systems

MDPI

Article

Integration of SysML and Virtual Reality Environment: A Ground Based Telescope System Example

Mostafa Lutfi * and Ricardo Valerdi

Department of Systems and Industrial Engineering, The University of Arizona, Tucson, AZ 85721, USA; rvalerdi@arizona.edu
* Correspondence: mostafalutfi@arizona.edu

Abstract: In recent years, Model Based Systems Engineering (MBSE) has continued to develop as a standard for designing, managing, and maintaining increasingly complex systems. Unlike the document centric approach, MBSE puts the model at the heart of system design. Among the various MBSE language development efforts, "Systems Modeling Language (SysML)", is the most anticipated and broadly utilized in the research and in industrial practice. SysML originated from Unified Modeling Language (UML) and follows the Object-Oriented Systems Engineering Method (OOSEM). SysML diagrams help users create various systems engineering artifacts, including requirements, use cases, operational concepts, system architecture, system behaviors, and parametric analyses of a system model. In the early days of implementation, MBSE languages, including SysML, typically relied on static viewpoints and limited simulation support to depict and analyze a system model. Due the continuous improvement efforts and new implementation approaches by researchers and organizations, SysML has advanced vastly to encompass dynamic viewpoints, in-situ simulation and enable integration with external modeling and simulation (M&S) tools. Virtual Reality (VR) has emerged as a user interactive and immersive visualization technology and can depict reality in a virtual environment at different levels of fidelity. VR can play a crucial role in developing dynamic and interactive viewpoints to improve the MBSE approach. In this research paper, the authors developed and implemented a methodology for integrating SysML and VR, enabling tools to achieve three dimensional viewpoints, an immersive user experience and early design evaluations of the system of interest (SOI). The key components of the methodology being followed in this research paper are the SysML, a VR environment, extracted data and scripting languages. The authors initially developed a SysML for a ground-based telescope system following the four pillars of SysML: Structure, Requirements, Behavior and Parametrics. The SysML diagram components are exported from the model using the velocity template language and then fed into a virtual reality game engine. Then, the SysML diagrams are visualized in the VR environment to enable better comprehension and interaction with users and Digital Twin (DT) technologies. In addition, a VR simulation scenario of space objects is generated based on the input from the SysML, and the simulation result is sent back from the VR tool into the model with the aid of parametric diagram simulation. Hence, by utilizing the developed SysML-VR integration methodology, VR environment scenarios are successfully integrated with the SysML. Finally, the research paper mentions a few limitations of the current implementation and proposes future improvements.

Keywords: virtual reality (VR); systems modeling language (SysML); model based systems engineering (MBSE); digital twin (DT)

Citation: Lutfi, M.; Valerdi, R. Integration of SysML and Virtual Reality Environment: A Ground Based Telescope System Example. *Systems* **2023**, *11*, 189. https://doi.org/10.3390/systems11040189

Academic Editors: Ed Pohl and Eric Specking

Received: 10 March 2023
Revised: 3 April 2023
Accepted: 5 April 2023
Published: 7 April 2023

1. Introduction

According to the International Council on Systems Engineering (INCOSE), "Systems Engineering is a transdisciplinary and integrative approach to enable the successful realization, use, and retirement of engineered systems, using systems principles and concepts, and scientific, technological, and management methods" [1]. Model Based Systems Engineering

(MBSE) focuses on formalized applications of modeling to support systems engineering artifacts development from the conceptual design phase throughout the end of the system of interest (SOI) lifecycle [2]. In his 1993 book, "Model Based Systems Engineering", Dr. Wayne Wymore first introduced the term "MBSE" [3]. Systems engineers are living in a generation wherein modern systems are growing in complexity [4,5]. MBSE is the proposed solution by the researchers to cope with system complexity in a variety of SOI [6,7]. Various MBSE methodologies have been developed in recent years. The most notable ones are Object Oriented Systems Engineering Method (OOSEM), Object Process Methodology (OPM), State Analysis, IBM Harmony and Arcadia-Capella [8,9]. Among these, OOSEM-based SysML has better tool support than any other MBSE methodologies available [10,11]. In fact, SysML has emerged as the de facto standard system modeling language for MBSE [12,13]. According to Friedenthal, Moore, and Steiner, "SysML is a general-purpose graphical modeling language that supports the analysis, specification, design, verification, and validation for complex systems" [14]. SysML is owned and published by the Object Management Group, Inc. (OMG) [15]. SysML is a derivation of UML, developed in the 1990s as a general-purpose language for software engineering [16]. SysML consists of nine diagram types, namely Package Diagram, Requirements Diagram, Block Definition Diagram (BDD), Internal Block Diagram (IBD), Use Case Diagram, Activity Diagram, State Machine Diagram, Sequence Diagram, and Parametric Diagram (see Figure 1) [15].

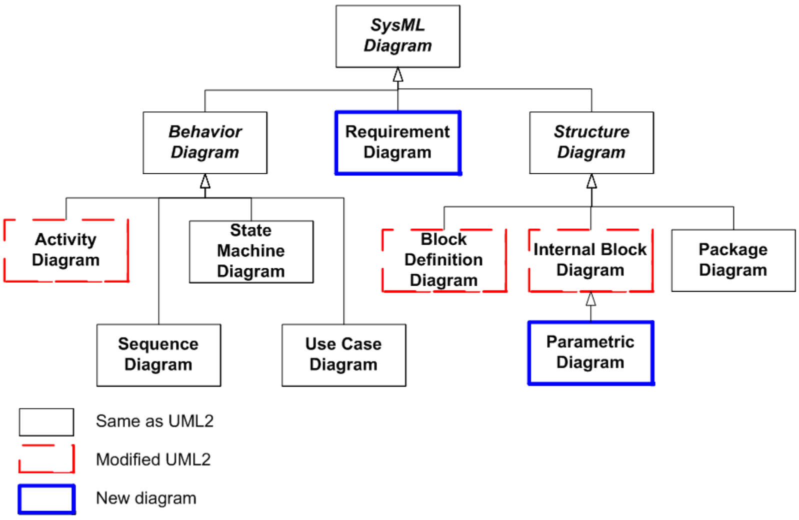

Figure 1. SysML diagram taxonomy [17].

VR is one of the major technologies with the utmost anticipated potential for growth, especially in the fields of engineering, education, gaming, cinematography and healthcare [18–25]. For example, different engineering fields leveraged VR technologies to accomplish various tasks, such as VR based Civil Engineering training, Industry 4.0 implementation, and Aerospace training [26–32]. Coates et al. defined VR as an electronic simulation of environments with head mounted display and wired outfit, which allows the end user to interact with lifelike 3D environments [33]. A typical VR system consists of the following key components: VR implementation software, Head Mounted Display (HMD), base station, tracking sensors, feedback devices, on board/external computers and users [34–36].

Though SysML application has been widespread in recent years, it has not been fully adopted by organizations. In addition, stakeholders may not have knowledge of SysML [37]. External product suppliers do not have SysML software knowledge and often ask for Excel/Word documents for data exchanges. Hence, system engineers still need to generate documents from the SysML system model to present the work in front of non-technical audiences. Therefore, there is a need for automatic data extraction from SysML models to facilitate model demonstration to technical and non-technical stakeholders alike.

To date, SysML models have mostly been used to define requirements, use cases, system architectures, system behaviors, and task sequences through static viewpoints and elements. In recent years, researchers have incorporated simulation analysis within SysML models to enable system requirements verification and design evaluation [38,39]. However, very few studies have focused on the integration of VR tools with SysML models. Recent studies indicate that VR-supported immersive design review allows participants to identify more issues/faults in design, improve collaborative engagement, focus and presence as compared to non-immersive methods [40,41]. Moreover, VR environments are beneficial for visualizing complex structures and provide higher levels of understanding and knowledge retention than the conventional approaches [42–44]. Recent case studies/evaluations by the researchers showed that VR environment translation of visual formalisms (similar to SysML) resulted in enhanced collaboration, learning and understanding among the users/stakeholders. For example, multiple studies have confirmed that 3D VR environments significantly increase the processes of knowledge retention and the collaborative process modeling experience of Business Process Model and Notation (BPMN) [45–47]. Similarly, modeling 3D UML diagrams in a virtual environment resulted in faster model comprehension and a more enjoyable modeling experience than 2D diagrams [48,49]. Based on the above study results, it can be elicited that visualization of 2D SysML diagrams in a 3D VR environment can improve model understanding, especially of complex systems, and facilitate a better collaborative environment among users and stakeholders.

VR technology can help visualize the system architecture, facilitate virtual storytelling, verify system requirements and evaluate product/service performance early in the lifecycle of a SOI. For example, a system engineer can evaluate system architecture (e.g., Telescope Mount System) in a 3D virtual environment and, based on their understanding of the system, modify/refine the architecture early in the design phase. In addition, external stakeholders with minimal technical knowledge can be immersed into a 3D virtual environment and experience use case scenarios of proposed designs. As a result, there is a need for interoperability between 3D VR models and SysML models to enable data exchange and review to systems engineering artifacts such as use cases, requirements, operational concepts, system architecture, etc.

In addition, use of Digital Twin (DT) technologies in systems engineering projects have been increasing in recent years. DT as a concept was first introduced by Michael Grieves as a virtual representation of an actual component that can be used to emulate the same behavior as that of its real world counterpart [50]. However, the term "Digital Twin" first appeared in National Aeronautics and Space Administration's (NASA) draft technological roadmap published in 2010 [51]. Over the last two decades, literature and research in the field has evolved to produce a wide variety of DT definitions and methodologies [52]. To simplify the ambiguity, a DT can be defined as a digital or virtual model of a real-world object (system, component, part, process, human, etc.) representative of the exact or future state of its physical twin via real time data exchange as well as keeping records of historical data [53]. Virtual models can be developed using a variety of methods, including VR, Augmented Reality (AR), Mixed Reality (MR), dashboard technologies and 2D simulations, etc. [54–59]. For example, developers can fabricate an entire machine in a digital environment and test their high-level control mechanism in a VR environment; manufacturing engineers can utilize dashboards to analyze the production process in real time and make necessary adjustments. Data exchanges between real and virtual counterparts of a DT infrastructure are achieved through internal/external sensors connected to physical systems and communication networks [58]. Recently, few DT frameworks have emerged where VR environments co-exist with other DT infrastructure and support validation of the digital/virtual configurations before the physical twin implementation [60,61]. The integration of MBSE tools and DT infrastructure can present several opportunities for systems engineers to test and visualize large models [62,63]. Both the physical and virtual models can be integrated with a shared MBSE repository (e.g., SysML model), which will act as a communication hub between DT infrastructure and systems engineering activities [63].

Moreover, MBSE efforts can introduce early DT development in a program's lifecycle [64]. Therefore, integration of SysML models with VR tools is one of the key steps in establishing synergy between DT and MBSE.

Hence, the authors have developed a methodology to integrate a SysML model with VR environments for a ground-based telescope system. The authors have chosen the above-mentioned system since telescope systems heavily rely on digital data products and generate virtual scenarios to analyze different aspects of scientific findings. The key contributions of this research are summarized as follows:

- A SysML model for a ground-based telescope system is developed, which will act as a reference model for the telescope MBSE practitioners.
- A detailed methodology to integrate SysML models with VR environments.
- Data interchange between SysML parametric diagram simulation and the VR scenario, which will enable DT and MBSE co-existence.
- SysML model diagram visualization in the interactive VR tool, which will enhance the systems modeling experience, model understanding and collaboration among stakeholders.

2. Literature Review

The authors believe there are limited studies available in MBSE research which emphasize integration of VR environments with SysML tools. In his thesis paper, Kande proposed an integration methodology of Virtual Engineering (VE) suite with the SysML model, providing a graphical interface to demonstrate the system of interest and operations [65]. He created a fermenter analysis model using Computer Aided Design (CAD) data and analyzed the effects of changes in design parameters on system performance. To enable effective communication of systems engineering artifacts to the non-engineering disciplines, Madni mapped systems engineering (SE) artifacts modeled in SysML to virtual worlds, within which different storylines can unfold [66]. In addition, he combined MBSE + frameworks (MBSE and storytelling in VR) and Experimental Design Language (EDL) to enable early participation and collaboration of the stakeholders in the system design [67]. Abidi et al. proposed an interactive VR simulation to encompass real time simulation of production flows in a lean manufacturing industry through communication between ARENA and Virtual Environment by utilizing SysML based transformation and Real Time Infrastructure (RTI) [68]. Mahboob et al. proposed a concept for a user- and task-centric model for product development in the VR environment. In that research proposal, SysML was used to separate isolated model components into products, actors and behaviors of a specific use case, and then those model components were combined to build the use case-specific scenario in VR [69]. Sanvordenker visualized a self-driving truck's SysML model in a VR environment by extracting SysML model diagrams, representing them in a browser, and finally embedding the web browser inside a Unity scene through a custom plugin [70]. Oberhauser demonstrated a solution concept for visualizing and interacting with SysML models as stacked hyperplanes in a VR environment [71].

However, in the above research there are some limitations. Namely, requiring extensive manual effort to set up each of the simulation components and frequent changes in the interface files (e.g., dynamic link library) to enable seamless data exchange. Furthermore, the VR diagrams used in these studies were more complex (e.g., stacked hyperplanes) instead of being easily comprehendible and their methodologies involved far-reaching conversions of models (XMI conversion, MASCARET, etc.). Those complex design techniques may hinder the mass use of VR-enabled SysML due to the time and effort needed to set up those systems.

3. Methodology

The overview of this study's methodology is summarized in Figure 2. Key components of the methodology are: SysML System Model, Virtual Environment, Scripting Languages and Extracted Data. Details of the components are described below.

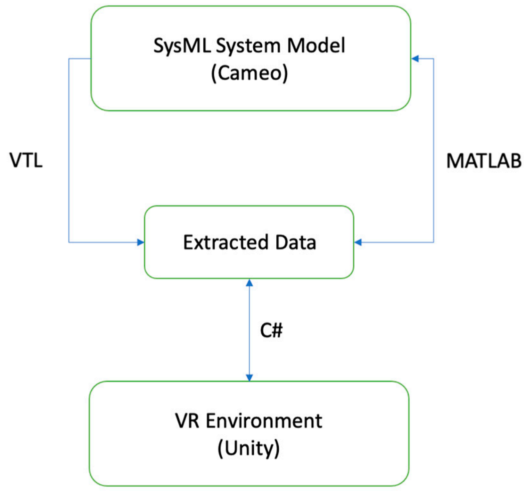

Figure 2. Methodology overview.

3.1. SysML System Model

An SysML system model functions as the central hub for all the systems engineering artifacts. In this research paper, the authors selected eight SysML diagram types to represent system hierarchy, requirements, structure, and behavior, which are as follows: Package Diagram, Requirement Diagram, Block Definition Diagram (BDD), Internal Block Diagram (IBD), Use Case Diagram, Activity Diagram, State Machine Diagram and Parametric Diagram. A *Requirement* Diagram represents different types of requirements (system level, interface, etc.) and their traceability with other requirements, model elements that satisfy them, and the test cases which verify them. *BDDs* are primarily used to depict a system's physical/logical architecture through the hierarchical relationship between the subsystem level components and classifiers. *IBD* is a diagram type which is owned by a particular block/system context and shows parts, properties, ports, interfaces and connectors within it. *Use Case* Diagrams define use cases and operational concepts by visualizing the relationship between the system's key functions and its actors (human/nonhuman). An *Activity* Diagram is a key SysML tool to represent system, subsystem and component level scenarios, and to represent behaviors through the sequence of data/control flows between actions. *State Machine* Diagrams are utilized to present information about dynamic system behaviors, transitions between system states, lifetime of a block through states, event triggers and transition elements. *Parametric* Diagrams are a specialized IBD diagram that help to combine structural and behavior model elements with engineering, simulation, or mathematical models. *Package* Diagrams facilitate the organization of a complex system model through container packages and their relationships. The authors used Cameo Systems Modeler (CSM) as the SysML tool because it has Cameo Simulation toolkit incorporated [72]. Cameo Simulation toolkit and Report Wizard tools support a wide variety of scripting languages including JavaScript, Python, MATLAB, Velocity Template Language (VTL), Modelica, etc. [73].

3.2. VR Environment

The authors propose to use Unity as the VR environment tool to be integrated with CSM [72]. The VR environment can be configured by utilizing the input from the SysML system model. The VR scene can incorporate human interaction/feedback, which can be returned to the SysML model using two-way communication. Unity can read the automatically generated excel files with the aid of a scripting language to extract the SysML model elements and import them into the VR environment. The major elements of a Unity scene that one must have knowledge of to understand this research are as follows: *GameObject*: fundamental objects in Unity that represent characters, props and scenery. *Components*: added to the game objects to provide real functionality. *Transform*: determines the Position, Rotation, and Scale of each object in the scene *Prefab*: acts as a template from which one can create new object instances in the scene [74]. For example, in this research paper, the authors used Cube Game Objects to represent the blocks and requirements. So, the authors attached scripts and collider components to a single instance of Cube. Then, the authors created a prefab of the Cube so that the authors can automate the production of similar game objects in the scene.

3.3. Extracted Data

For presentation purposes and communication of model data with external tools, reusable generic scripts are developed to automate the generation of SysML model output files in various formats (Word, Excel, XML, etc.) containing model elements and diagrams/viewpoints. The extraction is done through a custom script which can be generalized/modified for other project usage. Similarly, Unity simulation results are extracted as Excel files, which are in turn imported into the SysML model for analysis.

3.4. Scripting Languages

The authors have used VTL to create the report wizard template file in the CSM to export the SysML model diagram components (see Figures 3 and 4) [75]. CSM is also integrated with MATLAB to exchange simulation parameter values with the Unity engine. On the other hand, Unity installation comes with built-in C# scripting capabilities (see Figure 5). Hence, C# scripts are used to define the behavior of the game objects, import/export data values and user interaction methods (see Figure 5).

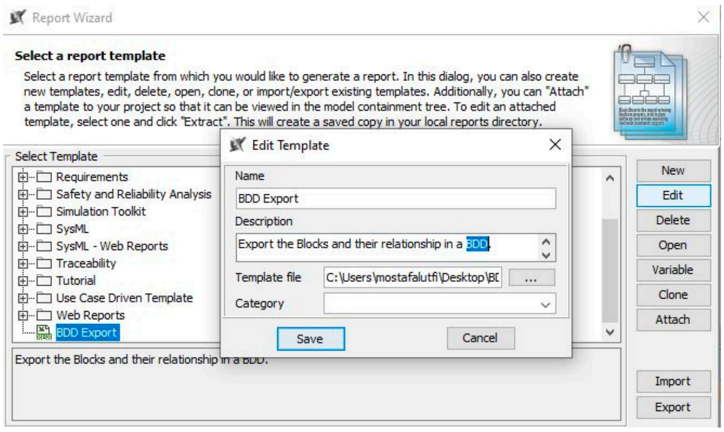

Figure 3. Report wizard setup.

```
#set($scopedPackage = $packageScope.get(0))
#set($child = $sorter.humanSort($report.getOwnedElementsIncludingAdditional($scopedPackage, true), “order:asc”))
#set($blocks = $array.createArray())
#foreach($el in $child)
#if(($report.containsStereotype($el, “Block”)))
#set($filler = $blocks.add($el))
#end
#end
#if($list.size($blocks) != 0)
#foreach($bl in $blocks)$bl.name
#end
#end
```

Figure 4. VTL script snippets.

```
void OnMouseOver()
{
    // Change the color of the GameObject to red when the mouse is hovering over GameObject
    m_Renderer.material.color = m_MouseOverColor;

    //Read the x position of the selected GameObject
    xPos = gameObject.transform.position.x;
    Debug.Log(xPos);
    detectObjects();
}
```

Figure 5. C# script snippets.

4. Ground Based Telescope System Model

The Ground Based Telescope System Model developed for this research is a large model consisting of multiple instances of the SysML diagram types. Hence, each of the diagram type examples are included here to demonstrate how SysML diagrams are used to model different system artifacts for the SOI.

Figure 6 shows the SysML package diagram of the system model, which is consists of "requirements", "structure", "behavior" and "parametric" packages. The requirements package consists of the requirements diagrams and tables. The structure package includes system architecture and communication interfaces. The behavior package contains use cases, operational scenarios, and system states. The parametric package includes the data exchange and integration components with the VR environment.

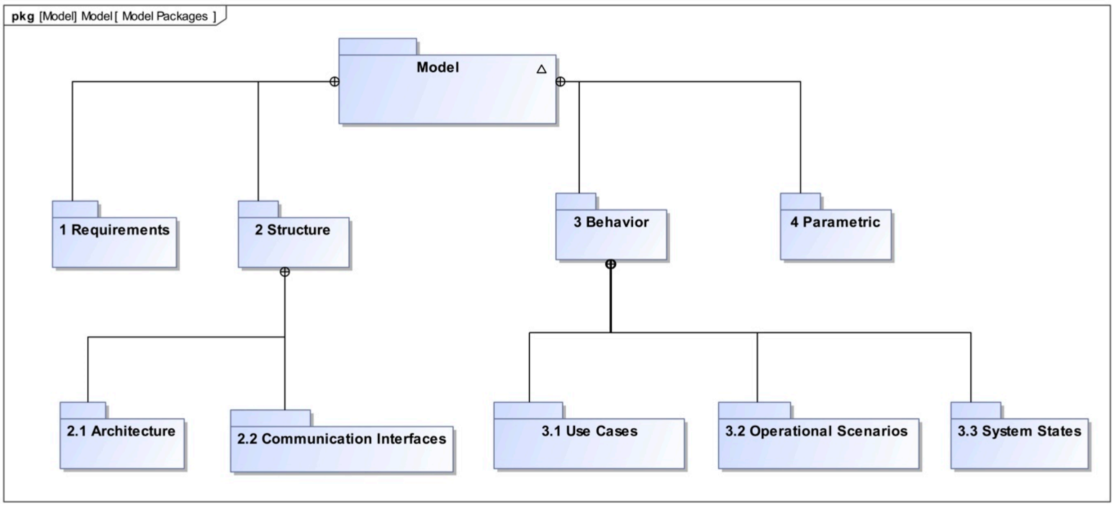

Figure 6. Package diagram of the telescope system model.

In this study, SysML requirement diagrams are used to represent the requirements specifications and their relationships with lower-level requirements. Relationship types between requirements and lower-level requirements and/or other related elements are "containment", "derive", "refine", "satisfy", "verify", and "trace". Figure 7 shows a SysML requirement diagram with a containment relationship between parent and child requirements for a telescope data collection system. Requirements can be displayed using a requirements table as well (Figure 8).

SysML BDDs are used to depict the system architecture and relationship between the components of the different subsystems of the ground-based telescope. Figures 9 and 10 demonstrate how the authors have decomposed different subsystems to a lower level with the BDD. Also, users can specify the multiplicity to define the number of components necessary for a given subsystem. The authors also included a navigator to the previous diagram (see Figure 10).

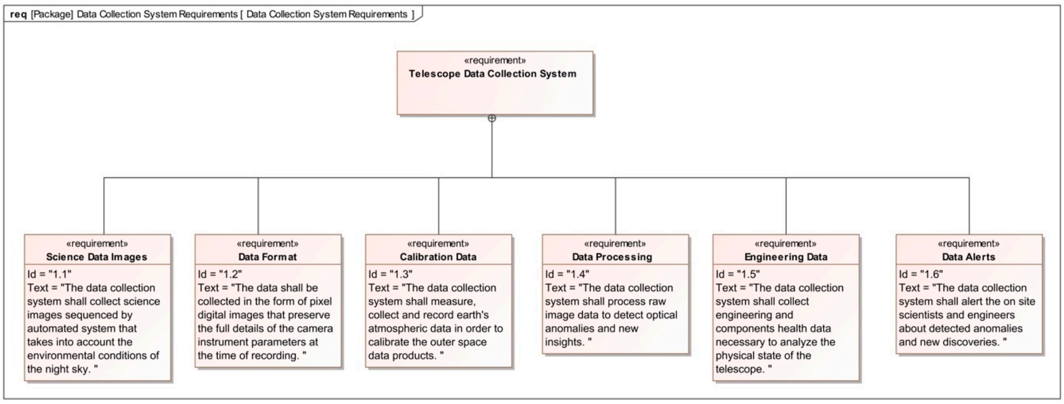

Figure 7. Requirements diagram for the data collection system.

#	Name	Text
1	1 Telescope Data Collection System	The telescope shall be developed with the necessary data collection subsystems in order to achieve the survey objectives.
2	1.1 Science Data Images	The data collection system shall collect science images sequenced by automated system that takes into account the environmental conditions of the night sky.
3	1.2 Data Format	The data shall be collected in the form of pixel digital images that preserve the full details of the camera instrument parameters at the time of recording.
4	1.3 Calibration Data	The data collection system shall measure, collect and record earth's atmospheric data in order to calibrate the outer space data products.
5	1.4 Data Processing	The data collection system shall process raw image data to detect optical anomalies and new insights.
6	1.5 Engineering Data	The data collection system shall collect engineering and components health data necessary to analyze the physical state of the telescope.
7	1.6 Data Alerts	The data collection system shall alert the on site scientists and engineers about detected anomalies and new discoveries.

Figure 8. Requirements table for the data collection system.

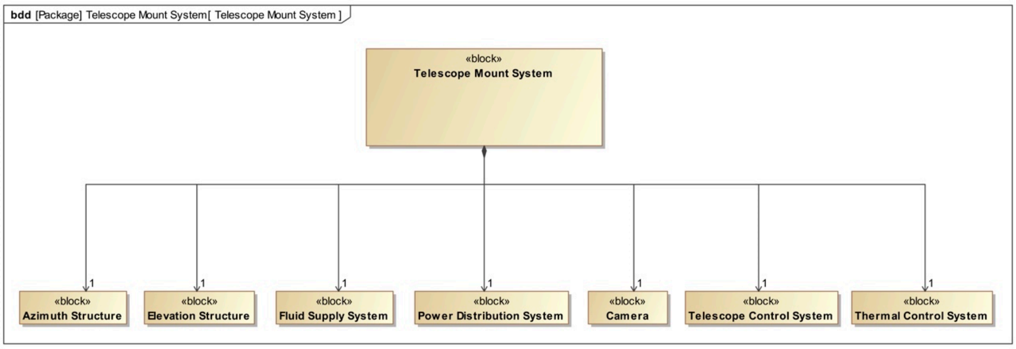

Figure 9. BDD of the telescope mount system.

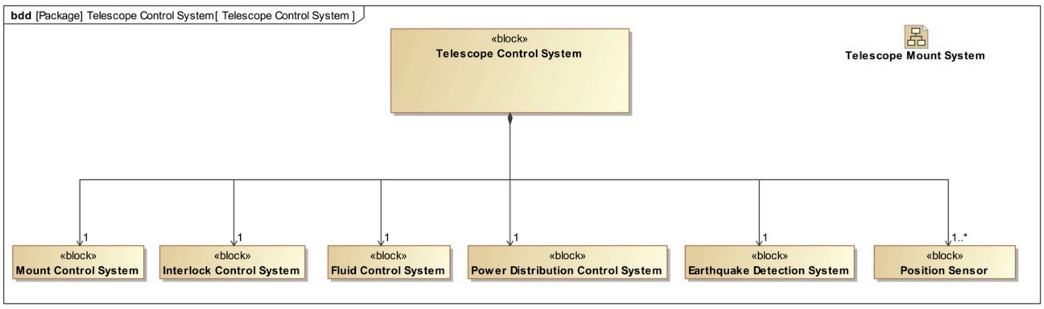

Figure 10. BDD of the telescope control system.

Communications and data flow between different subsystems are represented by SysML IBDs. Specifically, the IBDs represent interconnection, interfaces, and ports between the parts of a block. For example, Figure 11 depicts the telescope control system data flows. Signals are sent from one port to another through connectors to depict the transfer of data between the system components.

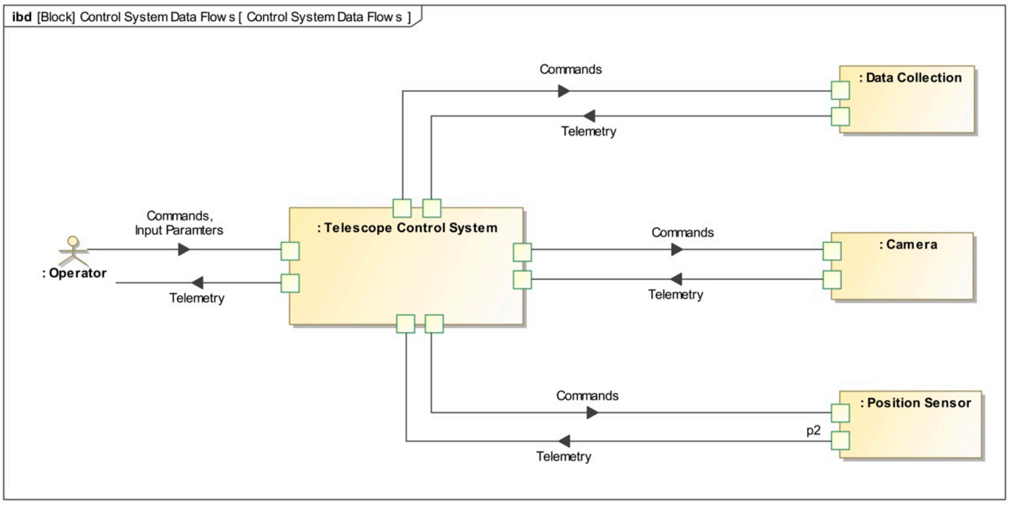

Figure 11. IBD showing telescope control system data flows.

SysML use case diagrams are a good way of visualizing operations concepts and use cases for different systems through the visualization of the interaction between actors and use cases. In Figure 12, a data analysis operational concept (OpsCon) is presented that consists of use cases, actors, and their relationships. As the actors are human in this case, they are represented by a human icon. Non-human actors are usually represented through block-like actors.

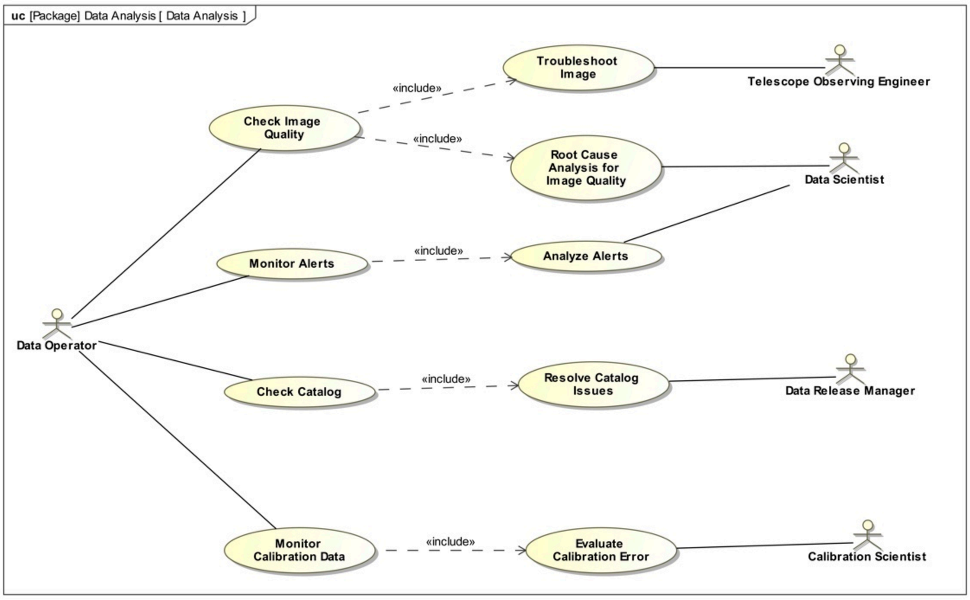

Figure 12. Data analysis OpsCon through use case diagram.

Activity diagrams are another type of behavior diagram used in this model to visualize different operational scenarios for a ground-based telescope system. Usually, an activity diagram starts with an initial node (a solid circle) to symbolize the start of an activity. The diagram connects different action types with control flows or object flows. Control flows are used to model the flow of control between actions, whereas object flows are used to model the flow of data. In Figure 13, an operational scenario on how to disable the telescope motions are depicted through an activity diagram.

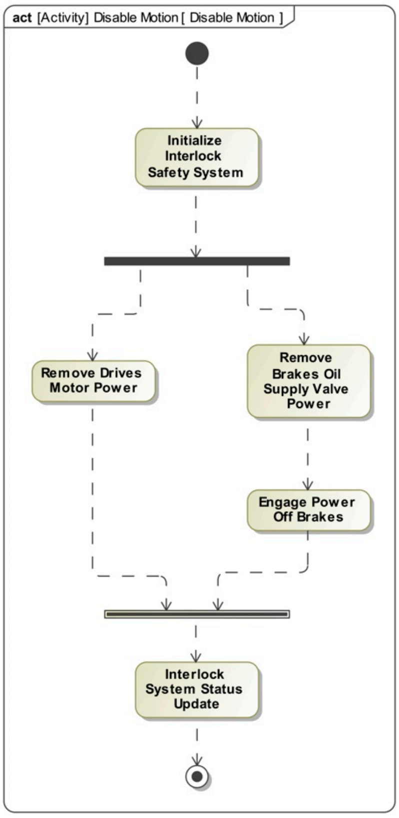

Figure 13. Activity diagram of "disable motion" scenario.

SysML state machine diagrams help to model different operational states and transitions between states triggered by events/signals/values, etc. State machine diagrams can be also used to define/build verification constraints of a system to verify the system requirements [76]. In Figure 14, different states and their transition paths with signals are displayed. For instance, when an *errrorIsTrue* signal is triggered, the system transitions to *FaultState* from *DisabledState* or *EnabledState*. If there is maintenance activity required, a *maintenance* signal will trigger a transition to *MaintenanceState* from *FaultState*.

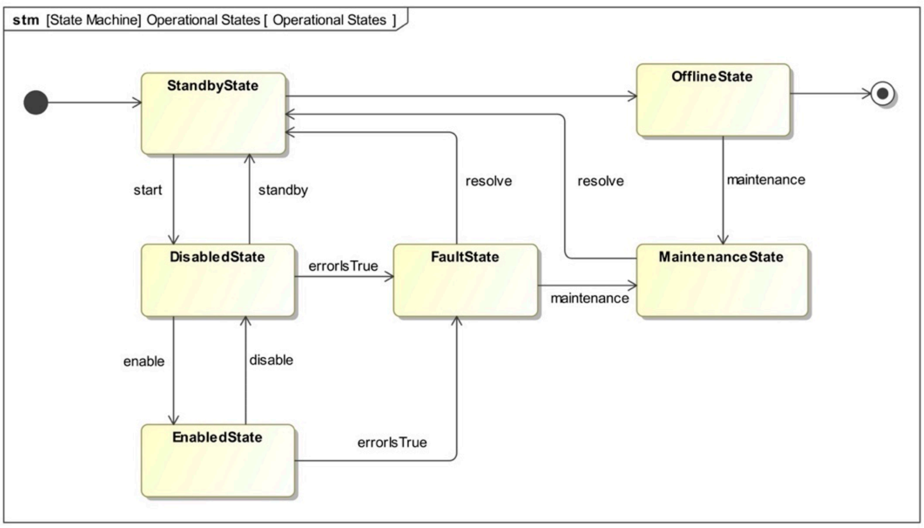

Figure 14. State machine showing operational states of the telescope mount system.

SysML parametric diagrams help the modeler to combine system models with engineering analysis models (e.g., performance, reliability, cost analysis, simulation). SysML activity diagrams with the aid of the *opaque* action can serve a similar purpose of integration with external analysis tools [77]. However, to serve large, complex system models, the using a parametric diagram is a more efficient way of trading with a high volume of data and calculations as compared to an activity diagram. In Figure 15, the authors incorporated a MATLAB function into a *constraint* block by dragging it into the parametric diagram. The function takes the *numObjects* (the number of space Objects to be created in a unity scene) variable as input and returns the *xPos* of a selected game object in the same unity scene by a user.

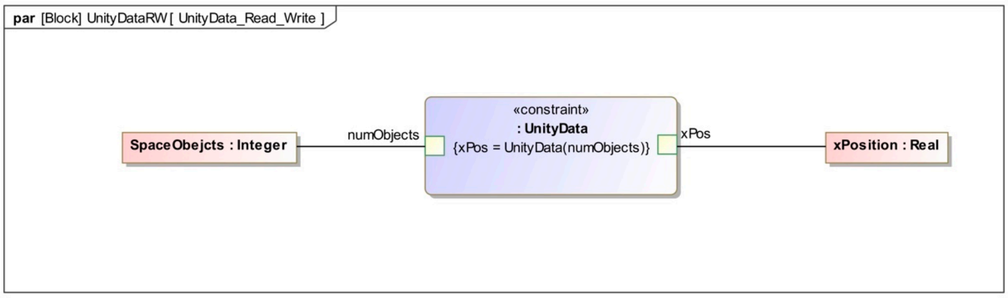

Figure 15. Parametric diagram for unity data exchange.

5. Integration with the VR Environment

In this research paper, a SysML system model is integrated with the VR environment in Unity using two different methods. First, the authors utilized CST in the CSM tool to simulate the parametric diagram in order to achieve data exchange between the model and the VR environment. The value properties *SpaceObjects* and *xPosition* are mapped to the *numObjects* and *xPos* variables from the MATLAB script, respectively (see Figure 15). Both the MATLAB function and Unity engine access the same excel file to read and write the data values being exchanged. As shown in Figure 16, the *SpaceObjects* value was set to 10 and the authors received a returned *xPosition* value of −2.1438 from the Unity scene. It is apparent from Figure 17 that C# script from Unity received the input from the model and generated 10 space objects in the Unity scene by instantiating the *Space Objects* prefab. Then, the *xPosition* value becomes populated in the CST console of the system model once the user selects a space object in the Unity scene by hovering the mouse over the object (see Figure 18). The value can be confirmed by comparing it to the unity console showing the xPosition value of the selected object (see Figure 19). The color of the object changes to red once the user selects a particular space object in the Unity scene (see Figure 18). Thus, two-way data exchange between the system model and the Unity environment is facilitated.

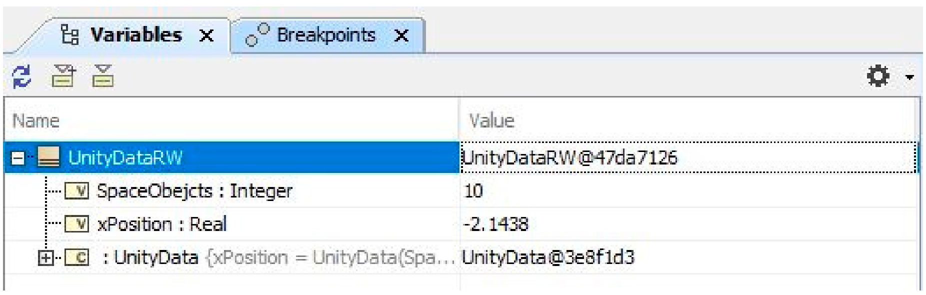

Figure 16. CST variables with values.

Figure 17. Objects created in the VR simulation scene.

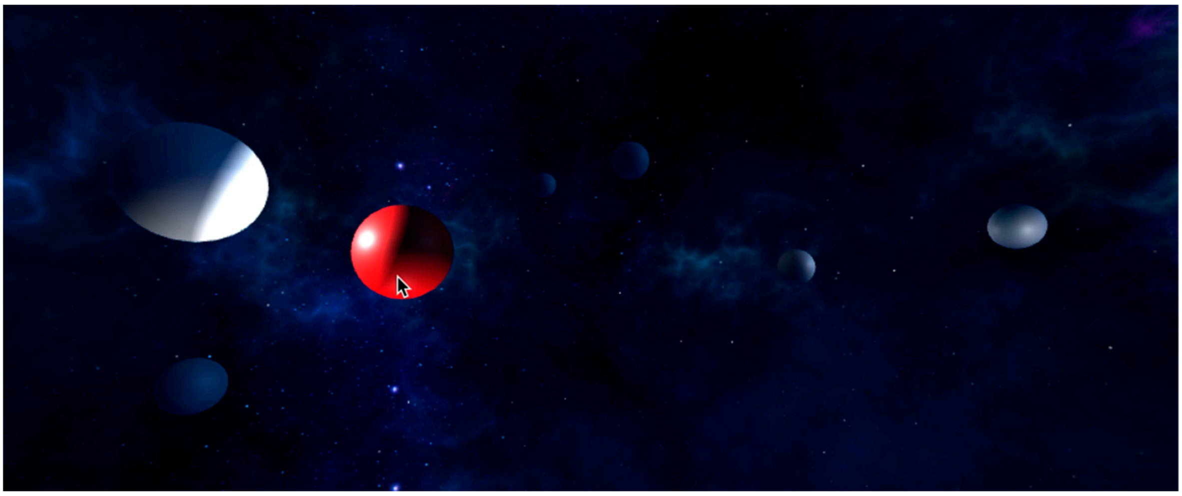

Figure 18. User selection of an object by hovering the mouse.

Figure 19. X-Position of the user selected object in the unity console.

Secondly, the authors used VTL and report wizard from CSM to export the SysML diagram components and their relationships to different Excel files which are shared with Unity C# scripts. The scripts written for different diagram types can take the information from the Excel files and generate visualization of the same diagram exported in the Excel files in the Unity scene. In Figure 20, the exact same BDD as modeled in the system model shown in Figure 10 can be seen. The authors used a Cube prefab to depict the blocks and cylinder prefabs to visualize the relationship in the Unity scene. Parent blocks are shown in a different color (cyan) compared to the child blocks (see Figure 20). Users can visualize the requirements diagram utilizing the same prefabs used in a BDD, as both are hierarchical diagrams. However, for a requirements diagram visualization, the authors only show the requirement details when the user selects a particular requirement by hovering the mouse over the selection to keep the diagram less cluttered (see Figure 21). This selection of a requirement also shows how users can design user interactions with diagrams inside the VR environment. Using a similar method, users can use different types of prefabs and organization patterns to visualize the different types of SysML diagrams in the VR environment (e.g., using human prefabs to depict actors in the use case diagram). Hence, with the aid of VTL and C# scripts attached to custom prefabs, users can visualize SysML diagrams inside Unity scenes to enable improved user interaction and communication.

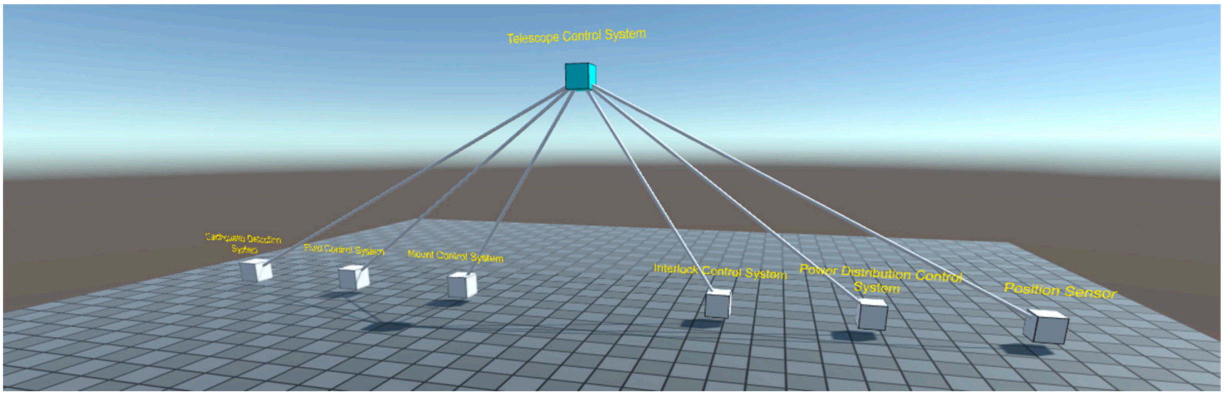

Figure 20. SysML BDD of the telescope control system in VR.

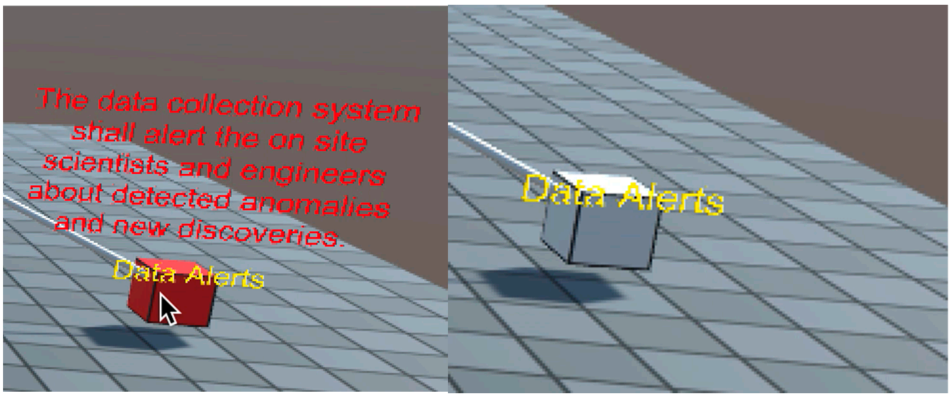

Figure 21. Requirement diagram showing requirement description on mouse hover.

6. Discussion

Telescope systems depend heavily on vast amounts of data collection and scenario visualizations to analyze different science aspects of existing space objects, consider the possibility of new discoveries and run educational programs for different institutions. So, integration of VR environments with the SysML system model will enable design/analysis of those visualization systems with reduced cost and effort, as alternative architectures can be evaluated within the system model. The system model developed in SysML will contribute to the literature on the application of MBSE in ground-based telescope systems. Model based system engineers will be able to use different SysML diagrams appropriately to model similar telescope systems. The system's integration capabilities with VR environments also facilitates future integration of DT technologies with MBSE tools. As DT's virtual components can be designed in a VR environment, SysML models can potentially communicate with the physical twin through the digital counterpart in the VR environment. Design/analysis parameters from SysML models can be fed into the VR environment. Then, those changes in VR environment will dictate the physical twin's behavior, as twins can be connected through data communication networks and sensors. In an alternative configuration, SysML models can be shared with both digital and physical twins, where the system receives sensor fed data from the physical twin and makes changes/updates in the digital twin or vice versa. As a majority of the VR environment tools are also equipped to develop AR/MR models, the proposed methodology can be expanded to enable integration of SysML models with AR/MR based DT infrastructure [55,78]. In addition, with the aid of predesigned prefab databases and ontological dictionaries, it will be possible to create DT scenes of more complex alternatives with multiple game objects seamlessly in the VR environment from the system model. Hence, the time, cost, and effort necessary to design the individual components of the whole MBSE-DT paradigm will be greatly reduced.

Early prototypes can be generated from the system model by utilizing the proposed methodology. Then, the responsible engineer can test the design of the planned system by analyzing the interaction results with the potential customer and other stakeholders. Finally, SysML diagrams are not easily comprehendible to all types of stakeholders involved in the development of a system. An end user may not have enough expertise to understand the different types of diagrams and their relationships within a SysML tool. By recreating the SysML diagrams in an interactive virtual environment, the learning curve present in understanding SysML models will be improved.

The developed methodology to integrate VR environments with SysML has few scopes for extension. The communication of SysML diagram visualizations in VR environments is implemented as one way, i.e., users can only update in the SysML model and then export for interactive visualization in the VR environment. However, the user in VR environments may want to change some blocks of a BDD and propose additional blocks to be added into the diagram. These updates can be exported from the VR scene and incorporated back to the system model upon approval of the systems engineer. Next, the authors have utilized Excel format for extracting the SysML diagrams to visualize in the VR environment. As researchers frequently use JSON and/or XML data interchange format as mediums for data interoperation between tools, the authors will consider these formats in the future expansion of this research. Moreover, since the scope of this research is to develop an easily replicable integration approach, the authors did not design the VR scenario generation scripts with the consideration of VR design parameters such as an end user's engagement level, fatigue, lighting conditions, etc. Finally, although the benefits of integration of VR and SysML are perceivable through the illustration of the proposed methodology and related studies (see introduction) discussed in this paper, formal measurement studies need to be performed to measure the benefits of the integrated MBSE approach.

7. Conclusions

This research paper demonstrated modeling of a ground-based telescope system utilizing the widely accepted MBSE language SysML. The system model was organized by the four pillars of SysML, which are the Requirements, Structure, Behavior and Parametrics aspects of the SOI. From a methodological viewpoint, this study is useful in illustrating step-by-step how SysML diagrams can be applied to model various design aspects of a telescope system's system artifacts including requirements, architectures, interfaces, use cases, operational concepts, operational scenarios, and system states. The system model was integrated with a virtual reality environment (Unity engine) to enable data exchange between SysML and VR scenarios through MATLAB and C#. Finally, the SysML diagrams were exported using VTL and recreated automatically through C# scripts as Unity scenes, where users can interact with the SysML diagrams. The proposed methodology fills the gap in the literature by achieving direct communication between SysML and VR environments through real time data exchange. In addition, unlike the previous efforts by the researchers, the proposed methodology did not involve intensive conversion of SysML components to multiple file types to facilitate the integration. In this research, the SysML diagrams visualized in the VR environment are more comprehendible and interactable. Hence, the developed integration methodology will be easy to understand and replicate by future users. The authors plan to expand upon this research in the future to overcome the limitations mentioned in the previous section. The capability to update SysML model diagrams from VR environments based on user feedback will be incorporated. In addition, the integration approach will be evaluated by analyzing the selected benefit types from the MBSE value and benefits review performed by Henderson and Salado [79].

Author Contributions: Conceptualization, M.L. and R.V.; methodology, M.L.; software, M.L.; validation, M.L. and R.V.; formal analysis, M.L.; investigation, M.L. and R.V.; resources, M.L.; data curation, M.L.; writing—original draft preparation, M.L.; writing—review and editing, M.L. and R.V.; visualization, M.L.; supervision, R.V.; project administration, M.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Data sharing not applicable—no new data generated data sharing does not apply to this article as no new data were created or analyzed in this study.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. INCOSE. System and SE Definitions. Available online: https://www.incose.org/about-systems-engineering/system-and-se-definition/system-and-se-definitions (accessed on 26 January 2023).
2. Hart, L. Introduction to model-based system engineering (MBSE) and SysML. In Proceedings of the Delaware Valley Chapter Meeting, Philadelphia, PA, USA, 26 April 2015; p. 43.
3. Wymore, L.A.W. *Model-Based Systems Engineering*; CRC Press: Boca Raton, FL, USA, 1993.
4. Honour, E. Systems Engineering and Complexity. *Insight* **2008**, *11*, 20–21. [CrossRef]
5. Calvano, C.N.; John, P. Systems engineering in an age of complexity. *Syst. Eng.* **2004**, *7*, 25–34. [CrossRef]
6. French, M.O. Extending model based systems engineering for complex systems. In Proceedings of the 53rd AIAA Aerospace Sciences Meeting, Kissimmee, FL, USA, 5–9 January 2015. [CrossRef]
7. Alvarez, J.L.; Metselaar, H.; Amiaux, J.; Criado, G.S.; Venancio, L.M.G.; Salvignol, J.-C.; Laureijs, R.J.; Vavrek, R. Model-based system engineering approach for the Euclid mission to manage scientific and technical complexity. *Model. Syst. Eng. Proj. Manag. Astron. VII* **2016**, *9911*, 99110C. [CrossRef]
8. Estefan, J.A. *Survey of Model-Based Systems Engineering (MBSE) Methodologies*; INCOSE MBSE Initiative: San Diego, CA, USA, 2008; p. 70.
9. Roques, P. MBSE with the ARCADIA method and the capella tool. In Proceedings of the 8th European Congress on Embedded Real Time Software and Systems (ERTS 2016), Toulouse, France, 27–29 January 2016. Available online: https://hal.archives-ouvertes.fr/hal-01258014 (accessed on 8 August 2021).
10. Object-Oriented SE Method, Default. Available online: https://www.incose.org/incose-member-resources/working-groups/transformational/object-oriented-se-method (accessed on 17 September 2021).
11. Free, Open Source & Commercial MBSE + SysML Tools—MBSE Tool Reviews. Available online: https://mbsetoolreviews.com/ (accessed on 17 September 2021).
12. Delligatti, SysML Distilled: A Brief Guide to the Systems Modeling Language | Pearson. Available online: https://www.pearson.com/us/higher-education/program/Delligatti-Sys-ML-Distilled-A-Brief-Guide-to-the-Systems-Modeling-Language/PGM259527.html (accessed on 8 August 2021).
13. Wolny, S.; Mazak, A.; Carpella, C.; Geist, V.; Wimmer, M. Thirteen years of SysML: A systematic mapping study. *Softw. Syst. Model.* **2020**, *19*, 111–169. [CrossRef]
14. Friedenthal, S.; Moore, A.; Steiner, R. *A Practical Guide to SysML: The Systems Modeling Language*; Morgan Kaufmann: Burlington, MA, USA, 2014.
15. OMG SysML Home | OMG Systems Modeling Language. Available online: https://www.omgsysml.org/ (accessed on 8 August 2021).
16. Erickson, J.; Siau, K. Unified modeling language: The teen years and growing pains. In *Human Interface and the Management of Information. Information and Interaction Design*; Yamamoto, S., Ed.; Springer: Berlin/Heidelberg, Germany, 2013; Volume 8016, pp. 295–304. [CrossRef]
17. What is SysML? | OMG SysML. Available online: https://www.omgsysml.org/what-is-sysml.htm (accessed on 26 June 2022).
18. Berni, A.; Borgianni, Y. Applications of Virtual Reality in Engineering and Product Design: Why, What, How, When and Where. *Electronics* **2020**, *9*, 1064. [CrossRef]
19. Williams, E.; Love, C.; Love, M. *Virtual Reality Cinema: Narrative Tips and Techniques*; Routledge: England, UK, 2021.
20. Dowling, D.; Fearghail, C.O.; Smolic, A.; Knorr, S. Faoladh: A case study in cinematic VR storytelling and production. In Proceedings of the Interactive Storytelling: 11th International Conference on Interactive Digital Storytelling, ICIDS 2018, Dublin, Ireland, 5–8 December 2018; Proceedings 11. pp. 359–362.
21. Haluck, R.S. Computers and Virtual Reality for Surgical Education in the 21st Century. *Arch. Surg.* **2000**, *135*, 786–792. [CrossRef]
22. Rivas, Y.C.; Valdivieso, P.A.V.; Rodriguez, M.A.Y. Virtual reality and 21st century education. *Int. Res. J. Manag. IT Soc. Sci.* **2020**, *7*, 37–44. [CrossRef]
23. Ma, M.; Jain, L.C.; Anderson, P. Future trends of virtual, augmented reality, and games for health. In *Virtual, Augmented Reality and Serious Games for Healthcare 1*; Ma, M., Jain, L.C., Anderson, P., Eds.; Springer: Berlin/Heidelberg, Germany, 2014; pp. 1–6. [CrossRef]

24. Papanastasiou, G.; Drigas, A.; Skianis, C.; Lytras, M.; Papanastasiou, E. Virtual and augmented reality effects on K-12, higher and tertiary education students' twenty-first century skills. *Virtual Real.* **2019**, *23*, 425–436. [CrossRef]
25. Wang, P.; Wu, P.; Wang, J.; Chi, H.L.; Wang, X. A critical review of the use of virtual reality in construction engineering education and training. *Int. J. Environ. Res. Public Health* **2018**, *15*, 1204. [CrossRef]
26. Dinis, F.M.; Guimarães, A.S.; Carvalho, B.R.; Martins, J.P.P. Virtual and augmented reality game-based applications to civil engineering education. In Proceedings of the 2017 IEEE Global Engineering Education Conference (EDUCON), Athens, Greece, 26–28 April 2017; pp. 1683–1688. [CrossRef]
27. Hilfert, T.; König, M. Low-cost virtual reality environment for engineering and construction. *Vis. Eng.* **2016**, *4*, 2. [CrossRef]
28. Bai, C.; Dallasega, P.; Orzes, G.; Sarkis, J. Industry 4.0 technologies assessment: A sustainability perspective. *Int. J. Prod. Econ.* **2020**, *229*, 107776. [CrossRef]
29. Salah, B.; Abidi, M.H.; Mian, S.H.; Krid, M.; Alkhalefah, H.; Abdo, A. Virtual Reality-Based Engineering Education to Enhance Manufacturing Sustainability in Industry 4.0. *Sustainability* **2019**, *11*, 1477. [CrossRef]
30. Stone, R.J.; Panfilov, P.B.; Shukshunov, V.E. Evolution of aerospace simulation: From immersive Virtual Reality to serious games. In Proceedings of the 5th International Conference on Recent Advances in Space Technologies—RAST2011, Istanbul, Turkey, 9–11 June 2011; pp. 655–662. [CrossRef]
31. Fussell, S.G.; Truong, D. Using virtual reality for dynamic learning: An extended technology acceptance model. *Virtual Real.* **2022**, *26*, 249–267. [CrossRef]
32. Lee, H.; Woo, D.; Yu, S. Virtual Reality Metaverse System Supplementing Remote Education Methods: Based on Aircraft Maintenance Simulation. *Appl. Sci.* **2022**, *12*, 2667. [CrossRef]
33. Steuer, J. Defining Virtual Reality: Dimensions Determining Telepresence. *J. Commun.* **1992**, *42*, 73–93. [CrossRef]
34. Biocca, F. Virtual Reality Technology: A Tutorial. *J. Commun.* **1992**, *42*, 23–72. [CrossRef]
35. Maheepala, M.; Kouzani, A.Z.; Joordens, M.A. Light-Based Indoor Positioning Systems: A Review. *IEEE Sens. J.* **2020**, *20*, 3971–3995. [CrossRef]
36. Bamodu, O.; Ye, X.M. Virtual Reality and Virtual Reality System Components. *Adv. Mater. Res.* **2013**, *765–767*, 1169–1172. [CrossRef]
37. Patterson, E.A. *Utilizing SysML Viewpoints to Improve Understanding and Communication of Human Mental Models Within System Design Teams*; The University of Alabama in Huntsville: Huntsville, AL, USA, 2017.
38. Karban, R.; Jankevičius, N.; Elaasar, M. Esem: Automated systems analysis using executable SysML modeling patterns. In Proceedings of the INCOSE International Symposium, Edinburgh, Scotland, 18–21 July 2016; Volume 26, Number 1. pp. 1–24.
39. Karban, R.; Dekens, F.G.; Herzig, S.; Elaasar, M.; Jankevičius, N. Creating system engineering products with executable models in a model-based engineering environment. In Proceedings of the Modeling, Systems Engineering, and Project Management for Astronomy VII, Edinburgh, UK, 26–28 June 2016; Volume 9911, pp. 96–111.
40. Tea, S.; Panuwatwanich, K.; Ruthankoon, R.; Kaewmoracharoen, M. Multiuser immersive virtual reality application for real-time remote collaboration to enhance design review process in the social distancing era. *J. Eng. Des. Technol.* **2022**, *20*, 281–298. [CrossRef]
41. Wolfartsberger, J. Analyzing the potential of Virtual Reality for engineering design review. *Autom. Constr.* **2019**, *104*, 27–37. [CrossRef]
42. Shao, X.; Yuan, Q.; Qian, D.; Ye, Z.; Chen, G.; Le Zhuang, K.; Jiang, X.; Jin, Y.; Qiang, D. Virtual reality technology for teaching neurosurgery of skull base tumor. *BMC Med. Educ.* **2020**, *20*, 3. [CrossRef] [PubMed]
43. Brown, C.E.; Alrmuny, D.; Williams, M.K.; Whaley, B.; Hyslop, R.M. Visualizing molecular structures and shapes: A comparison of virtual reality, computer simulation, and traditional modeling. *Chem. Teach. Int.* **2020**, *3*, 69–80. [CrossRef]
44. Ferrell, J.B.; Campbell, J.P.; McCarthy, D.R.; McKay, K.T.; Hensinger, M.; Srinivasan, R.; Zhao, X.; Wurthmann, A.; Li, J.; Schneebeli, S.T. Chemical Exploration with Virtual Reality in Organic Teaching Laboratories. *J. Chem. Educ.* **2019**, *96*, 1961–1966. [CrossRef]
45. Pöhler, L.; Schuir, J.; Lübbers, S.; Teuteberg, F. Enabling collaborative business process elicitation in virtual environments. In Proceedings of the Business Modeling and Software Design: 10th International Symposium, BMSD 2020, Berlin, Germany, 6–8 July 2020; Proceedings 10. pp. 375–385.
46. Leyer, M.; Brown, R.; Aysolmaz, B.; Vanderfeesten, I.; Turetken, O. 3D virtual world BPM training systems: Process gateway experimental results. In Proceedings of the Advanced Information Systems Engineering: 31st International Conference, CAiSE 2019, Rome, Italy, 3–7 June 2019; Proceedings 31. pp. 415–429.
47. West, S.; Brown, R.; Recker, J. Collaborative business process modeling using 3D virtual environments. In Proceedings of the 16th Americas Conference on Information Systems: Sustainable IT Collaboration around the Globe, Lima, Peru, 12–15 August 2010; pp. 1–11.
48. Oberhauser, R. VR-UML: The unified modeling language in virtual reality—An immersive modeling experience. In Proceedings of the Business Modeling and Software Design: 11th International Symposium, BMSD 2021, Sofia, Bulgaria, 5–7 July 2021; Proceedings 11. pp. 40–58.
49. Reuter, R.; Hauser, F.; Muckelbauer, D.; Stark, T.; Antoni, E.; Mottok, J.; Wolff, C. Using augmented reality in software engineering education? First insights to a comparative study of 2D and AR UML modeling. In Proceedings of the 52nd Hawaii International Conference on System Sciences, Grand Wailea, HI, USA, 8–11 January 2019.
50. Grieves, M. Digital twin: Manufacturing excellence through virtual factory replication. *White Pap.* **2014**, *1*, 1–7.

51. Shafto, M.; Conroy, M.; Doyle, R.; Glaessgen, E.; Kemp, C.; LeMoigne, J.; Wang, L. Draft modeling, simulation, information technology & processing roadmap. *Natl. Aeronaut. Space Adm.* **2012**, *2010*, 11.
52. Barricelli, B.R.; Casiraghi, E.; Fogli, D. A Survey on Digital Twin: Definitions, Characteristics, Applications, and Design Implications. *IEEE Access* **2019**, *7*, 167653–167671. [CrossRef]
53. Singh, M.; Fuenmayor, E.; Hinchy, E.P.; Qiao, Y.; Murray, N.; Devine, D. Digital Twin: Origin to Future. *Appl. Syst. Innov.* **2021**, *4*, 36. [CrossRef]
54. Kaarlela, T.; Pieskä, S.; Pitkäaho, T. Digital twin and virtual reality for safety training. In Proceedings of the 2020 11th IEEE International Conference on Cognitive Infocommunications, (CogInfoCom), Mariehamn, Finland, 23–25 September 2020; pp. 000115–000120.
55. Tu, X.; Autiosalo, J.; Jadid, A.; Tammi, K.; Klinker, G. A Mixed Reality Interface for a Digital Twin Based Crane. *Appl. Sci.* **2021**, *11*, 9480. [CrossRef]
56. Sepasgozar, S.M.E. Digital Twin and Web-Based Virtual Gaming Technologies for Online Education: A Case of Construction Management and Engineering. *Appl. Sci.* **2020**, *10*, 4678. [CrossRef]
57. Choi, S.H.; Park, K.-B.; Roh, D.H.; Lee, J.Y.; Mohammed, M.; Ghasemi, Y.; Jeong, H. An integrated mixed reality system for safety-aware human-robot collaboration using deep learning and digital twin generation. *Robot. Comput. Manuf.* **2022**, *73*, 102258. [CrossRef]
58. Wang, K.-J.; Lee, Y.-H.; Angelica, S. Digital twin design for real-time monitoring—A case study of die cutting machine. *Int. J. Prod. Res.* **2021**, *59*, 6471–6485. [CrossRef]
59. Liu, Y.K.; Ong, S.K.; Nee, A.Y.C. State-of-the-art survey on digital twin implementations. *Adv. Manuf.* **2022**, *10*, 1–23. [CrossRef]
60. Pérez, L.; Rodríguez-Jiménez, S.; Rodríguez, N.; Usamentiaga, R.; García, D.F. Digital Twin and Virtual Reality Based Methodology for Multi-Robot Manufacturing Cell Commissioning. *Appl. Sci.* **2020**, *10*, 3633. [CrossRef]
61. Havard, V.; Jeanne, B.; Lacomblez, M.; Baudry, D. Digital twin and virtual reality: A co-simulation environment for design and assessment of industrial workstations. *Prod. Manuf. Res.* **2019**, *7*, 472–489. [CrossRef]
62. Brusa, E. Digital Twin: Toward the Integration Between System Design and RAMS Assessment Through the Model-Based Systems Engineering. *IEEE Syst. J.* **2021**, *15*, 3549–3560. [CrossRef]
63. Madni, A.M.; Madni, C.C.; Lucero, S.D. Leveraging Digital Twin Technology in Model-Based Systems Engineering. *Systems* **2019**, *7*, 7. [CrossRef]
64. Bickford, J.; Van Bossuyt, D.L.; Beery, P.; Pollman, A. Operationalizing digital twins through model-based systems engineering methods. *Syst. Eng.* **2020**, *23*, 724–750. [CrossRef]
65. Kande, A. Integration of Model-Based Systems Engineering and Virtual Engineering Tools for Detailed Design. Master's Thesis, Missouri University of Science and Technology, Rolla, MO, USA, 2011; p. 79.
66. Madni, A.M. Expanding Stakeholder Participation in Upfront System Engineering through Storytelling in Virtual Worlds. *Syst. Eng.* **2015**, *18*, 16–27. [CrossRef]
67. Madni, A.M.; Nance, M.; Richey, M.; Hubbard, W.; Hanneman, L. Toward an Experiential Design Language: Augmenting Model-based Systems Engineering with Technical Storytelling in Virtual Worlds. *Procedia Comput. Sci.* **2014**, *28*, 848–856. [CrossRef]
68. Abidi, M.A.; Lyonnet, B.; Chevaillier, P.; Toscano, R. Contribution of virtual reality for lines production's simulation in a lean manufacturing environment. *Simulation* **2016**, *9*, 11. [CrossRef]
69. Mahboob, A.; Weber, C.; Husung, S.; Liebal, A.; Krömker, H. Model based systems engineering (MBSE) approach for configurable product use-case scenarios in virtual environments, DS 87-3. In Proceedings of the 21st International Conference on Engineering Design (ICED 17) Vol. 3: Product, Services and Systems Design, Vancouver, BC, Canada, 21–25 August 2017.
70. Sanvordenker, R. Visualization and Testing of an Autonomously Driving Truck's Sysml Models in a Virtual 3D Simulation Environment. Master's Thesis, Eindhoven University of Technology, Eindhoven, The Netherlands, 13 August 2020.
71. Oberhauser, R. VR-SysML: SysML model visualization and immersion in virtual reality. In Proceedings of the International Conference of Modern Systems Engineering Solutions (MODERN SYSTEMS 2022), IARIA, Nice, France, 24–28 July 2022; pp. 59–64.
72. No Magic Inc. Cameo Systems Modeler. Available online: https://www.nomagic.com/products/cameo-systems-modeler (accessed on 19 June 2020).
73. Cameo Simulation Toolkit—CATIA—Dassault Systèmes®. Available online: https://www.3ds.com/products-services/catia/products/no-magic/addons/cameo-simulation-toolkit/ (accessed on 16 August 2021).
74. Unity Technologies. Unity Real-Time Development Platform | 3D, 2D VR & AR Visualizations. Available online: https://unity.com/ (accessed on 19 June 2020).
75. Apache Velocity Engine—User Guide. Available online: https://velocity.apache.org/engine/1.7/user-guide.html (accessed on 27 January 2023).
76. Anwar, M.W.; Rashid, M.; Azam, F.; Kashif, M.; Butt, W.H. A model-driven framework for design and verification of embedded systems through SystemVerilog. *Des. Autom. Embed. Syst.* **2019**, *23*, 179–223. [CrossRef]
77. Lutfi, M.; Valerdi, R. Executable modeling of a cubesat-based space situational awareness system. In *Recent Trends and Advances in Model Based Systems Engineering*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 475–484. [CrossRef]

78. Koca, B.A.; Çubukçu, B.; Yüzgeç, U. Augmented reality application for preschool children with unity 3D platform. In Proceedings of the 3rd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), Ankara, Turkey, 11–13 October 2019; pp. 1–4.
79. Henderson, K.; Salado, A. Value and benefits of model-based systems engineering (MBSE): Evidence from the literature. *Syst. Eng.* **2021**, *24*, 51–66. [CrossRef]