

# A Holistic Integral Safety Assurance Approach to Aircraft Digital System Development Process <sup>†</sup>

Loris Dal Lago <sup>1,\*</sup>  and Janet J. Liu <sup>2</sup><sup>1</sup> Collins Aerospace, Via Praga, 5-Loc. Spini, 38121 Trento, Italy<sup>2</sup> Collins Aerospace, 400 Collins Rd. NE M/S 108-206, Cedar Rapids, IA 52498, USA; janet.liu@collins.com

\* Correspondence: loris.dallago@collins.com

<sup>†</sup> Presented at the 14th EASN International Conference on “Innovation in Aviation & Space towards sustainability today & tomorrow”, Thessaloniki, Greece, 8–11 October 2024.

**Abstract:** The maturation of system engineering practices to support the regulatory landscape and handle novel application domains, such as hybrid-electric distribution systems, calls for new ways to support development assurance objectives and end-to-end safety. In this paper, we build on the concept of a knowledge base of safety assurance methods to guide users in framing the problem space and discover existing and novel digital solutions to tackle safety assurance challenges. The proposed approach offers users with different skills, roles, objectives and experience ways to identify relevant use case solutions for traditional and novel aerospace applications.

**Keywords:** end-to-end safety; safety assurance; development assurance; aircraft digital systems; digital development; knowledge base

## 1. Introduction

The aerospace industry is driving towards the development of clean and sustainable aircrafts, with increasing demand to integrate new technologies, including hybrid-electric or hydrogen propulsion systems [1–3], as well as more complex systems that implement different levels of autonomy [4]. As the regulatory documents evolve, new challenges to the development of trustworthy methods for system design and development arise in addressing new aspects of aircraft designs [5,6], still requiring the safety aspects for aircraft equipment and systems to be considered, coordinated and maintained from the earliest phases of development. As an example, the definition of hybrid-electric aircrafts poses new challenges to their safety assurance. It also increases the option space for trade-off analyses due to the different choices for achieving aircraft safety [7]. For this reason, companies are required to build strong cases for airworthiness in new contexts, adapting existing practices to the challenges of an evolving technological and new regulatory landscape, with the intent of identifying as much as possible safety-related risks early in the development lifecycle.

Under these premises, in this paper, we describe a holistic and integral approach to supporting safety assurance practices for aircraft system development. The approach is holistic, as it views and integrates different methods and technologies end to end, accounting for the safety considerations at different phases of development. It is integral in that it enables the incorporation of safety aspects in lockstep with system development practices, also supporting system evolution, as exemplified in ARP4754B (Section 5).

To address the emerging landscape of safety assessments, our proposed approach collects existing challenges and solutions to forge them together in a holistic inventory



Academic Editors: Spiros Pantelakis,  
Andreas Strohmayer and Nikolaos  
Michailidis

Published: 14 March 2025

**Citation:** Lago, L.D.; Liu, J.J. A  
Holistic Integral Safety Assurance  
Approach to Aircraft Digital System  
Development Process. *Eng. Proc.* **2025**,  
*90*, 46. <https://doi.org/10.3390/engproc2025090046>

**Copyright:** © 2025 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/>).

available to users. We implement this kind of inventory with an evolving and searchable knowledge base that combines information from different application domains from past and existing programs. The knowledge base will consist of use cases, activities and their associated inputs (needs), outputs (deliverables), preconditions (dependencies) and constraints (limitations) and the tools needed to execute given tasks. The retrieval of information from the knowledge base guides users to recognize suitable solutions and to derive plans to satisfy the given safety assurance objectives, supporting effective reasoning about the context of operation, intended functions and failure mode definitions, with their propagation and effects in an aircraft system and its equipment.

The intended user base for the proposed approach includes system engineers, safety engineers and program managers. The first objective is to facilitate users with different skills, roles and objectives in identifying the relevant use cases for their applications. The knowledge base is used to frame the problem space and to select the recommended processes, tools and digital technologies. The second objective is to support users in the definition of new unexplored methods and investigate partially known or unfamiliar settings, such as those of the aforementioned novel technology contexts. We will provide an example of how the inspection and navigation of the knowledge base can support the definition of new methods and show how they can be later included in the knowledge base so that they can be made available to the next similar application.

Given the prominent growth of digital methods for handling the increasing complexity of systems, the specific knowledge base discussed in this paper is captured in the SysML language to facilitate its adoption by the digital engineering community with a familiar language and integrate it with existing emerging frameworks for safety analyses [8–10]. The idea of using SysML to answer engineering questions is not new. In [11], Graves uses axiom sets to represent questions and make them amenable to automatic reasoning, including the high-level validation of aircraft performance or the validation of consistency during design change. Our approach does not relate to questions about designs and systems under development but on the strategies and methods for assessing their safety. Also, our approach uses few restricted concepts to create the knowledge base, with possibilities for its extension. In [12], the authors consider systemic and socio-technical aspects in their holistic approach to safety assurance, with concepts similar to those in [13,14]. Although this is not the focus of our proposed work, it would be interesting to assess our methods with the use cases identified by that line of work and to see whether they could support the identification of suitable methods.

This paper is organized as follows: Section 2 overviews the proposed method, discussing the needs, structure, use and update of the knowledge base to support the holistic digital safety approach. Section 3 discusses an example with a prototype implementation of the knowledge base, arguing how the SysML language is instrumental to capturing the relevant concepts and being able to navigate the model efficiently. Section 4 is dedicated to discussions, and Section 5 concludes this paper.

## 2. Definition of the Method

### 2.1. The Master Steps of the Knowledge Base Approach

As discussed in the introduction, the method proposed in this paper builds upon the use of a knowledge base as an inventory to guide users to select the approach to finding the solution for a given use case with the required process steps and tool selection. The knowledge base is set to capture safety information consistently in a structured, formalized way and to consolidate problem resolution patterns based on experience from previous and ongoing company programs. In return, it provides means for users to understand the use of proposed technologies in the context of their previous applications and for program

managers to plan efforts based on experience data, increasing the predictability of program executions. Once defined and maintained, the knowledge base is to be considered an evolving artifact, with updates coming from more recent experiences, tools, methods and processes to keep the knowledge base up to date.

Under this paradigm, the typical steps for a user would constitute (1) knowledge retrieval, to identify the problem space and extract the methods and tools for performing safety analyses of the corresponding needs; (2) knowledge use, to adapt the retrieved methods and integrate them with the specific needs at hand, generating problem solutions; and (3) knowledge update, to improve the knowledge base with the new problem definition and proposed solution for future retrieval and use. We reference steps 1–3 as the *master steps* of the knowledge base approach.

## 2.2. Structure of the Knowledge Base

As part of the definition of the challenges to be addressed by the safety assurance methods and corresponding solutions, the knowledge base defines a number of concepts, which are represented in Figure 1. Definitions of all of the represented concepts are detailed in Table 1, along with examples, and commented on below in this section. Note that the account of *user stories* and *use cases* is inspired by their long-standing use in agile software development practices [15], whereas all of the other concepts are considered novel to the present work.

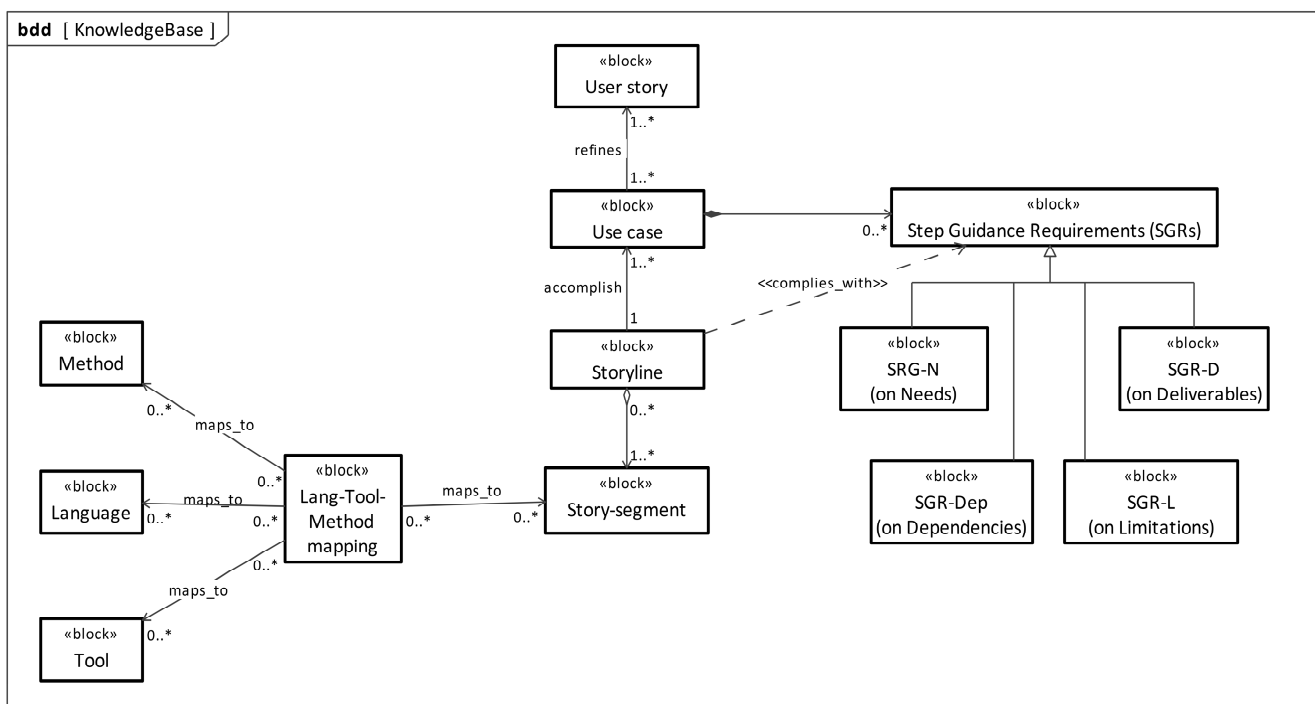


**Figure 1.** Main concepts available of the knowledge base and their relationship.

*User stories* help to pin down user needs using generic and simple statements, which are then refined into more complex needs by the definition of *use cases* and their associated *Step Guidance Requirements (SGRs)*. *Use cases* are accomplished through *Storylines*, each of which realizes the narrative of one possible solution to realize that *use case* and is required to comply with the corresponding *use case SGRs*. We note that depending on the internal company processes and policies, the level of detail at which *SGRs* and *Storylines* are represented may vary. In general, it is a program-specific exercise to find a good fit for these items. The *use case* and the *SGRs* exemplified in Table 1 are intentionally simple.

**Table 1.** Definitions and examples of the concepts delineated in this section.

| Concept                            | Definition and Example                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| User Story                         | High-level description of what the user of a knowledge base may want or be assigned to deliver, expressing an end goal for a user using the pattern “As a . . . I want to . . . so that . . .”.<br><i>[UserStory1] As a safety engineer, I want to perform a PSSA (Preliminary System Safety Analysis) so that I can comply with ARP4761A</i><br><i>[UserStory2] As a system engineer, I want to perform safety-related trade-off analyses so that I can efficiently support product design and development</i>                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| Use Case                           | Program-specific usage scenario to determine the contingent, articulated and particular needs of users. Each <i>use case</i> refines one up to a multitude of <i>user stories</i> , and each <i>user story</i> may be refined by one or more <i>use case</i> .<br><i>[UC-HE-PSSA-01] Perform a PSSA for a hybrid-electric distribution system, involving legacy components for low-voltage distributions.</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Step Guidance Requirement (SGR)    | The unrolling of a <i>use case</i> into the steps required to achieve the described scenario, including the definition of inputs (needs), outputs (deliverables), preconditions (dependencies) and constraints (limitations). More than one SGR is typically constructed for any one given <i>use case</i> , although each SGR is assigned to at most one <i>use case</i> , contributing directly and uniquely to its definition.<br><i>[SGR01] (Input/Need) The PSSA shall consider failure conditions from the SFHA (System Functional Hazard Analysis) as the inputs, according to ARP4761A (Section D.2)</i><br><i>[SGR02] (Output/Deliverable) The PSSA shall generate preliminary fault trees as the outputs, according to ARP4761A (Section D.2)</i><br><i>[SGR03] (Precondition/dependency) After the failure condition functional mapping phase, the PSSA shall evaluate the system design against failure conditions ARP4761A (Section D.4)</i> |
| Storyline                          | Sequence and/or interleaving of <i>Story-segments</i> , aggregated based on user experience to accomplish narratives and realize solutions for one or more <i>use case</i> . Storylines are required to comply with the SGRs of each involved <i>use case</i> .<br><i>To accomplish use case [UC-HE-PSSA-01], the architecture of the hybrid-electric distribution system is modeled using Story-segment [SSeg-PSSA-FT]. The generated fault trees are improved iteratively as the components of the architecture are identified, in compliance with [SGR01] and [SGR02]. FMEA tables are created according to Story-segment [SSeg-PSSA-FMEA] and used to feed the basic events of the fault trees. Fault trees are reviewed in compliance with [SGR03].</i>                                                                                                                                                                                              |
| Story-segment                      | A sequence and/or interleaving of actions and individual steps, defined as unit in scope (self-contained), separately testable and reusable. <i>Story-segments</i> constitute the building blocks for the definition of <i>Storylines</i> , but they exist independently of <i>Storylines</i> .<br><i>[SSeg-PSSA-FT]The process for creating PSSA fault trees is as follows:</i><br>1. Identify the relevant failure conditions and hazards of the system;<br>2. Build a model description of the system;<br>3. Annotate the system components with faults/fault propagations;<br>4. Analyze the system fault propagations, building the fault tree top-down from the failure conditions and hazards to the faults events.                                                                                                                                                                                                                                |
| LTM (Language–Tool–Method) mapping | A mapping of <i>Story-segments</i> to the Languages (Ls), Tools (Ts) and Methods (Ms) available for performing the required steps. Each <i>Story-segment</i> may be realized via different choices of language, tools and methods, and each language, tool or method may support the realization of different <i>Story-segments</i> .<br><i>The creation of a fault tree for Story-segment [SSeg-PSSA-FT] can be supported by the AADL (Architecture Analysis and Design Language) [16], using the EVM2 standard [16] (Annex E), following the method in [17] (Section 4). Alternatively, it can be supported using an AADL tool editor that supports the AADL safety annex [18] (Section 3), using the method as derived in [19].</i>                                                                                                                                                                                                                    |

The definition of *Storylines* is obtained by sequencing and interleaving existing procedural building blocks, called *Story-segments*, which exist independent of *Storylines* and represent units of reusability. *Storylines* and *Story-segments* represent user solutions and are required to be language/tool/method-agnostic. *Languages, Tools* and *Methods* (LTMs) are separately mapped to *Story-segments*, which complete the implementation narrative when aggregated into *Storylines*. It is up to *Storylines* to harmonize the use of different LTMs organically to attain the prescribed goal. In a typical scenario, different *Story-segments* are associated with similar *LTM mappings*, with low variability in the languages (Ls) and tools (Ts) to make the harmonization between different *Story-segments* in a *Story-Line* easier and mainly based on aligning the underlying methods (Ms). The rationale and link to past programs may be included in the *LTM mappings* to highlight recommendations and the maturity benefits and cost of one *LTM* solution over another, allowing users to select the most suitable solution based on the program needs.

We further note that the separation between *use case* definitions, *Story-segments* and *LTM mappings* makes the approach suitable for identifying development opportunities or understanding how existing LTMs can be employed to accomplish existing or new *use cases*. In fact, in the process of aggregation, *Storylines* may identify the lack of *Story-segments* for the accomplishment of a given *use case* or gaps in the knowledge base. These gaps may be fixed by expanding the knowledge base with the relevant information to constitute new *Story-segments*. Similarly, *Story-segments* not associated with *LTM mappings* realize development opportunities, which can be handled separately in a company *LTM* development or adoption process. The specifics of this part, including aspects related to tool maturation and deployment, are considered out of scope for this work.

### 2.3. Mapping the Concepts to the Knowledge Base Master Steps

With the availability of the knowledge base structure, the master steps of Section 2.1 guide users to retrieve, use and update its content. First, *user stories* relevant to the needs at hand are identified, and one or more *use case* is written to refine them. In this context, existing *user stories* guide the identification of the relevant *use cases* and the corresponding SGRs, constraining the problem space definition. After that, the *Storylines* accomplishing the existing *use cases* are inspected to help determine a new *Storyline* for the *use case* at hand. This new *Storyline* may reuse and repurpose the *Story-segments* available in the knowledge base, along with the available *LTM*s. In this phase and depending on the novelty of the *use case*, it is possible for *Story-Segment* gaps and/or *LTM* development opportunities to be identified. Finally, information on the *use case*, the new *user stories* and *Storylines* and new and old *LTM mappings* is added to the knowledge base to complete the *knowledge update* master step.

Under this paradigm, every user has the possibility to search and employ the information of the knowledge base, query the knowledge base, retrieve the process steps and tool options for a particular use case, scout different alternatives and then update its content based on success stories from their application at hand. Clearly, in contrast to the use of free text in the illustrative examples in Table 1, pre-defined formal structures can help to coordinate the searchability, use and updates of the knowledge base. We discuss in the next section a prototype implementation of the knowledge base in the SysML language, with a corresponding use example.

## 3. The Prototype Implementation and Use Example

### 3.1. Handling Knowledge Base Complexity with SysML

The limited information contained in the simple example in Table 1 is enough to demonstrate the type of complexity that can arise when the data for the knowledge base are captured. For example, *use cases* and *Story-segments* may be referenced by *Storylines*, and the relationships between these notional concepts may be recorded to make the *knowledge retrieval* master step easier. To capture and maintain the information in the knowledge base, we captured the different concepts in the knowledge base digitally in the SysML language so that they could be navigated easily and be brought together consistently.

We selected the SysML language for several different reasons. As an industry standard for capturing system architectures, SysML is familiar to the engineering community, making it readily available to the user base. It also contains language structures and diagrams (e.g., “use cases” and “activity diagrams”) which can readily support the constructs defined in this work (*use cases* and *Storylines/Story-segments*, respectively). Moreover, it offers relevant features such as SSOT (Single Source Of Truth) representations of the involved elements and the possibility to relate them and handle them in digitized form. Customization features, such as profiles, allow us to define the required concepts swiftly, change them as necessary and create validation rules. Moreover, such concepts (such as the *use cases*, *Storylines* and *Story-segments*) can be directly associated with the program artifacts captured in SysML (such as *System Requirements*, *System Functions*, *System Architectures* or *Traceability information*). The capability to reference the same objects as SSOTs is key to consolidating knowledge with consistency and fostering alignment between the users of the knowledge base. Also, this facilitates querying the model to retrieve relevant data, such as the set of *Story-segments* involving the handling of a particular artifact. Lifecycle process information can be customized and captured as meta-data to determine the status of the involved *use cases* and steps in a *Storyline* or track to the maturity of *Story-segments* and relative *LTM*s. The identification of lifecycle steps depends on company policies and practices and thus is not considered in scope of this work.

### 3.2. The Use Example

For the reasons outlined in the previous section, we devised a SysML profile to capture the information on the knowledge base concepts and used it on several examples at a small scale. A simple example of this is displayed in Figure 2.

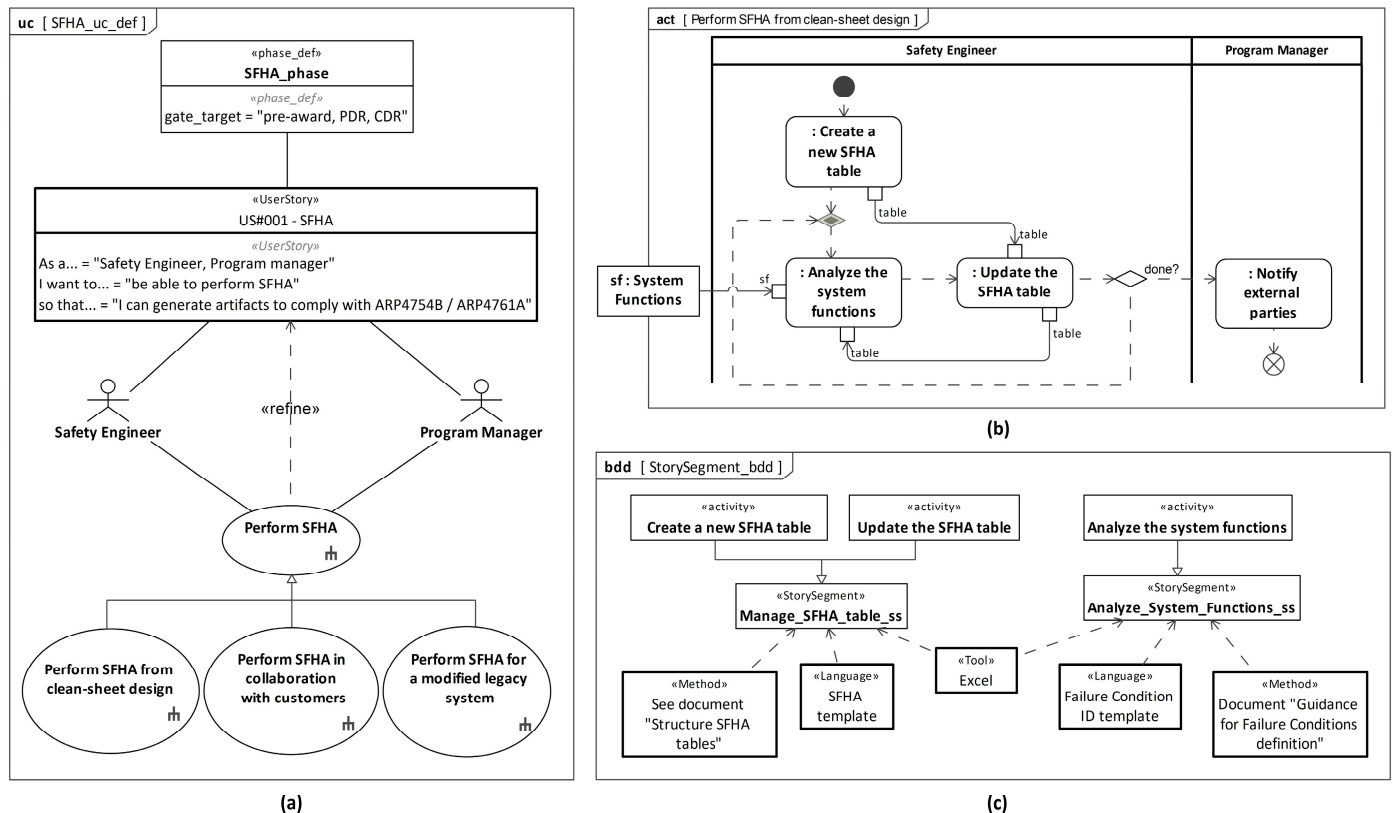


**Figure 2.** Example SysML representation of (a) System Functional Hazard Assessment (SFHA) use cases with the user story and phase, (b) the Storyline accomplishing the “Perform SFHA from clean-sheet design” use case and (c) mapping of the Storyline actions (b) to Story-segments and relative LTMs.

In Figure 2a, the problem definition is represented, with *user story* “US#001–SFHA” and three different use cases. The flexibility of the SysML language allows us to derive *use cases* from a common one, called “Perform SFHA”, which refined the *user story*.

Note that operating based on the expressivity of the language, we also added two actors, namely “Safety Engineer” and “Program Manager”, and related them to both the *user story* and the parent *use case* “Perform SFHA”. The user story is also associated with a specific phase in the safety process, called “SFHA\_phase”, in the diagram in Figure 2a, with a specific associated tag called “gate\_target”. This demonstrates how the proposed framework can also be used to relate elements outside of the knowledge base’s standard concepts. Engineers or program managers interested in performing the SFHA on a system of interest will compactly have all of the relevant information in this diagram. From here, they can navigate to the *Storylines* accomplishing the corresponding *use case* (following the rake symbol on the SysML use case), track the role played by the “SFHA\_phase” for the rest of the model or simply create new *use cases* previously not considered (for example, “Perform SFHA for a hybrid-electric distribution system”).

Figure 2b shows a possible *Storyline* accomplishing the *use case* “Perform SFHA from clean-sheet design”. This is represented as a SysML activity diagram (simplified from a more complete one for illustration purposes). Here, the roles of the involved actors are represented using the two swimlanes. Inputs (such as the “System Functions” in this case) are represented

as SysML Activity Parameters, coherently with the *Input-SGRs* of the accomplished *use case*. The “Safety Engineer” actions are expressed as subsuming the *Story-segments* represented in Figure 2c, which themselves are shown to be separately mapped to *Language*, *Tool*, and *Method* primitives. Noticeably, the running tool in this case is “Excel”. Being common, it is represented as an SSOT, helping to harmonize the sequencing of the *Storyline* in Figure 2b. The action “Notify external parties” in the *Storyline* only represents a contribution to the narrative of the *Storyline* and does not entail the existence of a corresponding associated *Story-segment*. This demonstrates how the approach is made flexible so as to allow compliance with the company-specific required steps, policies and rules.

#### 4. Discussions

The demonstrated example illustrates the capabilities of the presented SysML customization in capturing relevant information and reasoning about safety, as well as the methods for performing safety analyses. The current work includes *user stories* about trade-off analyses, the comprehensiveness of the analyses, the generation of specific safety artifacts, capture of reliability information and reporting capabilities, among others. In our experience, the presented set of concepts constitutes a minimal yet usable set that fosters flexibility and concept reusability while minimizing the burden on the users and maintainers of the knowledge base. The knowledge base concepts proposed in this work can be refined and expanded in the future as additional needs are identified. For example, concepts such as stakeholders, user roles and skills are not included in the current version of the framework, although possible extensions are considered part of future work.

This approach could be adapted to cases beyond safety. We started with safety because (1) it is a central concept to the aerospace industry, making it a primary candidate for holistic integral views; (2) various digital- and model-based approaches to supporting safety assessments at different levels exist in the literature to be combined and harmonized; and (3) novel technologies, such as hybrid-electric propulsion systems, would benefit from having a knowledge base built on past experience and proposing the recommended actions by adapting existing technologies to new domains. Of particular interest is the framing of the impact of complex conditions on the system behavior driven by external unpredictable factors, which may require analyses to account for aspects of SOTIF (Safety Of The Intended Functionality).

This paper presents a prototype, with future work aimed at enhancing its scalability and applying the approach to complex, industry-relevant case studies. Currently, the SysML implementation of knowledge base searchability is manual and based on model navigation. To address scalability, we plan to investigate techniques to increase the automation and improve user experience. Employing more sophisticated technologies for knowledge base management, such as relational databases [20], knowledge graphs [21] and graph databases [22], could make the approach more practical. Additionally, to efficiently query the knowledge base with engineering questions, we are exploring the use of ontologies [23], which could capture knowledge base concepts in a structured format, thereby enhancing the clarity and providing a basis for further automation opportunities.

The above enhancements may also support additional usability options. For example, by storing the weights associated with *Storylines* or *Story-segments* and by representing the time or cost for specific activities, program managers can query the knowledge base to access relevant time/cost information for their *use cases*. By querying the knowledge base, users can determine what is feasible within the given time or cost constraints, leveraging the captured experience. Consequently, users can estimate the progress achievable within a specified timeframe, such as in one month versus one year.

Finally, to support regulatory approval in certification processes better, we aim to refine the current approach of representing processes using *Storylines*. This could be achieved

by storing relevant certification objectives associated with the recommended approaches alongside the *Story-segments* in the knowledge base. This will enable users to exhibit these associations to authorities and generate reports for certification tasks.

## 5. Conclusions

With experience and historical know-how being at the basis of safety assessments practices, it is key to support development assurance objectives and end-to-end safety by recognizing solutions from expertise. The proposed approach helps to determine ways to use safety assessment methodologies, allowing users to find suitable technologies from the definition of the problem space, and identify solutions with more rigor, including them holistically as part of digital engineering practices.

This approach supports the integration of safety and development aspects into digital and traditional system engineering based on experience, where information is captured and iteratively updated in the knowledge base. For the prototype implementation of this framework, we used SysML as the front-end user language. Integrating knowledge graphs or relational databases could enhance the usability, streamline information retrieval and improve knowledge maintenance.

Reliable and user-friendly methods for accessing the costs, dependencies and required inputs to achieve a particular set of tasks or activities can support engineers and program managers in current programs, as well as in bidding on and planning and executing future programs, with the possibility of supporting the development of systems and safety arguments in novel domains. Users with different skillsets, roles and objectives can access the information according to their needs, finding solutions that can be part, holistically, of an all-encompassing integral safety view.

**Author Contributions:** Conceptualization: L.D.L. and J.J.L. Methodology: L.D.L.; Writing—original draft: L.D.L.; Writing—review & editing: L.D.L. and J.J.L.; Visualization: L.D.L.; Supervision: L.D.L. and J.J.L. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the European Union under GA no. 101101961—HECATE.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** No new data were created or analyzed in this study. Data sharing is not applicable to this article.

**Conflicts of Interest:** Author Loris Dal Lago and Janet J. Liu were employed by the company Collins Aerospace.

## References

1. AIR8678; Architecture Examples for Electrified Propulsion Aircraft. SAE International: Warrendale, PA, USA, 2022.
2. Rendón, M.; Sánchez R, C.; Gallo M, J.; Anzai, A.H. Aircraft Hybrid-Electric Propulsion: Development Trends, Challenges and Opportunities. *J. Control Autom. Electr. Syst.* **2021**, *32*, 1244–1268. [\[CrossRef\]](#)
3. Nishikawa, T.; Kenichi, R. A Comprehensive Study on Hydrogen and Hybrid Electric Aircraft with Distributed Electric Propulsion. In Proceedings of the AIAA SCITECH 2024 Forum, Orlando, FL, USA, 8–12 January 2024.
4. Aerospace Technology Institute. *The Journey Towards Autonomy in Civil Aerospace*; Aerospace Technology Institute: Bedfordshire, UK, 2020.
5. EASA. *Final Special Condition SC E-19—Electric/Hybrid Propulsion System—Issue 01*; EASA: Cologne, Germany, 2021.
6. EASA. *Proposed CM Ref. CM-21. A-004 Issue 01 on “Acceptable Approaches for the Certification of Electric/Hybrid Propulsion Systems”*; EASA: Cologne, Germany, 2024.
7. Cano, T.C.; Castro, I.; Rodríguez, A.; Lamar, D.G.; Khalil, Y.F.; Abiol-Tendillo, L. Future of Electrical Aircraft Energy Power Systems: An Architecture Review. *IEEE Trans. Transp. Electr.* **2021**, *7*, 1915–1929. [\[CrossRef\]](#)
8. Ross, R.; Hecht, M. *A SysML Profile for MIL-STD-882E (System Safety)*; INCOSE International Symposium: Detroit, MI, USA, 2022.

9. Biggs, B.; Post, K.; Armonas, A.; Yakymets, N.; Juknevičius, T.; Berres, A. *OMG Standard for Integrating Safety and Reliability analysis into MBSE: Concepts and Applications*; INCOSE International Symposium: Detroit, MI, USA, 2019.
10. Hect, M.; Chuidian, A.; Tanaka, T.; Raymond, R. Automated generation of FMEAs using SysML for reliability, safety, and cybersecurity. In *Proceedings of the 2020 Annual Reliability and Maintainability Symposium (RAMS)*, Palm Springs, CA, USA, 27–30 January 2020.
11. Graves, H. *Integrating Reasoning with SysML*; INCOSE International Symposium: Detroit, MI, USA, 2012.
12. Burton, S.; McDermid, J.A.; Garnett, P.; Weave, R. Safety, Complexity, and Automated Driving: Holistic Perspectives on Safety Assurance. *Computer* **2021**, *54*, 22–32. [[CrossRef](#)]
13. Leveson, N.G. *Engineering a Safer World: Systems Thinking Applied to Safety*; The MIT Press: Cambridge, MA, USA, 2016.
14. Wang, H.; Shao, W.; Sun, C.; Yang, K.; Cao, D.; Li, J. A Survey on an Emerging Safety Challenge for Autonomous Vehicles: Safety of the Intended Functionality. *Engineering* **2024**, *33*, 17–34. [[CrossRef](#)]
15. Cohn, M. *User Stories Applied: For Agile Software Development*; Addison-Wesley Professional: Boston, MA, USA, 2004.
16. *5506D*; Architecture Analysis & Design Language (AADL). SAE International: Warrendale, PA, USA, 2022.
17. Delange, J.; Feiler, P.H.; Gluch, D.P.; Hudak, J.J. *AADL Fault Modeling and Analysis Within an ARP4761 Safety Assessment (Tech. Rep. CMU/SEI-2014-TR-020)*; Software Engineering Institute, Carnegie Mellon University: Pittsburgh, PA, USA, 2014.
18. Stewart, D.; Liu, J.J.; Cofer, D.; Heimdahl, M.; Whalen, M.W.; Peterson, M. Architectural modeling and analysis for safety engineering. In *Model-Based Safety and Assessment: 5th International Symposium, IMBSA 2017, Trento, Italy, 11–13 September 2017*; Springer: Berlin/Heidelberg, Germany, 2017.
19. Stewart, D.; Liu, J.J.; Cofer, D.; Heimdahl, M.; Whalen, M.W.; Peterson, M. AADL-Based safety analysis using formal methods applied to aircraft digital systems. *Reliab. Eng. Syst. Saf.* **2021**, *213*, 107649. [[CrossRef](#)]
20. Harrington, J.L. *Relational Database Design and Implementation*; Morgan Kaufmann: Burlington, MA, USA, 2016.
21. Hogan, A.; Blomqvist, E.; Cochez, M.; d’Amato, C.; Melo, G.D.; Gutierrez, C.; Kiriene, S.; Gayo, J.E.; Navigli, R.; Neumaier, S.; et al. Knowledge graphs. *ACM Comput. Surv.* **2021**, *54*, 1–37. [[CrossRef](#)]
22. Ian, R.; Webber, J.; Eifrem, E. *Graph Databases: New Opportunities for Connected Data*; O’Reilly Media, Inc.: Sebastopol, CA, USA, 2015.
23. Uschold, M.; Gruninger, M. Ontologies: Principles, methods and applications. *Knowl. Eng. Rev.* **1996**, *11*, 93–136. [[CrossRef](#)]

**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.