Proceedings

Modeling Language for Systems Engineering in Defense Industry †

Yoothana Suansook * and Tanongsak Taweesri

Systems Engineering Department, Defence Technology Institute, Office of the Permanent Secretary of Defence (Chaengwattana), Ban Mai, Pak Kret, Nonthaburi 11120, Thailand; tanongsak.t@dti.or.th
* Correspondence: yoothana.s@dti.or.th

† Presented at the Innovation Aviation & Aerospace Industry—International Conference 2020 (IAAI 2020), Chumphon, Thailand, 13–17 January 2020.

Published: 30 December 2019

Abstract: Systems engineering work begins with the concept of product design, product development and product implementation. The relationships between each step are complicated. This article is presenting the symbolic language that develops for managing the complexity in system engineering. The applications of this language are assisting in explaining the relationships as a blueprint that describes the details of the various parts especially defense industry.

Keywords: systems engineering; defense industry; symbolic language; ontology

1. Introduction

The industry that manufactures and sells weapons and military technology are known as arms industry or defense industry. The defense industry differs from other industries due both to the nature of its products and to its significance for national survival and national security [1]. These industries often involved in research and development military products and weapons utilized by the *armed forces, including artillery included guns, hand grenades, landmines, artillery, ammunition, missile, military aircraft, military vehicles, surveillance ships, electronic systems i.e., night-vision devices, laser rangefinders and more.* They are also providing maintenance and operational support. Many industrialized countries have manufactured the local arms-industry products to supply their military forces [2,3].

Systems engineering has an important role in these types of industry. Since many products are far more complicate for local manufacturer to produce. Systems engineering is an interdisciplinary field engineering that aim on how to design and manage complex systems over product life cycles, which focuses on designing systems that ensure performance optimization, robustness, and reliability. The product life cycles begin with the products design according to the requirement, further by products implementation and products maintenance until disposal [4]. The term that explains products life cycle management and the relationship between each phase is known as a *V-shape model*. This term arises from software development. In the systems engineering viewpoint the model consists of the following phase i.e., *User requirement, System Design, Architecture Design, Detail Design, Implementation, Unit Testing, Integration Testing, System Testing, User Acceptance Testing*. The complexity in design and development in systems engineering life cycle can be simplified via *symbolic modeling language* [4,5].

2. Information Modeling

Today, many industries are heavily influenced and disrupted by advance in digital technologies [6]. Many new technologies are introduced include artificial intelligence, augmented reality [7]. The defense industries in Thailand are begun with the production of armaments and domestic production of ordnance items which not much complexity in productions. The present goal of Thai government's industrial development is focusing on the sophisticated or advanced technology industry development which included digital industry, the robotics industry, and avionics and defense industries. The modernization of military vehicle, aircraft engines, and guided missile are complicated in production. The engineering processes relevant to these industries are becoming more critical and complex. Therefore the rapid adaptation to new technological change is important to deal with [8].

In this section various modeling languages briefly described. The case study of modeling the defense industry product is provided. The modeling often uses to simplify or explain the behavior of a complex system. The information modeling is feasible to explain by symbolic language. The communication method that uses characters or images to represent concepts is known as a *symbolic language*. In systems engineering concept the symbolic language has been developed to assist in simplifying engineering management in the form of *pictorial diagrams* to represent engineering concepts. The logical semantics behind the diagram is based on *ontology* which consists of *objects* and their *relations* [9]. The following are information modeling that wildly uses in software engineering and computer science:

2.1. Entity—Relationship Model (ER-Model)

The early development of data modeling for relational database design is known as an *entity—relationship model (ER-model).* The model describes interrelated things of interest in a specific domain in the form of entity and its relationship. The model was developed by *Peter Chen* and published in a 1976. In database design concept, the entity-relationship model is commonly formed to represent things of consideration in business processes. The purpose of the design is to preserve the information details and reduce redundancy in storage because the cost of storage systems in the past was expensive [10,11].

2.2. Unified Modeling Language (UML)

The development of *Object-Oriented Analysis and Design* of information systems is assisting in creating the new modeling language called "*Unified Modeling Language (UML)*". This language is created by combining modeling method of *Grady Booch, Ivar Jacobson* and *James Rumbaugh* and incorporates concepts from other industry practice together. It is shaping the concept of relational database design in the form of "*objects*" rather than "*entities*" or the thing of interested. The model defines the object as a *class* and their relationship in object oriented behavior, i.e., *aggregation, composition, inheritance, polymorphism* etc. This modeling language is intended to provide a standard to visualize the system design in the field of software engineering. This modeling language is widely accepted as the *de facto standard* for software development. In 1997, UML was adopted as a standard by the Object Management Group (OMG) [12].

2.3. Systems Modeling Language (SysML)

The *Unified Modeling Language (UML)* has been developed further for systems engineering applications in the form of the *Systems Modeling Language (SysML). SysML* is defined as an extension of a subset of the *Unified Modeling Language (UML)* using UML's profile mechanism. The languages have provided many diagram types which designed to support systems engineering activities, and also support the *specification, analysis, design, verification* and *validation* in product life cycle management [13,14].

2.4. Object Process Methodology (OPM)

Object Process Methodology (OPM) is a conceptual modeling language and methodology for capturing knowledge and designing systems. It integrates the *object-oriented* and *process-oriented paradigms* into a single frame of reference. The elements of this modeling are *things, states* and *links*. The basic elements of any system are explained by *object* and *process*. *Objects* are *things* that exist, while *processes* are *things* that transform *objects*. At any specific point in time, an *object* can be exactly in one *state*, and *object states* are changed through occurrences of processes. Analogically, *links* can be structural or procedural. Structural *links* express static relations between *pairs of entities*, while procedural links connect entities (*objects, processes, and states*) to describe the behavior of a system. *OPM* was conceived and developed by *Dov Dori* and published in 1995. The *ontology* of *OPM* is identical to the *ontology* of *Navya-Nyāya* an ancient Hindu school of thought in India [15,16].

2.5. Defense Architecture Framework (DoDAF)

The *United States Department of Defense (DoD)* provides visualization infrastructure for specific stakeholder concerns through viewpoints organized by various views called the *Department of Defense Architecture Framework (DoDAF).* This framework assist in visualizing, understanding, and assimilating the broad scope and complexities of an architecture description through *tabular, structural, behavioral, ontological, pictorial, temporal, graphical, probabilistic,* or *alternative conceptual means.* This Architecture Framework is well suited to large system with complex integration and interoperability challenges [17].

2.6. Architecture Analysis and Design Language (AADL)

The *Architecture Analysis & Design Language (AADL)* is an architecture description language standardized by SAE (*Society of Automobile Engineers*). AADL was first developed in the field of avionics, and was known formerly as the *Avionics Architecture Description Language.* This architecture model can then be used either as a design documentation, for analyses i.e., flow control or code generation (of the software portion), like UML. This symbolic language is suitable for software mission and *safety-critical systems,* such as avionics systems in aircraft. The language addresses common problems arise in systems development, such as mismatched assumptions about physical system, and their interaction that can result in too late detected problems in the product development life cycle [18].

3. Materials and Methods

The military products often comprise several of interacting subcomponents that independently and collectively constraint to the complex set of performance requirements. In this article, we considered the *Multiple Launch Rocket System (MLRS)* as a modeling instance of product in the defense industry. The MLRS is an armored, self-propelled, multiple rocket launcher; a type of rocket artillery. The generic system consists of the following major components *Launcher (Vehicle), Rocket Canister, Fire Control, Power System, Mechanical System* etc. The simple relationships between components are illustrated with UML. The best practice in design the model is to create a simple model rather than create the complex model and filled in the details later on. The simplify form of a system is described as a *class* and their *relationship*. The proper model can extend to other related diagrams in systems engineering i.e., *Package diagram, Sequence diagram, Parametric diagram* etc. To capture the information about this type of armament product in modeling form is depicted as in Figure 1.

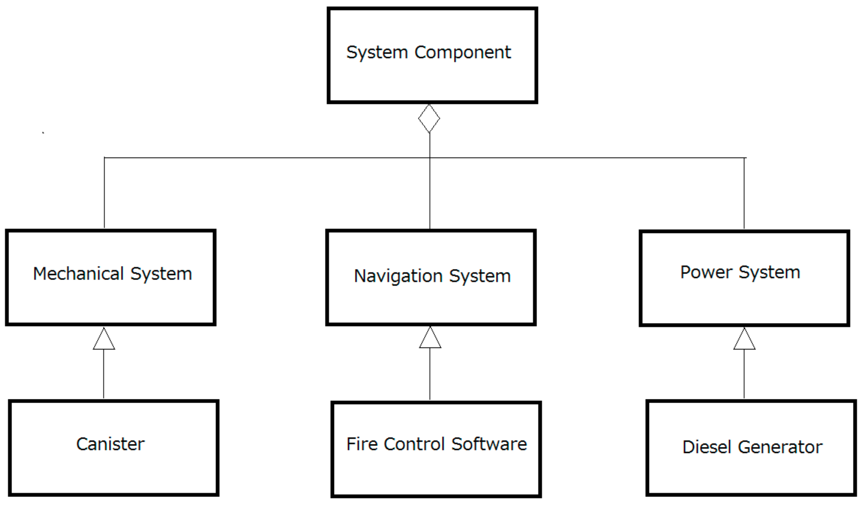

Figure 1. Illustrate the simplified modeling of the *Multiple Launch Rocket System (MLRS)* system with UML. This simple modeling system is done by capturing the related information concern, which not intended for use as database design.

4. Conclusions

Traditional systems engineering management is based on documentation base, which not able to capture the entire picture as a whole. The complexity in design and development in systems engineering life cycle can be simplified via *graphical symbolic modeling language*. In this article the widely recognized modeling languages briefly described. A simple model of military product has been presented with UML. Modern systems engineering management need modeling language to alleviate the transition from analysis to design and from design to implementation. These models can be transformed or converted between the models. Since the foundation of the modeling language are based on the *ontology* which representation in the form of *objects* and their *relationship* [19].

Acknowledgements: The authors gratefully acknowledge the financial support from the Defence Technology Institute, Thailand.

Conflicts of Interest: The authors declare that there is no conflict of interests regarding the publication of this article.

References

1. Kopač, E. Defense Industry Restructuring: Trends in Europe an and U.S. Defense Companies. *Transit. Stud. Rev.* **2006**, *13*, 283–296.
2. Chiang, J.-T. Defense conversion and systems architecture: Challenges to Taiwan's aircraft industry. *Technol. Soc.* **1999**, *21*, 263–274.
3. Arms Industry Webpage. Available online: https://en.wikipedia.org/wiki/Arms_industry (accessed on 28 November 2019).
4. *INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*; Wiley: Hoboken, NJ, USA, 2015.
5. De Niz, D. *Diagrams and Languages for Model-Based Software Engineering of Embedded Systems: UML and AADL*; Software Engineering Institute: Pittsburgh, PA, USA, 2007.
6. McQuivey, J. *Digital Disruption: Unleashing the Next Wave of Innovation*; Forrester Research, Incorporated: Amsterdam, The Netherlands, 2013.
7. Donald,M.; Donald, M. *Organisational Implications of Artificial Intelligence, Leading and Managing Change in the Age of Disruption and Artificial Intelligence*; Emerald Publishing Limited: Bingley, UK, 2019; pp. 57–74.
8. Jamshidi, M. System of systems engineering—New challenges for the 21st century. *IEEE Aerosp. Electron. Syst. Mag.* **2008**, *23*, 4–19.

9. Valente, A.; Holmes, D.; Alvidrez, F.C. Using a Military Information Ontology to Build Semantic Architecture Models for Airspace Systems. In Proceedings of the 2005 IEEE Aerospace Conference, Big Sky, MT, USA, 5–12 March 2005.
10. Chen, P.P.-S. The Entity Relationship Model—Toward a Unified View of Data. In *Pioneers and Their Contributions to Software Engineering;* Springer: Berlin, Germany, 1976; pp. 205–234.
11. Leung, C.M.R.; Nijssen, G.M. Relational database design using the NIAM Conceptual Schema. *Inf. Syst.* **1988**,*13*,219–227.
12. Booch, G.; Rumbaugh, J.; Jacobson, I. *The Unified Modeling Language User Guide*; Addison-Wesley Professional: Boston, MA, USA, 1998.
13. OMG, OMGSysML Specifications Webpage. Available online: https://www.omgsysml.org/specifications.htm (accessed on 28 November 2019).
14. Grobshtein, Y.; Dori, D. Generating SysML views from an OPM model: Design and evaluation. *Syst. Eng.* **2011**, *14*, 327–340.
15. Dori, D. *Object-Process Methodology: A Holistic Systems Paradigm*; Springer: Berlin/Heidelberg, Germany, 2002.
16. Dori, D.; Goodman, M. From object-process analysis to object-process design. *Ann. Softw. Eng.* **1996**, *2*, 25–50.
17. Department of Defense Architecture Framework Webpage. Available online: https://en.wikipedia.org/wiki/Department_of_Defense_Architecture_Framework (accessed on 28 November 2019).
18. Architecture Analysis and Design Language. *Software Engineering Institute, Carnegie-Mellon University, Pittsburgh*; Architecture Analysis and Design Language: Pennsylvania, PA, USA. Available online: https://www.sei.cmu.edu/research-capabilities/all-work/display.cfm?customel_datapageid_4050=191439 (accessed on 28 November 2019).
19. Ontology (Information Science). Available online: https://en.wikipedia.org/wiki/Ontology_(information_science) (accessed on 28 November 2019).