Consistent information flow between stakeholders in an development environment using a central model in SysML v2

Marvin Manoury[1,] , Jesko Lamm [2], Tim Weilkiens[2], Till Tschiltschke[1] and Theresa Riedelsheimer [1,]

[1] Fraunhofer IPK; marvin.manoury@ipk.fraunhofer.de
[2] FAS-Arbeitsgruppe der GfSE; fas@gfse.de

Abstract: Model-based systems engineering (MBSE) is intended to enable the transdisciplinary realization of a product across its entire life cycle. This paper presents the use of the SysML v2 API for integrative work across requirements, functional and physical architecture, MCAD design, and life cycle assessment (LCA). To demonstrate feasibility, based on the reference implementation of SysML v2 and existing CAD, LCA, and architecture tools, the necessary extensions were implemented to realize all of the aforementioned steps from different workstations around a central SysML v2 repository. The demonstration with these tools and an example system shows that SysML v2 is suitable for realizing the stated transdisciplinary vision.

Comment/Material: This paper was created in June 2024. Further material is published at https://publica.fraunhofer.de/handle/publica/483736

1. Introduction

A competitive systems development project requires state-of-the-art development work across all participating disciplines and, at the same time, a shared understanding, which is fostered, for example, by Systems Engineering (SE). The transdisciplinary information flows required across all lifecycle phases are, in Modellbasierten Systems Systems Engineering (MBSE), ideally ensured via a shared model repository to which various tools of the development disciplines — for example, programs for mechanischen Computer-Aided Design (MCAD) — are connected, which is possible, for instance, via the Application Programming Interface (API) of SysML v2 [1]. In this paper, an end-to-end development centered around a model repository is investigated using an example system and several representative development tasks for a set of exemplary lifecycle phases. A commercially available consumer drone serves as the example system, whose system analysis and architecture definition as well as mechanical design and preparation of an Ökobilanzierung (LCA) are to be carried out as a case study in a model-based manner using SysML v2. After refining the objective of this paper, the chosen approach is explained and the results of the case study are presented and evaluated.

2. Approach

To achieve a realistic scenario with both geographically distributed and heterogeneously configured stakeholder workstations, the development tools to be used were distributed across the authors' computers (workstations). A reference implementation of the SysML v2 API running on the internet served as the central repository (a publicly accessible server operated by Intercax http://sysml2.intercax.com:9000). It was accessed for read and write operations from workstations in Germany and Switzerland. The various tasks in the exemplary development were distributed among the members of the author team to achieve a representative approximation of reality in a company with different departments. One team member worked on the definition of the baseline architecture, the requirements, and the physical architecture; a second team member handled the use case analysis and the application of the FAS method to derive the functional architecture; and a third team member took care of CAD and LCA tasks. Thus, over multiple write transactions, the model in the repository alternately served the work of different team members. The development tools used comprised the following software solutions:

- Two tools developed specifically for this work in Jupyter Notebook and Python for creating a new project in the model repository and for processing models specified in the textual notation of SysML v2. Using these tools, the baseline architecture was first written to the repository (see Figure 1), and later the system requirements and the physical architecture were added to the same project in the repository, in each case by copying the relevant lines in textual notation from a publicly available model of the example system [2] and pasting them into an input field of the tool. The second tool has an additional function for generating traceability from system elements of the physical architecture to the functional blocks of the functional architecture. The corresponding mappings are provided to the tool as input, whereupon, when saving the physical architecture to the repository, it also writes the relationships to the functional architecture into the same project.

doi: 10.24406/publica-4229

We assume that in the future SysML v2 modeling tools will offer these functions integrated.

- The FAS plugin, prototypically implemented for SysML v2 according to [3], which in a first step models the graphically entered use-case activities of the use cases as well as functional groups in SysML v2 and, in a second step, according to the FAS method, derives the functional architecture from them and establishes traceability to the functional groups using "dependency" relationships. The resulting artifacts are written to the same project in which the baseline architecture created in the first step described above already resides.
- A specifically adapted integration for the FreeCAD CAD program, based on [1], which can read SysML v2 projects and write adapted values to the model repository.
- The life cycle assessment software openLCA, with its associated open Python libraries olca-ipc and olca-schema

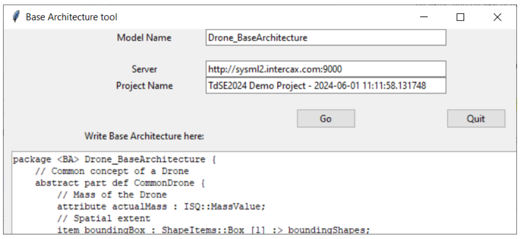

Figure 1: Repository editing tool with pasted baseline architecture in textual notation)

3. Results

At this point, in a top-down approach, the systems engineer's activity ends, and after the baseline architecture of the consumer drone has been created and written to the repository (Figure 1), the steps of the Functional Architectures for Safety (FAS) method for developing the functional architecture are carried out using the prototypical FAS plugin for SysML v2. The method and the plugin are described in detail in [4, 3]. Figure 3 shows the functional model of the consumer drone in an informal graphical notation according to [3]. From this input, the plugin can automatically derive the functional architecture.

In parallel with the development of the functional architecture, the system requirements and the physical architecture are modeled. The resulting model can be viewed online [2]. With one of the already mentioned Python tools, the new model elements are added to the project in the repository and traceability to the functional architecture is established (Figure 4).

At this point, for a top-down approach, the systems engineer's activity ends and the work continues within the domains. These domains typically do not possess systems engineering knowledge and therefore have an incomplete understanding of the system architectures. Since, in MBSE, these form the foundation for interdisciplinary collaboration and contain information relevant to the various domains, it is necessary to integrate this information as far as possible into the domains' native applications, thereby supporting the achievement of digital continuity [5]. In the following, the mechanical and life cycle assessment domains are considered. In the case of MCAD, in the FreeCAD application a dialog for interaction with the model is invoked via a macro (see Figure 2). In this dialog, the requirements with target values for the mass and maximum height of the drone are read. After the mechanical design of the drone is completed while meeting the height requirement, the exact dimension values are written into the SysML v2 model. Subsequently, based on the maximum mass value of the drone, the material for the body is determined by testing various materials for the drone body with respect to their impact on mass. The selected material (aluminum) is written to the SysML model together with the mass of the main body (1609 g). In the SysML v2 modeling tool, it can be checked whether the requirements on dimensions and mass are satisfied. In SysML v2, requirements are special evaluable assertions. In the sustainability assessment domain, an LCA is to be performed for the consumer drone. It is important to note that the SysML integration is not intended to perform an automated LCA, but to prepare the elaboration of an LCA. An LCA quantifies the environmental impact of a product system and its entire value chain on the environment [6]. A simple procedure for an LCA can proceed as described here [7, 8]:

1. First, the lifecycle phases considered in the LCA and the assessment goal (considered impact categories) are defined. Common delimitations of lifecycle phases are Cradle-to-Gate (from raw material extraction to production), Cradle-to-Grave (from raw material extraction to end-of-life of the product), or Cradle-to-Cradle (reuse in a new lifecycle). By selecting the impact categories such as global warming potential, eutrophication, acidification potential, and ozone depletion, the environmental impacts to be considered are specified. Since the example concerns a consumer drone and usage data cannot be evaluated, a Cradle-to-Gate boundary is appropriate. For the focused impact category, global warming potential is selected, as this is relevant as part of CO_2 reporting.
2. Next is the definition of the functional unit, which specifies the reference unit of the system under consideration. This ensures comparability of different product systems, which can be scaled to the same

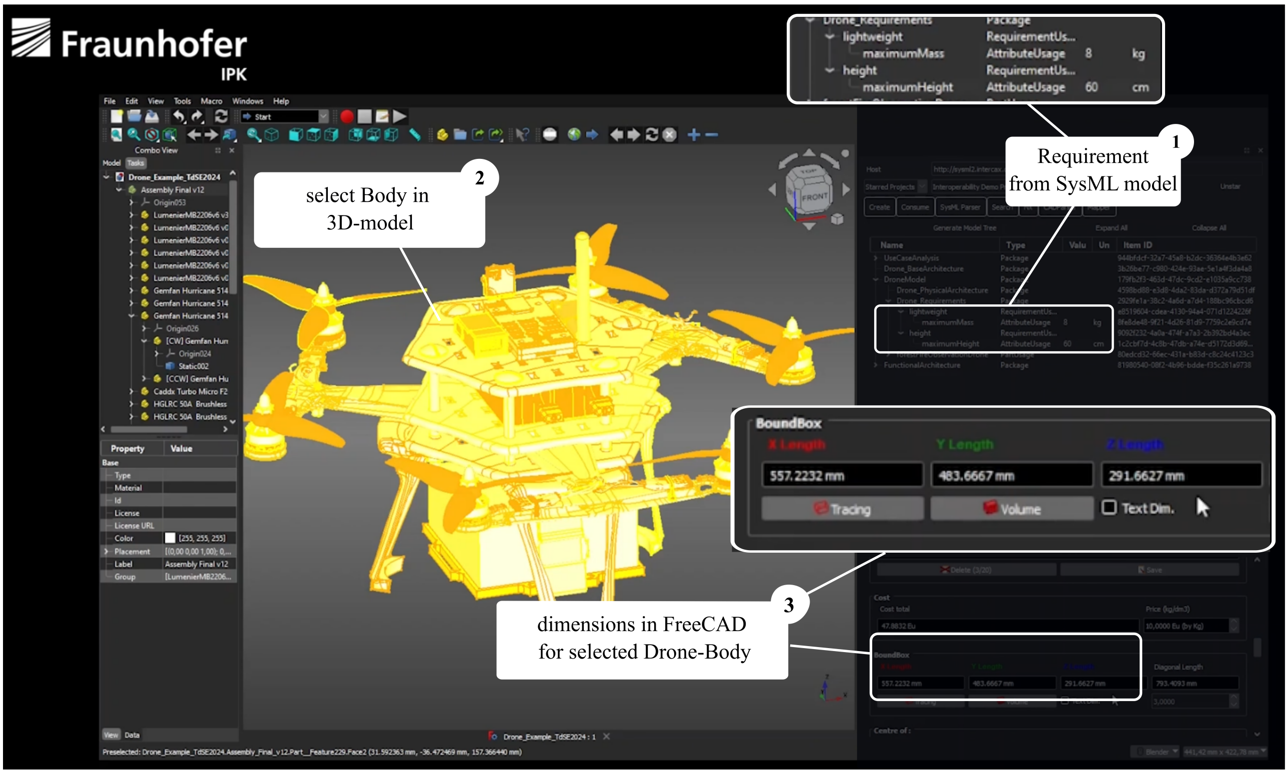

Figure 2: MCAD application FreeCAD with a dialog for interacting with the SysML v2 model on the Intercax server via the API.

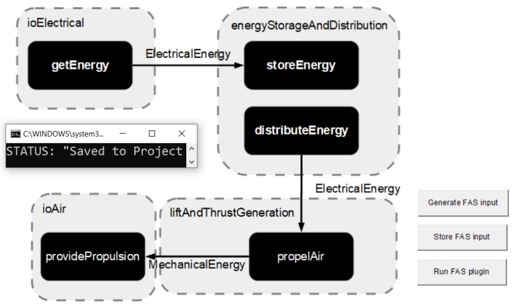

Figure 3: The prototypical FAS plugin for deriving and storing the functional architecture

reference unit within an LCA. A conceivable functional unit for the system considered here would be: The production of a drone with a maximum flight time of 1 hour and a total service life of 2000 flight hours.

3. For the functional unit, a Lifecycle Inventory (LCI) is created, which includes the inputs and outputs during the considered lifecycle. Creating the LCI comprises the actual data collection for the LCA. The ecological inputs and outputs are captured and modeled as flows. In openLCA [9], as an example application, a distinction is made between

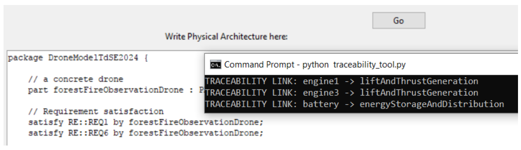

Figure 4: The prototypical FAS plugin for deriving and storing the functional architecture

- Elementary flows: elementary connections such as materials (e.g., aluminum) or emissions (e.g., carbon dioxide) (material or energy that has not been processed by humans),
- Waste flows: any substances and by-products that must be disposed of and leave the product system (e.g., plastic waste), and
- Product flows: flows that connect process phases, e.g., the exchange of energy between an energy generation process and an energy consumption process.

While generic lifecycle data, such as raw material extraction and supply chain (e.g., CO2 emissions along transport routes by ship and truck), are obtained from databases such as ecoInvent or GaBi, specific lifecycle data, such as the type and quantity of materials used, can be derived from the system

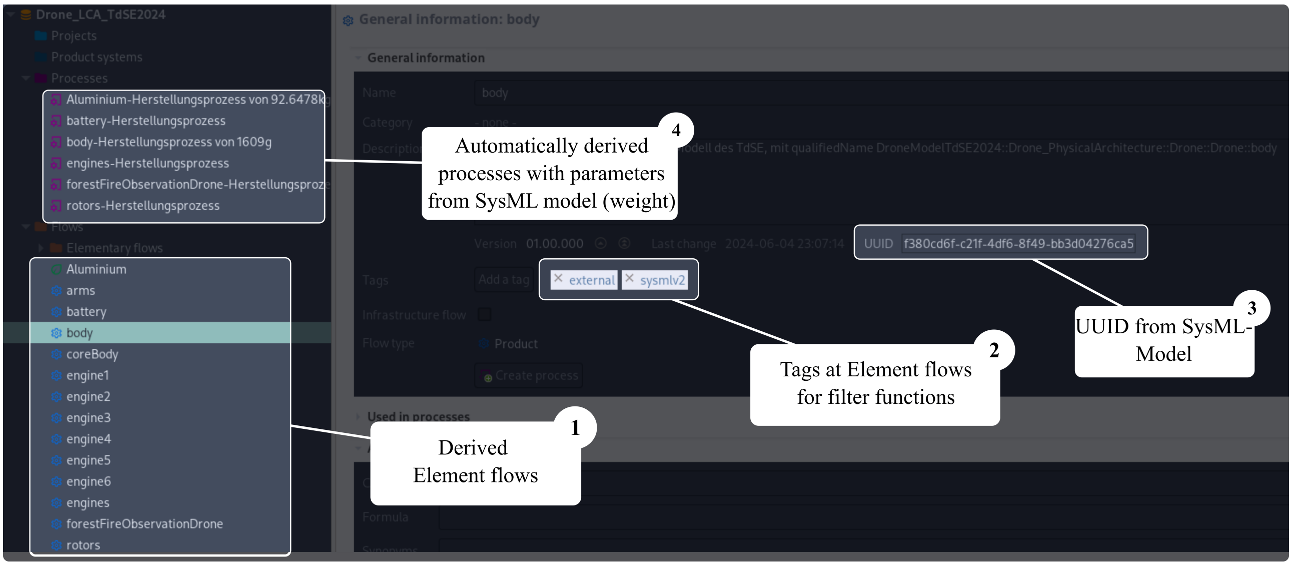

Figure 5: View in openLCA with flows created from the SysML model

model. This is where the integration in the example comes into play.

4. In the subsequent phase of impact assessment, which—after the integration shown in the example—is carried out manually, the contributions of the captured elementary flows to the selected impact categories are quantified. For this purpose, scientifically grounded characterization factors are used, which quantify the contribution of an elementary flow (e.g., methane) to the formation of an impact indicator (e.g., kg CO_2-equivalent). The accumulation of impact indicators in openLCA is performed by including so-called impact assessment methods, which contain all relevant characterization factors. After selecting the method, the calculation runs automatically.

5. Finally, as part of evaluation and interpretation, the LCA results are assessed. For this, openLCA offers extensive capabilities to compare the contributions of individual process phases to the considered impact categories and to identify corresponding hot spots. A plausibility check regarding expected results and statements is also advisable. Subsequently, the LCA results are transferred into the intended reporting format.

Via a Python tool, all instances modeled as parts are read from the SysML model. In the openLCA tool, an IPC server is started that enables direct communication with the model. These instances are initially integrated into the openLCA model as product flows via this server. The identical ID is used in the SysML model and the LCA model, and a “sysml” tag is included to indicate the origin of the element and enable potential updates.

These can be seen in Figure 5. For the material flows, the material captured as an attribute is read from the model, and the material as well as processes for the production of the products are created. On this basis, further processes can subsequently be created manually and an LCA prepared. The result of the LCA can be used to provide recommendations on material choice or material quantities. For these steps, reference is made to the LCA literature, such as Bassam et al. [10] for the integration of LCA and MBSE via SysML v1.

4. Discussion, Conclusions, and Outlook

This paper has shown how systems engineers and domain disciplines can fundamentally collaborate via the API. However, some points remain open.

There is a multitude of additional information that is relevant to individual domains but should not be modeled explicitly in the SysML model one by one so as not to overload it. These include, for example, data sheets with performance data of physical components or material compositions. Here, the concept of external references should be applied to integrate pointers in the SysML model as the single source of truth, while not overburdening the model itself.

The example in this paper is structured in a waterfall manner — no iteration currently takes place. In the LCA and MCAD domains, brief examples were given of how iteration can look or be supported (e.g., by using identical UUIDs), but the actual implementation and testing remain subjects of future work. In the collaboration, the topic of roles and access rights management was intentionally omitted. This is currently a relevant topic among the tool vendors for SysML v2 and can be found in other contributions at TdSE and from the implementation partners.

References

[1] Marvin Michael Manoury and Christian Muggeo. “Praktische Anwendung der SysML v2 API am Beispiel von MCAD und Simulation”. In: *Tag des Systems Engineering - Würzburg 15.-17. November 2023* (2023). Ed. by Walter Koch et al. DOI: 10.24406/PUBLICA-2394.

[2] Weilkiens, Tim. *Modell Einer Freizeitdrohne in SysML V2*. 2024. URL: https://github.com/MBSE4U/SysMLv2JupyterBook/blob/master/TdSE2024/DroneModelTdSE2024.ipynb.

[3] Tim Weilkiens, Jesko G. Lamm, and Axel Berres. “Systemfunktionen Mit Der FAS-Methode Im Griff Behalten: Bewährtes Aus Der Praxis Und Neues Aus Der Theorie"”. In: 2023.

[4] Tim Weilkiens et al. *Model-Based System Architecture, Second Edition*. Wiley, Apr. 2022. DOI: 10.1002/9781119746683.

[5] Marvin Manoury. “Functional Architecture for Solution Independent Realizations of Digital Continuity in System Development”. In: *2023 International Interdisciplinary PhD Workshop (IIPhDW)*. 2023 International Interdisciplinary PhD Workshop (IIPhDW). Wismar, Germany: IEEE, May 3, 2023, pp. 1–6. DOI: 10.1109/IIPhDW54739.2023.10124410.

[6] Thomas Kruschke, Uwe Kaufmann, and Kai Lindow. “Kann Systems Engineering eine nachhaltige Produktentwicklung unterstützen”. In: Tag des Systems Engineering 2022. 2022.

[7] Brian M. Neuberger. “An Exploration of Commercial Unmanned Aerial Vehicles (UAVs) Through Life Cycle Assessments”. In: 2017.

[8] Sinéad Mitchell et al. “Evaluating the Sustainability of Lightweight Drones for Delivery: Towards a Suitable Methodology for Assessment”. In: *Functional Composite Materials* 4.1 (Apr. 2023). DOI: 10.1186/s42252-023-00040-4.

[9] GreenDelta GmbH. *Flows - openLCA 2 Manual*. URL: https://greendelta.github.io/openLCA2-manual/flows/ (visited on 10/12/2025).

[10] Hamza Bassam et al. “A Model-Based Methodology for Life Cycle Assessment from Cradle-to-Grave Early in Product Design”. In: *Procedia CIRP*. 34th CIRP Design Conference 128 (Jan. 1, 2024), pp. 662–667. DOI: 10.1016/j.procir.2024.07.058.