


Article

SysML4GDPSim: A SysML Profile for Modeling Geometric Deviation Propagation in Multistage Manufacturing Systems Simulation

Sergio Benavent-Nácher * , Pedro Rosado Castellano and Fernando Romero Subirón

Department of Industrial Systems Engineering and Design, Universitat Jaume I. Av. Vicent Sos Baynat, s/n, 12006 Castellón de la Plana, Spain; rosado@uji.es (P.R.C.); fromero@uji.es (F.R.S.)

* Correspondence: benavens@uji.es

Abstract: In recent years, paradigms like production quality or zero-defect manufacturing have emerged, highlighting the need to improve quality and reduce waste in manufacturing systems. Although quality can be analyzed from various points of view during different stages of a manufacturing system's lifecycle, this research focuses on a multidomain simulation model definition oriented toward the analysis of productivity and geometric quality during early design stages. To avoid inconsistencies, the authors explored the definition of descriptive models using system modeling language (SysML) profiles that capture domain-specific semantics defining object constraint language (OCL) rules, facilitating the assurance of model completeness and consistency regarding this specific knowledge. This paper presents a SysML profile for the simulation of geometric deviation propagation in multistage manufacturing systems (SysML4GDPSim), containing the concepts for the analysis of two data flows: (a) coupled discrete behavior simulation characteristic of manufacturing systems defined using discrete events simulation (DEVS) formalism; and (b) geometric deviation propagation through the system based on the geometrical modeling of artifacts using concepts from the topologically and technologically related surfaces (TTRS) theory. Consistency checking for this type of multidomain simulation model and the adoption of TTRS for the mathematical analysis of geometric deviations are the main contributions of this work, oriented towards facilitating the collaboration between design and analysis experts in the manufacturing domain. Finally, a case study shows the application of the proposed profile for the simulation model of an assembling line, including the model's transformation to Modelica and some experimental results of this type of analysis.

Keywords: MBSE; SysML; model consistency; manufacturing system simulation; geometric deviation analysis; TTRS



Citation: Benavent-Nácher, S.; Rosado Castellano, P.; Romero Subirón, F. SysML4GDPSim: A SysML Profile for Modeling Geometric Deviation Propagation in Multistage Manufacturing Systems Simulation. *Appl. Sci.* **2024**, *14*, 1830. <https://doi.org/10.3390/app14051830>

Academic Editors: Zoran Jurković, David Ištoković and Janez Gotlih

Received: 18 January 2024

Revised: 20 February 2024

Accepted: 20 February 2024

Published: 23 February 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In recent years, manufacturing systems design, in alignment with initiatives like Industry 4.0, has promoted different operational paradigms like production quality [1] or zero-defect manufacturing [2] to address quality improvement and waste reduction. The production quality bases propose an integration of quality, productivity, and maintenance evaluation, an orientation adopted in this paper to analyze, jointly, production and quality. Among product quality characteristics, this work focuses on the geometric quality of both manufacturing resources and processed products, combined with productivity analysis using multidomain simulation model during early design stages.

During the design and development of a system, and especially during the verification and validation stages of the adopted solution principles, the development of mathematical models enables the analysis of the referent system's behavior and performance. In the case of simple systems, an analytical solution can usually be obtained, but complex systems analysis, such as of manufacturing systems, usually requires the development

of simulation software systems to support their high uncertainty and/or nonlinearity, combining quantitative/qualitative features and continuous/discrete behaviors [3]. Although specific tools can be used for the simulation of certain domains (including the digital manufacturing tools focused on the manufacturing systems domain), the adoption of generic and/or standardized tools and languages (e.g., Modelica) facilitates the modeling of multi-domain simulation systems during initial design stages, an orientation aligned with model-based system engineering principles that promote model integration and consistency assurance [4]. Moreover, simulation models have been a widely explored solution, as evidenced in the increase in the research works in this field [5].

In this context, the capability of generic simulation languages (e.g., Modelica) to verify and validate simulation models is limited to assure the syntactic correctness of the models; for example, checking that the set of variables and mathematical relationships is sufficient to obtain a solution. An interesting alternative to overcome these limitations is the adoption of domain-specific modeling languages (DSML) [6], which include the specific semantics of a studied domain, which would enable the model consistency validation based on these specific semantics. Generic descriptive modeling languages such as SysML [7] support the development of DSML, defining specific profiles that extend the language by adding stereotypes and detailing the semantics through OCL expressions [8]. OCL defines a set of functions to define queries on the model without side effects. The results of these queries are processed by logical functions to check the accomplishment of specific conditions (invariants). Their implementation as part of stereotypes definition at the profile level enables the evaluation of stereotyped model constructions and the detection of inconsistencies. In addition to the creation of profiles, the use of SysML and the adoption of a systems modeling approach also allows for the linking of the simulation models with other models of the referent system (e.g., specification or design models), checking consistency between models to support the collaboration between design and analysis experts. The design and validation of the simulation model using SysML have been studied in previous works, such as [9]. This work also addresses the subsequent model transformation to executable models using languages like Modelica by applying the SysML4Modelica profile [10]. A similar transformation mechanism is proposed in [11], also based on the application of an SysML profile, to enable an automated transformation. Other authors have explored the use of SysML activity diagrams to describe workflows for connecting simulation and design models and facilitating collaboration between system architects and analysis experts [12,13].

In the manufacturing domain, there is a lack of developments adopting this approach, and it is hard to find works focused on geometric deviations and their propagation in multi-stage systems. In [14], SysML is used for the preliminary design of a multidomain simulation system that integrates the analysis of productivity and geometric quality to study how different control strategies influence performance measures. However, all the previously mentioned works use SysML as a modeling language only to define the descriptive models facilitating communication between engineers.

The present paper explores the SysML's capability to define DSMLs as SysML profiles, including domain-specific semantics to develop simulation software systems. Specifically, this research proposes a language with which to define simulation models for manufacturing systems, focused on a joint analysis of productivity and geometric quality performance. The proposed SysML profile (geometric deviation propagation Simulation—SysML4GDPSim) integrates the discrete behavioral aspects of a manufacturing system [15] and topologically and technologically related surfaces (TTRS) [16] concepts to model the geometrical quality characteristics of the products.

The content and structure of this document is summarized as follows. Section 2 briefly discusses the complexity and main characteristics of the supersystem to be analyzed (including the interactions between the product, the process, and the resources), the simulation systems for their analysis, and the deviated geometry modeling to support quality analysis. Section 3 is dedicated to the proposed SysML profile, presenting the metamodel description, the implementation of the profile, and its application during library model

definition. Section 4 presents a case study to exemplify the application of the proposal and validate its suitability in the context of manufacturing simulation. Finally, Section 5 presents a final discussion and Section 6 summarizes some conclusions and future works.

2. Modeling and Simulation of Product–Process–Resource Systems

A simulation system model is the model of (executable) software that allows for the performance analysis of a referent system in response to different conditions derived from the properties of the referent system and its environment. Simulation models are usually developed for complex referent systems. In the case of this research, the referent system is a system of systems, that is, a system that is made up of systems, according to the definition of [17]. This concept is also named a “supersystem” in other references, like [18]. Specifically, the term supersystem is adopted in [19] to express a temporal construction to describe the production lifecycle phase of a product, supporting links with other elements like manufacturing resources. In this paper, this orientation is adopted to define a system composed of resources (the manufacturing assets), the processes executed by these resources, and the manufactured products. All these elements together compound the PPR system that has been studied in previous works on manufacturing domain modeling, like [20]. In order to simulate PPR systems, it is necessary to acquire deep knowledge of the referent system and its representation in the simulation system, including all the aspects that have influence in the proposed analysis.

2.1. PPR Systems Basis

This research is focused on the study of discrete manufacturing systems that support multistage processes, with reconfigurable and automated resources. Therefore, the manufacturing system is composed of configurable resources supporting various manufacturing processes with small changes. In this context, a resource is any mechatronic system that plays a role in the production of goods (products).

Resources in a manufacturing system are structured in a multilevel hierarchy in which simple resources are part of complex ones. In fact, the manufacturing system as a whole is a complex resource. The modeling of the manufacturing system requires the establishment of an adequate decomposition level to support the analysis goals. Considering the interests of this research, this atomic level, according to the classification proposed in [21], is the workstation level. A workstation is a mechatronic system with a behavior that directly influences the final geometry of the product. In a broader sense, the term resource is also applied to simpler elements such as fixtures, tools, etc., but in this work, these elements are treated as physical artifacts without their own behavior, and they will be considered for the geometrical modeling of process assemblies without granting them the category of resource.

Moreover, resources in a manufacturing system can be classified into processing and control resources. The processing resources change the properties of the processed batches, while control resources merely monitor the batches and make decisions to improve their properties. Processing resources can be classified into: (a) logistical resources that modify the batch properties as a unit; and (b) transformer resources that modify the properties (especially geometric features) of individual products of the batch.

The transformer resources (mainly assembling and machining workstations) can support several configurations to execute alternative manufacturing processes. In this case, it is necessary to consider a setup stage (before the processing) to introduce the necessary changes (e.g., changes in fixtures or tools, loading an NC code, etc.).

Another basic element of the PPR system is the product to be manufactured. The product type can have representations from different viewpoints and degrees of complexity. In this research, products are represented as entities with the necessary attributes to describe the geometric deviations from the product specification. These products are grouped in batches which have their own properties, such as size, start time of manufacturing, waiting or storage times, processing duration, etc.

The third element of the PPR system is the manufacturing process. Each product type has a generic process plan, defining the necessary manufacturing processes, and a native process plan which details the specific resource assigned for each process stage. Although the process plan can be established with different levels of detail, in this research, the selected atomic level is the subphase, which aggregates all the operations executed in a machine using the same clamping (part-fixture assembly).

Each subphase of the native process plan establishes an interaction between the product (in a certain state) and specific resources of the manufacturing system. This research is focused on the physical interactions between the product and the manufacturing system that produce the new geometric features of the product, as studied in [22]. Specifically, fixture–workpiece–machine interactions are studied in [23] in order to characterize the geometric deviations on these types of assembly. This type of construction is also promoted in [24], which is focused on inspection planning. In this paper, this assembly model is named “process assembly”, and it is defined to support the analysis of the geometric deviations resulting from the execution of a certain manufacturing subphase. The propagation of these deviations to subsequent stages must be simulated in parallel to the material flow. In this way, the geometric quality of the final product and the indicators related to PPR system productivity can be jointly assessed, as promoted by the production quality paradigm.

2.2. Simulation System Modeling

As mentioned above, a simulation system is a soft system that emulates the behavior of a referent systems to analyze its response to certain conditions. These conditions are generally defined by a set of parameters whose values are assigned for each experiment in order to quantify the influence of conditions on the performance metrics. Simulation models are built from a set of software objects that share data through connecting ports and perform calculations. These objects are permanent entities, called “resources” in simulation [25], that have existence throughout simulated time (i.e., their life is not finite). This characteristic is one of the main differences compared to flow units (also called “entities” in [25]), which are elements that flow through the system and have a finite life (they are created and destroyed at certain moments). In PPR system simulation, permanent entities primarily emulate manufacturing system resources, while flow units represent individual products, product batches, manufacturing orders, etc.

Moreover, permanent entities can include the definition of a certain behavior, expressed in the form of algorithms and mathematical equations. In the scope of this research, different behaviors are identified corresponding to the typical functional units that participate in manufacturing the system simulation, supporting both the simulation of the referent system and its environment and other functionalities specific to the proposed analysis. These main functional units address issues such as the following:

- Generation of flow units based on a defined schedule or behavior: Objects with this functionality represent the beginning of the flow, and they have ports with which to send the data of the generated flow units.
- Processing flow units: Objects with this functionality have at least two ports with which to send and receive flow units. Moreover, a certain behavior can introduce changes in the data of the flow units. In the simulation of PPR systems, this function is mainly linked to the emulation of transforming resources (e.g., assembly or machining workstations) and logistic resources (stores or transportation resources).
- Destruction of flow units: Objects with this functionality represents the end point of the flow, where flow units are destroyed. These objects, generally named sinks, must have at least one port to receive flow units.
- Monitoring data and decisions making: Objects with this functionality have ports with which to receive data about key performance information. If they process data to make control decisions, they have ports with which to send data about these decisions. In a PPR systems simulation, these objects are linked to the control resources representation.

All these types of functional units are characterized by a discrete behavior described by the principles of the discrete event system (DEVS) formalism [26]. According to DEVS, the behavior of a complex system (coupled behavior) emerges from the behavior of its simplest components and their interactions. In the same way, the complex behavior simulation of the PPR system emerges from the simpler behavior simulation of its components. Although there are various ways to represent this type of discrete behavior, one of the most common alternatives is the use of state machines, where a set of system states are defined and linked by transitions that are triggered by certain events. In addition, detailed behaviors can be defined when a component enters, exits, or stays in each state.

2.3. Representation and Calculation of Geometric Deviations

All the components of the process assembly (products, fixtures, tools, etc.) are imperfect realities whose geometric elements have deviated from the nominal ones specified in their designs. The analysis of these geometric deviations is mainly supported by the study of the interactions between product and resource devices during each single process. The representation of geometric deviations has been addressed in different specific and standardized languages (ISO, Ansi, etc.) to represent the maximum allowable deviations, specified as tolerance zones. Beyond tolerance representation, different mathematical models have been also proposed to deal with tolerance analysis, such as the Jacobian–Torsor model [27], the small displacement torsor [28], T-map [29], or matrix transformation [30], among others summarized in [31].

SysML models have also been used for tolerance representation, although they are not as widespread. In [32], TTRS concepts are adopted to describe the tolerance specifications with a SysML model, but it does not include the mathematical constructions to support a quantitative analysis. To overcome this limitation, [33] presents the SysML for tolerance analysis (SysML4TA) profile, which defines a DSML to support the TTRS-based mathematical characterization of geometric surfaces and their intra- and inter-part relationships. The work in question also details the mathematical operations necessary to compute deviations and verify the specification fulfilment. In [34], this orientation is adopted for developing a Modelica library for the geometric analysis of mechanical assemblies or parts. These works also highlight the compatibility of TTRS with geometric dimensioning and tolerancing (GD&T) standards [35], allowing the specification of parts to be addressed and unambiguously transferring the dimensioning scheme to a mathematical formulation suitable for simulation.

This paper adopts the principles and concepts formulated in a SysML4TA profile [33] to support the geometric modeling of the process assemblies necessary to solve each single-stage problem (subphase analysis). Additionally, the simulation system must have sufficient elements to transmit the resulting product geometric data to subsequent stages, as commented on in Section 3.

3. Proposed SysML4GDPSim Profile

This section presents the SysML4GDPSim profile, first describing the concepts and relationships considered in the metamodel of the developed language, depicted using some conceptual diagrams with UML notation. The Section 3.2 introduces some brief notes on the profile implementation, whose stereotypes are described in detail in Appendix A. Finally, the Section 3.3 briefly describes the SysML libraries developed to facilitate the modeling of the simulation systems under study and their transformation to Modelica executable models.

3.1. Metamodel Description

Assuming many of the concepts and bases discussed in Section 2, this subsection presents the metamodel description, commenting on the concepts related to the basic structure of the simulation system, the representation of the manufacturing system and the products, and finally, the artifacts modeling for the analysis of geometric quality.

3.1.1. Simulation System Basic Structure

According to the bases commented on in Section 2.2, Figure 1 represents the basic components of the simulation system (*SimulationSystem4MS*), which simulates the behavior of a manufacturing system (*MS_sim*) in a specific environment (defined by the rest of the elements of the simulation system). Before the description of the *MS_sim*, which constitutes the central component of the simulation system, the different concepts that define the simulation environment supporting specific functionalities are presented below:

- *Configurator* identifies the component that groups all the characteristic parameters and offers them to the other components (*MS_sim*, *InputGenerator*, *OutputCollector*, etc.) through the reference relationships shown in Figure 1.
- *InputGenerator* represents components that create the *FlowUnits*, representing material supply (inputs for the *MS_sim*). An *InputGenerator* is the initial point of the data flow that represents the logistics flow.
- *OutputCollector* represents components that destroy *FlowUnits*; so, the *OutputCollector* is a sink or end point of the data flow that represents the materials flow.
- *ResultsManager* represents components that collect and process the information of simulation executions to compute the desired performance metrics.

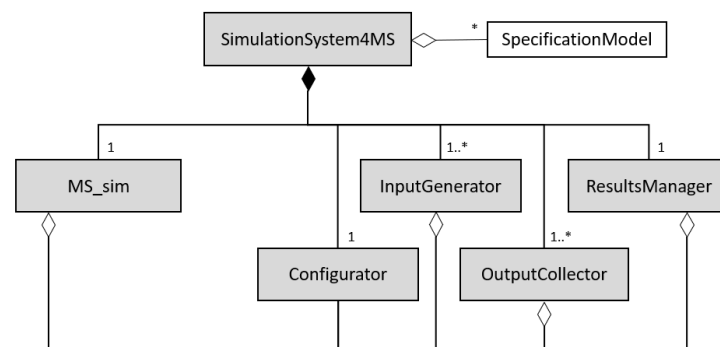


Figure 1. Abstract syntax of simulation systems. The notations “1” and “1..*” are multiplicities (“only one” and “one or more” respectively).

As represented in Figure 1, a *SimulationSystem4MS* is composed of at least one *MS_sim*, a *Configurator*, a *ResultsManager*, and at least one *InputGenerator* and *OutputCollector*. Moreover, the *SimulationSystem4MS* can have an aggregation relationship with the referent system specification to represent the necessary collaboration between the design and analysis tasks and facilitate the assurance of consistency between both models.

3.1.2. The Manufacturing System in the Simulation System

As mentioned, the *MS_sim* identifies the central component of the simulation system that emulates the behavior of the manufacturing system, transforming the input materials to obtain the output products with different geometrical properties. As represented in Figure 2, the *MS_sim* is a complex transformer resource (*TransformResource_sim*) because material units (a) flow through the resource (*ProcessingResource_sim* specialization) and (b) are transformed (i.e., there are changes in their physical characteristics).

A manufacturing system (and its simulation system) involves different types of resources; so, the metamodel includes different concepts as represented in Figure 2 and briefly described below:

- *ManufResource_sim* (abstract) represents any resource of the simulated manufacturing system, i.e., both processor and control resources, *ProcessingResource_sim* and *ControlResource_sim*, respectively.
- *ProcessingResource_sim* is a specialization of the *ManufResource_sim* to emulate a resource through which batches of products flow, executing processes that modify some properties of the batches (e.g., location, flow times, etc.) or the contained products (e.g., its state or its geometric characteristics). A *ProcessingResource_sim* can be composed of

other *ManufResource_sim* (i.e., both processing and control resources) and must have at least two *FU_Ports* to send and receive *FlowUnits*.

- *TransformResource_sim* is a specialization of the *ProcessingResource_sim* that emulates a transformer resource that modifies the properties of the *FlowUnits*; so, incoming *FlowUnit* type (input product) is different from the outgoing *FlowUnit* type (processed product).
- *LogisticResource_sim* is a specialization of the *ProcessingResource_sim* that emulates a logistical resource, participating in the materials flow without modifying *FlowUnit* properties so the type of incoming and outgoing *FlowUnits* is the same.
- *ControlResource_sim* is a specialization of the *ManufResource_sim* that emulates a control resource. *ControlResource_sim* does not participate in the logistics flow but exchanges data to support monitoring and decision making.
- *FlowUnit* represents a flowing unit, that is, a batch of products. Therefore, it will be composed of one or more *Product_sim*, which represents each material or product unit.
- *DataPort* represents a generic port as an interaction point to exchange data through connections that specify relationships between components of the simulation system.
- *FU_Port* is a specialization of the *DataPort* that supports the transfer of data on the simulated *FlowUnits* (product batches), supporting the logistics flow. As represented via a dependency relationship in Figure 2, this type of port must be typed by a *FlowUnit*.
- *C_Port* is a specialization of the *DataPort* that supports data transfer in the form of communication between resources to synchronize tasks and behaviors.
- *Product_sim* represents a simulated product unit characterized by properties that include the deviations for its key geometric characteristics. As a part of the *FlowUnit*, it supports geometric deviation propagation through the simulated manufacturing stages.

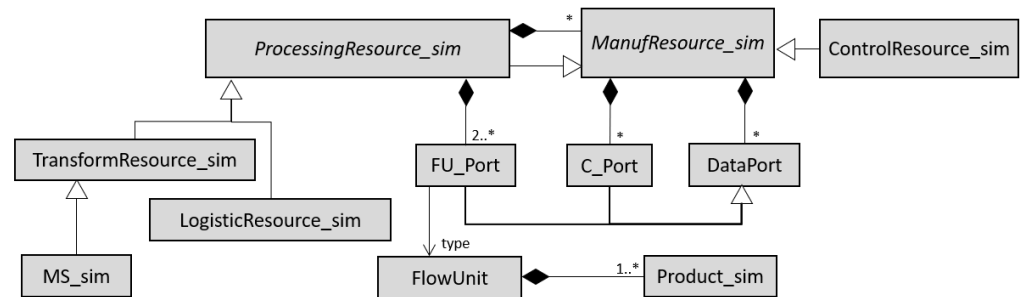


Figure 2. Abstract syntax of simulated manufacturing systems.

3.1.3. Behavior Modeling in the Simulation System

In addition to identifying previously commented concepts, it is important to clarify the modeling of the behavior of some of them (*MS_sim*, *InputGenerator*, *OutputCollector*, etc.). Although various approaches to modeling discrete behaviors could be valid, with each one focused on certain aspects (events, activities, processes, etc.), in this proposal, the DEVS formalism is adopted and described using state machines. According to DEVS formalism, a behavior can be defined by coupling simpler behaviors; so, the behavior of composite elements emerges from their behavioral components and their interactions. In this metamodel, the *BehavioralElement_sim* concept represents the generalization of any component with behavior, differentiating between atomic and coupled behavior with the *isAtomic* Boolean property (Figure 3). It is established that atomic elements (when *atomic = true*) must have an explicitly defined behavior and cannot be composed by parts of *BehavioralElement_sim* type (i.e., parts with their own behavior). This atomic behavior must be defined as a state machine, represented by *STM_MainBehavior*. In contrast, composite elements (when *atomic = false*) have no explicitly defined behavior, but they must be composed of other *BehavioralElement_sim* (atomic or not). Both cases are depicted as composite relationships in Figure 3, but they are mutually exclusive.

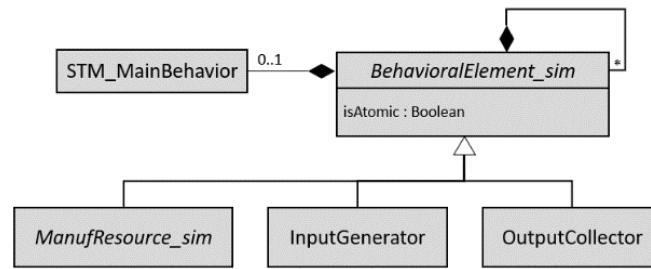


Figure 3. Abstract syntax of elements with behavior.

3.1.4. The Product in the Simulation System

Another fundamental element in the simulation of PPR systems is the processed product (*Product_sim*). As shown in Figure 4, the *Product_sim* has a reference relationship with product specification (*ProductSpecif_data*), which includes at least two descriptions of its possible transformation states (*StateSpecif*) corresponding to the initial and final product states.

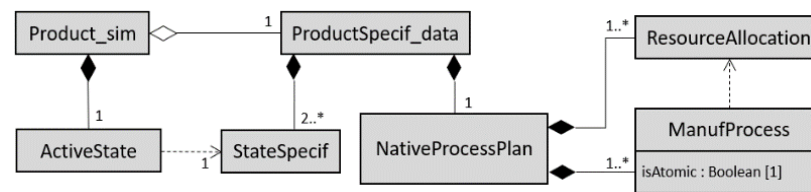


Figure 4. Abstract syntax of emulated product and its specification.

Moreover, *ProductSpecif_data* is composed by a native process plan (*NativeProcessPlan*) described in an activity diagram. Each *NativeProcessPlan* is composed of one or more actions (*ManufProcess*) that represent manufacturing processes. Atomic *ManufProcess* (identified by the Boolean attribute *isAtomic*) correspond to subphases. To support the resource assignment, the *NativeProcessPlan* has *ResourceAllocations* to define a specific resource for each atomic *ManufProcess*, as shown by the dependence relationship.

On the other hand, *Product_sim*, which represents each product unit, is characterized with some basic properties, including the identification of the current state (*ActiveState*), which points to one of the *StateSpecif*, as shown by the dependence relationship.

3.1.5. Product–Resource Geometric Interaction in the Simulation System

One of the fundamental aspects addressed in the proposed simulations is the analysis of the geometric quality of the emulated products, an issue that is based on the geometrical modeling of products and resources and the physical interactions between them.

On the one hand, the geometric modeling of the products is addressed in Figure 5. The product specification (*ProductSpecif_data*) must include at least one *ProductArtifact* (specialization of the *Artifact* concept from SysML4TA) to define the nominal geometries and tolerances according to the SysML4TA profile constructions. *ProductDeviations* is defined as a dataset (vector or matrix) that stores the deviations of each of the key geometric features identified in the specification of a simulated product unit (*Product_sim*). *ProductDeviations* specializes the *DeviationVector*, a concept of the SysML4TA profile.

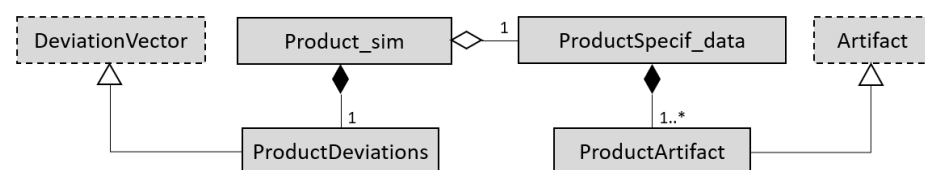


Figure 5. Abstract syntax of emulated products incorporating geometric deviations.

On the other hand, in a similar way, the geometric modeling of the transformer resources is addressed in Figure 6. A resource specification (*ManufResourceSpecif_data*) is composed by at least one *ConfiguredMachine*. Each *ConfiguredMachine*, which is a specialization of the *Artifact* concept from SysML4TA, describes the resource configuration for a subphase, including the geometric specification of the machine, fixtures and/or tools, and the assembly relationships between them. The particular deviations of each *TransformResourceSim* are defined by a *ResourceDeviations* (specialization of *DeviationVector* from the SysML4TA profile).

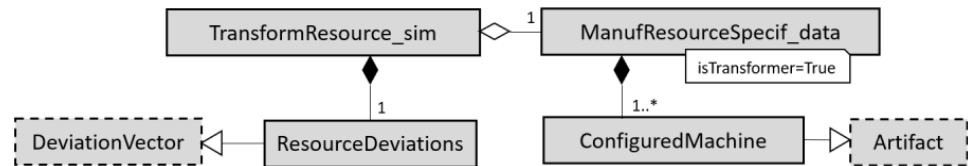


Figure 6. Abstract syntax of simulating resources with geometric deviations.

As shown in Figure 7, the *ProcessAssembly* is an artefact owned by an atomic *TransformResource_sim* which has references to at least one *ProductArtifact* and a *ConfiguredMachine* and includes the assembly relationships between these artifacts. Therefore, the *ProcessAssembly* involves the mathematical expressions necessary to obtain the quality characteristics of the resulting product in a single-stage problem. These results obtained in a workstation are transmitted to subsequent workstations to support the propagation of the in-process product geometric deviations, influencing other *ProcessAssembly* definitions until the final product quality is obtained.

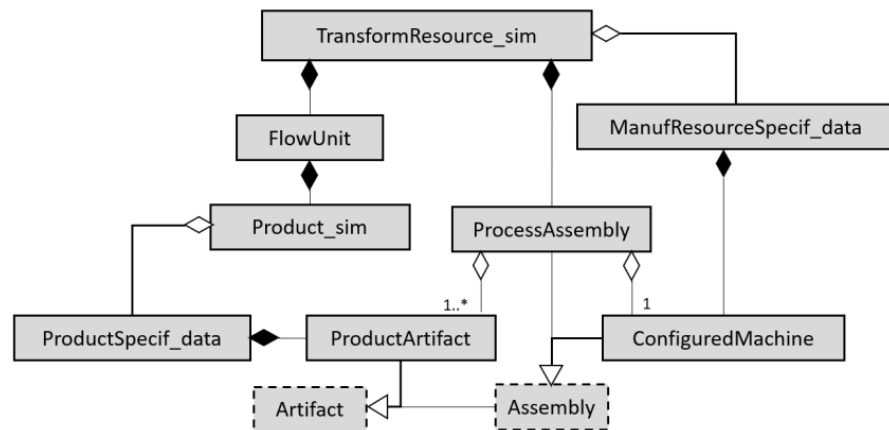


Figure 7. Abstract syntax of emulated process assembly.

3.2. SysML4GDPSim Profile

From the described metamodel (i.e., concepts and their relations and restrictions), the SysML4GDPSim profile has been developed. In order to define the profile, each concept of the metamodel has been defined as a stereotype, transferring part of the abstract syntax through extension relationships with UML metaclasses or specializations of the SysML profile, as shown in Figure 8. The rest of the relationships and semantic considerations have been included in the profile through the implementation of OCL rules for each SysML4GDPSim stereotype. These OCL rules are identified in the description of the stereotypes included in Appendix A and exemplified in Appendix B.

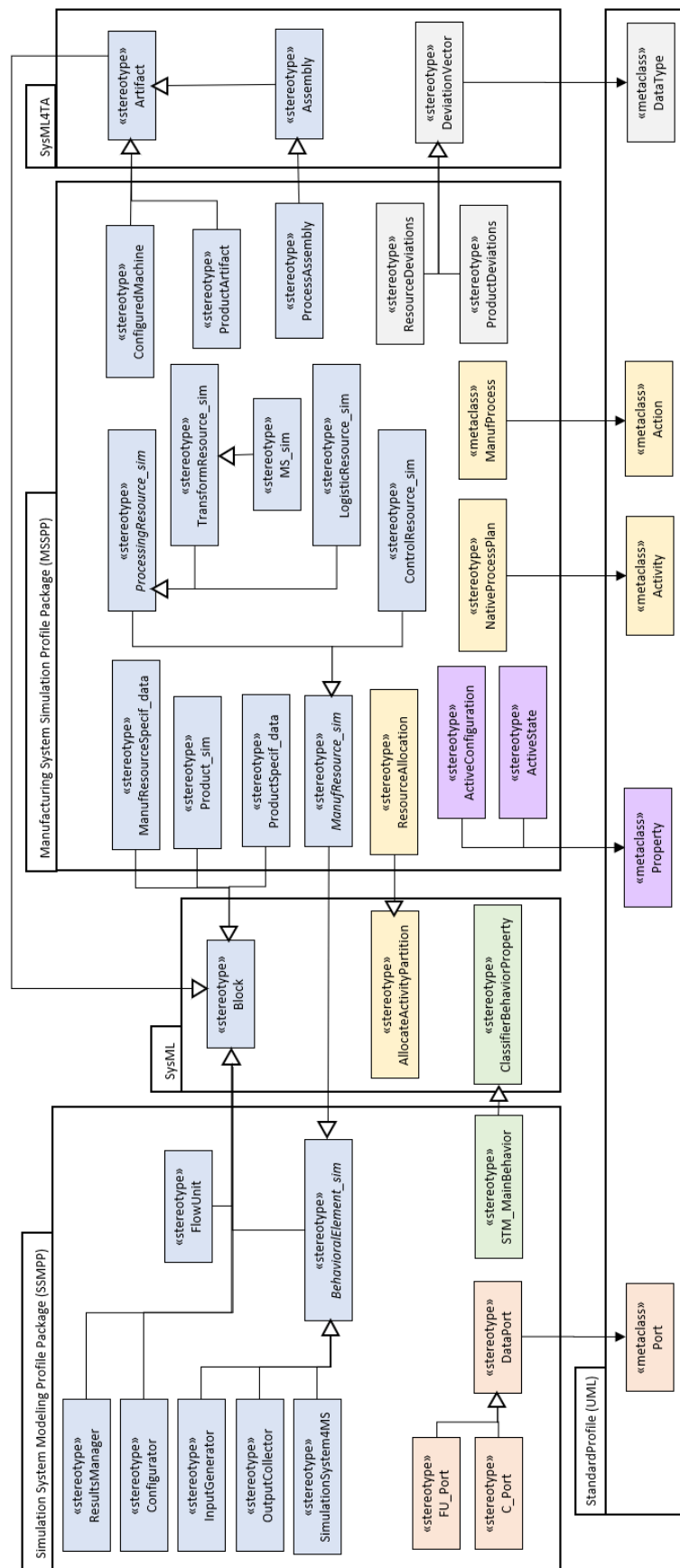


Figure 8. SysML4GDPsim profile package diagram. The color code groups stereotypes that extend the same UML metaclass, or specialize the same SysML stereotype.

It should be noted that in order to structure the content of this profile, two main packages have been proposed (Figure 8). The simulation system modeling profile package (SSMPP) includes the most general concepts of the simulation system; so, in future work, this package and its stereotypes can be used to develop other alternative simulation systems. The manufacturing system simulation profile package (MSSPP) includes specific stereotypes for the simulation of manufacturing systems, and in particular, for the analysis of productivity and geometric quality. As mentioned in the metamodel description and shown in Figure 8, some stereotypes are imported from the SysML4TA profile, which enables the nominal geometric specification of artifacts (i.e., product, fixtures, ...) and assembly relationships in order to compute the geometric deviations on the process assemblies.

3.3. Developed Libraries

Following the SysML4GDPSim profile definition, some modeling libraries have been developed to facilitate user modeling tasks. Both the SysML4GDPSim profile edition and the libraries definition have been carried out in the Papyrus environment, one of the most widespread free environments for modeling with UML and SysML in academia. Figure 9 shows a screenshot of this modeling environment, showing an error identified in the defined model. During profile development, some inconsistencies are forcibly introduced to verify the correct functioning of the OCL rules. In this case, rule “C8” is defined to assure that a «SimulationSystem4MS» block has a part typed by a «MS_sim» Block. This rule is violated in the model (i.e., multiplicity must be “1”, and it is defined as “1..*”).

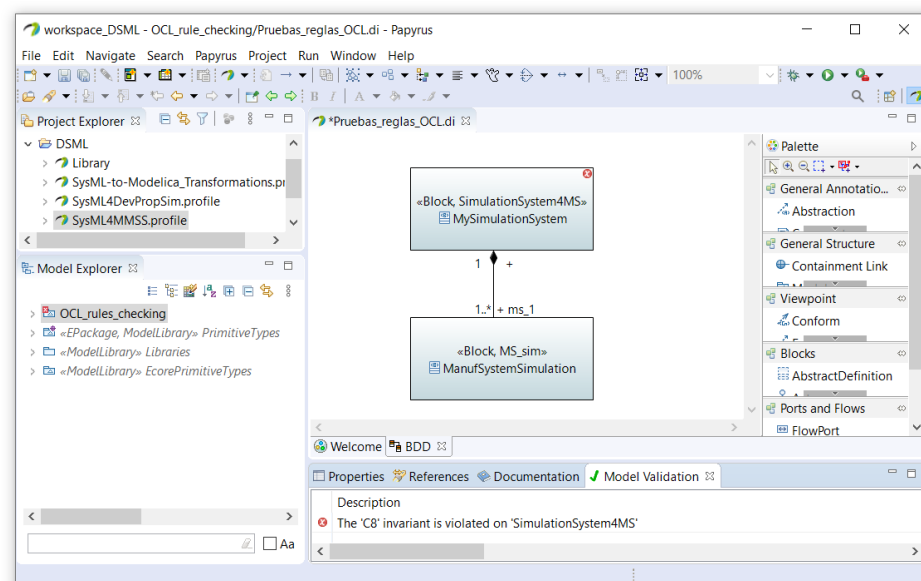


Figure 9. Papyrus environment and errors identified during model validation.

Additionally, equivalent libraries have been developed using Modelica. These Modelica libraries have the same package structure and elements as the SysML libraries, and, as explored in [11], they can be obtained via manual or automatic model transformation, which is beyond the scope of this paper. In this way, when a consistent simulation model is defined using the SysML libraries, it can be transformed into a Modelica model ready to be executed in an easy and quick way.

The package structure of these libraries, presented in detail in [11], is shown in Figure 10. As can be observed, there are three main packages with elements for (a) the analysis of material flow in multistage manufacturing systems; (b) the mathematical modeling of artifacts passed into TTRS concepts; and (c) the analysis of material flow and the propagation of geometric deviations in manufacturing systems.

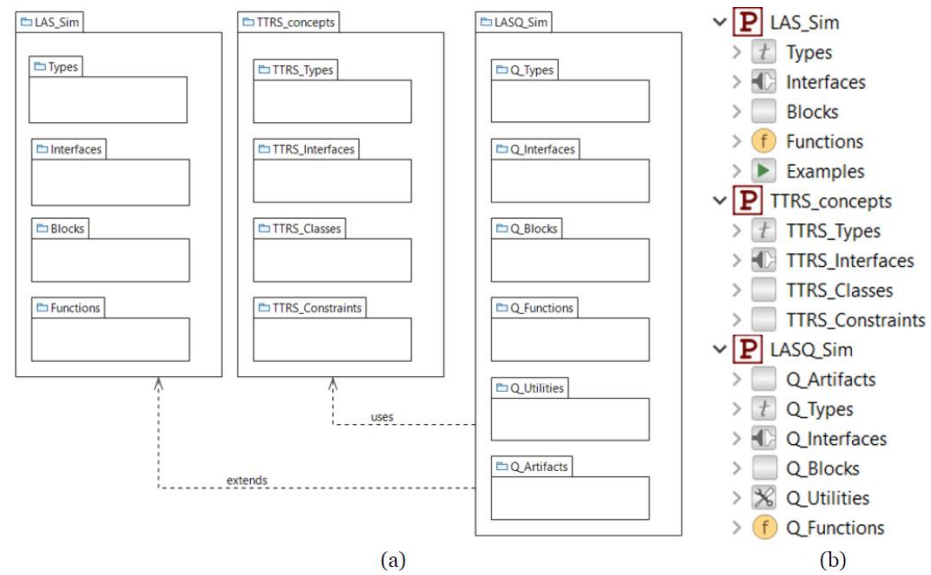


Figure 10. Libraries packages developed in SysML (a) and Modelica (b).

4. Case Study

This section presents a case study to exemplify the SysML modeling of a simulation system applying the SysML4GDPSim profile, as well as the description of some results obtained after its transformation and execution. It has been decided to limit the case to a 2D analysis of a multistage assembly process using a simple isostatic localization pattern for each assembly stage. Moreover, the methodology proposed in [11] is applied, whose procedure encompasses the tasks of modeling the referent system and modeling the simulation system (including the geometric modeling of artifacts based on TTRS), the definition of experiments, and their to-be-executed transformation to Modelica.

4.1. Referent System Description

This case is focused on the analysis of the manufacturing of the product depicted in Figure 11, a frame manufactured from the union of four metal parts. Each part has one or more holes where the axles of a series of gears are inserted. The manufacturing system (MyLAS) designed is made of three welding workstations, each one composed of a welding station and two stores for the incoming parts, as shown in the block definition diagram (BDD) presented in Figure 12. The resource connections, shown in the internal block diagram (IBD) of Figure 13, supports a linear flow of the product batches.

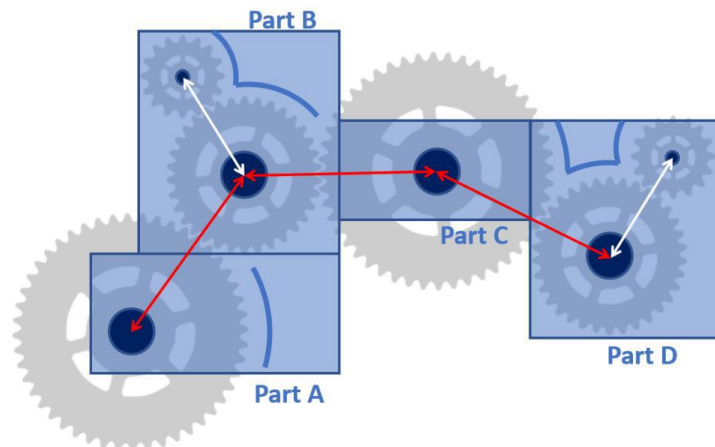


Figure 11. Graphic representation of the analyzed product.

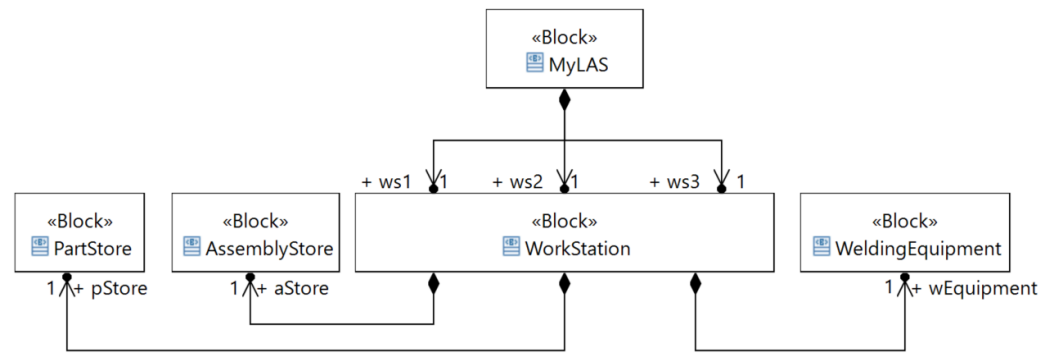


Figure 12. BDD to represent the structure of the linear assembly system.

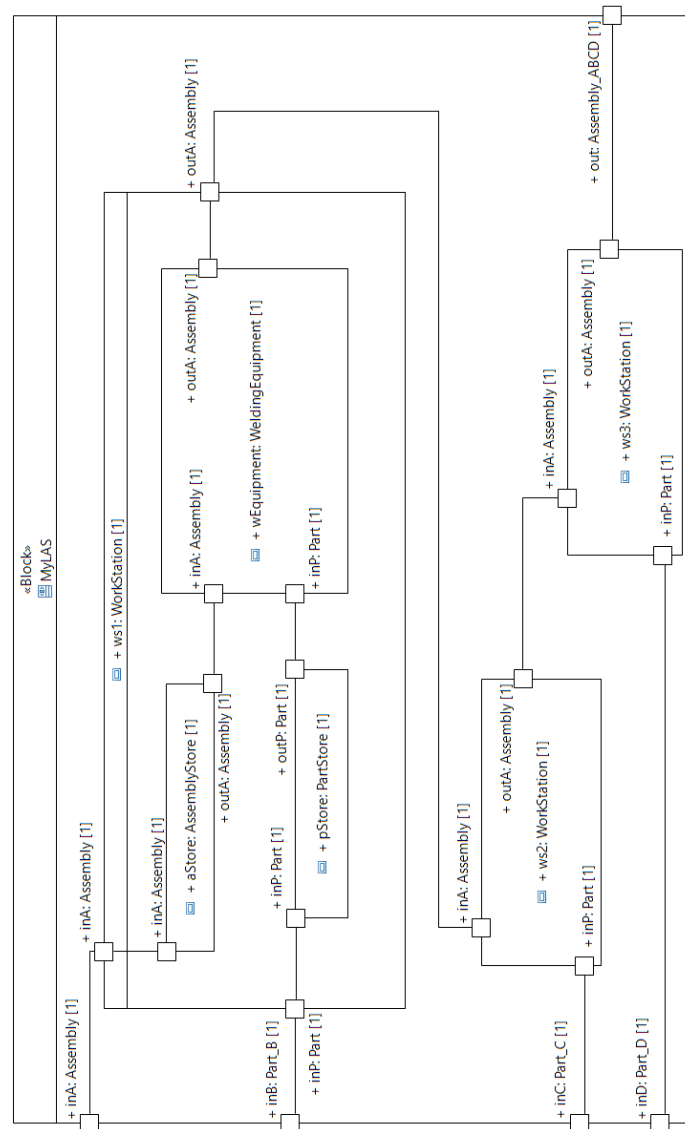


Figure 13. Internal structure of the specified assembly line (MyLAS block).

The main goal of the proposed study is the modeling of a simulation system to analyze its productivity and product quality performance. The native process plan for the assembly process is represented in Figure 14 by means of an activity diagram where the three assembling stages and intermediate storage stages are shown, each one assigned to a specific workstation of the manufacturing system.

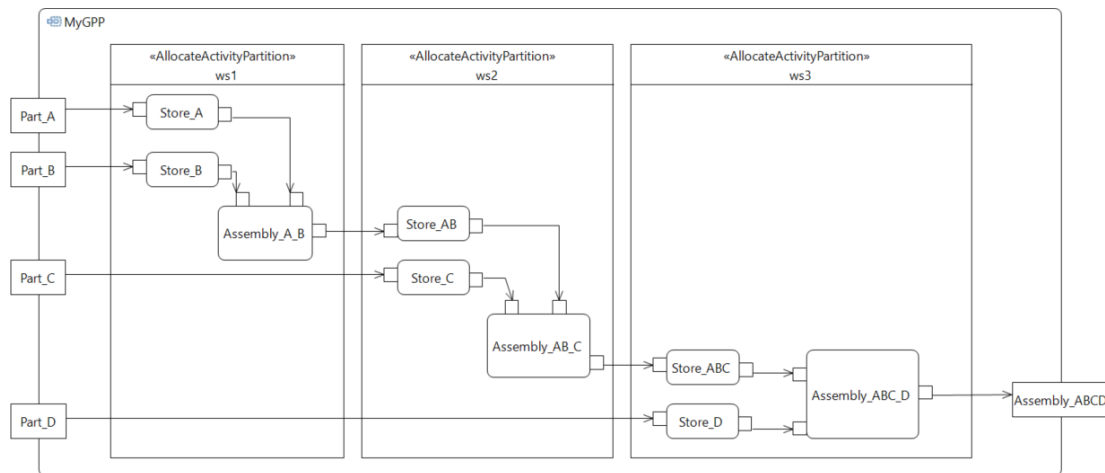


Figure 14. Activity diagram of the generic process plan specification (MyGPP).

In Figure 15, the process assemblies for each process stage are shown. As can be observed, for each process stage, the incoming parts or subassembly are located on the fixture using the same isostatic localization pattern: a four-way locator (p1, p3) and a two-way-locator (p2, p4).

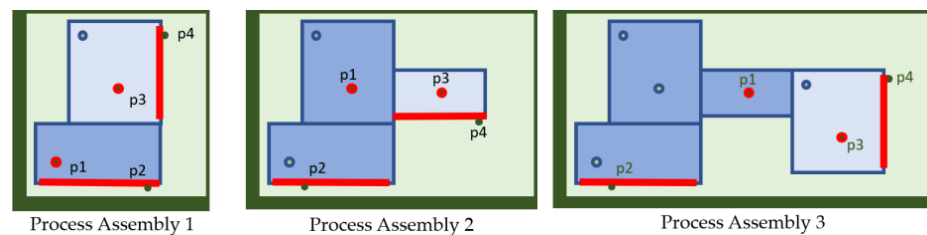


Figure 15. Process assembly corresponding to each of the stages considered.

4.2. Simulation System Modeling

Figure 16 shows a BDD of the simulation system (MySimulationModel) proposed for analyzing the performance of the manufacturing system described.

MySimulationModel includes the following: (a) a block to hold the parameter values («Configurator»); (b) the input materials generators («InputGenerator»); (c) the simulated manufacturing system itself («MS_sim»); and (d) the end of the material flow («OutputCollection»). These elements, as well as the connections that support the data flow between them, are also represented in the IBD depicted in Figure 17.

4.3. Geometric Artifact Modeling in the Simulation System

To support the analysis of geometric deviations in the assembled products, each of the blocks that emulates an assembly stage has a process assembly model defining the geometrical representation the artifacts (i.e., fixtures and parts) involved, as well as the assembly relations between them. This artifact model is defined based on TTRS concepts using the SysML4TA profile [33]. Figure 18 shows an IBD of ProcessAssembly1, defined for the first assembly stage where two parts (saA and pB) are positioned on the fixture (f1) using the pattern previously described, i.e., a four-way location (assembly relationships c9_a and c9_b) and a two-way location (assembly relations c11_a y c11_b).

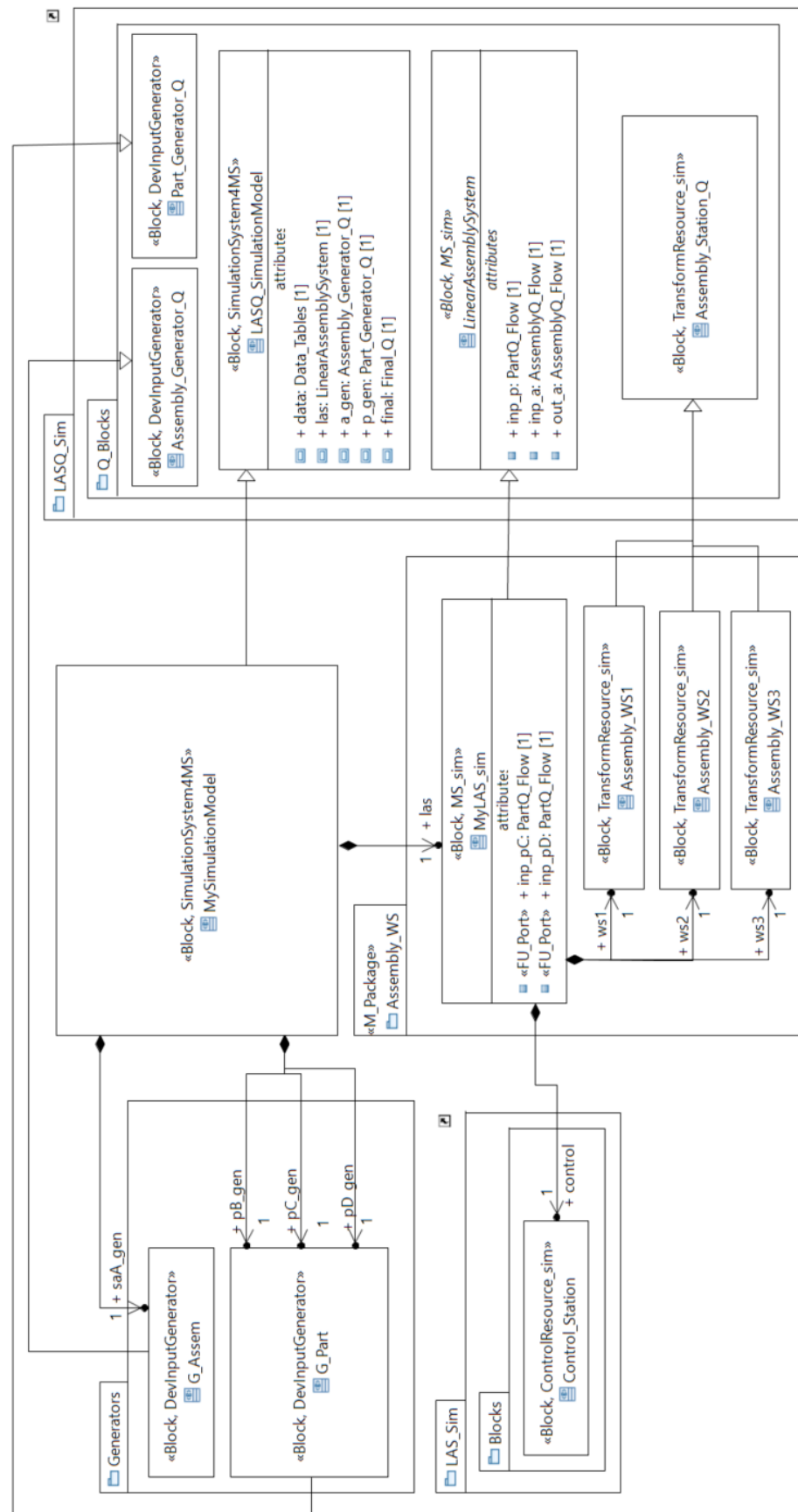


Figure 16. Block definition diagram of the simulation system (MySimulationModel).

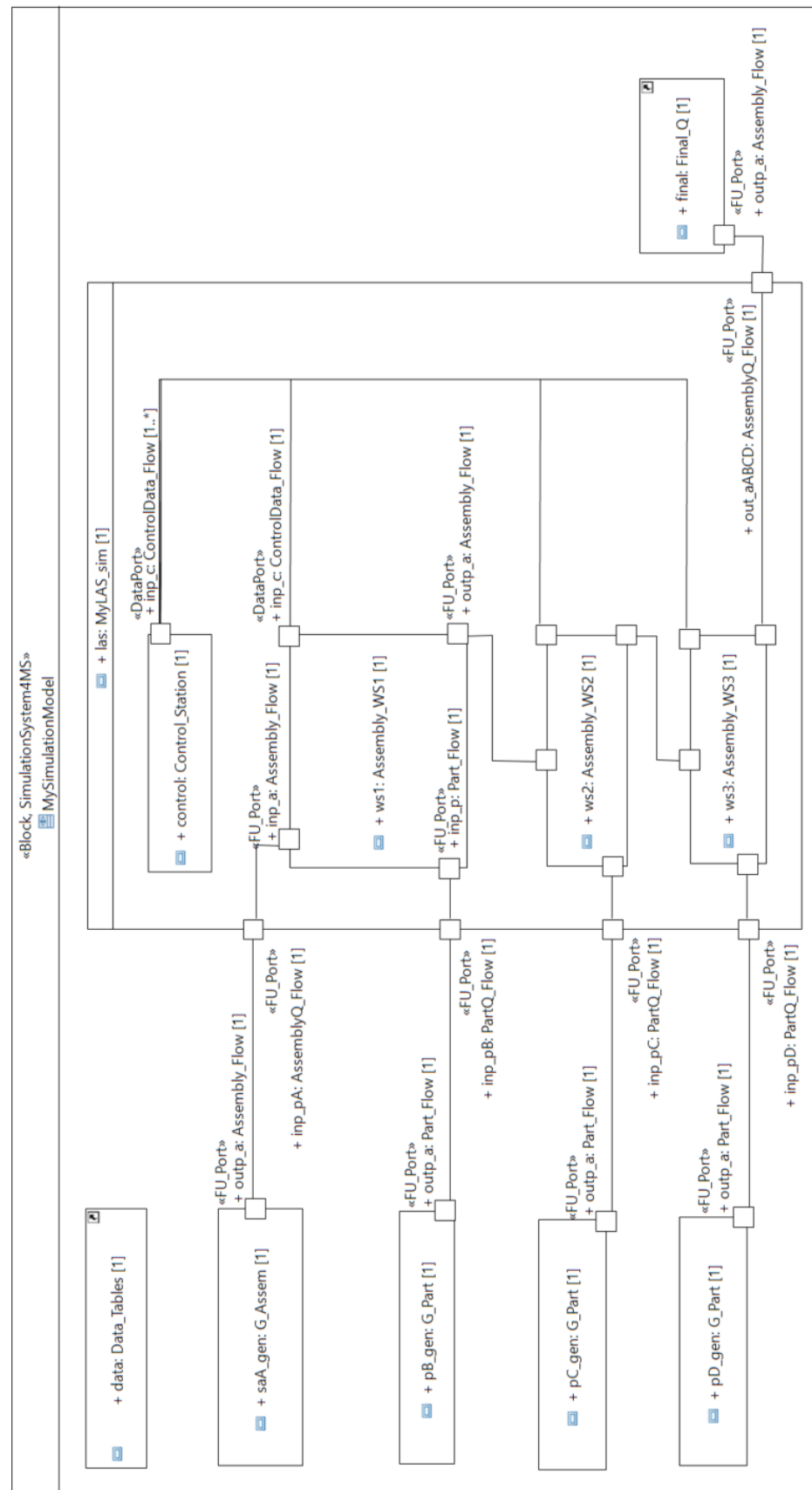


Figure 17. Internal block diagram of the simulation system (MySimulationModel).

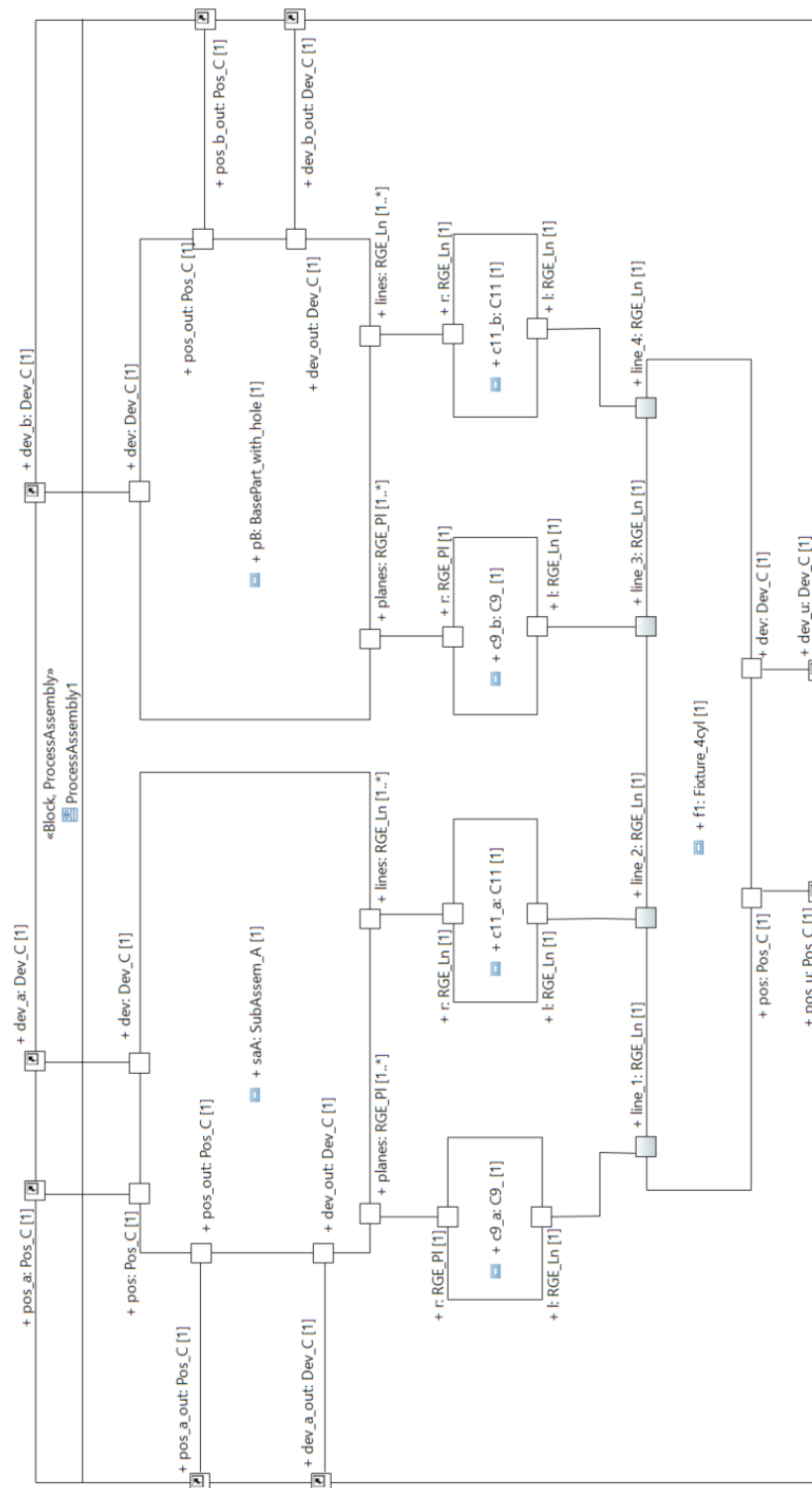


Figure 18. Internal block diagram of ProcessAssembly1.

4.4. Experiment Definition and Model Transformation

Once the structural modeling of the simulation system has been completed, and considering that the atomic behaviors are previously defined in the employed library elements, it is time to model the experiment to be executed; so, specific values are defined to each simulation parameter. In SysML, the particular values of the instantiated elements are defined by means of instance specifications. Figure 19 shows an example of a SysML block (`Data_Tables`) instantiation.

```

data_inst: Data_Tables
slots
max_parts : Integer = 4
part_models : Integer = 4
total_stages : Integer = 4
assembly_stages : Integer = 3
part_by_assembly : Integer = 4
max_steps : Integer = 4
Plan_data : Integer = {{2, 1, 2, 1, 200, 20, 10}, {2, 3, 4, 2, 190, 19, 9}, {2, 5, 6, 3, 190, 19, 9}, {2, 7, 7, 4, 60, 6, 3}}
Assemblies_data : Integer = {{1, 0, 0, 0}, {2, 0, 0, 0}, {1, 2, 0, 0}, {3, 0, 0, 0}, {1, 2, 3, 0}, {4, 0, 0, 0}, {1, 2, 3, 4}}
part_arrival_data : Integer = {1 / 20.25, 1 / 15.67, 1 / 35.67, 1 / 50}
part_arrival_data : Integer = {1 / 20.25, 1 / 15.67, 1 / 35.67, 1 / 50}

```

Figure 19. Instance specification for the Data_Tables block.

Finally, the model validation is executed, checking that all the OCL rules defined in the profile are met (see Figure 9). After that, the validated SysML model is transformed to Modelica model using model-to-model and model-to-text transformations (MMT and MTT). MMT is mainly supported by the application of another SysML profile (such as SysML4Modelica [10]), while automated MTT requires the execution of an algorithm. This content is out of the scope of this paper and can be consulted in [11]. Figure 20 shows a screenshot of the resulting textual model in the OpenModelica simulation environment.

```

394 model ManufSystemSimulation
395   import LAS_Sim.*;
396   import LASQ_Sim.*;
397   import TTRS_concepts.*;
398
399   inner LAS_Sim.Blocks.Data_Tables_LAS data annotation( (..) );
401   inner LAS_Sim.Blocks.Data_evolution_LAS evolution annotation( (..) );
403
404   LASQ_Sim_CASO.Generators.G_Assem a_gen1 annotation( (..) );
406   LASQ_Sim_CASO.Generators.G_Part p_gen2 annotation( (..) );
408   LASQ_Sim_CASO.Generators.G_Part p_gen3 annotation( (..) );
410   LASQ_Sim_CASO.Generators.G_Part p_gen4 annotation( (..) );
412
413   LASQ_Sim_CASO.WorkStations.ManufSystem manufSystem annotation( (..) );
415
416   LASQ_Sim.Q_Blocks.Final_Q final_G annotation( (..) );
418 equation
419
420   connect(a_gen1.outp_a, manufSystem.saA) annotation( (..) );
422   connect(p_gen2.outp_p, manufSystem.pB) annotation( (..) );
424   connect(p_gen3.outp_p, manufSystem.pC) annotation( (..) );
426   connect(p_gen4.outp_p, manufSystem.pD) annotation( (..) );
428 //Final connection
429   connect(manufSystem.aABCD, final_G.inp_a) annotation( (..) );
431 protected

```

Figure 20. Modelica executable model.

4.5. Experiment Results

In order to exemplify some of the many results offered by the proposed analysis, Figure 21 shows a comparison of the orientation deviation obtained for the last assembled part (part_D), calculated with two alternative experiments in which the position of the fixture locators has been modified. In Case 1, fixture locators are defined in the positions shown in Figure 15. In Case 2, two-way locators (p2 and p4) are positioned closer to the four-way locators (p1 and p3), reducing the distance between them. Data shown in Figure 21 correspond to the orientation deviation in the third assembly station; so, the deviation has no value until the first finished product is processed. After this point, each value change corresponds to a different processed product. As expected, the obtained results show that in Case 1, the deviations of the incoming product parts results in a greater orientation deviation of the resultant assembly (higher mean value) compared to Case 2. However, a closer locator position (Case 2) subtly increases the variability (standard deviation) of this orientation deviation in the simulated products.

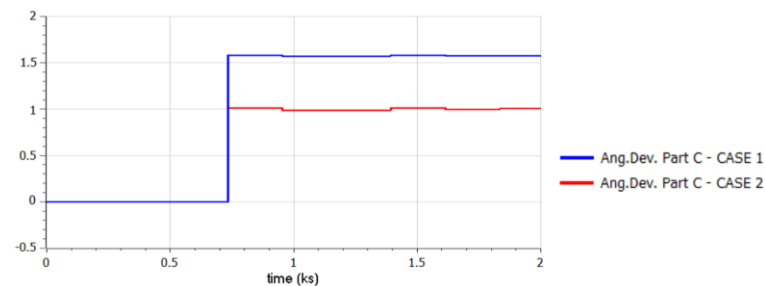


Figure 21. Orientation deviation of part D in the final product. Comparison between cases.

The simulations also provide data related to the productivity of the analyzed system, such as the throughput, blocked-station times, simulating stops for maintenance or repair of breakdowns, etc. Furthermore, these results can be crossed with aspects related to quality, for example, establishing limits on deviations for product acceptance and calculating the productivity of parts that meet the specifications. Table 1 summarizes some results of the Case 1 experiment execution with a simulation time of 5000 s (14 h approximately). During this time, 203 products were finished, which represent a throughput of 14.6 products per hour. Considering the geometric specifications and the calculated deviations for each simulated product, only 164 units meet these specifications, obtaining a real throughput of 11.8 products per hour.

Table 1. Main productivity results of the Case 1 experiment execution.

Variable	Results
Finished products	203
Throughput (Prod./h)	14.61
Finished products meeting specifications	164
Throughput with products meeting specifications (Prod./h)	11.81

5. Discussion

The research presented in this paper is focused on the multidomain simulation of multistage manufacturing systems, integrating the analysis of productivity and geometric quality. These types of analysis have been widely studied separately and are reported in multiple works. On the one hand, [36] presents the state of the art on the use of simulation models for manufacturing systems analysis during their design, and especially on process planning and material flow analysis. On the other hand, [37] is a review of tolerance-related works, ranging from tolerance specification to their mathematical analysis. Although paradigms like production quality promote the joint analysis of these aspects, in the revised literature, few works have proposed a detailed solution adopting this approach.

This orientation has already been explored by our research group in previous works. In [14], the simulation of the material flow is enriched with geometric data to support the productivity and geometric quality analysis. These multidomain simulation models integrate different control logics and/or strategies for a more realistic analysis of the system, quantifying the quality improvement and the influence of measurement processes and control decisions on productivity indicators. Specifically, the quality analysis proposed in [14] adopts of the stream of variation (SoV) technique [38,39], a mathematical model based on the state space formalism [40], to simulate the propagation of geometric deviations in multistage systems. However, the adoption of SoV has certain limitations, highlighted in [11]. To overcome these limitations, the SysML4TA profile proposed in [33], based on TTRS concepts, is adopted in this paper. The concepts of SysML4TA enable the definition of geometric artifact models and the mathematical expressions necessary to simulate geometric deviations without assumptions or simplifications.

Another substantial improvement over [14] is the use of SysML for the definition of a profile that considers the specific semantics of modeling simulation systems and supports

consistency checking based on these semantics. Some previous works have addressed the development of SysML profiles for geometric modeling or tolerance analysis, such as [41–43]. These DSML are basically a set of concepts (tags), but they do not take advantage of SysML's capabilities to formalize the semantics of the proposed concepts.

During the definition of the proposed SysML4GDPSim profile, certain difficulties or limitations have been detected in SysML, alongside its capabilities with respect to supporting the modeling of certain aspects of the system. For example, since SysML is fundamentally a descriptive language, it does not have sufficient elements to address the modeling of detailed behaviors, usually defined as opaque expressions implemented with other modeling languages, or even describing the behaviors in a natural language. The content of these opaque expressions is not based on any metamodel, so they are defined as annotations that can only be interpreted by users but without the necessary formalism to be interpreted by computers. At the current stage of this research, user models are created from library components, assuming that their behavior is well-defined, using opaque expressions written with other languages (like Modelica). However, other formal ways of defining detailed behaviors should be explored to support the consistent validation and subsequent transformation of user models into executable models. Despite these limitations, the SysML4GDPSim profile has proven to be a valid language for simulation systems design.

6. Conclusions and Future Work

The developed SysML4GDPSim profile supports the necessary concepts for modeling simulation systems to analyze manufacturing systems, especially those focused on the analysis of productivity and geometric quality. The formalization of these specific domain semantics supports the consistency assurance of models developed to analyze products and manufacturing systems during their design. Although the current state of the proposal addresses mainly intra-model consistency, this approach can also be applied to manage inter-model consistency between simulation models and specification models of the referent system. During this experience, both SysML and OCL have proven to be valid languages for creating DSML and supporting domain-specific semantics, implemented through rules defined in each concept.

The proposed profile has been successfully applied in the development of libraries of reusable elements, which have been used in the development of a case study, highlighting the assurance of consistency supported by the proposed profile. Furthermore, since the proposed profile is a well-defined DSML with its own metamodel, it is possible to establish relationships with other languages and perform automatic transformations. Although this issue is out of the scope of this paper, transformation mechanisms presented in referenced works have been applied during the case study to obtain the executable simulation models.

Based on the presented proposal, some future work lines are proposed to continue this research line. For example, the behavioral modeling in SysML must be deeply explored, reaching the formal and detailed level necessary to support the complete definition of simulation systems and to enable automatic transformation to executable models. Moreover, the proposed DSML can be extended to support alternative analysis, facilitating the definition of multidomain simulations integrating productivity analysis and other key aspects of manufacturing systems design. In a similar way, libraries can also be extended to support the analysis of other PPR systems, including other manufacturing processes (such as machining) or non-linear systems, in which more complex flows are considered. Finally, another future work line is the definition of the metamodel concepts with an ontological language such as ontology web language (OWL). This ontological approach would allow for consistency validation at both model and instance level (individuals in an ontological model) and the use of reasoners.

Author Contributions: Conceptualization, S.B.-N., P.R.C. and F.R.S.; methodology, S.B.-N. and F.R.S.; software, S.B.-N. and P.R.C.; validation, S.B.-N. and P.R.C.; formal analysis, S.B.-N. and F.R.S.; investigation, S.B.-N. and F.R.S.; resources, S.B.-N.; data curation, S.B.-N.; writing—original draft preparation, S.B.-N., P.R.C. and F.R.S.; writing—review and editing, P.R.C.; visualization, S.B.-N.; supervision, P.R.C. and F.R.S.; project administration, P.R.C.; funding acquisition, P.R.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by “Pla de Promoció de la Investigació a l’UJI” of UNIVERSITAT JAUME I. Project title: “Metodología para el modelado e implementación del gemelo digital de un sistema de fabricación multietapa orientado a la optimización del funcionamiento”. Reference: UJI-B2022-49.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request.

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

Appendix A

Table A1. Description of main SSMPP stereotypes.

Estereotype	Description
«BehavioralElement_sim»	Abstract stereotype that specializes the «Block» to represent any behavioral part of the simulation system (or the whole simulation system). A Boolean attribute (isAtomic) differentiates atomic and composite behavioral elements. Atomic behavioral elements have a «STM_MainBehavior» behavior (C1) and they cannot own behavioral parts (C2). Composite behavioral elements cannot have an explicitly defined behavior (C3) but they must own at least one behavioral part (C4).
«STM_MainBehavior»	«ClassifierBehaviorProperty» specialization to represent the behavior of an atomic «BehavioralElement_sim» Block, defined as a «StateMachine» (C5).
«SimulationSystem4MS»	«Block» representing the whole simulation system. A «SimulationSystem4MS» Block own a part typed by a «Configurator» block (C6), a part typed by a «ResultsManager» block (C7) a part typed by a «MS_sim» block (C8), at least one part typed by a «InputGenerator» block (C9), and at least one part typed by a «OutputCollector» block (C10).
«Configurator»	«Block» for data structuration and parameter definition in a simulation system.
«InputGenerator»	«BehavioralElement_sim» specialization to identify a block in which a flow unit starts, generating the unit flows with a periodicity defined in its behavior. An «InputGenerator» block must own at least one «FU_Port» port (C11).
«OutputCollector»	«BehavioralElement_sim» specialization to identify a block in which a Flow unit finishes. An «OutputCollector» block must own at least one «FU_Port» port (C12).
«ResultsManager»	«Block» defined to compute the performance measures from simulation data. A «ResultsManager» block must own at least one «DataPort» port (C13) to receive data from other simulation system parts.
«DataPort»	«Port» defined for the data exchange with other simulation system parts.
«FU_Port»	«DataPort» specialization to identify ports exchanging flow units. A «FU_Port» ports must be typed by a «FlowUnit» block (C14).
«C_Port»	«DataPort» specialization to identify ports defined for exchanging data related with the communication and processes synchronization.
«FlowUnit»	«Block» defined to represent a flow unit, representing a product batch. A «FlowUnit» block is composed by at least one part typed by «Product_sim» Block (C15).

Table A2. Description of main MSSPP stereotypes.

Stereotype	Description
«ManufResource_sim»	Abstract stereotype that specializes the «BehavioralElement_sim» to identify any manufacturing resource. A block stereotyped by a «ManufResource_sim» specialization must have an aggregation relationship (reference) with a «ManufResourceSpecif_data» block (C16).
«ManufResourceSpecif_data»	«Block» defined to support specification data about a manufacturing resource type. A boolean property (isTransformer) identifies the specifications about transformer resources. A transformer resource specification (isTransformer = True) must have at least one part typed by a «ConfiguredMachine» block (C17).
«ConfiguredMachine»	«Artifact» specialization to define the TTRS_based representation of a specific configuration for a transformer resource.
«ProcessingResource_sim»	Abstract stereotype that specializes the «ManufResource_sim» to identify a processing resource definition, that is, a resource through which material units flow. A block stereotyped by a «ProcessingResource_sim» specialization must have at least two «FU_Port» ports (C18).
«TransformResource_sim»	«ProcessingResource_sim» specialization to represent transformer resources where product characteristics are modified. A «TransformResource_sim» block must own an «ActiveConfiguration» property (C19) and at least one part typed by «ResourceDeviations» data type (C20), and its two «FU_Port» ports must be typed by different blocks (C21).
«MS_sim»	«TransformResource_sim» specialization to identify the block that emulates the whole manufacturing system.
«ActiveConfiguration»	«Property» owned by a «TransformResource_sim» block (C22) identifying the current configuration of a transformer resource.
«ResourceDeviations»	«DeviationVector» specialization to define deviation values for the key geometric characteristics in a resource artefactual representation.
«LogisticResource_sim»	«ProcessingResource_sim» specialization to represent logistic. A «LogisticResource_sim» block cannot own any party typed by a «TransformResource_sim» block (C23), and its two «FU_Port» ports must be typed by the same block (C24).
«ControlResource_sim»	«ManufResource_sim» specialization to represent a control resource, that is, a resource that supports the monitoring, control and decision-making functionality. A «ControlResource_sim» block cannot have any «FU_Port» port (C25), but it must have at least one «C_Port» or «DataPort» port (C26).
«Product_sim»	«Block» defined to support data about product units. A «Product_sim» block have an aggregation relationship (reference) with a «ProductSpecif_data» block (C27) and an «ActiveState» property (C28).
«ProductDeviations»	«DeviationVector» specialization to define deviation values for the key geometric characteristics in a product artefactual representation.
«ActiveState»	«Property» of a «Product_sim» block (C29) used to define the current product state.
«ProductSpecif_data»	«Block» defined to support specification data about a product type considered in the simulation system. A «ProductSpecif_data» block must include a behavior defined by a «NativeProcessPlan» activity (C30) and at least two parts typed by different «ProductArtifact» blocks (C31) to support the artefactual representations of the product at different manufacturing states.
«ProductArtifact»	«Artifact» specialization to define the TTRS_based representation of a specific state for a product type.
«NativeProcessPlan»	«Activity» defined to establish the manufacturing stages of a product and the resources where they are executed. All the activities included in a «NativeProcessPlan» activity must be stereotyped as «ManufProcess» (C32) and they must be contained in a «ResourceAllocation» allocate activity partition (C33).
«ManufProcess»	«Action» owned by a «NativeProcessPlan» activity (C34) representing a manufacturing stage. A Boolean attribute (isAtomic) identifies the atomic processes, in this case, the subphases.

Table A2. Cont.

Stereotype	Description
«ResourceAllocation»	Specialization of the «AllocateActivityPartition» to assign particular resources to each «ManufProcess». It must be defined in a «NativeProcessPlan» Activity (C34). Every contained action must be stereotyped as «ManufProcess».
«ProcessAssembly»	«Assembly» specialization to define the TTRS_based representation of a process assembly, so a «ProcessAssembly» block has at least one reference to a «ProductArtifact» block and another reference to a «ConfiguredMachine» block.

Appendix B

Table A3. OCL expressions corresponding to some described rules.

Rule	OCL Expression
C2	if self.isAtomic=true then self.base_Class.allAttributes()->select(a a.type.oclsKindOf(UML::Class)). type.oclsAsType(UML::Class).getAppliedStereotypes().allParents()->select(b b.name = 'BehavioralElement_sim')->isEmpty() endif
C5	self.oclsKindOf(UML::StateMachine)
C11	self.base_Class.allAttributes()->select(a a.type.oclsKindOf(UML::Port)). getAppliedStereotypes()-> select(b b.name = 'FU_Port').size() = 1

References

- Colledani, M.; Tolio, T.; Fischer, A.; Iung, B.; Lanza, G.; Schmitt, R.; Váncza, J. Design and management of manufacturing systems for production quality. *CIRP Ann.* **2014**, *63*, 773–796. [CrossRef]
- Psarommatis, F.; May, G.; Dreyfus, P.A.; Kiritsis, D. Zero-defect manufacturing: State-of-the-art review, shortcomings and future directions in research. *Int. J. Prod. Res.* **2020**, *58*, 1–18. [CrossRef]
- Zhang, L.; Zhou, L.; Ren, L.; Laili, Y. Modeling and simulation in intelligent manufacturing. *Comput. Ind.* **2019**, *112*, 103123. [CrossRef]
- Henderson, K.; Salado, A. Value and benefits of model-based systems engineering (MBSE): Evidence from the literature. *Syst. Eng.* **2021**, *24*, 51–66. [CrossRef]
- Ferreira, W.D.; Armellini, F.; Santa-Eulalia, L.A. Simulation in industry 4.0: A state-of-the-art review. *Comput. Ind. Eng.* **2020**, *149*, 106868. [CrossRef]
- Vještica, M.; Dimitrieski, V.; Pisarić, M.; Kordić, S.; Ristić, S.; Luković, I. An application of a DSML in Industry 4.0 production processes. In Proceedings of the IFIP International Conference on Advances in Production Management Systems (APMS), Novi Sad, Serbia, 30 August 2020.
- OMG Systems Modeling Language (SysML), v. 1.6. Available online: <https://sysml.org/.res/docs/specs/OMGSysML-v1.6-19-11-01.pdf> (accessed on 17 January 2024).
- OMG Object Constraint Language (OCL), v. 2.4. Available online: <https://www.omg.org/spec/OCL/> (accessed on 13 January 2023).
- Gauthier, J.-M.; Bouquet, F.; Hammad, A.; Peureux, F. Toolled process for early validation of SysML models using Modelica simulation. In Proceedings of the Fundamentals of Software Engineering (FSEN 2015), Tehran, Iran, 22–24 April 2015.
- OMG, SysML-Modelica Transformation, Version 1.0, Object Management Group. Available online: <https://www.omg.org/spec/SyM/1.0/PDF> (accessed on 6 February 2024).
- Benavent-Nácher, S. Modelado y Simulación Híbrida de Sistemas de Fabricación Multietapa Orientado a la Evaluación de la Calidad Geométrica y la Productividad. Ph.D. Thesis, Universitat Jaume I, Castelló de la Plana, Spain, 2024.
- Höpfner, G.; Jacobs, G.; Zerwas, T.; Drave, I.; Berroth, J.; Guist, C.; Rumpe, B.; Kohl, J. Model-based design workflows for cyber-physical systems applied to an electric-mechanical coolant pump. In Proceedings of the 19th Drive Train Technology Conference (ATK 2021), Aachen, Germany, 9–11 March 2021.
- Wagner, H.; Zuccaro, C. Collaboration between system architect and simulation expert. In Proceedings of the IEEE International Symposium on Systems Engineering, Vienna, Austria, 24–26 October 2022.
- Benavent-Nácher, S.; Rosado, P.; Romero, F. Multidomain simulation model for analysis of geometric variation and productivity in multi-stage assembly systems. *Appl. Sci.* **2020**, *10*, 6606. [CrossRef]

15. Lefeber, E.; Rooda, J.E. Modeling and analysis of manufacturing systems. In *Handbook of Dynamic System Modeling*, 1st ed.; Chapman and Hall/CRC: New York, NY, USA, 2007.
16. Clément, A. The TTRSs: 13 Constraints for dimensioning and tolerancing. In *Geometric Design Tolerancing: Theories, Standards and Applications*; Springer: Boston, MA, USA, 1998; pp. 122–131.
17. INCOSE. *Systems Engineering Handbook: A Guide for Systems Life Cycle Processes and Activities*, 4th ed.; John Wiley & Sons: Hoboken, NY, USA, 2015.
18. Dictionary by Merriam Webster. Available online: <https://www.merriam-webster.com/dictionary/supersystem> (accessed on 6 February 2024).
19. Gedell, S.; Claesson, A.; Johannesson, H. Integrated product and production model—Issues on completeness, consistency and compatibility. In Proceedings of the 18th International Conference on Engineering Design (ICED 11), Lyngby/Copenhagen, Denmark, 15–19 August 2011.
20. Kathrein, L.; Meixner, K.; Winkler, D.; Lüder, A.; Biffl, S. A meta-Model for representing consistency as extension to the formal process description. In Proceedings of the 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Zaragoza, Spain, 10–13 September 2019.
21. Biffl, S.; Lüder, A.; Gerhard, D. *Multidisciplinary Engineering for Cyber-Physical Production System*; Biffl, S., Lüder, A., Gerhard, D., Eds.; Springer: Cham, Switzerland, 2017.
22. Michaelis, M.T. Function and process modeling for integrated product and manufacturing system platforms. *J. Manuf. Syst.* **2015**, *36*, 203–215. [\[CrossRef\]](#)
23. Souilah, M.; Tahan, A.; Abacha, N. A small displacement torsor model to evaluate machining accuracy in the presence of locating and machine geometric errors. In Proceedings of the Canadian Society for Mechanical Engineering International Congress 2021, Charlottetown, PE, Canada, 27–30 June 2021.
24. Bruscas-Bellido, G. Modelo Basado en Elementos Característicos Para la Planificación Supervisora de la Inspección. Ph.D. Thesis, Universitat Jaume I, Castelló de la Plana, Spain, 2015.
25. Law, A.M. *Simulation Modeling and Analysis*, 5th ed.; Mc Graw Hill: Tucson, AZ, USA, 2015.
26. Kim, T.G.; Zeigler, B.P. The DEVS formalism: Hierarchical, modular systems specification in an object-oriented framework. In Proceedings of the 1987 Winter Simulation Conference, Atlanta, GA, USA, 14–16 December 1987.
27. Tian, A.; Liu, S.; Chen, K.; Mo, W.; Jin, S. Spatial expression of assembly geometric errors for multi-axis machine tool based on kinematic Jacobian-torsor model. *Chin. J. Mech. Eng.* **2023**, *36*, 44. [\[CrossRef\]](#)
28. Mu, X.; Yuan, B.; Wang, Y.; Sun, W.; Liu, C.; Sun, Q. Novel application of mapping method from small displacement torsor to tolerance: Error optimization design of assembly parts. *J. Eng. Manuf.* **2022**, *236*, 955–967. [\[CrossRef\]](#)
29. Wang, H.; Lin, Y.; Yan, C. T-Maps-based tolerance analysis of composites assembly involving compensation strategies. *ASME J. Comput. Inf. Sci. Eng.* **2022**, *22*, 041007. [\[CrossRef\]](#)
30. Jiang, Q.; Ou, Y.; Zou, Y.; Zhou, C.-G.; Huang, S.; Qian, C.-Q. Analysis and optimization of tolerance design for an internal thread grinder. *Int. J. Adv. Manuf. Technol.* **2023**, *125*, 5369–5383. [\[CrossRef\]](#)
31. Umara, E. A New Method of Rigid Assemblies Stochastic 3D Tolerance Analysis Including Thermal Performance. Ph.D. Thesis, Universidade de São Paulo, Sao Paulo, Brazil, 2022.
32. Monica, F.D.; Patalano, S.; Choley, J.; Mhenni, F.; Gerbino, S. A hierarchical set of SysML model-based objects for tolerance specification. In Proceedings of the IEEE International Symposium on Systems Engineering, Edinburgh, UK, 3–5 October 2016.
33. Benavent-Nácher, S.; Rosado Castellano, P.; Romero Subirón, F.; Abellán-Nebot, J.V. SYSML4TA: A SysML profile for consistent tolerance analysis in a manufacturing system case application. *Appl. Sci.* **2023**, *13*, 3794. [\[CrossRef\]](#)
34. Aguilera-Antolí, D.; Rosado-Castellano, P.; Benavent-Nácher, S. A Modelica library to simulate geometrical and dimensional deviations in process assemblies. In Proceedings of the 10th Manufacturing Engineering Society International Conference, Sevilla, Spain, 28–30 June 2023.
35. ASME. *Dimensioning and Tolerancing. Y14.5*; American Society of Mechanical Engineers: New York, NY, USA, 2019.
36. Mourtzis, D. Simulation in the design and operation of manufacturing systems: State of the art and new trends. *Int. J. Prod. Res.* **2020**, *58*, 1927–1949. [\[CrossRef\]](#)
37. Hallmann, M.; Schleich, B.; Wartzack, S. From tolerance allocation to tolerance-cost optimization: A comprehensive literature review. *Int. J. Adv. Manuf. Technol.* **2020**, *107*, 4859–4912. [\[CrossRef\]](#)
38. Zhou, S.; Huang, Q.; Shi, J. State space modelling of dimensional variation propagation in multistage machining process using differential motion vectors. *Trans. Robot. Autom.* **2003**, *19*, 296–309. [\[CrossRef\]](#)
39. Abellán-Nebot, J.V.; Liu, J.; Romero, F. Design of multi-station manufacturing processes by integrating the stream-of-variation model and shop-floor data. *J. Manuf. Syst.* **2011**, *30*, 70–82. [\[CrossRef\]](#)
40. Delchamps, D.F. *State Space and Input-Output Linear Systems*; Springer: New York, NY, USA, 2011.
41. Barbedienne, R.; Penas, O.; Choley, J.Y.; Rivière, A.; Warniez, A.; Della Monica, F. Introduction of geometrical constraints modeling in SysML for mechatronic design. In Proceedings of the 10th Europe-Asia Congress on Mechatronics, Tokyo, Japan, 27–29 November 2014.

42. Kernschmidt, K. Interdisciplinary Structural Modeling of Mechatronic Production Systems Using SysML4Mechatronics. Ph.D. Thesis, Technische Universitat Munchen, Munchen, Germany, 2019.
43. Nachmann, I.; Rumpe, B.; Wortmann, A.; Berroth, J.; Hoepfner, G.; Jacobs, G.; Spuetz, K.; Zerwas, T.; Guist, C.; Kohl, J. Modeling mechanical functional architectures in SysML. In Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, Virtual Event, Canada, 16–23 October 2020.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.