CATIA
...
Techniques
Unified Modeling Language (UML)
Building Information Modeling (BIM)
Computer Aided Design (CAD)
Discrete-Event Simulation (DES)
Finite Element Method (FEM)
Digital Mokck-Up Technology (DMuT)
Model Integration (MI)

Figure 9. Modeling and Simulation Techniques and Tools (adapted from [114]).

2.2.3. Hiding the Data-Layer Complexity—(H4)

Digitally transformed systems exhibit a dramatic growth in volumes and diversity of generated, stored, retrieved, and processed data harvested from different sources and preserved in a distributed data repository (a collection of multiple interconnected data chunks, physically spread across hosting computer network infrastructure, forming a complex data layer). The emerging complexity demands efficient handling and represents one of the most challenging aspects of successful software development.

Engineering rationality prefers data layer homogeneity with uniformity (*All nodes utilize the same DBMS software and possess identical database schemas*), consistency (*Changes made to the database on one node are automatically propagated to other nodes, ensuring data consistency*), and simplicity (*The same DBMS software exists throughout the data layer*) as the primary beneficial features.

Unfortunately, the evolution of digitalization in cyber-physical-socio-technology systems has led to natural heterogeneity (*the coexistence of structured, semi-structured, and unstructured data*). Heterogeneous data repositories are characterized by flexibility (*Nodes can employ different DBMS software, such as distributed file system, relational databases, document-oriented, key-value, column-based, and graph-oriented databases*), schema mapping (*Heterogeneous databases necessitate mapping between different schemas to ensure interoperability between nodes*), and data transformation (*Data might need to be transformed or translated between different formats or encodings to maintain consistency*). The ability to process and analyze heterogeneous data provides valuable insights, predictive analytics, and decision-making, which is crucial in the era of Big Data [115]. The usual approach to cope with the data layer complexity emerging from intensive heterogeneity assumes service-directed modular design (data chunks are clustered according to the specific service needs with possible increases in data redundancy but reducing service dependency), robust data normalization (data is stored in a logical manner optimizing the logical data layer), the efficient management of application programming interfaces (APIs) through API gateways, wrapping data with the additional abstract layers (serving as data encapsulation façade), the use of machine learning algorithms to detect query patterns and enable automatic query optimization, the use of AI for the advanced optimizations (performance monitoring, predictive refactoring, and mitigating the risks related to the data layer change propagation), and providing real-time feedback from physical date layer (physical twin) to logical data layer (data models aka virtual twin) to enable their dynamic synchronization.

The role of the Data Abstraction Layer (DAL) is to create a data virtualization that simplifies repository handling by providing a unique, higher abstraction layer that mitigates the concrete implementation details [116]. It is crucial in software development due to the higher usability and functionality resulting from the creation of a logical data access layer, which ultimately fosters the scalability and maintainability of software systems and can ensure a smooth implementation. When implementing DAL, the frameworks and tools used for a particular project engineering necessitate the highest possible alignment with the project requirements.

DAL usually contains two generic concepts: a generic interface with two methods (load and store) that hide technically diverse repository features from the referencing services, thereby enhancing the usability and functionality of the encapsulated data layer, and the abstract model of the generic data chunk, addressed in this research as an abstract information resource (AIR) generalized by the meta-specification of a large data object (LDO). From the architecture aspect, it is essential to separate the external repository from the internal dynamic storing and presentation layers. This separation hides the particular characteristics of a persistent LDO form from its operational counterparts.

The computational complexity of high-dimensional LDO processing is the main obstacle for real-time or near-real-time applications. Consequently, reducing the complexity of multidimensional LDOs is one of the highly promising approaches to system/problem simplification. As less complex LDOs are easy to navigate, explore, visualize, and analyze in different contexts, they are more suitable for effective machine learning.

Hypothesis 4 (H4) states that the natural heterogeneity of the underlying data layer demands a more sophisticated approach to Data Engineering (DatEng) and Data Analytics (DatAna). Consequently, the use of the abstract information resource concept (AIR) as a Large Data Object's (LDO) generalization is fully justified. Raising the abstraction level in such a way has a direct payback in hiding the data-layer implementation details emerging from the heterogeneous associations of arbitrary complex LDOs [47].

2.2.4. Meta-specification driven generative support—(H5)

DGT tools that support creativity in forward-looking and critical thinking in backward-chaining cognition are highly desirable in complex engineering/reengineering endeavors. With the proposed trinity of Systems, Software, and Operations Engineering, the DGTHyF needs to creatively combine distinct features and reduce possible redundancy while integrating traditionally separated development environments. Every trinity component relies on specification-based engineering (SBE) principles, with specification as a generic document that more or less formally defines relevant aspects of an engineered system at an arbitrary detailed level. Traditionally, even the programming code represents the most detailed specification of the underpinning problem's solution. Consequently, model-driven engineering (MDE) is a form of SBE where the specification document emerges from the modeling practices. Modeling is traditionally recognized as a domain complexity mitigation approach, while models (the products of modeling activities) represent scaled multimedia specifications (virtual twins) of problem domain concepts (real twins) [48]. Meta specification is a specification of a specification. It raises the abstraction level of a concrete specification to mitigate the complexity further and establishes the foundations for the automatic generation of context-dependent specification instances [46,47]. Both concepts (specification and meta-specification) are documents with generative potentials that demand collaborative document management tools. They are digital platforms that enable real-time collaboration/cooperation among stakeholders working on shared documents of arbitrary granularity and abstraction levels [117].

Hypothesis 5 (H5) states that the explicit support for H1, H2, H3, and H4 has to be a meta-specification driven and integrated into the DGT hyper-framework (DGTHyF). To mitigate the domain heterogeneity and support domain adaptability, meta-specification-driven dynamic building and biding of the generic DGTHyF features are mandatory. The shared document instance defines a virtual twin's content, with time and modal markers, representing a hyper-document that interrelates meta-specifications (Life Cycle, Architecture, Component, Component configuration, Data abstraction layer, and Presentation layer), corresponding derived specifications, and real-time execution instances. Meta-specifications, interpreted by the strategy pattern with extendible Generative Artificial Intelligence-supported concrete strategies, incline to Industry 6.0 and Beyond.

3. Results

This research combines digital twinning, multidimensional cognition processes, and systems engineering, software engineering, and operational engineering methodology to build an extendable, generic, configurable, and usable hyper-framework prototype that supports the sustainable digital transformation of cyber-physical-socio-technology systems DGT through the entire lifecycle of the transformed system. In our previously published research [48], we have proposed the conceptual framework for digital twin (DT) verification and validation, based on a quintuple helix model, as a paradigm for DGT hyper-framework (DGTHyF) prototype specification and development.

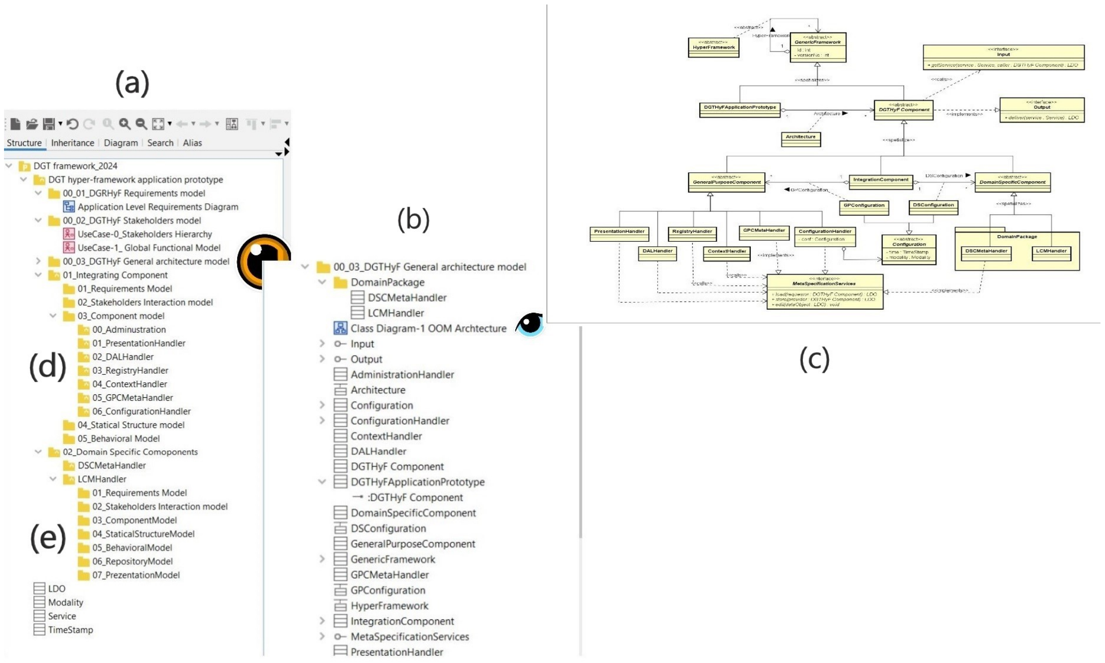
Due to its inherent complexity, we propose the evolution prototyping, meta-specification-based generation of hyper-document specification instances (models and code), and continuous integration and deployment support through a staged model of composing component systems. We claim that DGTHyF specification, modeling, and prototyping, compliant with the hypothesis (H1-H5), enable its incremental development, tailoring, verification, deployment, and operational validation in the lined agile DGT process.

Currently, there exists a large family of commercially available Integrated Development Environments (IDEs) representing interactive, GUI (Graphical Users Interface) oriented event-driven, software systems commonly supporting multiple programming languages, syntax-sensitive code editing, compiler or interpreter support, symbolic debugger, extendible library support, code search, version control, and code integration features. The proposed DGT Hyper-Framework (DGTHyF) is a prototype software system with configuration-based architecture, components, and service orchestration abilities. It comprises the extendible set of component frameworks, each supporting distinct closed systems (closed for modification but opened for extension) with internally extendible functionality.

Following the Model-based-engineering approach, we have defined a starting hyper-document represented as the AstahProfessional modeling tool component-based project suite (Figure 10). It is composed of the extendible collection of underpinned hyper-documents that represent either atomic-grained (non-hyper-linked) or course-grained (hyper-linked) large data objects representing the specifications (models) of corresponding building blocks. The overall project suite represents a hyper-graph (a graph whose nodes are graphs) and is a starting meta-specification (meta-model) of the proposed DGTHyF, which we address as a DGTHyF generic virtual twin. Its highest granulation level is presented in Figure 10 (a) and augmented by the package-contained example of the DGTHyF generalized architecture model (Figure 10 -(b)) as an object-oriented model (class diagram—Figure 10—(c)). The initial repertoire of General Purpose Components (Figure 10—(d)) and Life Cycle Model Handler a Domain Specific Component (Figure 10—(e)) represent the highest granularity specification of the corresponding hyper-document model.

The *GenericFramework* is a recursively orchestrated abstract *HyperFramework,* at the highest hierarchy level of the generic architecture meta-model, that specifies common structural and behavioral characteristics of an arbitrary hyper-framework. The DGTHyFApplicationPrototype is a *GenericFramework* that aggregates an open set of abstract *DGTHyFComponent*abstractions that either belong to the General Purpose Components Registry (a configuration of abstract *GeneralPurposeComponent*specializations) or Domain Specific Components Registry (a configuration of abstract *DomainSpecificComponent* specializations) and forms a dynamic Architecture configuration. Each DomainPackage is a stand-alone container composed of DSCMetahandler (Domain Specific Component Meta-Handler) supporting the dynamic generation of domain-specific context and configuration and an arbitrary domain-specific component that encapsulates domain-specific services, in this version represented by a Life Cycle Model Handler (LCMHandler).

The generic architecture specification (Figure 11) serves as a referent meta-model for the automatic generation of the initial prototype-shell code (Figure 12), thereby illustrating the whole spectrum of features announced by the stated hypothesis (H1-H5).



DGT framework_2024
DGT hyper-framework application prototype
00_01_DGRHyF Requirements model
Application Level Requirements Diagram
00_02_DGTHyF Stakeholders model
UseCase-0_Stakeholders Hierarchy
UseCase-1_ Global Functional Model
00_03_DGTHyF General architecture model
01_Integrating Component
01_Requirements Model
02_Stakeholders Interaction model
03_Component model
00_Adminustration
01_PresentationHandler
02_DALHandler
03_RegistryHandler
04_ContextHandler
05_GPCMetaHandler
06_ConfigurationHandler
04_Statical Structure model
05_Behavioral Model
02_Domain Specific Comoponents
DSCMetaHandler
LCMHandler
01_Requirements Model
02_Stakeholders Interaction model
03_ComponentModel
04_StaticalStructureModel
05_BehavioralModel
06_RepositoryModel
07_PrezentationModel
LDO
Modality
Service
TimeStamp

00_03_DGTHyF General architecture model
DomainPackage
DSCMetaHandler
LCMHandler
Class Diagram-1 OOM Archtecture
Input
Output
AdministrationHandler
Architecture
Configuration
ConfigurationHandler
ContextHandler
DALHandler
DGTHyF Component
DGTHyFApplicationPrototype
:DGTHyF Component
DomainSpecificComponent
DSConfiguration
GeneralPurposeComponent
GenericFramework
GPCMetaHandler
GPConfiguration
HyperFramework
IntegrationComponent
MetaSpecificationServices
PresentationHandler

Figure 10. DGTHyF hyper-document (AstahProfessional project).

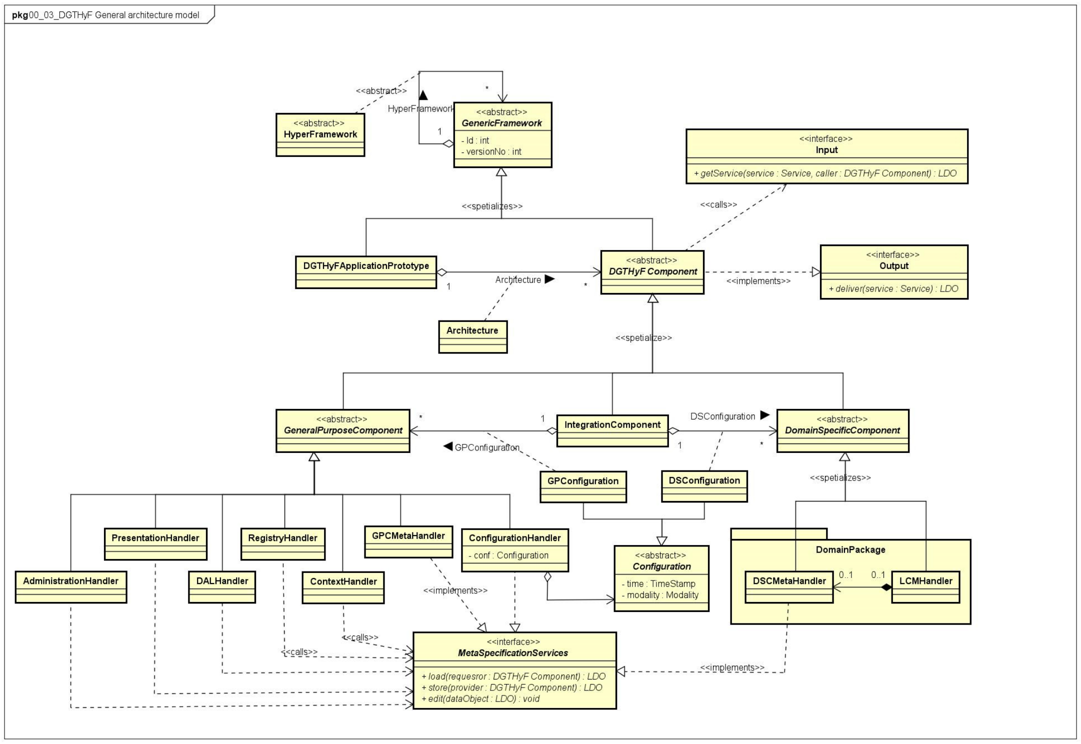
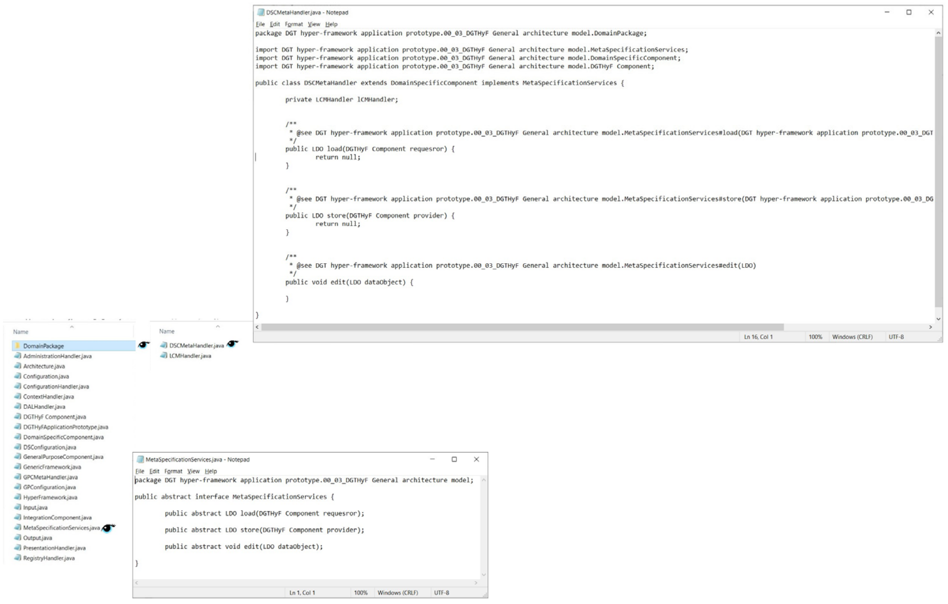


Figure 11. DGTHyF Generic Architecture Specification—class model (details of Figure 10—(c)).

doi:10.20944/preprints202410.2523.v1



DSCMetaHandler.java - Notepad
File Edit Format View Help
Name
DomainPackage
AdministrationHandler.java
Architecture.java
Configuration.java
ConfigurationHandler.java
ContextHandler.java
DALHandler.java
DGTHyF Component.java
DGTHyFApplicationPrototype.java
DomainSpecificComponent.java
DSConfiguration.java
GeneralPurposeComponent.java
GenericFramework.java
GPCMetaHandler.java
GPConfiguration.java
HyperFramework.java
Input.java
IntegrationComponent.java
MetaSpecificationServices.java
Output.java
PresentationHandler.java
RegistryHandler.java
Name
DSCMetaHandler.java
LCMHandler.java
MetaSpecificationServices.java - Notepad
File Edit Format View Help

```
package DGT hyper-framework application prototype.00_03_DGTHyF General architecture model.DomainPackage;

import DGT hyper-framework application prototype.00_03_DGTHyF General architecture model.MetaSpecificationServices;
import DGT hyper-framework application prototype.00_03_DGTHyF General architecture model.DomainSpecificComponent;
import DGT hyper-framework application prototype.00_03_DGTHyF General architecture model.DGTHyF Component;

public class DSCMetaHandler extends DomainSpecificComponent implements MetaSpecificationServices {

        private LCMHandler lCMHandler;


        /**
         * @see DGT hyper-framework application prototype.00_03_DGTHyF General architecture model.MetaSpecificationServices#load(DGT hyper-framework application prototype.00_03_DGT
         */
        public LDO load(DGTHyF Component requesror) {
                return null;
        }


        /**
         * @see DGT hyper-framework application prototype.00_03_DGTHyF General architecture model.MetaSpecificationServices#store(DGT hyper-framework application prototype.00_03_DG
         */
        public LDO store(DGTHyF Component provider) {
                return null;
        }


        /**
         * @see DGT hyper-framework application prototype.00_03_DGTHyF General architecture model.MetaSpecificationServices#edit(LDO)
         */
        public void edit(LDO dataObject) {

        }

}
```

```
Ln 16, Col 1    100%    Windows (CRLF)    UTF-8
```

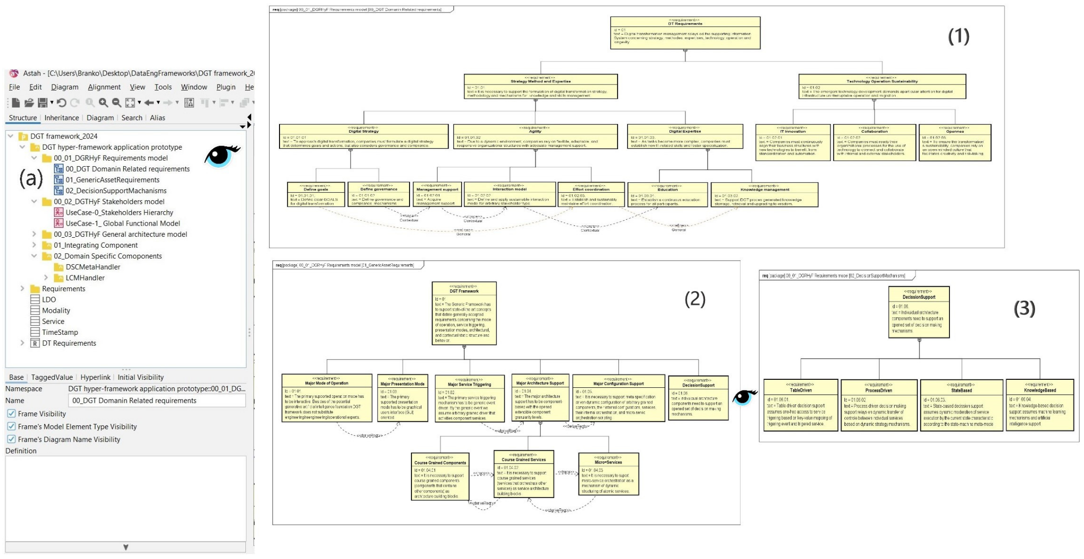
```
package DGT hyper-framework application prototype.00_03_DGTHyF General architecture model;

public abstract interface MetaSpecificationServices {

        public abstract LDO load(DGTHyF Component requesror);

        public abstract LDO store(DGTHyF Component provider);

        public abstract void edit(LDO dataObject);

}
```

```
Ln 1, Col 1    100%    Windows (CRLF)    UTF-8
```

Figure 12. Sample Java code generated from OOM-metaspecification (model) (Figure 11).

3.1. Multy-Staged Hyper-Framework Requirements Model

The high-level generic requirements model of the proposed DGTHyF defines a universal proto-iterative starting point of an arbitrary evolution prototype as an interactive, event-driven, graphical users interface (GUI) oriented, component-based, meta-specification configurable, software tool with flexible, profile-based, architecting features (Figure 13), designated as a hyper-document node 00-01-DGRHyF Requirements Model (see Figure 13—(a)).



Astah - [C:\Users\Branko\Desktop\DataEngFrameworks\DGT framework_2
File Edit Diagram Alignment View Tools Window Plugin He
Structure Inheritance Diagram Search Alias
DGT framework_2024
DGT hyper-framework application prototype
00_01_DGRHyF Requirements model
(a)
00_DGT Domanin Related requirements
01_GenericAssetRequirements
02_DecisionSupportMachanisms
00_02_DGTHyF Stakeholders model
UseCase-0_Stakeholders Hierarchy
UseCase-1_ Global Functional Model
00_03_DGTHyF General architecture model
01_Integrating Component
02_Domain Specific Comoponents
DSCMetaHandler
LCMHandler
Requirements
LDO
Modality
Service
TimeStamp
DT Requirements
Base TaggedValue Hyperlink Initial Visibility
Namespace DGT hyper-framework application prototype::00_01_DG...
Name 00_DGT Domanin Related requirements
Frame Visibility
Frame's Model Element Type Visibility
Frame's Diagram Name Visibility
Definition
(1)
DT Requirements
Strategy Method and Expertise
Technology Operation Sustainability
Digital Strategy
Agility
Digital Expertise
Innovation
Collaboration
Openness
Define goals
Define governance
Management support
Interaction model
Effort coordination
Education
Knowledge management
(2)
DGT Framework
Major Mode of Operation
Major Presentation Mode
Major Service Triggering
Major Architecture Support
Major Configuration Support
DecisionSupport
Course Grained Components
Course Grained Services
Micro-Services
(3)
DecisionSupport
TableDriven
ProcessDriven
StateBased
KnowledgeBased

Figure 13. A sample meta-model of DGTHyF Generic Requirements Package(00_01_DGRHyF Requirements model).

In the current stage, the generic requirements (Figure 13—(a)) are composed of three specification documents models: 00_DG_Domain related requirements (specifies the highest level DGT domain related (Systems) requirements, Figure 13—(1)—with detailed representation shown in Figure 13), 01_GenericAssetRequirements (the highest level software requirements, Figure 13—(2)—with detailed representation shown in Figure 14), and further hyper document layer

02_DecisionSupportMechanisms (specifies component's internal decision making mechanisms, Figure 13—(3)—with detailed representation shown in Figure 15).

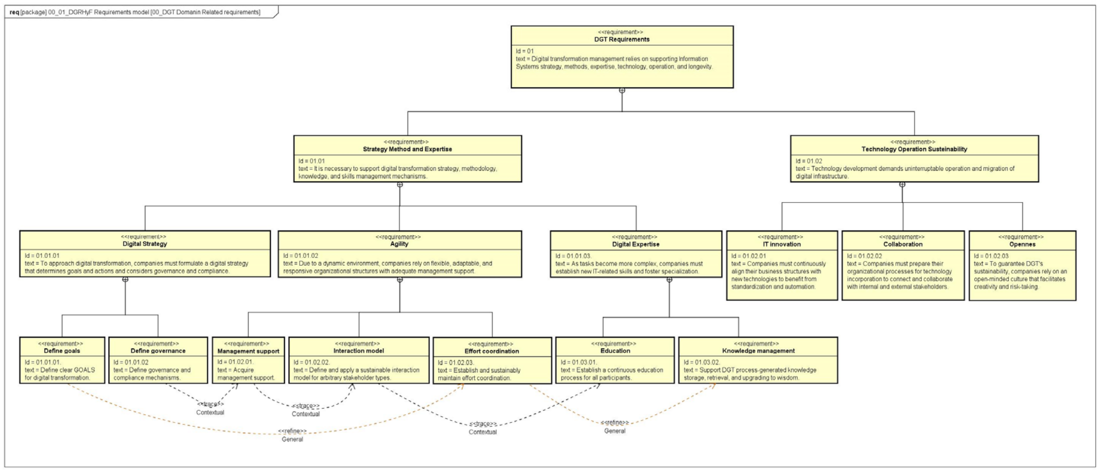

req [package] 00_01_DGRHyF Requirements model [00_DGT Domanin Related requirements]
<<requirement>> DGT Requirements
Id = 01
text = Digital transformation management relies on supporting Information Systems strategy, methods, expertise, technology, operation, and longevity.
<<requirement>> Strategy Method and Expertise
Id = 01.01
text = It is necessary to support digital transformation strategy, methodology, knowledge, and skills management mechanisms.
<<requirement>> Technology Operation Susteinability
Id = 01.02
text = Technology development demands uninterruptable operation and migration of digital infrastructure.
<<requirement>> Digital Strategy
Id = 01.01.01
text = To approach digital transformation, companies must formulate a digital strategy that determines goals and actions and considers governance and compliance.
<<requirement>> Agility
Id = 01.01.02
text = Due to a dynamic environment, companies rely on flexible, adaptable, and responsive organizational structures with adequate management support.
<<requirement>> Digital Expertise
Id = 01.01.03.
text = As tasks become more complex, companies must establish new IT-related skills and foster specialization.
<<requirement>> IT innovation
Id = 01.02.01
text = Companies must continuously align their business structures with new technologies to benefit from standardization and automation.
<<requirement>> Collaboration
Id = 01.02.02
text = Companies must prepare their organizational processes for technology incorporation to connect and collaborate with internal and external stakeholders.
<<requirement>> Opennes
Id = 01.02.03
text = To guarantee DGT's sustainability, companies rely on an open-minded culture that facilitates creativity and risk-taking.
<<requirement>> Define goals
Id = 01.01.01.
text = Define clear GOALS for digital transformation.
<<requirement>> Define governance
Id = 01.01.02
text = Define governance and compliance mechanisms.
<<requirement>> Management support
Id = 01.02.01.
text = Acquire management support.
<<requirement>> Interaction model
Id = 01.02.02.
text = Define and apply a sustainable interaction model for arbitrary stakeholder types.
<<requirement>> Effort coordination
Id = 01.02.03.
text = Establish and sustainably maintain effort coordination.
<<requirement>> Education
Id = 01.03.01.
text = Establish a continuous education process for all participants.
<<requirement>> Knowledge management
Id = 01.03.02.
text = Support DGT process-generated knowledge storage, retrieval, and upgrading to wisdom.
<<trace>> Contextual
<<trace>> Contextual
<<refine>> General
<<trace>> Contextual
<<refine>> General

Figure 14. The detailed requirements model (Figure 13—(1)).

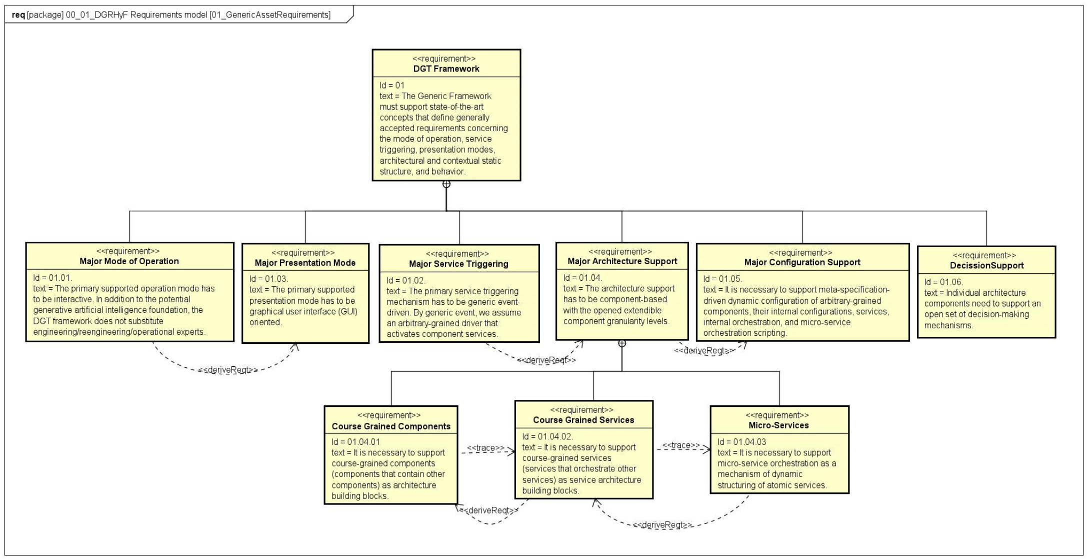

req [package] 00_01_DGRHyF Requirements model [01_GenericAssetRequirements]
<<requirement>> DGT Framework
Id = 01
text = The Generic Framework must support state-of-the-art concepts that define generally accepted requirements concerning the mode of operation, service triggering, presentation modes, architectural and contextual static structure, and behavior.
<<requirement>> Major Mode of Operation
Id = 01.01.
text = The primary supported operation mode has to be interactive. In addition to the potential generative artificial intelligence foundation, the DGT framework does not substitute engineering/reengineering/operational experts.
<<requirement>> Major Presentation Mode
Id = 01.03.
text = The primary supported presentation mode has to be graphical user interface (GUI) oriented.
<<requirement>> Major Service Triggering
Id = 01.02.
text = The primary service triggering mechanism has to be generic event-driven. By generic event, we assume an arbitrary-grained driver that activates component services.
<<requirement>> Major Architecture Support
Id = 01.04.
text = The architecture support has to be component-based with the opened extendible component granularity levels.
<<requirement>> Major Configuration Support
Id = 01.05.
text = It is necessary to support meta-specification-driven dynamic configuration of arbitrary-grained components, their internal configurations, services, internal orchestration, and micro-service orchestration scripting.
<<requirement>> DecissionSupport
Id = 01.06.
text = Individual architecture components need to support an open set of decision-making mechanisms.
<<deriveReqt>>
<<deriveReqt>>
<<deriveReqt>>
<<requirement>> Course Grained Components
Id = 01.04.01
text = It is necessary to support course-grained components (components that contain other components) as architecture building blocks.
<<requirement>> Course Grained Services
Id = 01.04.02.
text = It is necessary to support course-grained services (services that orchestrate other services) as service architecture building blocks.
<<requirement>> Micro-Services
Id = 01.04.03
text = It is necessary to support micro-service orchestration as a mechanism of dynamic structuring of atomic services.
<<trace>>
<<trace>>
<<deriveReqt>>
<<deriveReqt>>

Figure 15. The detailed requirements model (Figure 13—(2)).

Specified systems requirements (Figure 14) correspond to the systems engineering virtual twin at the highest abstraction level and split overall responsibilities into two hierarchies: the Strategy Method and Expertise (01.01) and Technology Operation Sustainability (01.02). They are mainly process-oriented. DGT Framework requirements model (Figure 15) is a virtual twin corresponding to the highest abstraction level of a product specification belonging to the software engineering solution domain. On the other hand, the Decision Support requirements model corresponds to the virtual twin that specifies the internal control mechanisms of arbitrary component decision-making core related to the highest level of the implementation domain.

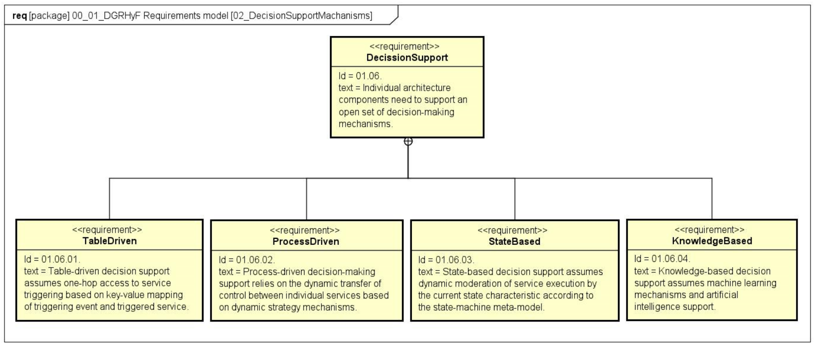

req [package] 00_01_DGRHyF Requirements model [02_DecisionSupportMachanisms]
<<requirement>> DecissionSupport
Id = 01.06.
text = Individual architecture components need to support an open set of decision-making mechanisms.
<<requirement>> TableDriven
Id = 01.06.01.
text = Table-driven decision support assumes one-hop access to service triggering based on key-value mapping of triggering event and triggered service.
<<requirement>> ProcessDriven
Id = 01.06.02.
text = Process-driven decision-making support relies on the dynamic transfer of control between individual services based on dynamic strategy mechanisms.
<<requirement>> StateBased
Id = 01.06.03.
text = State-based decision support assumes dynamic moderation of service execution by the current state characteristic according to the state-machine meta-model.
<<requirement>> KnowledgeBased
Id = 01.06.04.
text = Knowledge-based decision support assumes machine learning mechanisms and artificial intelligence support.

Figure 16. The detailed requirements model (Figure 13—(3)).

3.2. DGTHyF Generic Behavioral Model

Based on hypothesis 3 (H3), through the entire lifecycle, the digitally transformed system contains architectural building blocks (component/service/micro-service) that are in one of the following stages: not transformed (performing in a problem domain without digital support), in transition (digital transformation is in progress), transformed (digitally supported, verified, and validated), deployed (installed and usable), operational with refactoring abilities (used with user experience feedback abilities), and occasionally temporarily (disabled) or permanently removed (retired). The DGTHyF generic behavioral model, presented as a UML state diagram (AstahProfessional modeling tool), is shown in Figure 17.

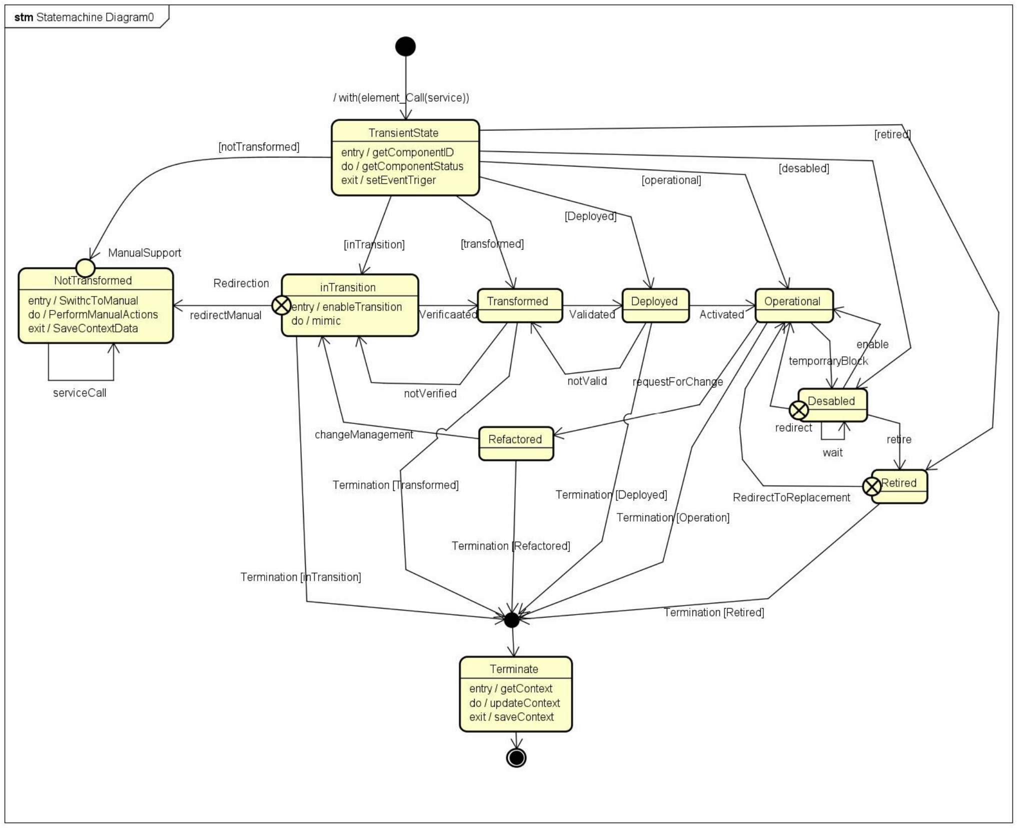


Figure 17. Component/Service/Microservice generic DGTHyF behavioral model.

These components are dynamically configured and support the intended services in a technically possible and time-dependent manner or form. Depending on the particular component service stage, the transformation activities belong to the operational, engineering, or reengineering category.

Consequently, the service discovery mechanism demands the establishment of traceable navigation paths reflecting the overall architecture. It assumes the creation and continuous maintenance of multi-layered hyper-structures of self-contained autonomous systems with arbitrary granularity levels (from atomic-grained to globally-grained), exclusively interrelated through interfacing points that enable the dynamic creation of horizontal (connecting different type ingredients) and vertical (connecting same type ingredients) associations.

On the other hand, the complexity mitigation of the service invocation mechanism assumes the stage-based polymorphism reduced to the meta-invocation defined by the statement:

with(context-bb: *BuildingBlock*; current-stage: *Stage*; invoke-se: *Service*)

that decuples the service invoker from the service discovery mechanism and service provider Building Block's context.

Figure 18 defines a meta-object-specification model corresponding to the specified generic behavioral model (Figure 17) as a further model refinement towards the executable specification (a programming code).

The sample Java code, generated from the OOM class model, extracted from the further refinements of the behavioral model's transformation to Java programming code, is presented in Figure 19.

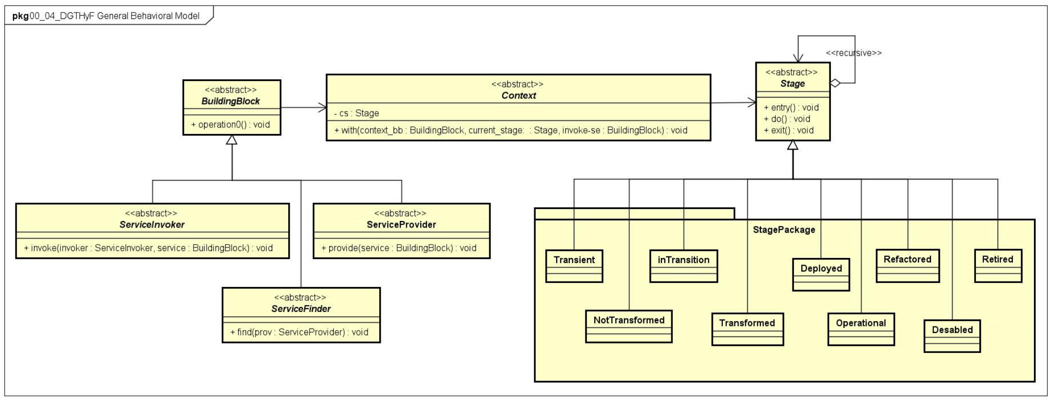
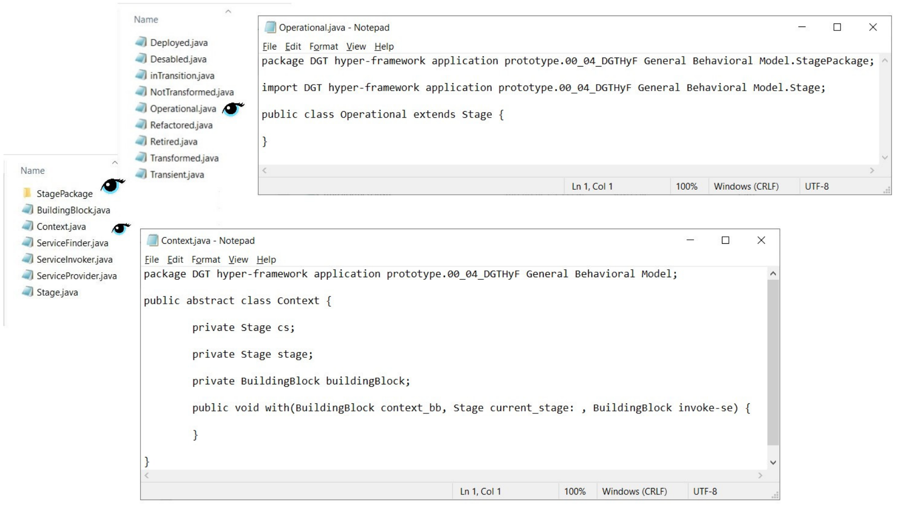


Figure 18. OOM-class model representation of generic DGTHyF behavioral model (Figure 16).



```
package DGT hyper-framework application prototype.00_04_DGTHyF General Behavioral Model.StagePackage;

import DGT hyper-framework application prototype.00_04_DGTHyF General Behavioral Model.Stage;

public class Operational extends Stage {

}
```

```
package DGT hyper-framework application prototype.00_04_DGTHyF General Behavioral Model;

public abstract class Context {

        private Stage cs;

        private Stage stage;

        private BuildingBlock buildingBlock;

        public void with(BuildingBlock context_bb, Stage current_stage: , BuildingBlock invoke-se) {

        }

}
```

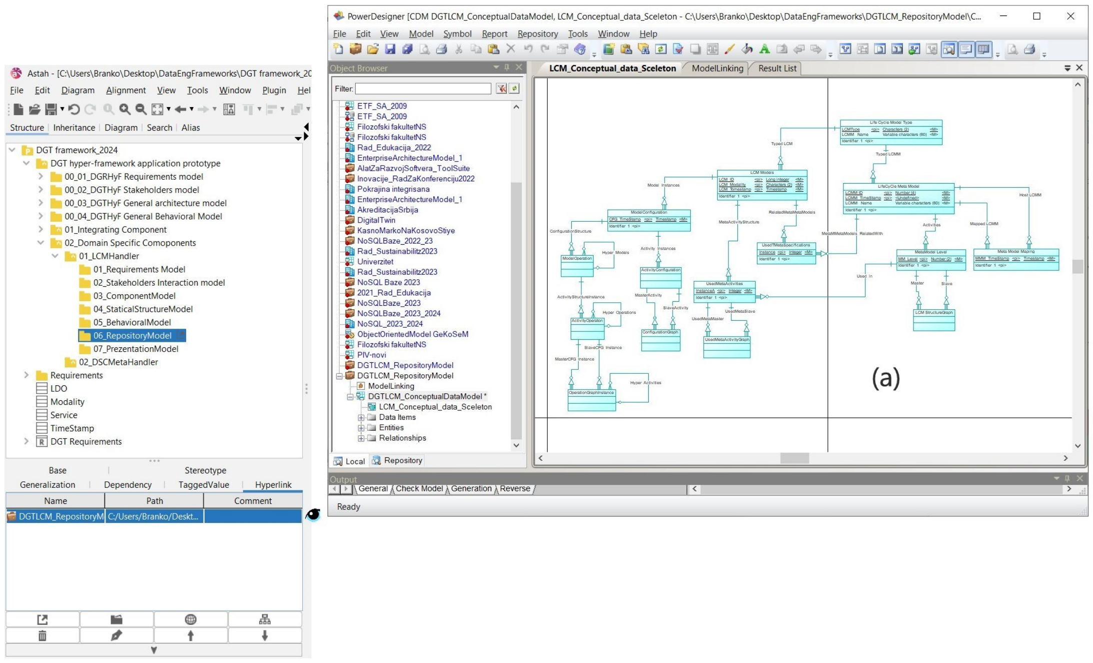
Figure 19. Sample Java code generated from OOM class model (Figure 18).

Currently, various formalisms rely on transforming behavioral models into corresponding analytical models. While these formalisms offer flexibility due to the range of available analytical languages and notations, they introduce additional complexity and require proper handling. The interoperability of different tools, general-purpose or domain-specific languages, and integrated production environments supporting simulations is one of the most challenging issues for further research and engineering directions. If we define software and technical infrastructure as systems, it is possible to raise the abstraction level and concatenate similarities in the higher abstract concept whose specializations redefine or add specific state and behavior refinements.

3.3. DGT Life Cycle Model Handling Component

According to hypothesis 1 (H1), the DGT Life Cycle Model (DGTLCM) Handling Component represents a central domain-specific DGTHyF component for this article's mission. The DGTLCM is a hyper-document that sublimates conceptual, logical, and physical repository models serving as meta-specifications canvas for data-driven generation of arbitrary DGTLCM instances.

The model specification development environment used for data layer modeling illustrates interoperable collaboration/cooperation (Figure 20) between the AstahProfessional UML2 modeling tool that hosts Object-Oriented Models and the Power Designer Data Architect Version 16.1 modeling tool that hosts data layer models in platform-independent form (conceptual and logical data layer model Figure 21) and platform dependent specialization (physical model Figure 22) with MySQL relation database as a hosting database server (Appendix A—Sample Data Definition Language script).



Astah - [C:\Users\Branko\Desktop\DataEngFrameworks\DGT framework_20
DGT framework_2024
DGT hyper-framework application prototype
00_01_DGRHyF Requirements model
00_02_DGTHyF Stakeholders model
00_03_DGTHyF General architecture model
00_04_DGTHyF General Behavioral Model
01_Integrating Component
02_Domain Specific Comoponents
01_LCMHandler
01_Requirements Model
02_Stakeholders Interaction model
03_ComponentModel
04_StaticalStructureModel
05_BehavioralModel
06_RepositoryModel
07_PrezentationModel
02_DSCMetaHandler
Requirements
LDO
Modality
Service
TimeStamp
DGT Requirements
DGTLCM_RepositoryM
C:/Users/Branko/Deskt...
PowerDesigner [CDM DGTLCM_ConceptualDataModel, LCM_Conceptual_data_Sceleton - C:\Users\Branko\Desktop\DataEngFrameworks\DGTLCM_RepositoryModel\C...
Object Browser
LCM_Conceptual_data_Sceleton
ModelLinking
Result List
DGTLCM_RepositoryModel
DGTLCM_ConceptualDataModel *
LCM_Conceptual_data_Sceleton
Data Items
Entities
Relationships
(a)

Figure 20. Hyper-linked Tools (AstahProfessional—PowerDesigner Data Architect).

Conceptually, the presented model possesses three of four Meta Object Facility (MOF) data layer abstraction levels (meta, model, and instance-configuration) represented with related ER sub-models, with naming convention related to the data abstraction layer (DAL) specification discussed in Section 2.

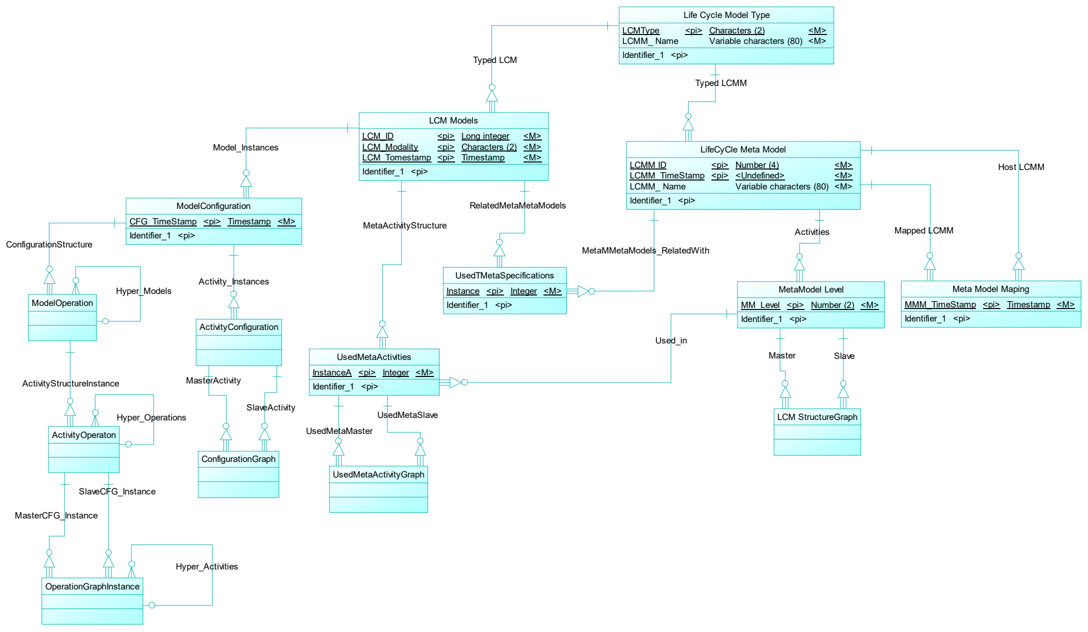


Figure 21. Multidimensional Conceptual LCM data model (Details from Figure 20—(a)).

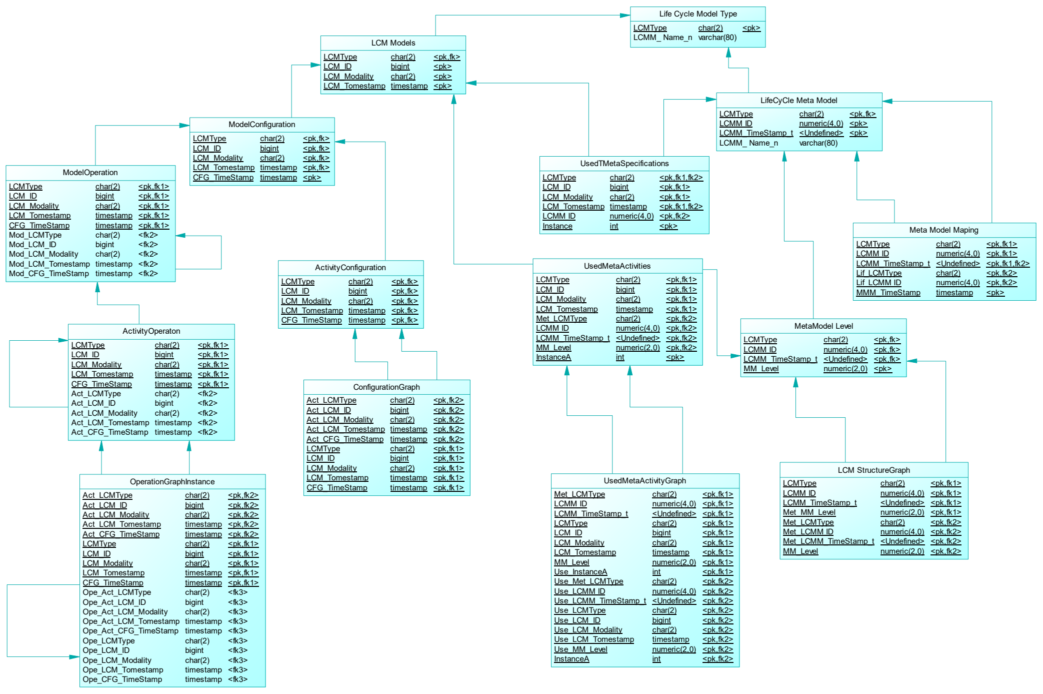


Figure 22. Multidimensional Physical model (MySQL targeting database management system).

The multidimensional data models rely on the data modeling principle of the deepest possible primary key propagation, which better suites database queering purposes and fosters model-based transformation from relational to arbitrary NoSQL counterparts (document, graph, column table, or key-value) due to the substitution of multilevel joins by embedded selections exclusively with the primary key, from bottom layer to higher layers perspective. In such a model, all queries start on the lowest hierarchy level where all related primary keys exist as composite primary identifiers (See Figure 22, UsedMetaActivityGraph table example).

4. Discussion and Related work

The DGTHyF initial meta-specifications frame the development of a software-supported development environment (tool) with generic impacts on future DGT endeavors. Such a tool has to support a wide range of problem domains and sustain through the entire life cycle of digitally transformed systems. Due to its inherent trinity, the superimposing of traditional Systems Engineering, Software Engineering, and Operation Engineering Life Cycle models [118,119] appear as the ultimate starting point for DGTHyF verification and validation purposes (hypothesis 1). The Collaboration/Cooperation of heterogeneously staged components (systems)—(hypothesis 2), Executable Virtualization Ability—(hypothesis 3), hiding the data-layer complexity (hypothesis 4), and the overall meta-specification-based generativity (hypothesis 5) creatively incorporated in the specification and modeling of the extendible integrated Life Cycle Model management domain specific DGTHyF component (Section 3). The DGTHyF is not a replacement for the existing or future incoming frameworks but a self-extendible hub for an open set of collaborative frameworks.

In the discussion section, we emphasize the collaborative potentials of mainstream meta-specifications, related hypotheses, and further perspectives (beyond the current stage).

Software Development Environments of arbitrary kinds are usually targeted in model-driven software development (MDSD) or model-based system engineering (MBSE) suits. They are closely related to the general architecture modeling paradigms and constitute the core of different contemporary Enterprise Architecture frameworks (EAF).

In the proposed approach, the DGT project is specified by a model-based hyper-document (currently defined as an AstahProfessional project) supporting the open, collaborative interoperability with arbitrary external tools (currently illustrated with the Power Designer Data Architect modeling tool hyper-linking) that may be either containerized or virtualized over the underlining digital technology infrastructure. Consequently, we direct the discussion and related work analysis to the conceptual or software-empowered hyper-document-based collaborative frameworks framed by the specified five hypotheses in the contemporary DGT context.

The elaborated DGTHyF is modeled and specified as an evolution prototype of heterogeneous collaborative component frameworks belonging to an ontology-based or tool-based group of EAFs (Figure 23), integrated over a virtual twin hyper-document instance.

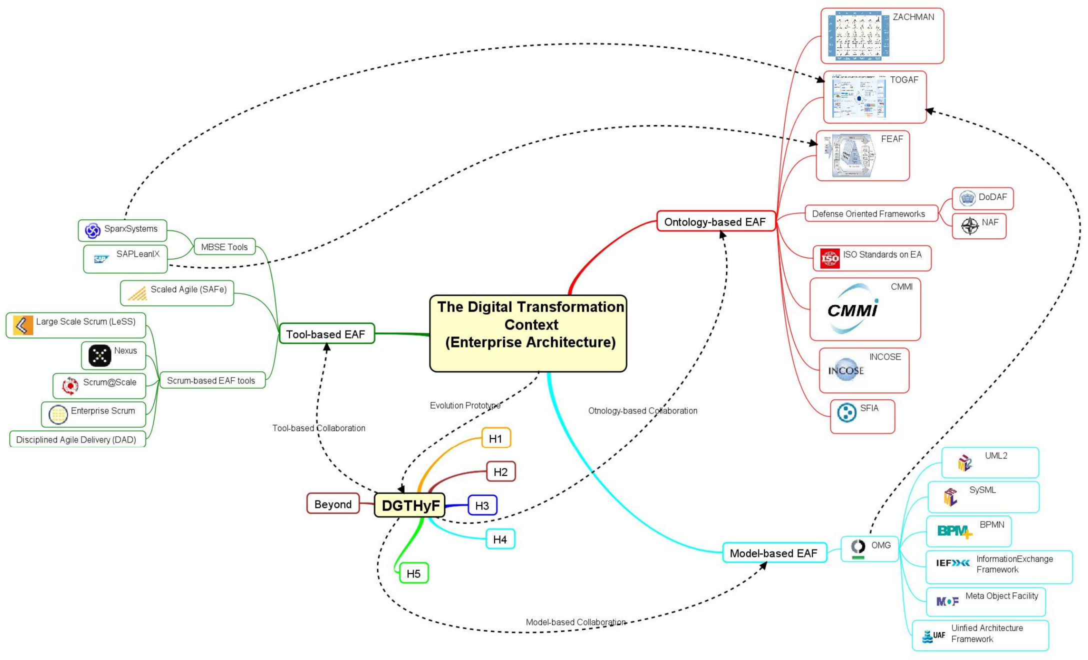


Figure 23. DGTHyF in the contemporary Digital Transformation Context.

The selected EAFs belong to three general groups: Ontology-based, Model-based, and Tool-based EAFs. Although a more detailed discussion of all available and selected frameworks is far beyond the scope of this article, it is necessary to highlight their core foundations.

The ontology-based group has a long development tradition focused on the fundamental principles and practices that form the boundaries of the addressed problem domain and serve as a starting point for concept clarification purposes. These specifications form the data structure skeleton of software tools supporting systems, software, and operations engineering activities. The most referenced Ontology-based EAF are: the Zachman Framework (ZF) [120], the Open Group Architecture Framework (TOGAF) [121], the Federal Enterprise Architecture Framework(FEAF) [122], the Defense oriented group of frameworks (US Department of Defense—DoDAF [123], NATO Architecture Framework—NAF [124]), International Standards Organization on EAF [125]), the Capability Maturity Model Integrated (CMMI) Framework [126], the International Council On Systems Engineering (INCOSE) framework discussed in the introductory section [1], and the Skills Framework for the Information Age (SFIA) [127].

The model-based EAF group directly impacts the current stage of DGTHyF regarding its inherent model-based orientation. It is used similarly as an ontology-based group but with extensive augmentation by the formal modeling languages. We have solely appointed the Object Management Group (OMG) [128] due to its relevancy, with the most referenced members being the Unified Modeling Language (UML2), Systems Markup Language (SySML), Business Process Modeling (BPM+), Information Exchange Framework (IEF), Meta-Object Facility Framework (MOF), and the Unified Architecture Framework (UAF).

The selected subset of Tool-based EAFs splits into three main groups. First, the Ontology-supported EAF tools (the Sparx System EAF [1,129] and SAP LenIX [130], with dominantly TOGAF and FEAF ontology foundation). The second, general agile (the Scaled Agile Framework [131] heavy-weight tool), and the third, Scrum-based EAF light-weight tools group (the Large Scale Scrum (LeSS) [132], Nexus [133], Scrum@Scale [134], and the Enterprise Scrum [135]) (see Figure 23).

The DGTHyF is a software-empowered heterogeneous hub promoting interoperability as an essential requirement. Although cooperative interoperability is usually assumed while developing tool-based frameworks, we predict the collaborative interoperability approach as a further universal direction.

Consequently, the briefly discussed sample context justifies the proposed DGTHyF's fundamental hypothesis as fostering mechanisms that place the collaborative heterogeneity in the core of arbitrary DGT context.

5. Conclusions

The addressed research topic frames a complex System-of-Systems specification, modeling, development, verification, validation, deployment, refactoring, and retirement life-long DGT support, with component systems persisting at arbitrary advanced DGT stages. In this article, we propose an initial software-empowered hyper-framework (DGTHyF) prototype specification compliant with five stated hypotheses (H1-H5) that define an extendible multidimensional context for systems-thinking-principles-based critical analysis and creative reasoning as the root of all innovativeness in process specification and product generation. The cognition-based foundation of the Proposed Context is compliant with estimated Industries 4.0. 5.0, 6.0, and current-Beyond. We claim it determines the promising prototype framework of future digital transformations.

The inherent complexity demands the development of appropriate mitigation frameworks. With this article, we have made an initial hyper-scratch of the underlying problem and hope the presented results will inspire a broader initiative for further research endeavors.

In the context of this article, the future research directions include but not restricted to the full application support for the core DGTHyF principles explicated through the stated hypothesis (H1-H5), the formation of heterogeneous hyper-data repository transparently supporting the relational, document-oriented, graph-oriented, distributed file-system oriented, column-table oriented, and key-value oriented counterparts interoperated over the meta-specification driven data abstraction layer (DAL) support, and further spreading of the underlining functionality. One of the most ambitious further directions is making the foundation for the Open Source Community Project—Open Source Digital Transformation Hyper-Framework (OSDGTHyF).

Author Contributions: Conceptualization, Ana Perisic and Branko Perisic; Data curation, Ana Perisic and Branko Perisic; Formal analysis, Branko Perisic; Investigation, Ana Perisic; Methodology, Ana Perisic and Branko Perisic; Supervision, Branko Perisic; Visualization, Ana Perisic; Writing—original draft, Branko Perisic; Writing—review & editing, Ana Perisic.

Funding: This research received no external funding.

Data Availability Statement: "The data used to support the findings of this study are included in the article."

Conflicts of Interest: Authors claim no conflict of interest.

Appendix A. Sample Data Definition Language script genereted from physical model (Figure 22)

```
drop table if exists UsedMetaActivityGraph;
/*==============================================================*/
/* Table: UsedMetaActivityGraph */
/*==============================================================*/
create table UsedMetaActivityGraph
(
MET_LCMTYPE        char(2) not null,
LCMMM_ID   numeric(4,0) not null,
LCMM_TIMESTAMP_T        char(10) not null,
LCMTYPE                 char(2) not null,
LCM_ID                  bigint not null,
LCM_MODALITY       char(2) not null,
LCM_TOMESTAMP              timestamp not null,
MM_LEVEL              numeric(2,0) not null,
USE_INSTANCEA      int not null,
USE_MET_LCMTYPE           char(2) not null,
USE_LCMMM_ID       numeric(4,0) not null,
USE_LCMM_TIMESTAMP_T          char(10) not null,
USE_LCMTYPE        char(2) not null,
USE_LCM_ID                     bigint not null,
USE_LCM_MODALITY         char(2) not null,
USE_LCM_TOMESTAMP     timestamp not null,
USE_MM_LEVEL       numeric(2,0) not null,
INSTANCEA                        int not null,
primary key (USE_MET_LCMTYPE, MET_LCMTYPE, USE_LCMMM_ID, USE_LCMM_TIMESTAMP_T,
LCMMM_ID,                                    LCMM_TIMESTAMP_T, USE_LCMTYPE, USE_LCM_ID,
USE_LCM_MODALITY, USE_LCM_TOMESTAMP,                             LCMTYPE, LCM_ID,
LCM_MODALITY, LCM_TOMESTAMP, USE_MM_LEVEL, MM_LEVEL, USE_INSTANCEA,
        INSTANCEA)
                );
alter table UsedMetaActivityGraph add constraint FK_MetaMaster foreign key (MET_LCMTYPE, LCMMM_ID,
LCMM_TIMESTAMP_T,          LCMTYPE, LCM_ID, LCM_MODALITY, LCM_TOMESTAMP, MM_LEVEL,
USE_INSTANCEA)
        references UsedMetaActivities (MET_LCMTYPE, LCMMM_ID, LCMM_TIMESTAMP_T, LCMTYPE, LCM_ID,
LCM_MODALITY,           LCM_TOMESTAMP, MM_LEVEL, INSTANCEA) on delete restrict on update restrict;
alter table UsedMetaActivityGraph add constraint FK_MetaSlave foreign key (USE_MET_LCMTYPE,
USE_LCMMM_ID,                    USE_LCMM_TIMESTAMP_T,      USE_LCMTYPE,      USE_LCM_ID,
USE_LCM_MODALITY, USE_LCM_TOMESTAMP,      USE_MM_LEVEL, INSTANCEA)
references UsedMetaActivities (MET_LCMTYPE, LCMMM_ID, LCMM_TIMESTAMP_T, LCMTYPE, LCM_ID,
LCM_MODALITY,   LCM_TOMESTAMP, MM_LEVEL, INSTANCEA) on delete restrict on update restrict;
```

References

1. INCOSE—International Council on Systems Engineering, Systems Engineering Vision 2035, available at https://www.incose.org/publications/se-vision-2035, (accessed on 19 of September 2024)
2. SEBoK Guide to the Systems Engineering Body of Knowledge, Fundamentals for Digital Engineering, available at https://sebokwiki.org/wiki/Fundamentals_for_Digital_Engineering, (accessed on 20 of September 2024)

3. World Economic Forum, Digital Transformation: Powering the Great Reset, **2020**, https://www.weforum.org/publications/digital-transformation-powering-the-great-reset/, (accessed on 15 of August 2024)
4. Roblek, V.; Dimovski, V. Essentials of 'the Great Reset' through Complexity Matching. Systems **2024**,12, 182. https://doi.org/10.3390/systems12060182
5. Marvin Cheung. 5 Ideas from Global Diplomacy, System-wide Transformation Methods to Close the Compliance Gap and Advance the 2030 Sustainable Development Goals, Ground Zero Books LLC New York, NY, 2024, Doi: https://doi.org/10.17613/xzxt-7567
6. Standardization Council Industry 4.0, https://www.sci40.com/english/thematic-fields/rami4-0/, (accessed on 12 of August 2024)
7. Zamora Iribarren, M.; Garay-Rondero, C.L.; Lemus-Aguilar,I.; Peimbert-García, R.E. A Review of Industry 4.0 Assessment Instruments for Digital Transformation. Applied Sciience.**2024**, 14, 1693. https://doi.org/10.3390/app14051693
8. Hinterhuber, A.; Vescovi, T.; Checchinato, F. (Eds.). Managing Digital Transformation: Understanding the Strategic Process (1st ed.). *Routledge*.**2021**, https://doi.org/10.4324/9781003008637
9. Zhang, Y.; Zhuang, G.; Lv, H.;i Li, Q. Research on Adaptive Intelligent Decision Algorithm in Industry 4.0 Digital Economy and Management Transformation Based on Clustering Algorithm. *Tehnički vjesnik, 31* (3), 870-880.**2024**. https://doi.org/10.17559/TV-20231205001174
10. Dossou, Paul-Eric.; Laouénan, G.; Didier, Jean-Yves. Development of a Sustainable Industry 4.0 Approach for Increasing the Performance of SMEs. *Processes*, **2022**, 10, 1092.https://doi.org/10.3390/pr10061092
11. X. X. Wang; J. Yang; Y. T. Wang; Q. H. Miao; F.-Y. Wang; A. J. Zhao; J.-L. Deng; L. X. Li; X. X. Na; L. Vlacic. Steps toward Industry 5.0: Building "6S" parallel industries with cyber-physical-social intelligence, *IEEE/CAA Journal of Automatica Sinica*, **2023**, Vol. 10, Iss. 8, pp.1692–1703, doi: 10.1109/JAS.2023.123753.
12. Alojaiman, B. Technological Modernizations in the Industry 5.0 Era: A Descriptive Analysis and Future Research Directions. *Processes,* **2023**, Vol.11, 1318. https://doi.org/10.3390/pr11051318
13. Madsen, D.Ø.; Slåtten, K. Comparing the Evolutionary Trajectories of Industry 4.0 and 5.0: A Management Fashion Perspective. *Applied System Innovation*, **2023**, Vol.6, 48, https://doi.org/10.3390/asi6020048
14. João Barata; Ina Kayser. Industry 5.0—Past, Present, and Near Future, *Procedia Computer Science*,**2023**, Vol.219, pp.978-788, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2023.01.351
15. Franziska Hein-Pensel; Heiner Winkler; Anja Brückner; Mandy Wölke; Iren Jabs; Ines Jasmin Mayan; Amit Kirschenbaum; Julia Friedrich; Christian Zinke-Wehlmann. Maturity assessment for Industry 5.0: A review of existing maturity models, *Journal of Manufacturing Systems*, **2023,**Vol. 66, pp.200-210, ISSN 0278-6125, https://doi.org/10.1016/j.jmsy.2022.12.009.
16. Mariia Golovianko; Vagan Terziyan; Vladyslav Branytskyi; Diana Malyk. Industry 4.0 vs. Industry 5.0: Co-existence, Transition, or a Hybrid, *Procedia Computer Science*, **2023,** Vol. 217, pp.102-113, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2022.12.206.
17. Rahmani, R.; Jesus, C.; Lopes, S.I. Implementations of Digital Transformation and Digital Twins: Exploring the Factory of the Future.Processes **2024**, 12, 787. https://doi.org/10.3390/pr12040787
18. Duggal, A.S.; Malik, P.K.; Gehlot, A.; Singh, R.; Gaba, G.S.; Masud, M.; Al-Amri, J.F. A sequential roadmap to industry 6.0: Exploring future manufacturing trends. IET Community. Vol.16, pp.521–531, **2022**, https://doi.org/10.1049/cmu2.12284
19. Das, S.; Pan, Tanushree. A strategic outline of Industry 6.0: Exploring the Future, *Available at SSRN:* http://dx.doi.org/10.2139/ssrn.4104696, **2022,** (accessed on 09 of September 2024)
20. Tavakkoli-Moghaddam, R.; Nozari, H.; Bakhshi-Movahed, A.; Bakhshi-Movahed, A. A Conceptual Framework for the Smart Factory 6.0. *In M. Oskounejad & H. Nozari (Eds.), Advanced Businesses in Industry 6.0* (pp. 1-14). IGI Global.,**2024**, https://doi.org/10.4018/979-8-3693-3108-8.ch001
21. Hsiao, Ming-Hsiung. Resource integration and firm performance through organizational capabilities for digital transformation, *Digital Transformation and Society*, Vol. *ahead-of-print* No. *ahead-of-print*.,**2024**, https://doi.org/10.1108/DTS-07-2023-0050
22. Marcus Fischer; Florian Imgrund; Christian Janiesch, Axel Winkelmann. Strategy archetypes for digital transformation: Defining meta objectives using business process management, *Information & Management*, Volume 57, Issue 5, **2020**, 103262, ISSN 0378-7206, https://doi.org/10.1016/j.im.2019.103262.
23. Elia, G.; Solazzo, G.; Lerro, A.; Pigni, F. & Tucci, C. L. The digital transformation canvas: A conceptual framework for leading the digital transformation process. *Business Horizons*., **2024**, Volume 67, pp.381-398,https://doi.org/10.1016/j.bushor.2024.03.007
24. Popoola, O. A.; Adama H. E.; Chukwuekem D. O; Akinoso A.E. Conceptualizing agile development in digital transformations: Theoretical foundations and practical applications. *Engineering Science & Technology Journal*, Volumn *5*(4), pp. 1524-1541,**2024**, https://doi.org/10.51594/estj.v5i4.1080
25. Gregory Vial. Understanding digital transformation: A review and a research agenda, *The Journal of Strategic Information Systems*,Volume 28, Issue 2, **2019**, pp. 118-144,ISSN 0963-8687, https://doi.org/10.1016/j.jsis.2019.01.003.

26. Aurangzeab Butt; Faisal Imran; Petri Helo; Jussi Kantola. Strategic design of culture for digital transformation,*Long Range Planning*,Volume 57, Issue 2,**2024**,102415,ISSN 0024-6301,https://doi.org/10.1016/j.lrp.2024.102415.
27. Laura Ruiz; Jose Benitez; Ana Castillo; Jessica Braojos. Digital human resource strategy: Conceptualization, theoretical development, and an empirical examination of its impact on firm performance, *Information & Management*,Volume 61, Issue 4,2024,103966,ISSN 0378-7206,https://doi.org/10.1016/j.im.2024.103966.
28. Strohmeier, S. Digital human resource management: A conceptual clarification. *German Journal of Human Resource Management*, **2020**, Volumn 34(3), pp.345-365. https://doi.org/10.1177/2397002220921131
29. Ben Ghrbeia, S.; Alzubi, A. Building Micro-Foundations for Digital Transformation: A Moderated Mediation Model of the Interplay between Digital Literacy and Digital Transformation. Sustainability **2024,** 16, 3749. https://doi.org/10.3390/su16093749)
30. Zhu, L.; Wang, J.; Li, J. Exploring the Roles of Artifacts in Speculative Futures: Perspectives in HCI. *Systems,* **2024**, 12, 194. https://doi.org/10.3390/systems12060194
31. Bantay, L.; Abonyi, J. Machine Learning-Supported Designing of Human–Machine Interfaces. *Appl. Sci.* **2024**, 14, 1564. https://doi.org/10.3390/app14041564
32. Wasilewski, A. Functional Framework for Multivariant E-Commerce User Interfaces, Journal *of Theoretical and Applied Electronic. Commerce Research*. **2024**, 19, 412–430. https://doi.org/10.3390/jtaer19010022
33. Liu, Yingchia; Yang Xu; Runze Song. Transforming User Experience (UX) through Artificial Intelligence (AI) in interactive media design. *Journal of Research in Science and Engineering,* Volume 6, Issue 9, **2024**, pp.1-7, https://doi.org/10.53469/jrse.2024.06(09).01
34. Casteleiro-Pitrez, J. Generative Artificial Intelligence Image Tools among Future Designers: A Usability, User Experience, and Emotional Analysis. *Digital,* **2024**, Vol. 4, pp.316–332, https://doi.org/10.3390/digital4020016
35. Sethi, S. ; Panda, S. Transforming Digital Experiences: The Evolution of Digital Experience Platforms (DXPs) from Monoliths to Microservices: A Practical Guide. *Journal of Computer and Communications*, Vol.12, pp.142-155,**2024**, https://doi.org/10.4236/jcc.2024.122009
36. Jie Li; Hancheng Cao; Laura Lin; Youyang Hou; Ruihao Zhu; Abdallah El Ali. User Experience Design Professionals' Perceptions of Generative Artificial Intelligence. *In Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI '24). Association for Computing Machinery*, **2024**,New York, NY, USA, Article 381, pp.1–18. https://doi.org/10.1145/3613904.3642114
37. Al Naqbi, H.; Bahroun, Z.; Ahmed, V. Enhancing Work Productivity through Generative Artificial Intelligence: A Comprehensive Literature Review. *Sustainability* **2024**, 16, 1166. https://doi.org/10.3390/su16031166
38. Partarakis, N.; Zabulis, X. A. Review of Immersive Technologies, Knowledge Representation, and AI for Human-Centered Digital Experiences. *Electronics* **2024**, 13, 269. https://doi.org/10.3390/electronics13020269
39. Laine, T.H.; Suk, H.J. Investigating User Experience of an Immersive Virtual Reality Simulation Based on a Gesture-Based User Interface. *Appl. Sci.* **2024**, 14, 4935. https://doi.org/10.3390/app14114935
40. Wu, H.; Ji, P.; Ma, H.; Xing, L. A Comprehensive Review of Digital Twin from the Perspective of Total Process: Data, Models, Networks and Applications. *Sensors* **2023**, 23, 8306. https://doi.org/10.3390/s23198306
41. Rawashdeh, A.; Abdallah, A.B.; Alfawaeer, M.; Al Dweiri, M.; Al-Jaghbeer, F. The Impact of Strategic Agility on Environmental Sustainability: The Mediating Role of Digital Transformation. *Sustainability* **2024**, *16*, 1338. https://doi.org/10.3390/su16031338
42. Aldoseri, A.; Al-Khalifa, K.N.; Hamouda, A.M. AI-Powered Innovation in Digital Transformation: Key Pillars and Industry Impact. *Sustainability* **2024**, 16, 1790. https://doi.org/10.3390/su16051790
43. Zainuddin, A. A., Zakirudin, M. A. Z., Syafiq Zulkefli, A. S., Mazli, A. M., Mohd Wardi, M. A. S., Fazail, M. N., Mohd Razali, M. I. Z., & Yusof, M. H. Artificial Intelligence: A New Paradigm for Distributed Sensor Networks on the Internet of Things: A Review. *International Journal on Perceptive and Cognitive Computing, 10*(1), 16–28., **2024**, https://doi.org/10.31436/ijpcc.v10i1.414
44. W3C Recommendation, https://www.w3.org/TR/wot-architecture/, (accesed on 26 of August 2024)
45. Onopa, S.; Kotulski, Z. State-of-the-Art and New Challenges in 5G Networks with Blockchain Technology. Electronics **2024**, 13, 974.https://doi.org/10.3390/electronics13050974
46. Perišić, A.; Perišić, I.; Perišić, B. Simulation-Based Engineering of Heterogeneous Collaborative Systems—A Novel Conceptual Framework. *Sustainability*, *15*(11), 8804, **2023**, https://doi.org/10.3390/su15118804
47. Perišić, A.; Perišić, B. The Foundation for Open Component Analysis: A System of Systems Hyper Framework Model. *Chapter In Advances in Principal Component Analysis. IntechOpen. Pedro García Márquez, F. (Ed.),* **2022**, https://doi.org/10.5772/intechopen.103830
48. Perisic, A.; Perisic, B. Digital Twins Verification and Validation Approach through the Quintuple Helix Conceptual Framework. *Electronics*, **2024**, *13*(16), 3303. https://doi.org/10.3390/electronics13163303

49. Guide to the Systems Engineering Body of Knowledge (SEBoK), Version 2.10, *Editor's Corner,pp.2, What's the holdup with Digital Engineering?*, **2024**, available as a pdf dowenlodable version at https://sebokwiki.org/wiki/Guide_to_the_Systems_Engineering_Body_of_Knowledge_(SEBoK),
50. Ester Sloth. Multi-, inter-, and transdisciplinarity:what is what?, *Utrecht Univeristy, Education Development and Training*, available at https://www.uu.nl/en/education/educational-development-training/knowledge-dossiers/interdisciplinary-education-and-cel/multi-inter-and-transdisciplinarity-what-is-what, (accessed on 28 of September 2024).
51. Ebbs, P. J.; Ward, S. A. ISO 18404: A Model for Lean Transformation in an Alliance. In D. B. Costa, F. Drevland, & L. Florez-Perez (Eds.), Proceedings of the 32nd Annual Conference of the International Group for Lean Construction (IGLC32) (pp. 1255–1267), **2024**, doi.org/10.24928/2024/0225
52. Adams, K.M.; Ibrahim, I.; Krahn, S. Engineering Systems with Standards and Digital Models: Development of a 15288-SysML Grid. *Systems* **2024**, 12, 276. https://doi.org/10.3390/systems12080276
53. Guide to the Systems Engineering Body of Knowledge (SEBoK), Version 2.10, *pdf document*, *pp.83-731*, *Knowledge Areas*, avaliable at https://sebokwiki.org/wiki/Guide_to_the_Systems_Engineering_Body_of_Knowledge_(SEBoK), **2024**, (dowenloaded on 15 of September 2024)
54. INCOSE Systems Engineering Standards, available at https://www.incose.org/about-systems-engineering/se-standards, (accessed on 15 of September 2024)
55. Adler, R.; Elberzhager, F.; Falcao, R.; Siebert, J. Defining and Researching "Dynamic Systems of Systems". *Software* **2024**, 3, 183–205. https://doi.org/10.3390/software3020009
56. Wikipedia, Systems engineering, https://en.wikipedia.org/wiki/Systems_engineering, (accessed on 21 of August 2024)
57. Kunc, M. The Systems Thinking Approach to Strategic Management. *Systems* **2024**, 12, 213. https://doi.org/10.3390/systems12060213
58. Introduction to Systems Thinking, available at https://bigthinking.io/systems-thinking-innovation-overview/, (accessed on 21 of August 2024)
59. Paddeu, D.; Lyons, G. Foresight through developing shared mental models: The case of Triple Access Planning. *Futures*, *155*, 103295. **2023**, https://doi.org/10.1016/j.futures.2023.103295
60. Muller, G., Giudici, H. Social Systems of Systems Thinking to Improve Decision-Making Processes Toward the Sustainable Transition. *In: Salado, A., Valerdi, R., Steiner, R., Head, L. (eds) The Proceedings of the 2024 Conference on Systems Engineering Research. CSER 2024. Conference on Systems Engineering Research Series. Springer, Cham.***2024**, https://doi.org/10.1007/978-3-031-62554-1_21
61. Ryan Bell; Ryan Longshore; Raymond Madachy.Introducing SysEngBench: A Novel Benchmark for Assessing Large Language Models in Systems Engineering, *Acquisition Research Program, Acquisition Management;SYM-AM-24-072*, 2024, https://dair.nps.edu/handle/123456789/5135
62. Gregoriades, A.; Sutcliffe, A. Using Task Support Requirements during Socio-Technical Systems Design. *Systems* **2024**, 12, 348. https://doi.org/10.3390/systems12090348
63. M. Torkjazi; A. K. Raz. A Review on Integrating Autonomy Into System of Systems: Challenges and Research Directions, In *IEEE Open Journal of Systems Engineering*, Vol. 2, pp.157-178, **2024**, doi: 10.1109/OJSE.2024.3456037
64. Restrepo-Carmona, J.A.; Taborda, E.A.; Paniagua-Garcia, E.; Escobar, C.A.; Sierra-Perez, J.; Vasquez, R.E. On the Integration of Complex Systems Engineering and Industry 4.0 Technologies for the Conceptual Design of Robotic Systems. *Machines* **2024**, 12, 625. https://doi.org/10.3390/machines12090625
65. Boy, G.A.; Masson, D.; Durnerin, É.; Morel, C. PRODEC for human systems integration of increasingly autonomous systems, *INCOSE Systems Engineering*, Vol. 27, Issue 5, pp.805-826, **2024**,https://doi.org/10.1002/sys.21751.
66. Nilsson, J.; Javed, S.; Albertsson, K.; Delsing, J.; Liwicki, M.; Sandin, F. AI Concepts for System of Systems Dynamic Interoperability. *Sensors* **2024**, 24, 2921. https://doi.org/10.3390/s24092921
67. Wach P.; Topcu T.G.; Jung S.; Sandman B.; Kulkarni A.U.; Salado A. A systematic literature review on the mathematical underpinning of model-based systems engineering. *Systems Engineering*. **2024**;pp.1-20, https://doi.org/10.1002/sys.21781
68. Pierre Borque; Richard E. (Dick) Fairley Editors. SWEBOK V3.0, Guide to the Software Engineering Body of Knowledge, IEEE Computer Society, available at https://www.computer.org/education/bodies-of-knowledge/software-engineering, (accessed on 12 of August 2024)
69. Institute of Data. Understanding Standards and Guidelines in Software Engineering, available at https://www.institutedata.com/blog/standards-and-guidelines-in-software-engineering/, (accessed on 28 of September 2024)
70. S. McConnell; L. Tripp; Guest Editors; Introduction: Professional Software Engineering-Fact or Fiction?, *in IEEE Software*, Vol. 16, No. 06, pp. 13-18, **1999**, doi: 10.1109/MS.1999.805468

71. Joanna Leng. The development of research software engineering as a profession, *Open Access Government July 2023*, pp.278-279, Available at https://www.openaccessgovernment.org/article/research-software-engineering-as-a-profession-rse/161896/, (accessed on 01 of October 2024)
72. Dwivedi, K.; Haghparast, M.; Mikkonen, T. Quantum software engineering and quantum software development lifecycle: a survey. *Cluster Comput* **27**, 7127–7145, **2024**. https://doi.org/10.1007/s10586-024-04362-1
73. Arif Ali Khan; Aakash Ahmad; Muhammad Waseem; Peng Liang; Mahdi Fahmideh; Tommi Mikkonen; Pekka Abrahamsson. Software architecture for quantum computing systems — A systematic review, *Journal of Systems and Software*, Vol. 201, **2023**,111682,ISSN 0164-1212, https://doi.org/10.1016/j.jss.2023.111682.
74. Muhammad Azeem Akbar; Arif Ali Khan; Sajjad Mahmood; Saima Rafi. Quantum Software Engineering: A New Genre of Computing. In Proceedings of the 1st ACM International Workshop on Quantum Software Engineering: The Next Evolution (QSE-NE 2024). Association for Computing Machinery, New York, NY, USA, 1–6. **2024**, https://doi.org/10.1145/3663531.3664750
75. Salam, M.; Ilyas, M. Quantum computing challenges and solutions in software industry—a multivocal literature review, *IET Quant. Comm.* 1–24 ,**2024**. https://doi.org/10.1049/qtc2.12096
76. J. L. Hevia; G. Peterssen; C. Ebert; M. Piattini. Quantum Computing, in *IEEE Software*, Vol. 38, No. 5, pp.7-15, **2021**, doi: 10.1109/MS.2021.3087755.
77. Sarkar, A. Automated quantum software engineering, *Automated Software Engineering* **31**, 36, **2024**, https://doi.org/10.1007/s10515-024-00436-x
78. Ilyas M; Khan SU; Khan HU; Rashid N. Software integration model: An assessment tool for global software development vendors. *Journal of Software: Evolution and Process*, 36(4):e2540. **2024,** https://doi.org/10.1002/smr.2540
79. Bull, R. L.; Dudukovich, R. M.; Fraire, J. A.; Kortas, N.; Kassouf-Short, R.; Smith, A.; Schweinsberg, E. Network Emulation Testbed Capabilities for Prototyping Space DTN Software and Protocols. In *The 11th International Workshop on Computer and Networking Experimental Research using Testbeds (CNERT 2024)*, **2024**, available at https://ntrs.nasa.gov/api/citations/20240002839/downloads/CNERT2024.pdf
80. Houssam ElBouanani; Chadi Barakat; Walid Dabbous; Thierry Turletti. Fidelity-aware large-scale distributed network emulation, *Computer Networks*,Volume 250, **2024**,110531,ISSN 1389-1286, https://doi.org/10.1016/j.comnet.2024.110531.
81. Fachrur Rozi, N. R.; Nurhayati A.; Arandiant Rozano, S. Implementation OSPFv3 For Internet Protocol Verses 6 (IPv6) Based On Juniper Routers Use Emulator Virtual Engine—Next Generation (Eve-NG). *International Journal of Engineering Continuity*, *3*(1), 1–11., **2023**, https://doi.org/10.58291/ijec.v3i1.141
82. Zhu, M.; Zhang, J.; Hua, B. et al. Ultra-wideband fiber-THz-fiber seamless integration communication system toward 6G: architecture, key techniques, and testbed implementation. *Sci. China Inf. Sci.* Vol.66, 113301, **2023**, https://doi.org/10.1007/s11432-022-3565-3
83. Zhiqing Wei; Hanyang Qu; Yuan Wang; Xin Yuan; Huici Wu; Ying Du; Kaifeng Han; Ning Zhang; Zhiyong Feng. Integrated Sensing and Communication Signals Toward 5G-A and 6G: A Survey," in *IEEE Internet of Things Journal*, Vol. 10, no. 13, pp.11068-11092, **2023**, DOI: 10.1109/JIOT.2023.3235618.
84. A. Alkhateeb; S. Jiang; G. Charan. Real-Time Digital Twins: Vision and Research Directions for 6G and Beyond, in *IEEE Communications Magazine*, Vol. 61, no. 11, pp.128-134, **2023**, DOI:10.1109/MCOM.001.2200866.
85. C. -X. Wang et al. On the Road to 6G: Visions, Requirements, Key Technologies, and Testbeds, in *IEEE Communications Surveys & Tutorials*, Vol. 25, no. 2, pp.905-974, **2023**, DOI:10.1109/COMST.2023.3249835.
86. Muhammad Adil; Houbing Song; Muhammad Khurram Khan; Ahmed Farouk; Zhanpeng Jin. 5G/6G-enabled metaverse technologies: Taxonomy, applications, and open security challenges with future research directions,*Journal of Network and Computer Applications*, Vol. 223, **2024**,103828,ISSN 1084-8045, https://doi.org/10.1016/j.jnca.2024.103828.
87. S. Hafeez et al. Blockchain-Assisted UAV Communication Systems: A Comprehensive Survey, in *IEEE Open Journal of Vehicular Technology*, Vol. 4, pp.558-580, **2023**, DOI: 10.1109/OJVT.2023.3295208
88. Khaleel, M.; Ahmed, A. A.; Alsharif, A. Artificial Intelligence in Engineering, *Brilliance: Research of Artificial Intelligence*, Vol.*3*(1), pp.32-42,**2023**, https://doi.org/10.47709/brilliance.v3i1.2170
89. Z. Qin; F. Gao; B. Lin; X. Tao; G. Liu; C. Pan, A Generalized Semantic Communication System: From Sources to Channels, in *IEEE Wireless Communications*, Vol. 30, no. 3, pp.18-26, **2023**, DOI:10.1109/MWC.013.2200553C.
90. Chaccour, W. Saad; M. Debbah; Z. Han; H. V. Poor, Less Data, More Knowledge: Building Next Generation Semantic Communication Networks, in *IEEE Communications Surveys & Tutorials*, **2024**, DOI:10.1109/COMST.2024.3412852
91. G. Zhang; Q. Hu; Z. Qin; Y. Cai; G. Yu; X. Tao. A Unified Multi-Task Semantic Communication System for Multimodal Data, in *IEEE Transactions on Communications*, Vol. 72, no. 7, pp. 4101-4116, **2024**, DOI:10.1109/TCOMM.2024.3364990

92. Z. Yang; M. Chen; G. Li; Y. Yang; Z. Zhang. Secure Semantic Communications: Fundamentals and Challenges, in *IEEE Network*, **2024**, DOI:10.1109/MNET.2024.3411027
93. A. Mavromatis; C. Colman-Meixner; A. P. Silva; X. Vasilakos; R. Nejabati; D. Simeonidou. A Software-Defined IoT Device Management Framework for Edge and Cloud Computing, in *IEEE Internet of Things Journal*, Vol. 7, no. 3, pp.1718-1735, 2020, DOI:10.1109/JIOT.2019.2949629
94. Dionysis Zindros; Apostolos Tzinas; David Tse. Rollerblade: Replicated Distributed Protocol Emulation on Top of Ledgers,Cryptology Archive, Paper 2024/210, **2024**, available at https://eprint.iacr.org/2024/210, (accessed on 5 of October 2024)
95. Computer cluster, In *Wikipedia*, https://en.wikipedia.org/wiki/Computer_cluster, (accessed on 6 of October 2024)
96. Private network, In *Wikipedia*, https://en.wikipedia.org/wiki/Private_network (accessed on 6 of October 2024)
97. Public computer, In *Wikipedia*, https://en.wikipedia.org/wiki/Public_computer, (accessed on 7 of October 2024)
98. Computer network. In *Wikipedia*, https://en.wikipedia.org/wiki/Computer_network, (accessed on 7 of October 2024)
99. Computer science Wiki, Type of Networks, https://computersciencewiki.org/index.php/Types_of_networks, (accessed on 7 of October 2024)
100. Hoa T. Nguyen; Muhammad Usman; Rajkumar Buyya. QFaaS: A Serverless Function-as-a-Service framework for Quantum computing, *Future Generation Computer Systems*, Vol. 154,**2024**,pp.281-300, ISSN 0167-739X, https://doi.org/10.1016/j.future.2024.01.018.
101. Aakash Ahmad; Ahmed B. Altamimi; Jamal Aqib. A reference architecture for quantum computing as a service, *Journal of King Saud University—Computer and Information Sciences*, Vol. 36, Issue 6,**2024**,102094,ISSN 1319-1578, https://doi.org/10.1016/j.jksuci.2024.102094.
102. Saskia Witteborn. Digitalization, Digitization and Datafication: The "Three D" Transformation of Forced Migration Management, *Communication, Culture and Critique*, Vol. 15, Iss. 2, **2022**, pp.157–175, https://doi.org/10.1093/ccc/tcac007
103. Stefanescu, D.; Galán-García, P.; Montalvillo, L.; Unzilla, J.; Urbieta, A. Industrial Data Homogenization and Monitoring Scheme with Blockchain Oracles, *Smart Cities* **2023**, Vol. 6, pp.263–290, https://doi.org/10.3390/smartcities6010013
104. Leonidas Anthopoulos; Ioannis Nikolaou. Homogenizing Data Flows in Smart Cities: Value-driven use Cases in the Era of Citiverse. In *Companion Proceedings of the ACM Web Conference 2024 (WWW '24). Association for Computing Machinery, New York, NY, USA*, **2024**, pp.1367–1371, https://doi.org/10.1145/3589335.3651900
105. M.S. Islam; M.A. Rahman; M.A.B. Ameedeen; H. Ajra; Z.B. Ismail; J.M. Zain. Blockchain-Enabled Cybersecurity Provision for Scalable Heterogeneous Network: A Comprehensive Survey, *Comput. Model. Eng. Sci.*, Vol. 138, No. 1, pp. 43-123. **2024**. https://doi.org/10.32604/cmes.2023.028687
106. Mengistu, T. M.; Kim, T.; Lin, J. A Survey on Heterogeneity Taxonomy, Security and Privacy Preservation in the Integration of IoT, Wireless Sensor Networks and Federated Learning, *Sensors* **2024**, 24, 968, https://doi.org/10.3390/s24030968
107. Casamayor Pujol, V.; Morichetta, A.; Murturi, I.; Kumar Donta, P.; Dustdar, S. Fundamental Research Challenges for Distributed Computing Continuum Systems. *Information* **2023**, 14, 198, https://doi.org/10.3390/info1403019
108. Yuan Feng; Yaqian Qi; Hanzhe Li; Xiangxiang Wang; Jingxiao Tian. Leveraging federated learning and edge computing for recommendation systems within cloud computing networks, in *Proc. SPIE 13210, Third International Symposium on Computer Applications and Information Systems (ISCAIS 2024)*, 1321012, **2024**, https://doi.org/10.1117/12.3034773
109. Wang, J.; Chen, C.; Li, S.;Wang, C.; Cao, X.; Yang, L. Researching the CNN Collaborative Inference Mechanism for Heterogeneous Edge Devices. *Sensors* **2024**, 24, 4176. https://doi.org/10.3390/s24134176
110. Wu, C.; Fan, H.; Wang, K.; Zhang, P. Enhancing Federated Learning in Heterogeneous Internet of Vehicles: A Collaborative Training Approach. *Electronics* **2024**, 13, 3999.https://doi.org/10.3390/electronics13203999
111. B. A. Ferreira; T. Petrović; M. Orsag; J. R. Martínez-de Dios; S. Bogdan. Distributed Allocation and Scheduling of Tasks With Cross-Schedule Dependencies for Heterogeneous Multi-Robot Teams, in *IEEE Access*, Vol. 12, pp.74327-74342, **2024**, DOI: 10.1109/ACCESS.2024.3404823
112. IBM, What are containers?, https://www.ibm.com/topics/containers (accessed on 15 of October 2024)
113. IBM, What is virtualization?, https://www.ibm.com/topics/virtualization (accessed on 15 of October 2024)
114. Fei Tao; Xin Ma; Weiran Liu; Chenyuan Zhang. Digital Engineering: State-of-the-art and perspectives, *Digital Engineering*, Vol. 1, **2024**, 100007, ISSN 2950-550X, https://doi.org/10.1016/j.dte.2024.100007
115. I Made Putrama; Péter Martinek. Heterogeneous data integration: Challenges and opportunities, *Data in Brief*, Vol. 56, **2024**, 110853, ISSN 2352-3409, https://doi.org/10.1016/j.dib.2024.110853.
116. https://sageitinc.com/reference-center/what-is-data-abstraction-layer, (accessed on 16 of October 2024)

117. https://www.proprofskb.com/blog/best-document-collaboration-tools/, (accessed on 18 of October 2024)
118. Buede, Dennis M.; William D. Miller. The engineering design of system, in *Pierre Bourgue; Richard E.s: models and methods. John Wiley & Sons*, **2024**, ISBN: 978-1-119-98403-0, pp.3-7
119. Rhee, H. L. A New Lifecycle Model Enabling Optimal Digital Curation. *Journal of Librarianship and Information Science*, *56*(1), 241-266., **2024**, https://doi.org/10.1177/09610006221125956
120. Zachman Framework, https://www.zachman.com/resources/ea-articles-reference/327-the-framework-for-enterprise-architecture-background-description-and-utility-by-john-a-zachman**,** (accessed on 21 of October 2024)
121. The Open Group, The Open Group Architecture Framework, (TOGAF), https://www.opengroup.org/togaf, https://www.opengroup.org/architecture-forum, (accessed on 21 of October 2024)
122. FEA, https://www.feacinstitute.org/**,** (accessed on 21 of October 2024)
123. U.S. Department of Defense, DoDAF, https://dodcio.defense.gov/Library/DoD-Architecture-Framework/dodaf20_viewpoints/, (accessed od 21 of October 2024)
124. NATO Architecture Framework NAF, https://www.nato.int/cps/en/natohq/topics_157575.htm, (accessed on 21 of October 2024)
125. Internationa Standards Organization, ISO EAF, https://www.iso.org/obp/ui/es/#iso:std:iso:15704:ed-2:v1:en, (accessed on 20 of October 2024)
126. Carnagie Mellon University, Software Engineering Institute, Capability Maturity Model Integrated, https://cmmiinstitute.com/, (accessed on 21 of October 2024)
127. SFIA, The global skills and competency framework for the digital world, available at, https://sfia-online.org/en, (accessed on 22 of October 2024)
128. Object Management Group, https://www.omg.org/about/omg-standards-introduction.htm, (accessed on 20 of October 2024)
129. Sparx Systems EAF, https://www.sparxsystems.eu/enterprise-architect/, (accessed on 22 of October 2024)
130. SAP LenIX EAF, https://www.leanix.net/en/wiki/ea/enterprise-architecture-frameworks, (accessed on 22 of October 2024)
131. Scaled Agile Framework (SAFe®), https://scaledagile.com/what-is-safe/, (accessed on 22 of October 2024)
132. Large Scale Scrum (LeSS) Framework, https://less.works/less/framework/product, (accessed on 22 of October 2024)
133. Nexus, https://nexus.hexagon.com/, (accessed on 22 of October 2024)
134. Scrum@Scale, https://www.scrumatscale.com/scrum-at-scale-guide/, (accessed on 22 of October 2024)
135. Enterprise Scrum, https://www.scrumstudy.com/whyscrum/scrum-for-enterprise, (accessed on 22 of October 2024)