Towards a Methodology for the Definition and Evaluation of Enterprises through Modeling and Simulation: Application to Product Development

Sean Ryan Mueller*, Olivia J. Pinon Fischer†, Burak Bagdatli‡, Selcuk Cimtalay§ and Dimitri N. Mavris¶
School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA, USA

Alberto Faveto‖
Capgemini Engineering Research & Development, Aix-en-Provence, France

Benjamin Boulbene **
Capgemini Engineering, Bristol, UK

Antonio Mazzoldi†† and Nicolas Croue ‡‡
Capgemini Engineering, Boston, MA, USA

Enterprises today have to navigate a rapidly evolving landscape driven by technological advancements, intensifying global competition, and shifting market dynamics. To remain competitive, organizations must adopt agile and digital solutions that enhance operational efficiency, optimize processes, and accelerate time-to-market. This research introduces a comprehensive methodology designed to evaluate enterprise performance across a variety of scenarios, with a specific focus on product development processes. The proposed methodology rests on three pillars: modeling, simulation, and their integration. Leveraging Model-based Systems Engineering (MBSE) and the Unified Architecture Framework (UAF), the methodology provides a structured approach to model organizational elements such as processes, resources (tools, personnel), and goals, establishing a cohesive enterprise architecture. A holistic agent-based simulation facilitates a granular analysis of the enterprise operations. The simulation captures dynamic interactions and emergent behaviors, enabling quantitative evaluations of the individual impacts of digital tools, processes and human resources on organizational outcomes. The seamless integration between the modeling and simulation environments is ensured through the use of a centralized Authoritative Source of Truth (ASOT), maintaining consistency and traceability. This end-to-end approach enables scenario-based assessments and quantitative evaluations of alternative configurations, addressing the limitations of traditional EA frameworks by capturing dynamic interactions and emergent behaviors. This methodology therefore lays the groundwork for the further development of Digital Twins of Organizations (DTOs), one that facilitates strategic decision-making and optimization in complex enterprise environments.

*Graduate Research Assistant, Aerospace Systems Design Laboratory
†Principal Research Engineer, Aerospace Systems Design Laboratory, AIAA Associate Fellow
‡Research Engineer II, Aerospace Systems Design Laboratory, AIAA Member
§Senior Research Engineer, Aerospace Systems Design Laboratory
¶S.P. Langley NIA Distinguished Regents Professor and Director of Aerospace Systems Design Laboratory, AIAA Fellow
‖Product, System & PLM Engineer, Capgemini Engineering R&D
**Lead Digital Engineer, Capgemini Engineering
††Director of Systems Engineering, MBSE and MBE, Capgemini Engineering
‡‡Vice-President, Head of Digital Continuity, CTO, Capgemini Engineering

I. Nomenclature

ABS	=	Agent-Based Simulation
API	=	Application Programming Interface
ASOT	=	Authoritative Source of Truth
CDF	=	Cumulative Distribution Function
CSV	=	Comma Separated Values
DB	=	Database
DES	=	Discrete Event Simulation
DODAF	=	Department of Defense Architecture Framework
DTO	=	Digital Twin of an Organization
EA	=	Enterprise Architecture
JSON	=	JavaScript Object Notation
MBSE	=	Model-Based Systems Engineering
MODAF	=	Ministry of Defence Architecture Framework
M&S	=	Modeling and Simulation
NAF	=	NATO Architecture Framework
PD	=	Product Development
PDF	=	Probability Density Function
PLM	=	Product Lifecyle Management
QFD	=	Quality Function Deployment
SD	=	System Dynamics
SysML	=	Systems Modeling Language
TOGAF	=	The Open Group Architecture Framework
UAF	=	Unified Architecture Framework
XML	=	Extensible Markup Language

II. Introduction and Motivation

The contemporary business landscape is characterized by a relentless pace of change driven by technological advancements, evolving customer expectations, and intensifying global competition. To thrive in this dynamic environment, enterprises must embrace agility and digital transformation [1, 2]. Agility refers to an organization's ability to rapidly adapt to changing market conditions, swiftly adjust its strategies, and readily incorporate new technologies. Digital transformation encompasses the integration of digital technologies into all facets of a business, fundamentally altering how an enterprise operates and delivers value to its customers.

A. The Challenges of Digital Transformation

Digital transformation is driven by the continuous advancements in technologies such as cloud computing, big data analytics, mobile technologies, and artificial intelligence, which offer opportunities for increased efficiency, innovation, and customer engagement [3]. However, digital transformation is not merely about adopting and implementing new technologies; it is about fundamentally changing how an organization operates, involving the complex interplay between people, processes, technology, and data. As such, it requires a holistic approach that encompasses organizational change and a deep understanding of the strategic benefits and risks involved [2, 4]. Enterprises often face challenges in aligning digital strategies with business goals, managing resistance to change, developing the necessary skills and competencies within their workforce, and integrating new technologies into existing systems and processes [5]. This challenge is compounded by the difficulty of quantifying the impact that digital solutions at large have on key metrics of interest.

B. Leveraging Enterprise Modeling for the Successful Implementation of Digital Transformation Initiatives

Enterprise modeling helps to create a holistic view of an organization, encompassing its personnel, processes, data, and technology [6, 7]. This comprehensive understanding allows for identifying areas where digital technologies can be leveraged to achieve strategic objectives, such as increasing speed and agility or improving data-informed decision making.

Enterprise architectures (EA) serve as strategic frameworks to model an organization's structure, processes, systems, roles, and capabilities [8], providing decision-makers with valuable insights into an enterprise's current state and ability to meet future objectives. Organizational transformation through the use of EA is achieved by connecting various developmental initiatives cohesively, effectively translating strategic goals into actionable plans while ensuring cohesion among different enterprise elements. Such insights are valued in the industry, leading to an increase in the adoption of EA frameworks, as exemplified by the increase in EA-related publications, standards, and activities [9].

However, conventional EA frameworks, by their very nature, focus solely on representing and managing intricate structures through strategic decomposition and lack the capability for behavioral analysis at the enterprise level. Hence, while this approach is valuable for managing complexity and aligning tools, processes and (human) resources with business strategy, it often falls short in capturing the dynamic interactions and emergent behaviors that characterize real-world enterprises. This limitation can be addressed by coupling EA models with simulation models that mimic the behavior and dynamics of enterprise systems, allowing for their quantitative evaluation and the ability to make informed decisions about enterprise-wide changes [10, 11].

A recently proposed concept is that of Digital Twins of Organizations (DTOs). These offer a transformative approach to monitoring and managing enterprises by mirroring their resources, workforce, processes, and their complex interactions. Through simulation and advanced analytics, DTOs can transform how organizations identify inefficiencies, optimize processes, and make strategic data-driven decisions [12]. Previous implementations of DTOs in studies have shown potential for the use of DTOs in the context of product development [13, 14]. However, as mentioned, modeling the dynamic, non-deterministic, and continuously changing nature of socio-technical systems —such as human behavior, emergent interactions, and hidden dependencies such as those present in enterprises— presents significant challenges [15]. Addressing these complexities requires robust methodologies and accurate models to better understand, simulate, and optimize organizational behavior. Advancing these capabilities is essential to unlock the full potential of DTOs in enterprise modeling and simulation and pave the way for more adaptive and efficient organizations [12, 14, 15].

This paper is organized as follows. Section III provides relevant context to this paper's topic by defining core concepts as well as introducing relevant related works. Section IV describes the methods used along with the high-level modeling logic and simplifications/hypotheses made for addressing the problem of this paper. Section V details the implementation of the methodology, with the results analyzed in Section VI. Finally, Section VII summarizes the study's findings and offers recommendations for future research.

III. Background

This section provides a brief overview of existing enterprise architecture frameworks as well as a short review of relevant approaches to the simulation of new product development (PD) processes.

A. Enterprise Architecture Modeling

Modeling Enterprise Architectures (EA) is enabled through the synergistic use of Model-Based System Engineering (MBSE) tools and EA frameworks. Enterprise Architecture frameworks provide structured approaches for modeling complex organizations by organizing architectural elements into distinct domains and views. More specifically, these viewpoints and views help partition complex enterprise models into manageable segments, making them accessible to different stakeholder groups.

Many EA frameworks exist for defining enterprises and analyzing the relationships between their business capabilities, processes, information systems, and technology infrastructure. The most prominent ones are discussed briefly below.

DoDAF (Department of Defense Architecture Framework) and MoDAF (Ministry of Defence Architecture Framework) are both enterprise architecture frameworks that are used primarily in defense organizations [16]. DoDAF was developed by the U.S. Department of Defense to address system integration and interoperability challenges, particularly for long-lived systems. MODAF, created by the UK Ministry of Defence, provides a coherent set of rules and templates through various views to support defense planning, capability management, and acquisition.

The Unified Architecture Framework (UAF) emerged as an evolution of these frameworks, uniquely serving both commercial and defense sectors. UAF provides a comprehensive set of views and viewpoints for modeling various aspects of an enterprise, including strategic, operational, services, resources, and projects [17]. Other frameworks (e.g., TOGAF, Zachman) exist that have different emphases or coverage of architectural domains, often with a more isolated focus on IT itself.

A comparison of EA frameworks by *Bankauskaite* [16] showed that UAF has established itself as a leading framework

through its ability to create consistent enterprise architectures based on generic enterprise and system concepts with rich semantics. Its provision of rules for creating consistent enterprise architectures, combined with support for current DoDAF/MODAF/NAF requirements and future adaptability, makes it particularly suitable for complex enterprise modeling needs. For these reasons, UAF was selected as the EA framework for this research effort.

B. Simulation of Product Development Processes

Simulating product development (PD) processes presents unique challenges compared to business or manufacturing processes due to its inherently dynamic, parallelized, iterative, and uncertain nature [18]. Over the years, a variety of modeling and simulation (M&S) methods have been developed and employed, with the most notable approaches including discrete event simulation (DES), system dynamics (SD), and agent-based simulation (ABS) [19, 20].

Discrete Event Simulation (DES) - DES models represent a process as a sequence of events. In the context of PD, activities and tasks are modeled to reflect the overall process, which may be executed by organizational resources. DES has been commonly applied to analyze the impact of various task sequences on rework and iteration [21] as well as the effects of task overlapping and different work policies [22]. However, DES models typically rely on a predefined and rigid process flow. Efforts have been made to enhance adaptability in DES-based PD simulations [23], but such advancements remain limited.

System Dynamics (SD) - SD uses stocks and flows to model dynamic systems, with different variables influencing the size of stocks and flow rates. In the context of PD, it has been predominantly used to develop models of the "rework loop", originally proposed by *Cooper* [24]. These models abstract organizational elements, aggregating them into high-level variables—often termed management levers [20]. Although SD models provide valuable information, they lack granularity and are less suitable for capturing individual organizational components. Extensions to SD models include the addition of multiple loops for different stages or process steps [25], as well as the incorporation of causal loops to represent complex interdependencies among variables [26].

Agent-Based Simulation (ABS) - ABS provides a more holistic approach by modeling individual entities with distinct behaviors, decision-making capabilities, and interactions. While this adds complexity, it also offers greater flexibility for capturing diverse organizational aspects and evaluating the bottom-up impact of individual entities [27]. ABS has been used to study collaborative problem-solving behavior in product development [28, 29], as well as to build detailed models of organizations that represent individual people. For the latter, a distinction can be made between task-based ABS [30] and a recently proposed knowledge-based method by *Zhang and Thomson* [31]. Task-based ABS models use a process architecture akin to the ones used for DES, to guide the simulation execution with agents deciding what work to do and interacting with each other. In contrast, knowledge-based ABS models apply high-level process logic to a product architecture. The process then emerges based on the complexity and dependencies of the product, as well as the competencies and responsibilities of personnel.

The literature reveals a significant gap in holistic simulation methods that integrate all facets of PD—product, process, organization, and tools [19]. Most methods focus on one or two dimensions, often making assumptions about or neglecting the influence of others. This limitation frequently results in suboptimal simulation outcomes and restricted insights. Additionally, the simulation of the impact of tools on PD is rarely discussed in the literature. These gaps make studying the effects of changes on singular elements of PD difficult. As a result, existing M&S methods are seldom applied in industrial contexts [20].

This underscores the need for M&S methodologies capable of comprehensively modeling PD organizations while maintaining clarity, scalability, and applicability to industry. Especially, the impact of digital tools and their characteristics and interdependencies are never sufficiently represented in simulation models of PD, making accurate evaluations of digital transformation initiatives through EA simulation impossible. Addressing these gaps, this paper proposes a methodology to integrate all aspects of PD, ensuring a structured, comprehensible approach that facilitates the modeling of the numerous elements required for such comprehensive simulations.

C. Problem Statement

This research proposes a methodology to address gaps in enterprise modeling and simulation within the product development domain. The approach leverages SysML and model-based systems engineering (MBSE) tools to develop organizational models within established frameworks like the Unified Architecture Framework (UAF), recognized for their robust enterprise architecture (EA) capabilities.

The core objective of this effort is to create a balanced EA modeling approach that maintains equilibrium between complexity and utility - sophisticated enough to analyze interactions between various actors and elements, yet manageable

in terms of creation and maintenance. This model serves as the foundation for organizational simulation, enabling evaluation and optimization while providing managers with enhanced decision-making capabilities. More specifically, the methodology and M&S environment discussed below are leveraged to better quantify the impact that elements of digital transformation have on the enterprise of interest.

In addition, the proposed methodology incorporates customizable metric extraction capabilities to support various types of organizational assessment and analysis. By establishing procedures for enterprise modeling and simulation, along with frameworks for data sharing and integration, this research lays the groundwork for the development of Digital Twins of Organizations (DTOs). More broadly, this research augments the field of enterprise architecture development and simulation, with particular emphasis on applications within product development contexts, by contributing valuable insights to both theoretical frameworks and practical implementations in organizational modeling.

IV. Proposed Methodology

To effectively address the complexities of organizational design and interactions in the context of product development, a methodology is proposed that is founded on three interconnected pillars, as shown in Fig. 1:

- *Enterprise Definition (Left Pillar):* This pillar focuses on modeling all relevant enterprise elements and their interrelationships, including goals, requirements, processes, and resources. It allows for the creation and comparison of various configurations and what-if scenarios tailored to the specific objectives of the evaluation. This foundational step establishes the basis for subsequent simulation efforts.
- *Enterprise Simulation (Right Pillar):* This pillar involves simulating the modeled enterprise within specific contexts and configurations. It facilitates the quantitative evaluation and comparison of different scenarios by assessing their performance against defined metrics. Simulation outcomes provide critical insights, help verify goals and requirements, and support informed decision-making.
- *Modeling and Simulation Integration (Center Pillar):* This pillar ensures seamless integration between the modeling and simulation environments. It emphasizes data consistency, traceability, and digital continuity through an Authoritative Source of Truth (ASOT) [32]. In addition, centralized data storage supports traceability, efficient access, and version management, enabling streamlined updates and configurations across models.

The following sections describe each pillar (and associated methodologies) in more detail.

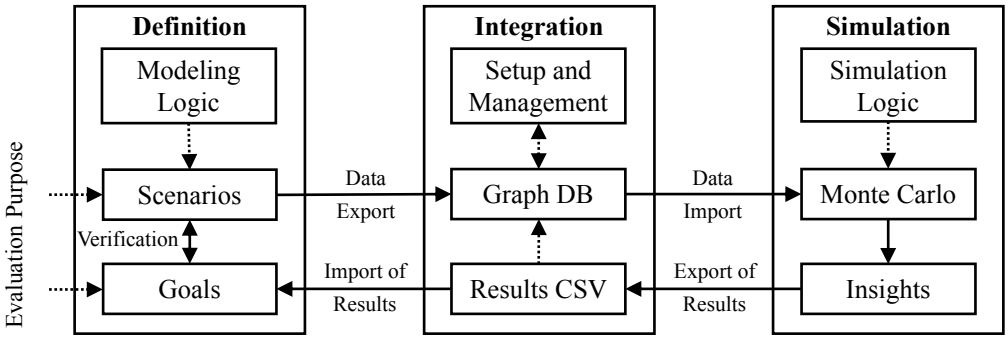

Fig. 1 Overarching Methodology

A. Modeling the Enterprise

Properly modeling an enterprise requires clearly identifying the purpose and intended use of the model at the outset. This ensures that only the necessary aspects of the system of interest are modeled [17]. As discussed, the Unified Architecture Framework (UAF) is used as it provides a comprehensive set of interconnected domains, model views, and viewpoints [33]. These serve as a foundation for creating domain-specific frameworks by selecting only the viewpoints relevant to a particular context. The development of UAF models follows a layered approach, as illustrated in Fig. 2. This approach will be partially applied to address the problem explored in this paper.

To model the enterprise in the context of product development, UAF is applied to an adaption of the ZOPH*

*ZOPH is a German acronym that stands for *Zielsystem* (goal system), *Objektsystem* (product system), *Prozesssystem* (process system), and *Handlungssystem* (agent system).

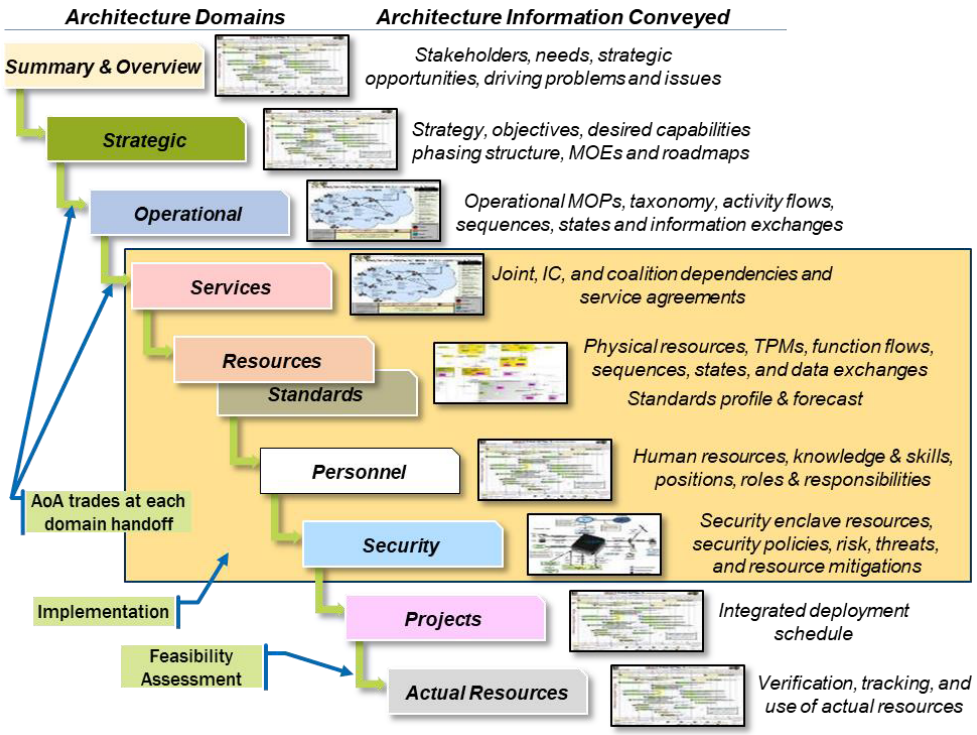

Fig. 2 Domain Layers guiding the Modeling Process in UAF [33]

framework [34], which divides product development into five distinct domains: product, process, organization, tools, and goals [18]. Figure 3 shows the high-level modeling logic that combines a scaled-down version of the UAF logic with the ZOPH framework, with the colors linking the domains of PD to the UAF domains.

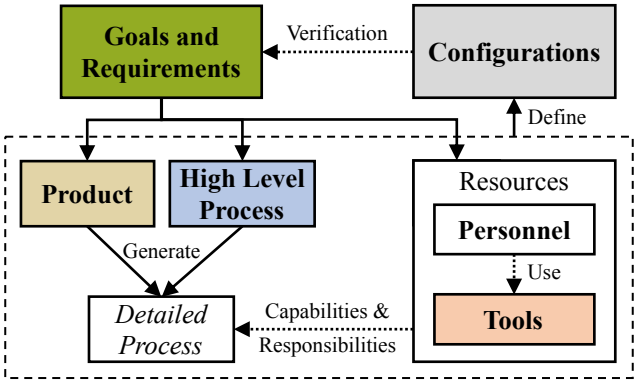

Fig. 3 High-level Modeling Architecture with Colors Representing the Corresponding UAF/SysML Domains

The Strategic layer begins by defining measures of goodness that outline the enterprise's goals and requirements within the goal domain. These measures can range from high-level metrics, such as cost, lead time, and quality [35, 36], to more detailed indicators, such as resource utilization, efficiency, bottlenecks, backlogs, adaptability, or breakdowns of high-level goals [30, 37, 38], depending on the evaluation's specific needs and objectives.

For instance, high-level requirements are suitable for comparing the performance of different configurations, whereas identifying bottlenecks necessitates more granular, lower-level requirements. In addition to project-specific goals, enterprise-level requirements can also be established to assess long-term organizational success, such as metrics related to knowledge management and organizational learning across the enterprise [39, 40].

The Resource layer of UAF is used to model the physical resources an organization uses [33]. To differentiate the

product being developed by the organization from the resources that help with this, the product will not be modeled within this layer. Therefore, the product is modeled using standard SysML elements.

Linking SysML elements with UAF elements is straightforward, as UAF is fundamentally built as an extension of SysML and UML. Such integration is particularly encouraged for scenarios requiring detailed alignment between product and enterprise models [41]. In addition, combining SysML and UAF offers a significant advantage: it facilitates seamless communication between systems engineers, who model the product in SysML, and enterprise architects or managers, who use UAF to plan enterprise capabilities and processes. This approach allows both groups to work within their preferred modeling frameworks while maintaining the ability to exchange data effectively.

With the goals and product defined, the project and its scope are established. Next the process is defined. This is only modeled at a high level in UAF's Operational layer to minimize the modeling effort. This approach avoids the need for detailed process definitions, which are often impractical for PD due to its inherent uncertainty [42]. A more detailed product development process can later be generated based on the high-level process, product definition, and available resources. Linking process generation to the product architecture is particularly effective, as activities in product development are often closely tied to specific subsystems or components [42].

The next step involves defining the resources—such as personnel and tools—required for product development. This is achieved using the Personnel and Resources layers of UAF, which capture the structure, measurements, and relationships of these resources. Establishing clear mappings between the resources and activities and product elements is needed to specify their capabilities and responsibilities effectively.

To create different configurations, the Actual Resources layer is utilized to instantiate elements from the previously defined layers. These instances specify values, measurements, structures, and relationships necessary for simulation. The simulation results are then integrated into the EA model, enabling requirements verification and completing the end-to-end methodology.

Because this paper focuses on product development, the selected layers are limited to the domains defined by the ZOPH framework, excluding additional layers or views such as the Service, Security, and Project domains. For a comprehensive EA model of the entire enterprise, additional views could be incorporated to address non-product-development aspects. Furthermore, the previously defined views can be adapted for more detailed definitions if necessary. A potential future extension could involve detailing tools more explicitly. For example, a testing department could be treated as a "tool" within product development. This could then be modeled in the Service layer and further decomposed to represent its structure and functionality.

B. Simulating the Enterprise

Leveraging the simulation toolkit within MBSE tools could simplify the implementation of the complete methodology. For simpler cases, simulations—or quantitative evaluations and calculations of Enterprise Architectures (EA) using UAF—can be performed directly within modeling environments (e.g., [43]). However, as noted in [44], which examined various methods for simulating enterprise architectures, more complex simulations are best conducted outside of MBSE tools. This is due to limitations such as insufficient support for UAF and issues with completeness and consistency when using functional mock-up units [44]. Due to the complexity and nature of the problem to be simulated, an external simulation approach is necessary. As a result, data export capabilities between the model and simulation environment also need to be developed.

Among the simulation approaches discussed in Section III.B, agent-based simulation (ABS) was selected for this research. ABS provides the highest level of flexibility needed to model the various entities, behaviors, and interactions outlined in the proposed EA modeling methodology. Specifically, the knowledge-based ABS approach by *Zhang and Thomson* [31] was chosen as the baseline model. This approach is recognized for its holistic integration of product, process, and organizational aspects in product development [19]. Moreover, as discussed in [45], it has already been applied to SysML for the purpose of developing a digital twin of system design processes, hence demonstrating its suitability for EA modeling for the problem at hand.

To address the absence of a tool domain and to enhance the scope of the process domain, this paper proposes substantial modifications to the framework developed by [31]. Furthermore, the integration of the PD goal domain into the simulation is improved by not only extracting high-level information but also by measuring data dynamically during simulation execution. A comprehensive description of the baseline modeling approach is provided below.

1. Description of the Baseline Simulation Logic

Zhang and Thomson's [31] simulation model employs an agent-based approach to model product development as a dynamic, nonlinear process that reflects the interplay of technical tasks, knowledge requirements, and interactions between agents. In their paper, the product is represented as a hierarchy of interdependent functions organized in a function tree, where child functions are developed first and subsequently integrated into parent functions. Each function is characterized by a technical complexity metric, determined by its knowledge requirements, and an integration complexity metric, derived from the number of interfaces and knowledge differences between interdependent functions.

Designer agents are assigned to specific functions and possess two types of knowledge: general knowledge, representing disciplinary expertise, and product-specific knowledge, reflecting their experience with similar products. The time required to execute tasks depends on the designer's competence and the complexity of the task. Upon task completion, several quality metrics are calculated, including:

- Technical Quality: The likelihood of producing error-free work.
- Interface Quality: The probability of successful integration with other components.
- Solution Goodness: The extent of progress toward achieving the desired outcome.

If errors or incompatibilities are detected, tasks may require rework. The effort needed for rework is mitigated by a learning curve effect, which reduces rework time as designers gain experience.

Key interactions in the model include communication, where agents exchange task outcomes through information agents, and consultation, where agents work together to address technical or integration issues. These consultations enhance the agents' knowledge levels, with their effectiveness determined by consultation efficiency and influenced by task complexity. Real-world constraints are reflected in the inclusion of communication delays and probabilistic consultation acceptance rates, capturing limitations in collaboration and information flow.

The simulation is stochastic, incorporating probabilistic distributions for task durations, communication delays, and rework probabilities. It proceeds iteratively, beginning with child functions and advancing to parent-level integrations only after input quality thresholds are satisfied. The outputs, including total project effort, span time, and quality metrics, emerge from the micro-level dynamics of agent behaviors and their interactions.

In this research, the baseline model described above has been adapted to provide a more holistic and realistic framework for evaluating organizational configurations and their individual elements. The modifications are detailed in the following sections.

2. Process Adaptations

The baseline model [31] assigns only a single activity to each architectural element: a design activity for components and an integration activity for subsystems. This simplification does not accurately reflect the realities of product development, where a variety of activities may exist for a single architectural element. At a high level, additional activities such as system design, simulation, and testing are examples that are currently omitted. As a result of this limitation, the baseline model's simulation results indicate that most work occurs early in the product development process, with only a few agents remaining active in later stages [31]. Expanding the model to include a broader range of activities allows the simulation to account for overlapping tasks, providing a more realistic representation of product development workflows.

To incorporate system design activities, a quality metric is introduced to evaluate how effectively requirements for lower-level subsystems or components have been formulated. Unlike the baseline model, which assumes a review process without accounting for time, the testing and simulation activities now require time to assess the quality of elements. These enhancements result in a more accurate representation of the engineering process [46]. Additionally, this updated framework provides flexibility to adapt the process, enabling the inclusion or exclusion of activities and supporting varied process structures.

3. Organization Adaptations

The baseline model [31] assigns one architectural element to a single individual, an extreme interpretation of the mirroring hypothesis [47] that does not accurately reflect real-world product development. In practice, individuals may focus on specific activities within an architectural element, multiple individuals may work on the same element and activity, or a single person may take on multiple activities. Introducing greater flexibility in modeling these scenarios is essential for capturing the complexities of organizational structures. To address the rigidity in the baseline model's representation of organizational structures and responsibilities, this paper extends the model to accommodate more realistic configurations. These extensions introduce manageable challenges to the simulation logic. For example,

when multiple individuals are assigned to the same architectural element and activity, a prioritization mechanism is needed to determine task assignment. Similarly, when a single individual is responsible for multiple activities, the model must decide which activity takes precedence. Additionally, organizational structure significantly affects decision-making speed and interdepartmental communication [29]. Many of these challenges have been effectively addressed in task-based ABS models [29, 30] and can be adapted for the knowledge-based ABS used in this research.

4. Addition of Tools

The baseline model [31] does not account for tools, particularly digital tools. However, to evaluate the digital transformation of product development (PD), tools and solutions must be explicitly integrated into the simulation model. This includes their benefits and efficiency improvements, as well as their implementation and operational costs, such as Total Cost of Ownership (TCO). The proposed simulation model addresses digital tools and solutions across several dimensions, as noted below.

- *Tools for Activity Execution:* The model incorporates tools explicitly used for specific activities, impacting the required competencies and the speed of task completion. Tools may require specialized training for effective use and vary in capabilities, influencing their suitability for different tasks and the overall resulting productivity. This approach enables the simulation of trade-offs, such as adopting newer tools that may initially reduce productivity due to training requirements versus continuing to use older tools [48]. These effects can be modeled using learning curves [49], as has been done for other aspects of PD process simulations. Additionally, "quantification tools" are included to simulate the varying certainties and accuracies of tools used for simulation and testing [50]. This contributes to capturing the impact of reducing reliance on physical tests through digital tools [51], where the use of advanced simulation tools enables faster and earlier iterations in the development process.
- *Knowledge Management and Collaboration:* Digital tools play a critical role in knowledge management by facilitating the storage, access, and sharing of data, information, and expertise both within organizations and across tools. Platforms such as PLM systems and collaborative tools enable real-time collaboration and the reuse of knowledge from past projects, improving productivity and decision-making [48, 51]. The effectiveness of these tools depends on their interoperability. High interoperability allows for digital continuity [32] and ensures fast data transfer. This, in turn, fosters efficient concurrent engineering and guarantees that all stakeholders work with consistent data [51].
- *Tools as Agents in the Simulation Model:* In the extended simulation model, digital tools are represented as agents that interact with personnel and other tools. Their capabilities and connectivity determine how they influence task execution, information flow, knowledge storage, and overall productivity. By incorporating these dimensions, the model reflects the trade-offs and benefits of digital tools, including decisions between customization and off-the-shelf solutions. This approach captures their role in enhancing collaboration, knowledge management, and efficiency in PD while accounting for their potential as cost drivers.

C. Integration of Modeling and Simulation

Integration between the modeling and simulation environment is necessary to ensure consistency through an ASOT as well as to manage and configure data. For this a method is proposed that allows to connect the SysML-based modeling tool with a simulation tool, using a graph database as a central integration point. This approach builds on the work of *Schummer and Hyba* [52], who highlighted the benefits of using a graph database for in-depth analyses of complex systems. NoSQL databases demonstrate significant advantages over traditional relational databases. In particular, graph databases, such as Neo4J, have proven to be highly efficient solutions for managing large complex and evolving datasets. In a recent study by *Monteiro* [53], which conducted an experimental evaluation of various graph databases, Neo4J has shown superior performance in both scalability and responsiveness. As a result, Neo4J offers the capacity to manage vast amounts of data with high scalability, making it an ideal choice to meet the substantial data demands of representing complex systems like an enterprise. For those reasons, Neo4J is used to store knowledge from the SysML model.

Using a Neo4J database preserves the complex semantic links between SysML model elements, thus enabling flexible analysis of relationships within the system and easy access for various other applications, including the simulation model used for this study. This architecture makes simultaneous data updates possible, thereby creating a true ASOT.

The advantage of this ASOT is significant, especially for large-scale projects where synchronization errors can be costly in terms of both time and resources. By centralizing the information in a graph database, all stakeholders have access to a unique, consistent representation of the model, thus avoiding ambiguities that may arise when multiple versions coexist. Moreover, this approach enables digital continuity between the model and the simulation. This

ensures that the same dataset and relationships are consistently used throughout different stages of analysis, avoiding misalignment between what is modeled and what is simulated. This enables shared responsibility of the EA and simulation among multiple stakeholders, rather than relying on a few key individuals, thereby improving collaboration and reducing the risk of knowledge silos.

V. Implementation

The proposed methodology is implemented and evaluated using the development of a simple drone as a proof of concept. The details of this implementation are discussed below.

A. Enterprise Architecture of a Product Development Project

The primary viewpoints of the architecture, as previously discussed, are the product, process, organization, tools, and goals. The following sections outline the structure of these viewpoints. Based on these definitions, instances are created for use in simulation. While the extensive number of input values used for these instances will not be detailed, insights into the data and its structure will be provided.

MagicDraw's System of Systems Architect was chosen for modeling the Enterprise Architecture (EA) due to its availability, widespread industry adoption, and compatibility with the UAF plugin.

1. Product Definition

The product architecture features a straightforward structure, illustrated in Fig. 4. The product is hierarchically decomposed into subsystems (`Architecture System Element`) and components (`Architecture Component Element`), forming the foundation for system design and integration activities during product development.

Components are connected through `Interface`s, defined by their type and severity. For this research, the following interface types were chosen: Geometric/Spatial, Structural, Energy, Material, and Information/Data [54]. Interfaces indicate where communication and information exchange must occur during product development [47].

Interface severity quantifies the extent of information exchange required, the complexity of integration, and the interface's criticality to the product's overall performance. Currently, the simulation aggregates these values to calculate interface complexity using the method outlined by *Zhang and Thomson* [31]. Future implementations could differentiate between interface types, assigning distinct expertise and capabilities during simulation.

Each architectural element includes several value properties that define its characteristics:

- `Knowledge Requirement`: Represented as a `Knowledge Vector`, this property specifies the required knowledge domains and levels for an element. This approach, adapted from *Zhang and Thomson* [31], defines the knowledge complexity of elements and interfaces. Unlike the baseline model, which focuses on abstract product functions, this research uses physical product definitions, enabling decisions about tools needed for simulation and testing.
- `Novelty`: This property measures how much of the element is new compared to previous versions. A value of 1 indicates entirely new development, while 0 signifies no change from a previous version. This metric determines the extent of new knowledge required versus existing knowledge that can be reused.
- `Importance`: This property indicates how critical an element is to overall system performance. It adds weight to architecture elements in calculating system quality and guides decision-making during simulation, prioritizing elements for rework if overall quality falls below acceptable thresholds. Importance values can be quantified through QFD (Quality Function Deployment) analysis, which links customer requirements to technical specifications and, ultimately, to development activities [55]. Future extensions could directly link these technical characteristics to elements within the EA model.

By incorporating these properties and interactions, the proposed architecture captures both the technical and organizational complexities of product development, supporting more realistic simulations and better-informed decision-making.

2. Development Process

The EA model defines a generic, high-level development process applied uniformly across the entire product or system, as shown in Fig. 5. This approach minimizes the modeling effort by avoiding the need to create numerous, similar activities for each architectural element.

Detailed activities are generated from this high-level framework based on the product architecture, replicating the decomposition, design, and integration steps of product development. Given the prevalent use of digital tools in modern

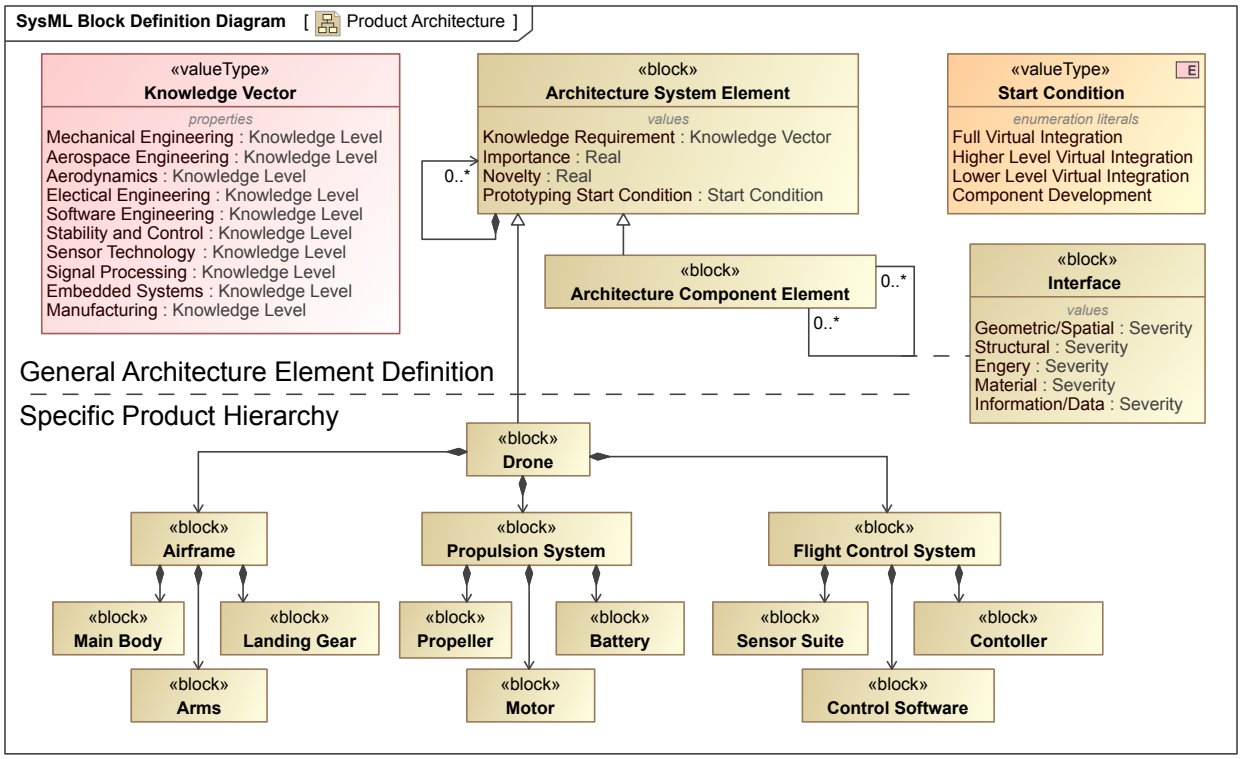

Fig. 4 Product Architecture Definition

PD, additional simulation and virtual integration activities are incorporated. These activities follow the structures outlined in adapted systems engineering reference processes, such as the W-model proposed by [56] and [57].

The model also employs process rules associated with activity and architecture element nodes to define overlapping and process flow patterns. These rules specify conditions that must be met before an activity can begin, such as overlap percentages or quality thresholds for preceding tasks. For prototyping, a specific rule (`Start Condition` in the top-right corner of Fig. 4) determines when prototyping can start for each architectural element. The earliest possible start occurs immediately after the element's design phase, while the latest start follows the complete virtual integration of the product.

However, the current version of the simulation model does not yet support sequential overlapping, so these rules are not yet applied in practice. Future enhancements could introduce additional rules to capture interdependencies between architectural elements, such as requiring the completion of one component's development before beginning another.

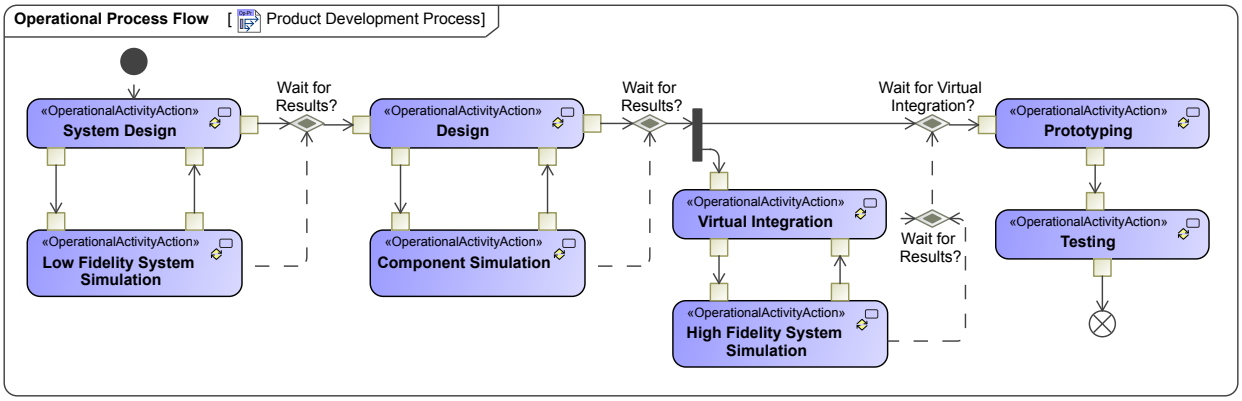

Fig. 5 High Level Process used for generating the detailed Product Development Process

Additionally, each activity in the model is associated with process variables for effort and learning rates. Effort is defined using a triangular distribution (minimum, median, maximum), which is scaled by the complexity of the corresponding architectural element. The distribution is then sampled to determine the actual effort required for each activity during a single simulation run. The learning rate, which quantifies the reduction in effort for repeated activities, is modeled using the power model [49].

In principle, activities can be generated for every architectural element. However, certain activities are exclusive to specific element types. For instance, `Design` and `Component Simulation` are unique to `Architecture Component Elements`, while `System Design`, `Low Fidelity System Simulation`, `Virtual Integration`, and `High Fidelity System Simulation` are exclusive to `Architecture System Elements`.

Furthermore, the capabilities of tools and the responsibilities of personnel, which are mapped to activities and architectural elements, influence activity generation. If no tool or employee is capable of performing a specific activity for an element, that activity will not be generated.

The generation of detailed activities is performed prior to the simulation but is not part of the architecture model itself. Figure 6 illustrates the detailed product development process generated for the defined baseline scenario. This process incorporates all the procedures and rules described, using the organizational structure and tools defined in Fig. 8 and Table 1. Overall, this approach for modeling the process significantly reduces the effort needed to set up a detailed product development process model while maintaining flexibility in the process flow.

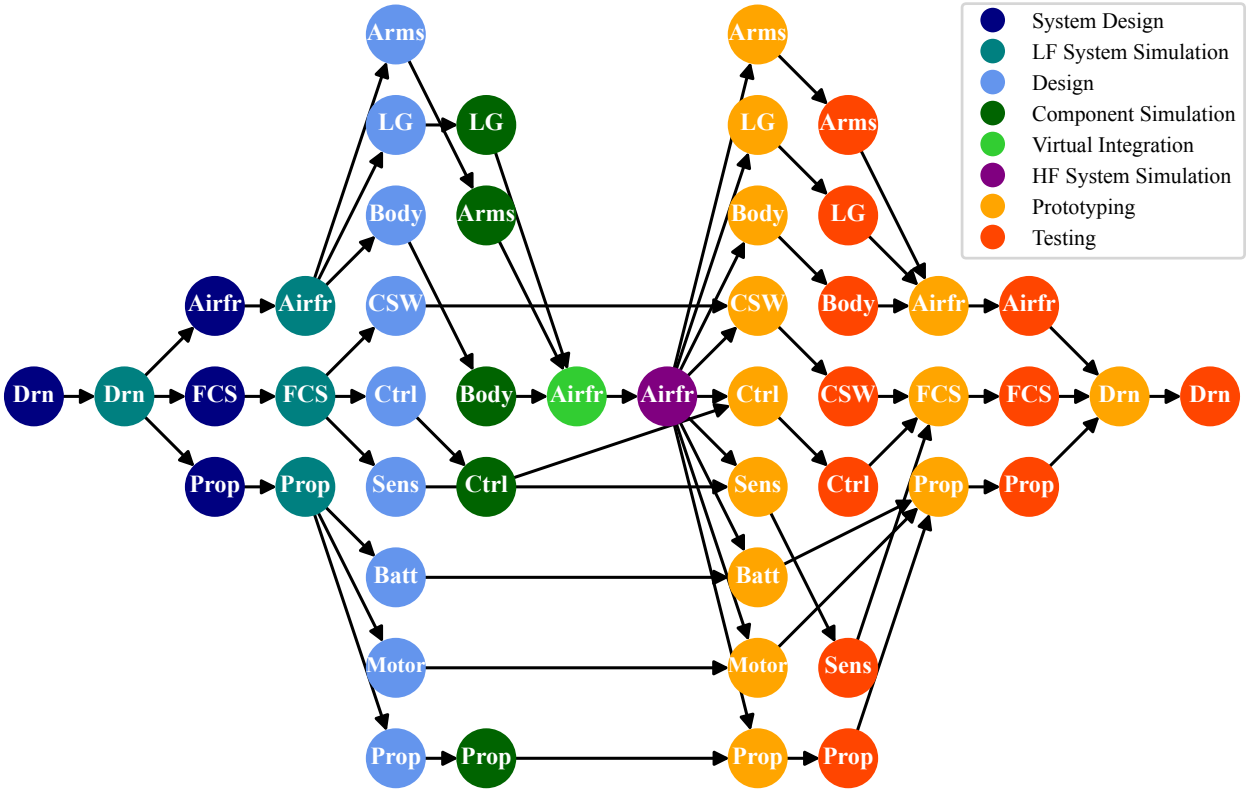

Fig. 6 Generated detailed PD Process for the Baseline Scenario

3. Organization

The definition of the organization (Fig. 7) focuses on personnel and their attributes, structured into `Teams` that establish decision-making pathways. `Decision Making` responsibilities are assigned by associating `Managers` with specific architectural elements. This association ensures that managers receive and handle decision requests related to the architectural elements they oversee. Similarly, the `Responsibility` for architectural elements and activities is assigned to `Engineers`, mapping them to the detailed activities involved as part of the product development (PD) process.

Several additional properties exist that describe the `Expertise` and competency of engineers. The `Expertise` includes a `Knowledge Vector` [31], representing the engineer's `General Knowledge`. This vector is structured identically to the one used by architectural elements, enabling the simulation to assess an engineer's suitability for developing a given element by evaluating the mismatch between the engineer's knowledge and the knowledge required by the element. Other components of expertise include `Experience with the Product` and `Digital Literacy`, as discussed below.

`Experience with the Product` captures an engineer's familiarity with product elements and their dependencies based on prior experience with similar or earlier product developments. During the simulation, this value is adjusted relative to the novelty of the architectural elements, reflecting how much prior knowledge is applicable. For highly novel elements, less prior experience applies compared to marginally new elements.

`Digital Literacy` [58] measures an engineer's proficiency with digital tools, models, and data. This metric influences various aspects of simulation, such as information handling and data processing. Engineers with higher digital literacy process and transform data more efficiently. A related property, `Tool Competency`, indicates an engineer's familiarity with specific tools. Similarly, `Familiarity` with Engineering Management and Support Tools describes how effectively an engineer navigates these tools to find necessary information. This competency directly impacts task efficiency and is particularly important for evaluating tool-driven workflows, which are detailed in the next section.

`Availability` specifies the percentage of time a person can dedicate to the project. In the current scope, which analyzes a single project, availability is defined as a fixed property of the individual. In scenarios involving multiple projects, availability would need to be associated with specific person-project relationships. While fixed for the project duration in this study, real-world availability often fluctuates. For more dynamic scenarios, the UAF Personnel-Roadmap viewpoint can model personnel availability, evolution, and forecasting [33]. In the simulation, availability introduces noise, occasionally blocking agents from working on tasks [30].

For the baseline configuration, a simple organization, consisting of a project team and four sub-teams, was modeled. This organizational structure was instantiated using the definition blocks in UAF's actual resources domain. Figure 8 summarizes the organizational setup.

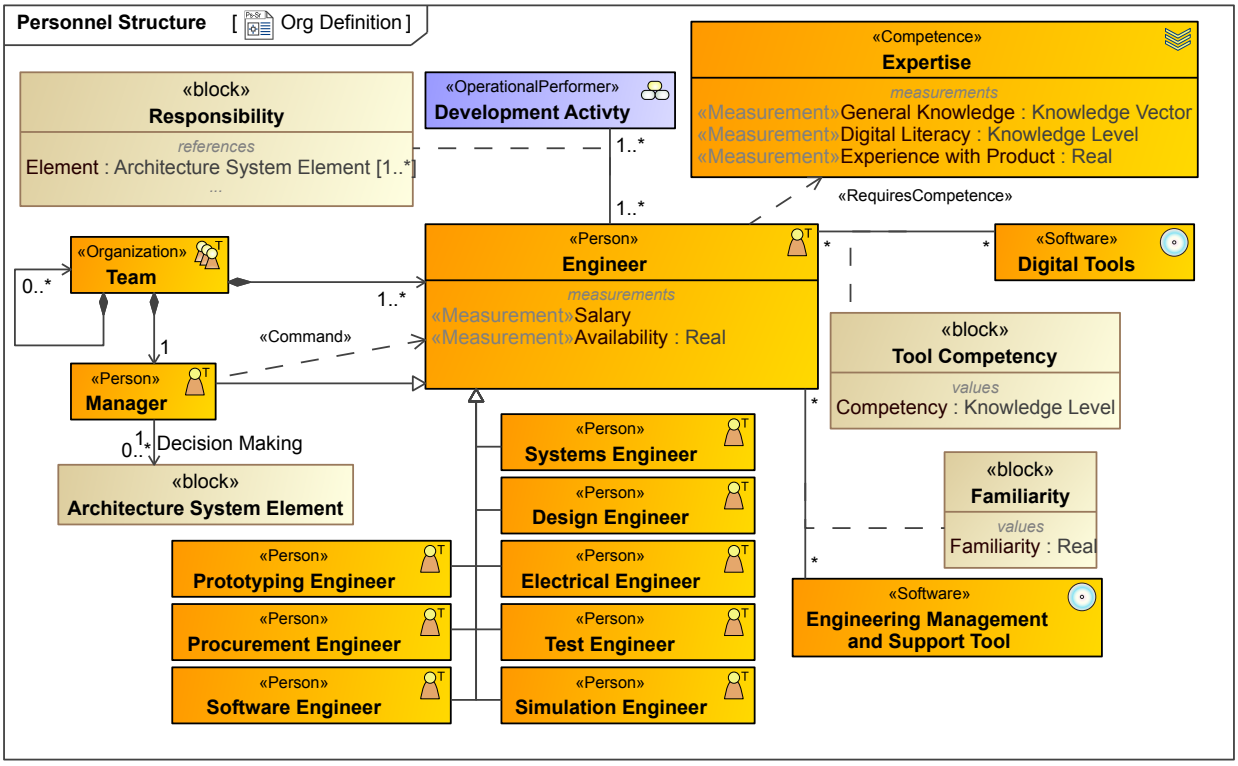

Fig. 7 Organization and Personnel Definition

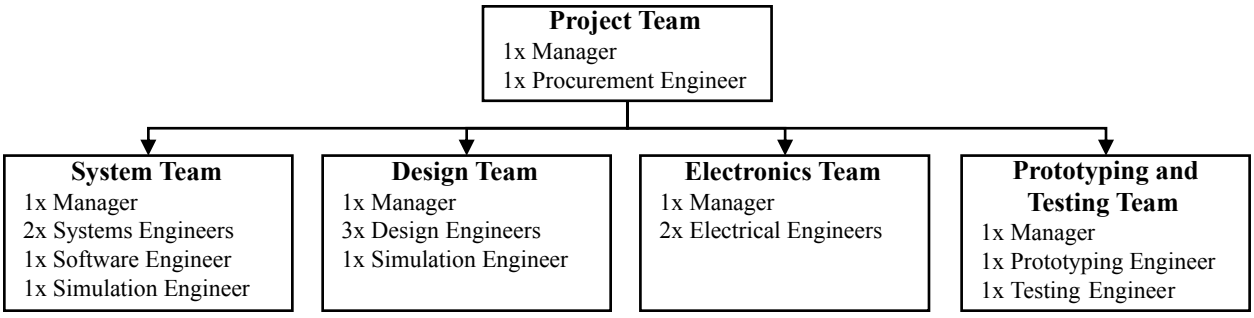

Fig. 8 Baseline Organizational Structure

4. Tools

Tools play a critical role in product development by assisting or enabling specific activities. To categorize their various uses during the process, tools are divided into `Physical Tools` and `Digital Tools`. Similar to personnel responsibilities, tools are mapped to activities and architectural elements through their defined `Capability` (Figure 9).

`Digital Tools`: Digital tools are characterized by several value properties that describe their usability and performance. These include:

- `Tool Complexity`, which reflects how easy the tool is to use.
- `Productivity`, which indicates how quickly activities can be completed with the tool.
- `Cost per Month`, which encompasses licensing and infrastructure costs.

The `Interoperability` property measures how effectively data from one tool can be used by another. This relates to standards, APIs, and automated data exchange within the digital thread framework [32]. A value of 0 indicates no data compatibility, while a value of 1 represents seamless, real-time data sharing. Most tools fall between these extremes, requiring some manual effort to transfer and utilize data between systems.

To address specific uses, digital tools are further categorized into `Simulation Tools`, `Development Tools`, and `Engineering Management and Support Tools`:

- `Development Tools` and `Simulation Tools` are used for specific tasks, such as CAD for designing components. Simulation and Testing Tools also include an `Accuracy` property, which quantifies their ability to predict product quality. Lower accuracy may lead to overestimated quality, resulting in rework during later development stages.
- `Engineering Management and Support Tools` are used to store, exchange, and connect data, information, and knowledge between tools. These tools are mapped to architectural elements to define their `Completeness`, representing how much data from previous versions or similar products is available. This property, similar to an engineer's experience, determines how effectively information can be reused for developing a new product, particularly for novel components.

`Physical Tools`: Physical tools have additional value properties, including:

- `Cost per Hour`, which accounts for the total usage cost, including technician labor.
- `Availability`, which specifies how much time the tool can be dedicated to the project.

In the baseline configuration, a limited number of tools were defined. Their capabilities are summarized in Table 1. Currently, only one tool can be defined for any activity-element combination. Future enhancements could allow for more flexibility by associating tools with specific technical characteristics of architectural elements (e.g., FEM for mechanical analysis, CFD for aerodynamics).

In addition, because procurement activities are not explicitly included in the current simulation, a "Supplier Tool" serves as a proxy for generating these activities. Finally, the baseline EA includes a PLM tool for managing knowledge and product data. This tool also acts as a central hub, linking other tools based on their interoperability.

5. Evaluation Results and Requirements Validation

To evaluate the enterprise, requirements and goals are defined in the strategic views of UAF, as shown in Fig. 10. In this case, high-level requirements are established for `Development Cost`, `Lead Time`, and `Overall Technical Quality`. Additionally, a lower-level requirement is specified for `Human Resource Utilization`.

Since cost, lead time, and quality are often interdependent and involve trade-offs [35, 36], the simulation uses the defined technical quality goal as input and converges towards this target. The results of the simulation then predict the

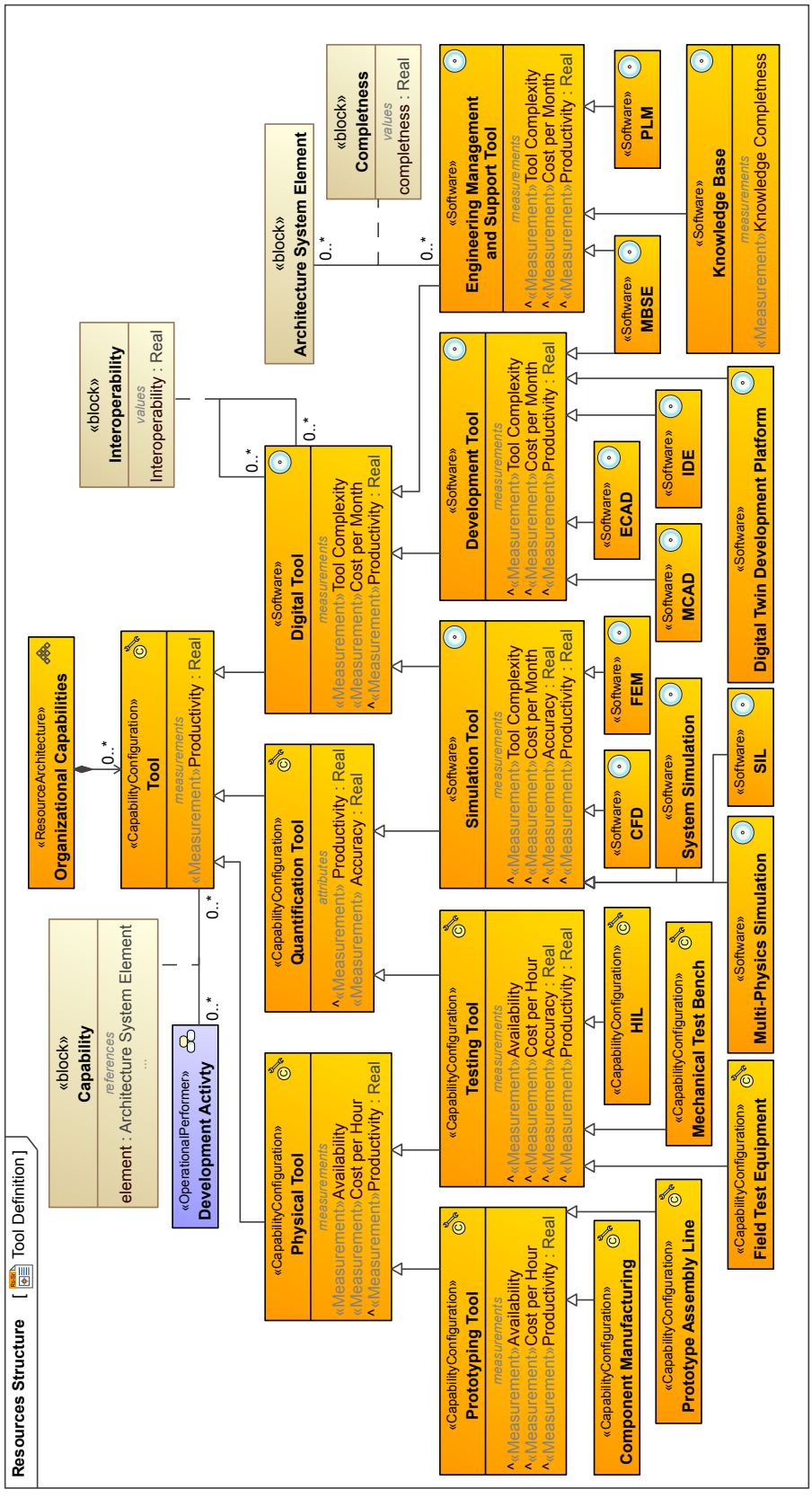

Fig. 9 Tool Definition

Table 1 Capabilities of the Defined Tools

Tool	**Capabilities** (*Activities*: Architecture Elements)
MBSE	*System Design*: Drone, Propulsion System, Flight Control System, Airframe
System Simulation	*Low Fidelity System Simulation*: Drone, Propulsion System, Flight Control System
CAD	*Design*: Arms, Main Body, Landing Gear, Propeller *Virtual Integration*: Airframe
ECAD	*Design*: Motor, Battery, Sensor Suite, Controller
IDE	*Design, Prototyping, Testing*: Control Software
FEM	*Low/High Fidelity System Simulation*: Airframe *Component Simulation*: Arms, Main Body, Landing Gear, Propeller
Circuit Simulation	*Component Simulation*: Controller
Prototype Manufacturing Equipment	*Prototyping*: Arms, Main Body, Landing Gear, Propeller, Propulsion System, Airframe, Flight Control System, Drone
Supplier	*Prototyping*: Motor, Battery, Sensor Suite, Controller
Mechanical Test Bench	*Testing*: Arms, Main Body, Landing Gear, Propeller
HIL Equipment	*Testing*: Sensor Suite, Controller, Flight Control System
System Testing Facility	*Testing*: Drone, Propulsion System

lead time and development cost required to achieve the specified quality goal for a given enterprise configuration.

In the future, a more comprehensive understanding of the enterprise's performance could be obtained by introducing more detailed technical requirements and linking them to the product's technical characteristics. Doing so would enable a deeper evaluation of multiple requirements and their trade-offs.

Since the simulation is non-deterministic due to the uncertain nature of PD, a metric for the `Overall Development Risk` is introduced that combines the probability of cost or lead time overrun with its impact [21]. To measure the `Impact` of overruns, a risk factor and type of function (constant, linear, or quadratic) is defined. In this case, economic utility curves are used to quantify this risk in terms of lost revenue by scaling the risk functions accordingly. As an example of these functions, Eq. 1 shows the calculation of the impact of lead time overrun that is defined by a quadratic utility curve, and Eq. 2 shows how the risk is calculated based on multiple simulation runs, where κ_{LT} is the risk factor, LT is the lead time, T_{LT} is the targeted lead time and $f(LT)$ is the PDF of the lead time results [21]:

$$I_{LT} = \kappa_{LT}(LT - T_{LT})^2 \,, \qquad LT > T_{LT} \tag{1}$$

$$R_{LT} = \kappa_{LT} \int_{T_{LT}}^{\infty} f(LT) \cdot (LT - T_{LT})^2 \, dLT \tag{2}$$

Eventually, the goal would also be to define the impact of technical quality on development risk. This would enable dynamic decision-making during a simulation where agents can make trade-offs between reworking to increase quality and not reworking to reduce lead time and cost during the simulation. This has previously been done on a DES of PD, as discussed in [23].

To define different configurations to be evaluated by the simulation model, instances of the previously mentioned elements have to be created. These instances can have different structures, relationships, and values depending on the kind of scenarios to be investigated. To ensure that the correct instances are associated with each other, a `Development Project` is defined (Fig. 11). A project is composed of instances of the `Development Process`, `Team`, `Organizational Capabilities` (tools), `Drone` (product), and `Simulation Results` (goals). The `Simulation Results` instance links data from the simulation output to the EA model's requirements, enabling requirements verification. This integration fully realizes the implementation of the ZOPH framework [18, 34], proposed in Sec. IV.A as a guide for EA modeling. As a result, detailed representations of product development projects can be effectively created.

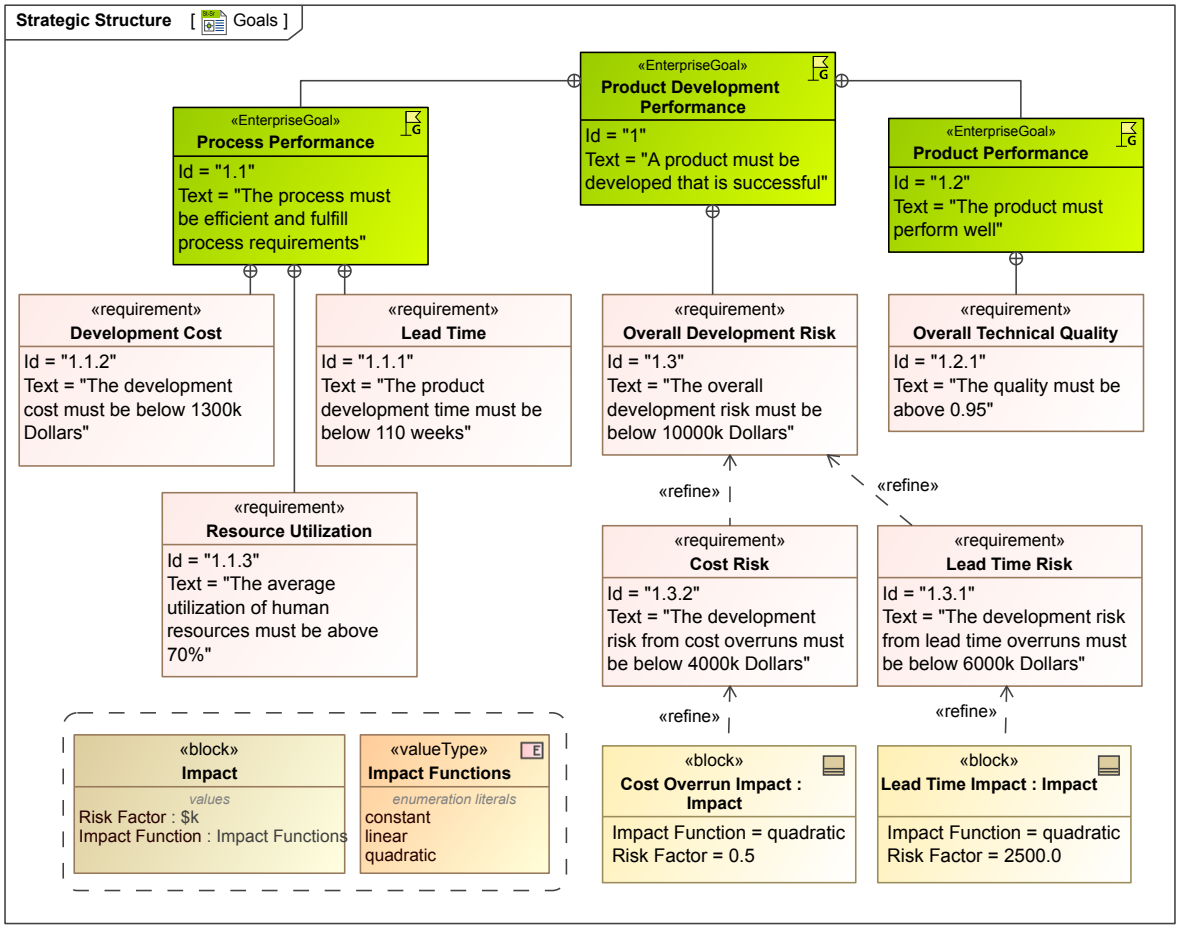

Fig. 10 Goals and Requirements of the Enterprise

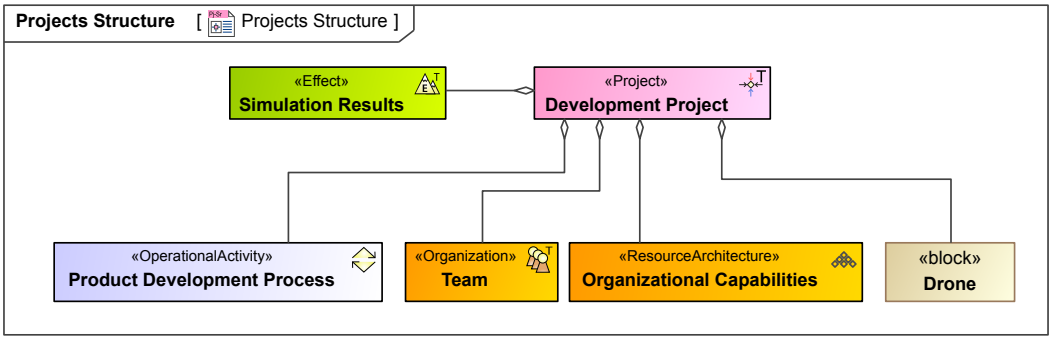

Fig. 11 Configuration of a Project to define Consistent and Traceable Instances for Simulation

B. Graph Database Export

The graph database export was partially implemented. For this, the XML file of the created model is uploaded and then a script parses specific elements, with the current focus being on Block Definition Diagrams, Internal Block Diagrams, and Activity Diagrams. This parsing allows us to describe both the system's taxonomy and its behavior, which is subsequently transferred to the simulation tool.

Figure 12 illustrates the graph based on the baseline described in the previous paragraphs. In particular, the dark green node labeled "Model" serves as the root of the graph, from which three main branches originate: Product Definition, Product Instance, and Product Development Process. The Product Definition branch represents various elements of the drone, as shown in Fig. 4 represented as yellow nodes in the graph. The red nodes represent the instances of the blocks, illustrating how each component instance interacts within the overall system. Finally, the Product Development Process diagram represents the high-level process depicted in Fig. 5, highlighting control flows in red and object flows in yellow. The initial nodes are represented in pink, final nodes in blue, activities in brown, and gateways in gray. Each of these visual elements plays a specific role in defining the development process and stores the necessary data for properties and relationships to other elements.

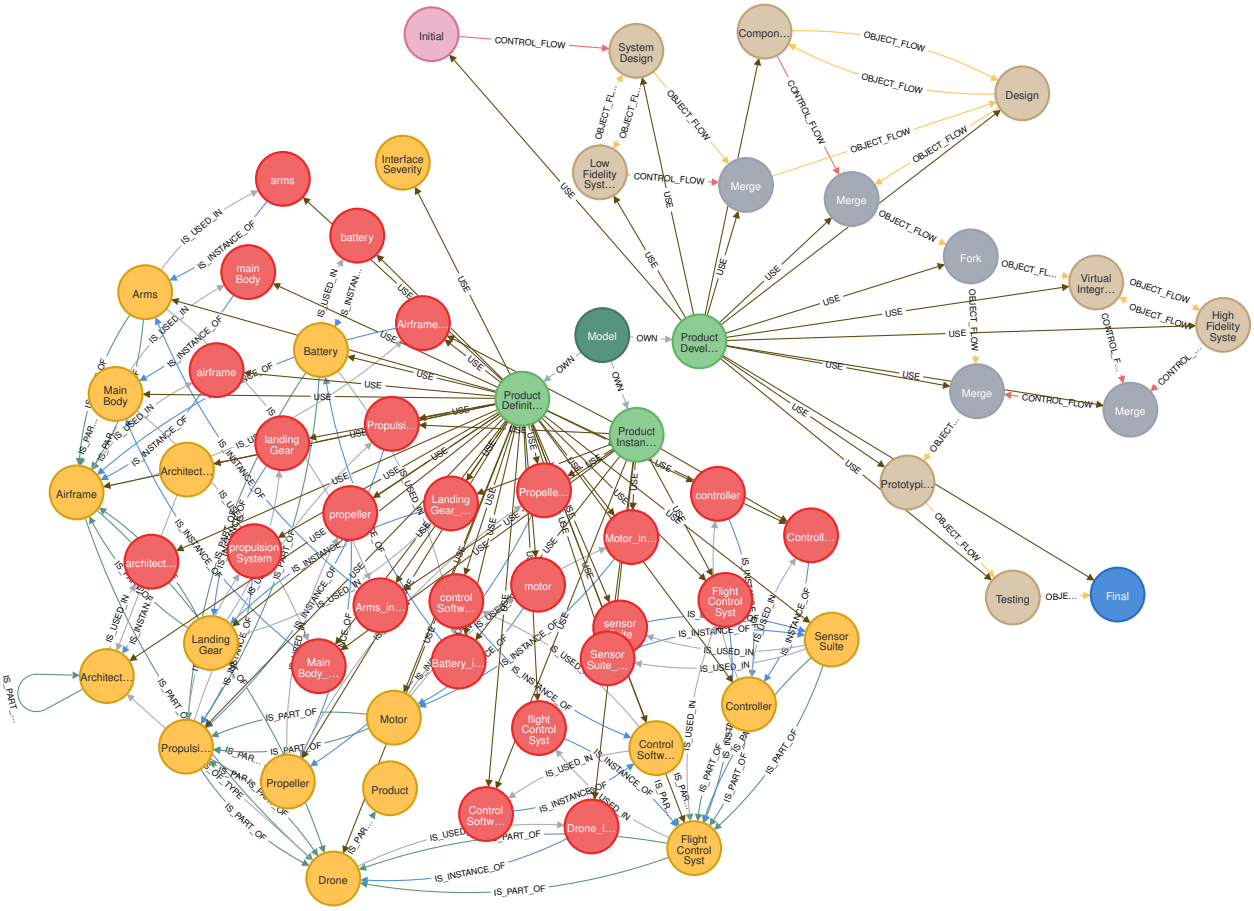

Fig. 12 Neo4J Knowledge Graph for the exported Product and Process View of the EA model

It should be noted that, in its current state, the export and integration functionality only exports selected parts of the model and has not yet been applied to the full EA. Future developments will aim to provide comprehensive coverage of all the necessary diagrams and elements. To connect all the necessary data to the simulation model, instance tables and generic tables were created and exported as CSV files for the missing data. The simulation model then reads these files, ensuring that the data used for simulation is consistent with the user-defined data in the EA model. Data exported from MagicDraw, either through the XML or CSV export, is then transformed into a readable format (JSON) for the simulation through an additional Python script.

C. Simulation Model

The ABS model was implemented in Python using standard libraries. While the current version of the model is substantial in size, it remains under development and does not yet incorporate all aspects of the proposed simulation methodology and logic. Further development of the simulation model is ongoing.

As this paper focuses on the end-to-end modeling and simulation (M&S) methodology, the specific implementation details of the simulation model are not discussed here. A subsequent publication, based on the finalized simulation model, will provide a comprehensive description and analysis of the simulation approach and its implementation.

For context, the most critical aspects of the simulation model relevant to a general understanding have already been outlined in Sections IV.B and V.A.

VI. Results & Discussion

The implemented methodology was evaluated through initial simulations of the defined baseline, focusing on analyzing the results generated for a single configuration. The baseline EA model was then modified to explore alternative configurations and potential changes enterprises could adopt to improve performance, with an emphasis on digital transformation. This evaluation highlights the methodology's potential capabilities.

A. Simulation of the PD Enterprise Architecture

Once configurations are created in the EA model and exported to the simulation model, the simulation process is initiated. This involves transforming the input data to construct all agents and their relationships, as well as generating a detailed process based on the high-level process and product architecture. The simulation is then executed, producing a large amount of data. Each agent—whether representing a person, tool, task, or product architecture element—can be tracked in terms of state and interactions. Figures 13 and 14 illustrate a subset of the simulation results from a random single run of the baseline configuration. Figure 13 shows the Gantt chart of that simulation. In the baseline not all possible simulation tools are included leading to errors being discovered later in the process. This creates two large reworks loops that lead to substantial delay.

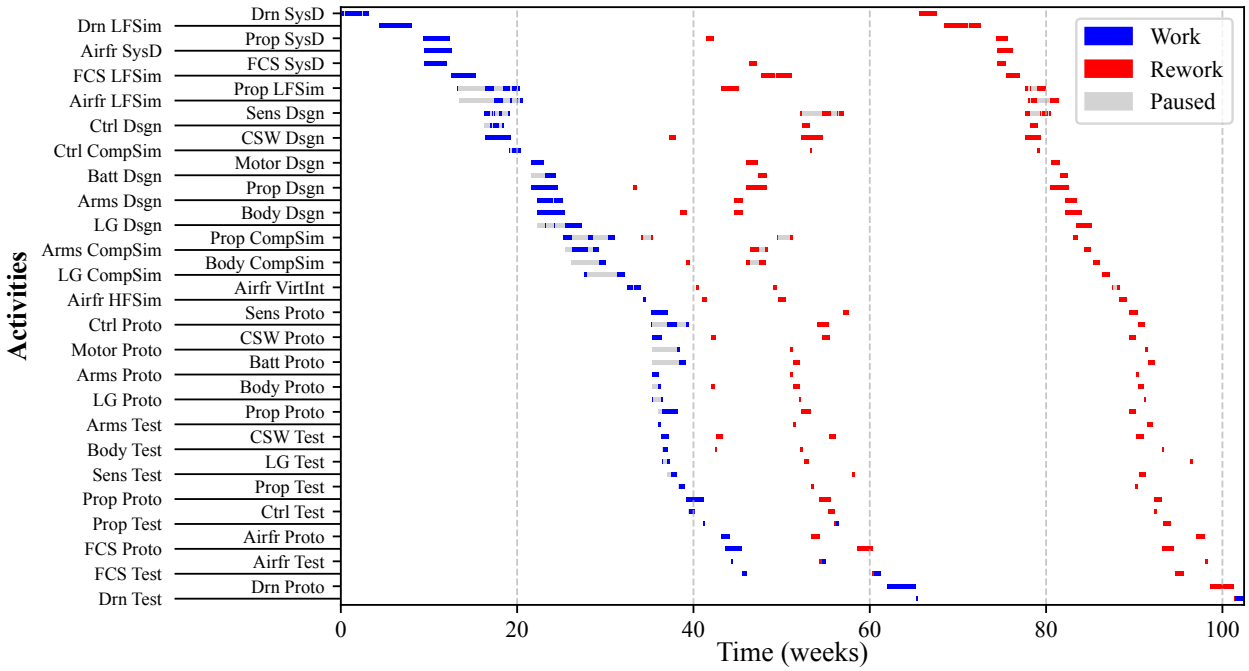

Fig. 13 Gantt Chart Generated by a Single Simulation Run

Figure 14 shows data for the organization and the system team that were defined in 8. Initially, their effort backlog is high, which aligns with their primary focus on system design activities. Two smaller increases in backlog occur later, caused by rework resulting from testing activities that uncovered issues in the system design. These rework activities,

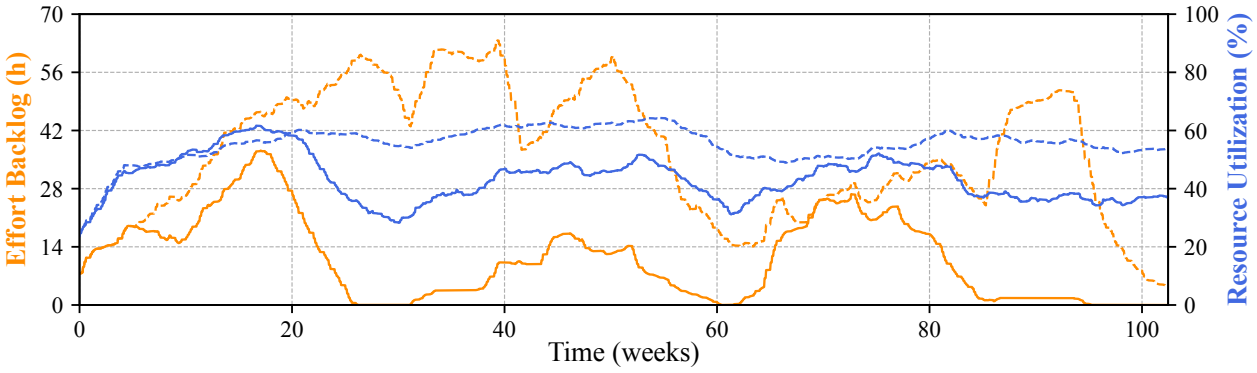

Fig. 14 Effort Backlog and Resource Utilization over Time for a Single Simulation Run (Solid: System Team; Dashed: Organization)

however, are executed more quickly due to the effects of learning curves. For the overall organization, spikes can be seen in the backlog curve on a few occasions. These are in line with bottlenecks that are created due to the sequential process architecture. This causes all testing activities to start sometime after the virtual integration. It can, therefore, be seen that modifications, such as overlapping or earlier starts of some activities, lead to improvements in the PD process. Resource utilization of the organization never drops to zero, reflecting noise introduced to simulate involvement in other projects.

High-level outputs, such as total cost and lead time, are also exported as single values extracted from agent states. For this simulation run, the total development cost was $2.967 million, the lead time was 103 weeks, and the average resource utilization was 55.4%. Effort, cost, and product quality could be further broken down to provide more detailed insights. Additional process characteristics, such as activity repetitions or information flow, could also be analyzed, though this has not yet been explored.

A Monte Carlo simulation is executed due to the stochastic nature of the problem. Consequently, simulation results are described using distributions. These probabilistic factors include effort distributions, decisions during product development (e.g., quality checks, compatibility checks), and the occurrence of unexpected problems, among others, as detailed in earlier sections. Currently, the Monte Carlo simulation primarily tracks high-level results and metrics that can be easily extracted as single values at the end of a single simulation run.

To determine the number of runs to be executed, convergence is evaluated by monitoring the mean and variance of key metrics such as cost and lead time. Figure 15 illustrates the relative changes in these values over 1,500 simulation runs. The mean converges relatively quickly, while the variance stabilizes more slowly, with minor, non-concerning oscillations. During the Monte Carlo simulation, the stability of the distribution is checked every 100 runs. If the change in the mean or variance compared to the previous check is below 2%, the distribution is considered stable, and the simulation is deemed to have converged. All results presented in this paper meet this convergence criterion.

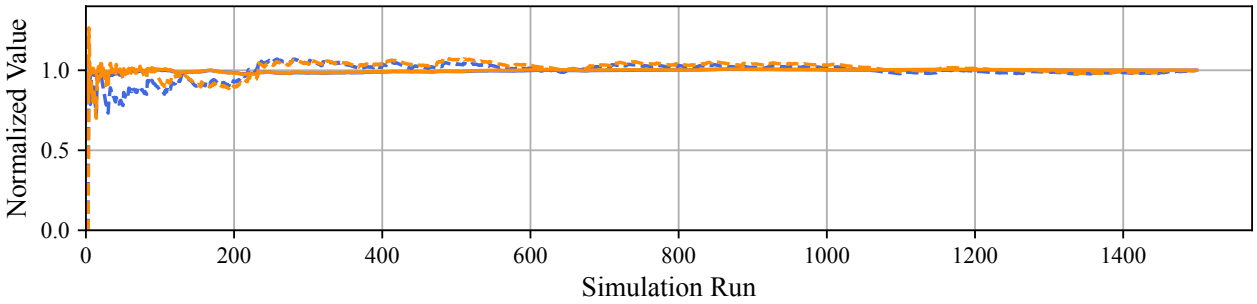

Fig. 15 Relative Convergence of Cost (blue) and Lead Time (orange) during the Baseline Monte Carlo Simulation (Solid: Mean; Dashed: Variance)

After completing the Monte Carlo simulation, distributions for the extracted results are obtained. Figure 16 shows the cumulative distribution function (CDF) and probability density function (PDF) for cost and lead time in the baseline simulation. Notably, there is a significant spread in the predictions, likely due to the incomplete implementation

of information exchange events in the simulation, which results in additional iterations and rework to meet quality requirements.

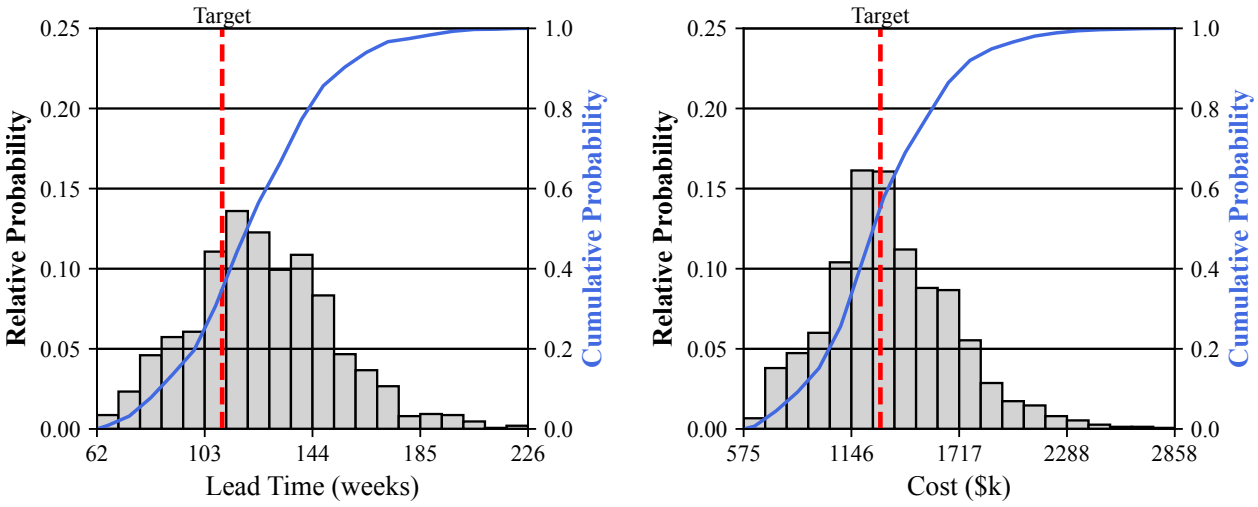

Fig. 16 PDF (gray) and CDF (blue) of Lead Time and Cost for the Baseline Monte Carlo Simulation

The average development cost is $1.362 million, the average lead time is 126 weeks, and the average overall resource utilization is 55.7%. Using the risk curves defined in Fig. 10, the overall risk of lost revenue is estimated at $209.72 million, divided into $10.616 million for cost overrun risk and $199.104 million for lead time overrun risk. These results indicate that the baseline configuration does not meet the previously established requirements (Fig. 10).

B. Comparison and Evaluation of different Configurations

To evaluate the current state of the developed methodology, it is applied to compare and assess different organizational scenarios. Specifically, the baseline model is analyzed alongside four new configurations, each incorporating various incremental changes. The changes primarily focus on various digital transformation initiatives that enterprises might implement to enhance performance.

The first scenario introduces a new digital tool capable of performing comprehensive virtual integration and accurate simulation of the drone. The second scenario enhances the digital ecosystem, facilitating easier, faster, and more frequent data transfer between tools and databases. The third scenario focuses on personnel training to improve their competencies. Table 2 summarizes the modifications made to the baseline model for each of these configurations.

Table 2 Overview of the Changes Made for the Simulated Configurations

Config.	Change	Description
1	*New Tool*	A new system simulation tool is introduced, enabling virtual integration and high-fidelity simulation of the propulsion system,flight control system, and the overall drone.
2	*Digital Ecosystem*	The interoperability and productivity of all digital engineering, and management and support tools is increased by 50%.
3	*Training*	The digital literacy and tool competency of all personnel is increased by 20% and their general knowledge by 10%.

Scenarios or configurations that involve changes to process structures, such as overlapping activities or modifications to the organizational structure, were excluded from the current analysis. These types of changes are not yet fully implemented in the simulation logic, and their inclusion would result in no impact on the outcomes, execution errors, or inaccurate results. The final version of the simulation model will be capable of handling such changes. Additionally, the product itself remains unchanged in this analysis, as the focus of this paper is on the organization developing the product

rather than the product itself.

After creating configuration instances in MagicDraw and exporting the data to the simulation tool, full Monte Carlo simulations were run for each configuration. The outcomes are summarized as box plots for lead time and development cost, shown in Fig. 17. The stability of the distributions was verified for all simulations, with additional runs performed as needed to ensure convergence.

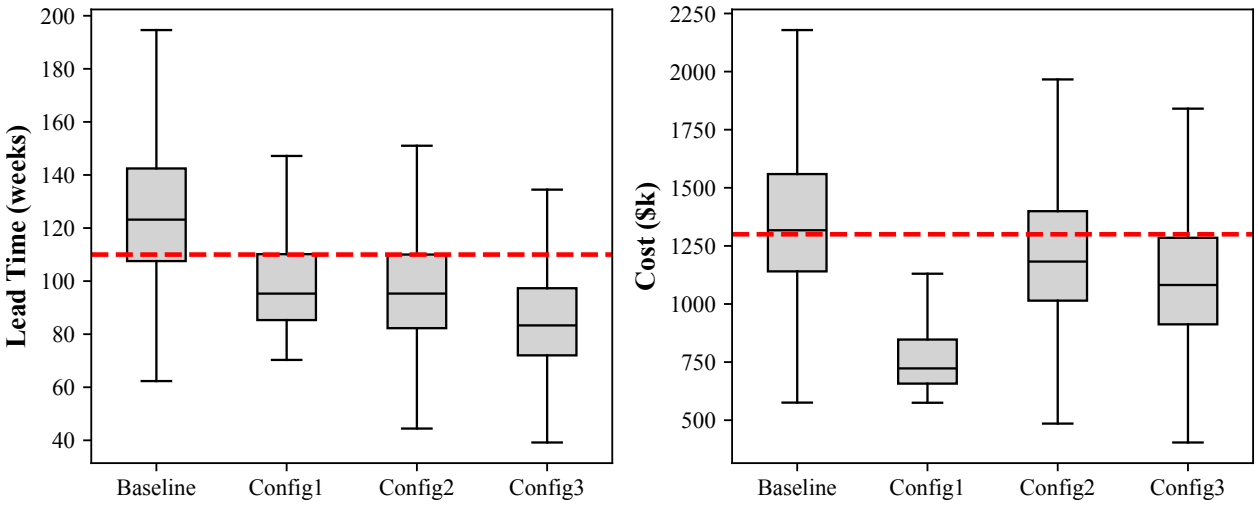

Fig. 17 Box Plots for Cost and Lead Time of the Simulated Configurations

The cost and lead time results, along with other simulation outputs such as average resource utilization and risk values, are exported into a CSV file. This file is then imported into MagicDraw to facilitate requirements verification for the different configurations. Figure 18 illustrates the requirements verification status for all configurations, displayed within an instance table of the EA model, using the simulation-generated results.

#	Name	Mean Lead Time : weeks	Mean Development Cost : $k	Resource Utilization : percent	Overall Development Risk : $k	Development Cost Risk : $k	Schedule Risk : $k
1	Baseline	126.1	1362.3	55.7	209720	10616	199104
2	Config1	100.3	796.4	55.8	22388	28	22360
3	Config2	97.9	1250	56.1	26485	4903	21583
4	Config3	84.7	1110.6	56	6332	1824	4508

Fig. 18 Instance Table with the Requirements Verification Status of the Created EA Configurations

Currently, the Monte Carlo simulation provides only high-level insights, limiting this analysis to a comparison of different scenarios. The results clearly show that each configuration significantly affects the product development (PD) process, demonstrating the impact of different changes. These findings can serve as a foundation for managers to evaluate which changes may be worth pursuing.

Since the simulation does not account for the cost of implementing these changes, the results must be weighed against such costs. For instance, if the introduction of a new tool requires minimal investment because it is readily available and training is unnecessary, but improving interoperability necessitates a comprehensive IT overhaul, managers might prioritize implementing the new tool first. Future simulation versions could incorporate aspects such as training, ramp-up periods, and long-term impacts of these changes.

The resource utilization results in Fig. 18 show minimal impact from the different configurations. This is likely because noise in the simulation has a greater influence on resource utilization than actual project work. Incorporating availability curves, as previously mentioned, could improve the realism of future simulations.

Depending on the specific evaluation needs and interests, additional data can be extracted to provide deeper insights. For instance, when implementing new digital tools, analyzing information flow patterns and durations could help identify tools requiring higher levels of interoperability due to bottlenecks or heavy information flow. Similar analyses could be applied to other aspects of the organization to gain a broader understanding of its dynamics.

VII. Conclusion and Future Work

This paper introduces a novel methodology for enterprise evaluation, leveraging integrated architecture modeling and simulation to analyze product development processes and support digital transformation. By combining organizational elements via the Unified Architecture Framework (UAF) with simulation capabilities, it enables the quantitative assessment of enterprise performance metrics across diverse scenarios. The methodology demonstrates its potential by effectively analyzing trade-offs between development costs and lead times using agent-based simulations. It uniquely captures the complex interplay among digital tools, processes, and personnel, allowing risk-free evaluation of digital transformation strategies prior to implementation. However, the current implementation has limitations that require further refinement. Validation against real-world data is essential, and process mining [59] offers a promising pathway for extracting data from enterprise processes. Its effectiveness is well-documented in product development [60, 61] and PLM systems [61, 62]. Reliable data integration mechanisms are critical for translating the methodology into practical applications.

While the framework supports advanced features such as sensitivity analysis and Design of Experiments (DOE), these capabilities are not yet fully implemented. Current Monte Carlo simulation results, though insightful, lack the granularity needed for robust decision support, underscoring the need for enhanced data extraction and calibration. Future development priorities include refining the simulation model to capture activity overlaps, technical characteristics, and improved information flow. Achieving seamless data transfer between enterprise architecture models and simulation environments is a critical element to maintain consistency across models.

Despite challenges in integrating modeling and simulation, the methodology holds significant promise for optimizing enterprise architectures. It supports agile strategic capacity management, simulating incremental changes to identify bottlenecks and capacity constraints in complex systems. Enhanced calibration with real-world data will further improve its predictive accuracy and uncertainty management. In the long term, this methodology is poised to deliver comprehensive benefits for enterprise architecture management, including informed decision-making, goal alignment, and scalable problem-solving. It lays a strong foundation for testing disruptive approaches, such as agile methodologies and digital engineering in product development but other phases of the product lifecycle as well. Finally, it contributes to the evolution of Digital Twins of Organizations (DTOs) by establishing a robust framework for model calibration and real-world data integration.

Acknowledgments

Georgia Tech gratefully acknowledges Dassault Systèmes for its CATIA Magic Academic Sponsorship. Capgemini Engineering gratefully acknowledges Dassault Systèmes for their partnership in Digital Engineering and MBSE, and provision of Magic Systems of Systems Architect.

References

[1] Costa Melo, I., Alves Junior, P. N., Queiroz, G. A., Yushimito, W., and Pereira, J., "Do we consider sustainability when we measure small and medium enterprises (SMEs') performance passing through digital transformation?" *Sustainability*, Vol. 15, No. 6, 2023, 4917. https://doi.org/10.3390/su15064917.

[2] Machado, C. G., Winroth, M., Almström, P., Ericson Öberg, A., Kurdve, M., and AlMashalah, S., "Digital organizational readiness: experiences from manufacturing companies," *Journal of Manufacturing Technology Management*, Vol. 32, No. 9, 2021, pp. 167–182. https://doi.org/10.1108/JMTM-05-2019-0188.

[3] Kao, L.-J., Chiu, C.-C., Lin, H.-T., Hung, Y.-W., and Lu, C.-C., "Unveiling the dimensions of digital transformation: A comprehensive taxonomy and assessment model for business," *Journal of Business Research*, Vol. 176, 2024, p. 114595. https://doi.org/10.1016/j.jbusres.2024.114595.

[4] Jha, R., "Analyzing the impact of digital transformation on business," Ph.D. thesis, Massachusetts Institute of Technology, 2020.

[5] Mugge, P., Abbu, H., Michaelis, T. L., Kwiatkowski, A., and Gudergan, G., "Patterns of digitization: A practical guide to digital transformation," *Research-Technology Management*, Vol. 63, No. 2, 2020, pp. 27–35. https://doi.org/10.1080/08956308.2020.1707003.

[6] Kassan, M. W., "The Utility of Modeling an Enterprise Architecture for Air Force Digital Transformation," Master's thesis, Air Force Institute of Technology, 2021.

[7] Medeiros, P., Santana, A., Lima, M., Moura, H., and da Silva, M. M., "An agile approach for modeling enterprise architectures," *Proceedings of the 23rd International Conference on Enterprise Information Systems*, Vol. 2, 2021, pp. 659–670. https://doi.org/10.5220/0010450506590670.

[8] Jonkers, H., Lankhorst, M., ter Doest, H., Arbab, F., Bosma, H., and Wieringa, R., "Enterprise architecture: Management tool and blueprint for the organization," *Information Systems Frontiers*, Vol. 8, 2006, pp. 63–66. https://doi.org/10.1007/s10796-006-7970-2.

[9] Tamm, T., Seddon, P. B., Shanks, G., and Reynolds, P., "How Does Enterprise Architecture Add Value to Organisations?" *Communications of the Association for Information Systems*, Vol. 28, No. 10, 2011, pp. 141–168. https://doi.org/10.17705/1CAIS.02810.

[10] Molina, A., and Medina, V., "Application of enterprise models and simulation tools for the evaluation of the impact of best manufacturing practices implementation," *Annual Reviews in Control*, Vol. 27, No. 2, 2003, pp. 221–228. https://doi.org/10.1016/j.arcontrol.2003.09.005.

[11] Zacharewicz, G., Pirayesh-Neghab, A., Seregni, M., Ducq, Y., and Doumeingts, G., "Simulation-Based Enterprise Management: Model Driven from Business Process to Simulation," *Guide to Simulation-Based Disciplines. Simulation Foundations, Methods and Applications*, edited by S. Mittal, U. Durak, and T. Ören, Springer, Cham, 2017, p. 261–289. https://doi.org/10.1007/978-3-319-61264-5_12.

[12] Becker, M. C., and Pentland, B. T., "Digital Twin of an Organization: Are You Serious?" *Business Process Management Workshops*, edited by A. Marrella and B. Weber, Springer, Cham, 2022, pp. 243–254. https://doi.org/10.1007/978-3-030-94343-1_19.

[13] Jonkers, R. K., and Shahroudi, K. E., "A Design Change, Knowledge, and Project Management Flight Simulator for Product and Project Success," *IEEE Systems Journal*, Vol. 15, No. 1, 2021, pp. 1130–1139. https://doi.org/10.1109/JSYST.2020.3006747.

[14] Moser, B. R., and Grossmann, W., "Digital Twins of Complex Projects," *The Digital Twin*, edited by N. Crespi, A. T. Drobot, and R. Minerva, Springer, Cham, 2023, pp. 677–702. https://doi.org/10.1007/978-3-031-21343-4_24.

[15] Lyytinen, K., Weber, B., Becker, M. C., and Pentland, B. T., "Digital twins of organization: implications for organization design," *Journal of Organization Design*, 2023. https://doi.org/10.1007/s41469-023-00151-z.

[16] Bankauskaite, J., "Comparative analysis of enterprise architecture frameworks," *CEUR workshop proceedings: IVUS 2019 international conference on information technologies: proceedings of the international conference on information technologies*, Vol. 2470, 2019, p. 61–64.

[17] Martin, J. N., "Problem framing: Identifying the right models for the job," *INCOSE International Symposium*, Vol. 29, No. 1, 2019, p. 1–21. https://doi.org/10.1002/j.2334-5837.2019.00586.x.

[18] Browning, T. R., Fricke, E., and Negele, H., "Key Concepts in Modeling Product Development Processes," *System Engineering*, Vol. 9, No. 2, 2006, pp. 104–128. https://doi.org/10.1002/sys.20047.

[19] Yassine, A. A., "Managing the Development of Complex Product Systems: An Integrative Literature Review," *IEEE Transactions on Engineering Management*, Vol. 68, No. 6, 2021, pp. 1619–1636. https://doi.org/10.1109/TEM.2019.2929660.

[20] Wynn, D. C., and Clarkson, P. J., *The Design and Development Process: Perspectives, Approaches and Models*, Springer Nature, Cham, 2024. https://doi.org/10.1007/978-3-031-38168-3.

[21] Browning, T. R., and Eppinger, S. D., "Modeling Impacts of Process Architecture on Cost and Schedule Risk in Product Development," *IEEE Transactions on Engineering Management*, Vol. 49, No. 4, 2002, pp. 428–442. https://doi.org/10.1109/TEM.2002.806709.

[22] Meier, C., Browning, T. R., Yassine, A. A., and Walter, U., "The Cost of Speed: Work Policies for Crashing and Overlapping in Product Development Projects," *IEEE Transactions on Engineering Management*, Vol. 62, No. 2, 2015, pp. 237–255. https://doi.org/10.1109/TEM.2015.2411514.

[23] Lévárdy, V., and Browning, T. R., "An Adaptive Process Model to Support Product Development Project Management," *IEEE Transactions on Engineering Management*, Vol. 56, No. 4, 2009, pp. 600–620. https://doi.org/10.1109/TEM.2009.2033144.

[24] Cooper, K. G., "Naval Ship Production: A Claim Settled and a Framework Built," *Interfaces*, Vol. 1, No. 6, 1980, pp. 20–36. URL https://www.jstor.org/stable/25060009.

[25] Kasperek, D., Maisenbacher, S., and Maurer, M., "Structure-based Analysis of Dynamic Engineering Process Behavior," *2014 IEEE International Systems Conference Proceedings*, 2014, pp. 83–88. https://doi.org/10.1109/SysCon.2014.6819240.

[26] Lyneis, J. M., and Ford, D. N., "System dynamics applied to project management: a survey, assessment, and directions for future research," *System Dynamics Review*, Vol. 23, No. 2/3, 2007, p. 157–189. https://doi.org/10.1002/sdr.

[27] Borshchev, A., and Filippov, A., "From System Dynamics and Discrete Event to Practical Agent Based Modeling: Reasons, Techniques, Tools," *22nd International Conference of the System Dynamics Society*, 2004.

[28] Lapp, S., Jablokow, K., and McComb, C., "KABOOM:anagent-based model for simulating cognitive style in team problem solving," *Design Science*, Vol. 5, No. 13, 2019. https://doi.org/10.1017/dsj.2019.12.

[29] Cardinal, L. B., Turner, S. F., Fern, M. J., and Burton, R. M., "Organizing for Product Development Across Technological Environments: Performance Trade-offs and Priorities," *Organization Science*, Vol. 22, No. 4, 2011, p. 1000–1025. https://doi.org/10.1287/orsc.1100.0577.

[30] Jin, Y., and Levitt, R. E., "The Virtual Design Team: A Computational Model of Project Organizations," *Computational & Mathematical Organization Theory*, Vol. 2, No. 3, 1996, pp. 171–196. https://doi.org/10.1007/BF00127273.

[31] Zhang, X., and Thomson, V., "Modelling the development of complex products using a knowledge perspective," *Research in Engineering Design*, Vol. 30, 2019, pp. 203–226. https://doi.org/10.1007/s00163-017-0274-3.

[32] AIAA Digital Engineering Integration Committee, "Digital Thread: Definition, Value, and Reference Model," Tech. rep., AlAA, AIA, and NAFEMS, 2023. URL https://www.aiaa.org/resources/digital-thread-white-paper.

[33] Martin, J. N., and O'Neil, D. P., "Enterprise Architecture Guide for the Unified Architecture Framework (UAF)," *INCOSE International Symposium*, Vol. 31, No. 1, 2021, pp. 242–263. https://doi.org/10.1002/j.2334-5837.2021.00836.x.

[34] Negele, H., Fricke, E., and Igenbergs, E., "ZOPH – A Systemic Approach to the Modeling of Product Development Systems," *System Engineering*, Vol. 7, No. 1, 1997, pp. 266–273. https://doi.org/10.1002/j.2334-5837.1997.tb02181.x.

[35] Cohen, M. A., Eliasberg, J., and Ho, T.-H., "New Product Development: The Performance and Time-to-Market Tradeoff," *Management Science*, Vol. 42, No. 2, 1996, pp. 173–186. https://doi.org/10.1287/mnsc.42.2.173.

[36] Bayus, B. L., "Speed-to-Market and New Product Performance Trade-offs," *Journal of Product Innovation Management*, Vol. 14, No. 6, 1997, pp. 485–497. https://doi.org/10.1016/S0737-6782(97)00062-3.

[37] Grimaldi, M., Greco, M., Cricelli, L., Rogo, F., and Scalvenzi, M., "How Can Productivity in Product Design and Engineering Be Assessed? Guidelines to Build a Dashboard of KPIs," *IEEE Transactions on Engineering Management*, Vol. 71, 2024, pp. 560–573. https://doi.org/10.1109/TEM.2021.3131387.

[38] Anderson, A. M., "A framework for NPD management: doing the right things, doing them right, and measuring the results," *Trends in Food Science & Technology*, Vol. 19, 2008, pp. 553–561. https://doi.org/10.1016/j.tifs.2008.01.015.

[39] Li, C.-R., Chua, C.-P., and Lin, C.-J., "The contingent value of exploratory and exploitative learning for new product development performance," *Industrial Marketing Management*, Vol. 39, 2010, p. 1186–1197. https://doi.org/10.1016/j.indmarman.2010.02.002.

[40] Law, C. C., and Ngai, E. W., "An empirical study of the effects of knowledge sharing and learning behaviors on firm performance," *Expert Systems with Applications*, Vol. 34, 2008, p. 2342–2349. https://doi.org/10.1016/j.eswa.2007.03.004.

[41] Martin, J. N., and Brookshier, D., "Linking UAF and SysML Models: Achieving Alignment between Enterprise and System Architectures," *INCOSE International Symposium*, Vol. 33, No. 1, 2023, pp. 1132–1155. https://doi.org/10.1002/iis2.13074.

[42] Eckert, C. M., and Clarkson, P. J., "Planning development processes for complex products," *Research in Engineering Design*, Vol. 21, 2010, p. 153–171. https://doi.org/10.1007/s00163-009-0079-0.

[43] Alghamdi, A., Torkjazi, M., Davila-Andino, A. J., and Zaidi, A. K., "Employing UAF Inter-Domain Traceability for Performance and Effectiveness Evaluation," *2023 IEEE Internactional Systems Conference*, 2023, pp. 1–8. https://doi.org/10.1109/SysCon53073.2023.10131056.

[44] Lepez Da Silva Duarte, N., Pinon Fischer, O. J., and Mavris, D. N., "Towards a Framework for Modeling and Simulating Complex Enterprises: The Case of an Academic Research Laboratory," *AIAA SciTech Forum*, 2024. https://doi.org/10.2514/6.2024-2698.

[45] Wang, H., Li, H., Wen, X., and Luo, G., "Unified modeling for digital twin of a knowledge-based system design," *Robotics and Computer Integrated Manufacturing*, Vol. 68, 2021. https://doi.org/10.1016/j.rcim.2020.102074.

[46] Ulrich, K. T., and Eppinger, S. D., *Product Design and Development*, 5th ed., McGraw-Hill, New York, 2012. https://doi.org/10.4236/ad.2014.22003.

[47] Colfer, L. J., and Baldwin, C. Y., "The mirroring hypothesis: theory, evidence, and exceptions," *Industrial and Corporate Change*, Vol. 25, No. 5, 2016, pp. 709–738. https://doi.org/10.1093/icc/dtw027.

[48] Marion, T. J., and Fixson, S. K., "The Transformation of the Innovation Process: How Digital Tools are Changing Work, Collaboration, and Organizations in New Product Development," *Journal of Product Innovation Management*, Vol. 38, No. 1, 2021, p. 192–215. https://doi.org/10.1111/jpim.12547.

[49] Loerch, A. G., "Learning Curves," *Encyclopedia of Operations Research and Management Science*, edited by S. I. Gass and M. C. Fu, Springer US, Boston, MA, 2013, pp. 871–874. https://doi.org/10.1007/978-1-4419-1153-7_526.

[50] Oberkampf, W. L., "Simulation Accuracy, Uncertainty, and Predictive Capability: A Physical Sciences Perspective," *Computer Simulation Validation: Fundamental Concepts, Methodological Frameworks, and Philosophical Perspectives*, edited by C. Beisbart and N. J. Saam, Springer, Cham, 2019, pp. 69–97. https://doi.org/10.1007/978-3-319-70766-2_3.

[51] Roucoules, L., and Anwer, N., "Coevolution of digitalisation, organisations and Product Development Cycle," *CIRP Annals - Manufacturing Technology*, Vol. 70, No. 2, 2021, pp. 519–542. https://doi.org/10.1016/j.cirp.2021.05.003.

[52] Schummer, F., and Hyba, M., "An Approach for System Analysis with MBSE and Graph Data Engineering," *CoRR*, 2022, pp. 1–37. https://doi.org/10.48550/arXiv.2201.06363.

[53] Monteiro, J., Sá, F., and Bernardino, J., "Graph databases assessment: Janusgraph, neo4j, and tigergraph," *Perspectives and Trends in Education and Technology: Selected Papers from ICITED 2022*, edited by A. Mesquita, A. Abreu, J. V. Carvalho, and C. H. P. de Mello, Springer, Singapore, 2023, pp. 655–665. https://doi.org/10.1007/978-981-19-6585-2_58.

[54] Eppinger, S. D., Joglekar, N. R., Olechowski, A., and Teo, T., "Improving the systems engineering process with multilevel analysis of interactions," *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, Vol. 28, 2014, pp. 323–337. https://doi.org/10.1017/S089006041400050X.

[55] Hung, H.-F., Kao, H.-P., and Juang, Y.-S., "An integrated information system for product design planning," *Expert Systems with Applications*, Vol. 35, 2008, p. 338–349. https://doi.org/10.1016/j.eswa.2007.07.030.

[56] Wagg, D. J., Worden, K., Barthorpe, R. J., and Gardner, P., "Digital twins: State-of-the-art and future directions for modelling and simulation in engineering dynamics applications," *ASME Journal of Risk Uncertainty Part B*, Vol. 6, No. 3, 2020. https://doi.org/10.1115/1.4046739.

[57] Nattermann, R., and Anderl, R., "The W-Model Using Systems Engineering for Adaptronics," *Procedia Computer Science*, Vol. 16, 2013, p. 937–946. https://doi.org/10.1016/j.procs.2013.01.098.

[58] Hutchison, N., Pepe, K., Blackburn, M., Tao, H. Y. S., Verma, D., Whitcomb, C., Khan, R., Peak, R., and Baker, A., "Developing the Digital Engineering Competency Framework (DECF) – Phase 2," Tech. Rep. SERC-2021-TR-005, Stevens Institute of Technology, 2021. URL https://sercproddata.s3.us-east-2.amazonaws.com/technical_reports/reports/1616668486.A013_SERC%20WRT%201006_Technical%20Report%20SERC-2021-TR-005_FINAL.pdf.

[59] van der Aalst, W., *Process Mining: Data Science in Action*, 2nd ed., Springer, Heidelberg, 2016. https://doi.org/10.1007/978-3-662-49851-4.

[60] Maruster, L., and Alblas, A., "Tailoring the Engineering Design Process Through Data and Process Mining," *IEEE Transactions on Engineering Management*, Vol. 69, No. 4, 2022, pp. 1577–1591. https://doi.org/10.1109/TEM.2020.3000861.

[61] Brock, J., von Enzberg, S., Kühn, A., and Dumitrescu, R., "Towards business use cases of process miningin product development and manufacturing: Deriving and classifying18 application scenariosin industry," *Procedia CIRP*, Vol. 128, 2024, pp. 268–273. https://doi.org/10.1016/j.procir.2024.06.022.

[62] Rigger, E., Vosgien, T., Bitrus, S., Szabo, P., and Eynard, B., "Enterprise Architecture Method for Continuous Improvement of PLM Based on Process Mining," *Product Lifecycle Management Enabling Smart X*, Vol. 594, 2020, pp. 563–575. https://doi.org/10.1007/978-3-030-62807-9_45.