[JGW+21] N. Jäckel, C. Granrath, L. Wachtmeister, A. Karaduman, B. Rumpe, J. L. Andert:
Feature-Driven Specification of VTOL Air-Taxis with the Use of the Model-Based Systems Engineering (MBSE) Methodology CUBE.
In: 77th Annual Vertical Flight Society Forum and Technology Display (FORUM 77)), pp. 2776-2784, Curran Associates, Inc., Mai 2021.
www.se-rwth.de/publications/

Feature-Driven Specification of VTOL Air-Taxis with the Use of the Model-Based System Engineering (MBSE) Procedure CUBE

Nicolas Jäckel
Systems Engineer
FEV Europe GmbH
Munich, Germany

Christian Granrath
Research Associate
RWTH Aachen University
Aachen, Germany

Louis Wachtmeister
Research Associate
RWTH Aachen University
Aachen, Germany

Abdulsamed Karaduman
Assistant Engineer
FEV Europe GmbH
Aachen, Germany

Bernhard Rumpe
Professor
RWTH Aachen University
Aachen, Germany

Jakob Andert
Professor
RWTH Aachen University
Aachen, Germany

ABSTRACT

The ground-bound transport systems in major cities are already reaching their limits. Vertical Take-Off and Landing systems, in short VTOL systems, possibly autonomous and electrified, are one preferred to use the urban space for public transport. Applying the methodology 'Compositional Unified system-Based Engineering', in short CUBE, enables the use of model-based systems engineering (MBSE) to develop appropriate VTOLs. This methodology can be used to develope from abstract artifacts to concrete architecture and from system to sub-systems development. In this paper, the CUBE methodology will be displayed with a feature-driven specification to demonstrate its general usability. The model-based description of the System of Interest (SoI, meaning VTOL) from an abstract use-case to a technical architecture is exemplarily showcased with three features. This methodology approaches to create system-based and feature-based models with a tailored level of abstraction for a better handling of complexity.

INTRODUCTION

The traffic situation in big cities has become more and more problematic in recent decades. Especially ground transport systems are already reaching their limits in many places with continuous global urbanization. However, individual mobility is a key factor for the quality of life in urban areas whereas the expected increase in demand will push conventional ground-based transportation to its limits. This status quo will result in pollution, noise, and traffic congestion. One solution could be urban air mobility in addition to existing transport systems. The use of aerial taxi services has the potential to unlock the urban sky for public transport. Vertical Take-Off and Landing (VTOL) systems with electrical propulsion are very promising candidates because of their relatively low noise emission and the little space requirements on the ground. The VTOL systems can be integrated into the existing urban traffic system to reduce the amount of ground traffic, while also reducing the time to travel a defined distance. The enablers for a VTOL breakthrough are recent technology improvements in battery technology, low noise emitting electric motors and automation technology, which enable a suitable system development and performance. The VTOL systems promise a safe, quiet, and significantly more cost-effective operation, especially in highly frequented routes, e.g., from downtown to the airport [1]. A systematic approach to improve the profitability of air taxi concepts has shown the many parameters influencing the economic performance [2]. The introduction of such air taxis in an existing urban infrastructure is depending on many different stakeholders and external boundary conditions, i.e., norms and legal questions, security and safety aspects, passenger comfort, and customer acceptance [3]. Further engineering is required to address the challenges arising in the development of such complex and networked systems [4]. Certain case studies have demonstrated the usability and feasibility of VTOLs to improve the quality of live in urban areas, i.e., in Dubai [4], Munich [5], San Francisco [6] and in the Silicon Valley area [7].

The application of Systems Engineering (SE) is a suitable approach to cope with the development complexity by addressing the totality of all requirements occurring in the different phases of the system life cycle [8].
An essential procedure for the appropriate handling of complex systems is partitioning. This means that the System of Interest (SoI), which contains all its elements, is embedded in the System of Systems (SoS) containing the context of the SoI described in other systems. In the presented case, the SoS is the existing air-taxi environment containing all

Presented at the Vertical Flight Society's 77th Annual Forum & Technology Display, Palm Beach, Florida, USA, May 11-13, 2021.

stakeholders like the operation control center, the airport, other aerial and ground objects and the people. The SoI is the VTOL air-taxi as a system. In the following we describe the methodology from an abstract description of the system to an exemplarily technical solution for one feature. This artifact-based development is one major advantage using systems engineering fundamentals [9].

CUBE METHODOLOGY

The model-based systems engineering (MBSE) methodology 'Compositional Unified system-Based Engineering' (CUBE) [10] helps to master the complexity of the problem and to build the right product to solve this challenge (Fig. 1). A systematic exploration of existing structures and the possibility of using the urban airspace are the first steps to understand constraints and new features to ensure the satisfaction of all relevant stakeholders [4]. This approach can be carried out to focus on certain specific parameters of the system or to ponder the impact of a design decision [11]. Moreover, the performance of the VTOL systems depending on the design and the typical speed of travel must be considered [12]. More general and more abstract views are also needed to gain more information about the system under development [13].

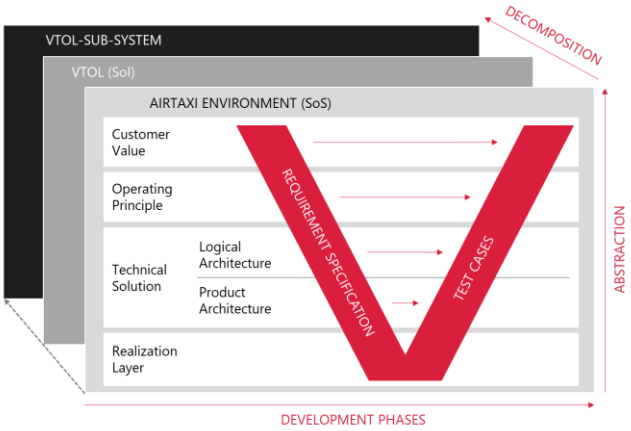

Figure 1. Methodology of CUBE [14] **with its three dimensions.**

The methodology CUBE was introduced with the three main dimensions development phases, abstraction, and decomposition [14]. In addition, CUBE is characterized by the innovative combination of feature-driven development with conventional systems engineering methods. A feature is understood in the further context as a small, functional, and independently editable part of the system, which provides added value for the customer. In parallel with improved complexity control, this combination enables continuous enhancement of the system specification. Another major advantage of CUBE is the cost reduction in the later development phases, since frontloading ensures a high quality of requirements from the outset. The requirements are directly traced to test cases in all abstraction layers. When design flaws occur, it is therefore easy to find all affected artifacts, and the sustainable troubleshooting of the fault is manageable. Further, the ability to test system parts and the entire integrated system separately reduces the time and costs for verification and validation. The domain-independent applicability has already been successfully demonstrated using the example of the development of automated vehicles [15] or also XiL simulation models of electric vehicles [16]

CUBE FOR VTOL SYSTEMS DESIGN

In the last paper of the authors at FORUM 76, high-level requirements in the most abstract view for SoS and SoI of VTOLs were introduced together with a model-based description in form of use-case diagrams [14]. With these requirements and use-cases, further development can be performed in the most effective way to integrate this new transportation technology into the existing urban transit system. In this paper, the practical application of the CUBE methodology regarding the SoI of a VTOL system in the more concrete layers is shown and the possibility of feature integration through detailed presentation of exemplary system features is illustrated. The requirements regarding the topics "VTOL positioning broadcast" and "flight state information" will be focused to show how the requirements of the most abstract level ('Customer Value') can be used to further refine the development.

The term frontloading describes the MBSE approach with more focus in the early developing artifacts and is a key to reduce verification and validation time in the later phases. Developing a system based on models requires more work in the beginning but saves much more time and money for testing [8]. Therefore, the generation of the most abstract artifacts is a major challenge to ensure a better design of the final product. Starting point is the generation of a use-case diagram, the identification of all relevant stakeholders for the air taxi environment, the SoS, and the creating of high-level requirements. Continuing in lateral direction of the CUBE, the first layer for the SoI is also described with a set of use-cases and derived requirements.

FEATURE-DRIVEN SPECIFICATION OF VTOL SYSTEMS BASED ON CUBE

Use-cases and high-level requirements

The full use-case diagram for the SoI contains 13 actors and 51 use-cases, described in the appendix figure A1 based on ref [14]. This illustrates the complexity of the problem and the need for a further systematic clustering. The use-cases are related to derived requirements, which are displayed in a requirements diagram (Fig. 2).

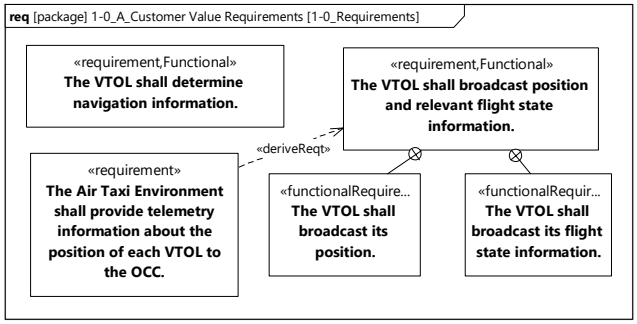

Figure 2. Requirement Diagram for navigation and position information broadcast requirements

The functional requirement 'The VTOL shall broadcast position and relevant flight state information' contains the two atomic requirements, displayed in the diagram with the nesting connection. The derived features are 'Broadcast Position' and 'Communication', based on the requirement for 'broadcast flight state information'. Another feature 'Determine Position for Navigation' is derived from the requirement 'The VTOL shall determine navigation information'. Those features can now be integrated into the use-case diagram and all relevant use-cases are allocated to the feature. This is exemplarily shown for the two features 'Determine Position for Navigation' (Fig. 3A) and 'Broadcast Position' (Fig. 3B).

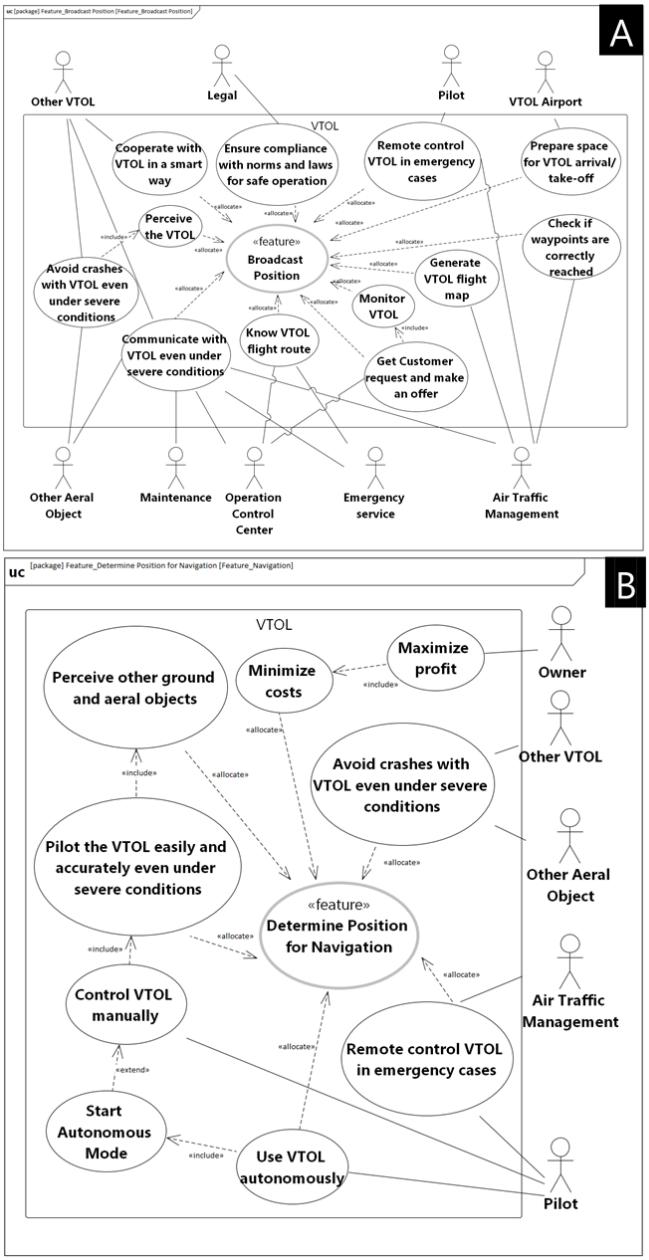

Figure 3. Use-Case Diagrams with relevant use-cases and stakeholders, and including the definition of features a: 'Broadcast Position' and b: 'Determine Position for Navigation'

With this allocation, there are also all relevant stakeholders for each feature identified and can be considered for the further feature-specific specification of the VTOL system.

Operating Principles for the VTOL Features

After definition of the relevant stakeholders and use cases as part of the customer value, the identified and allocated features are further specified in form of feature specific 'Operating Principles'. Following the SysML centric approach presented in the running example of this paper, for each feature allocated in the feature allocation presented in figure 4, a SysML activity diagram is to be modelled, to specify the features intended functionality.

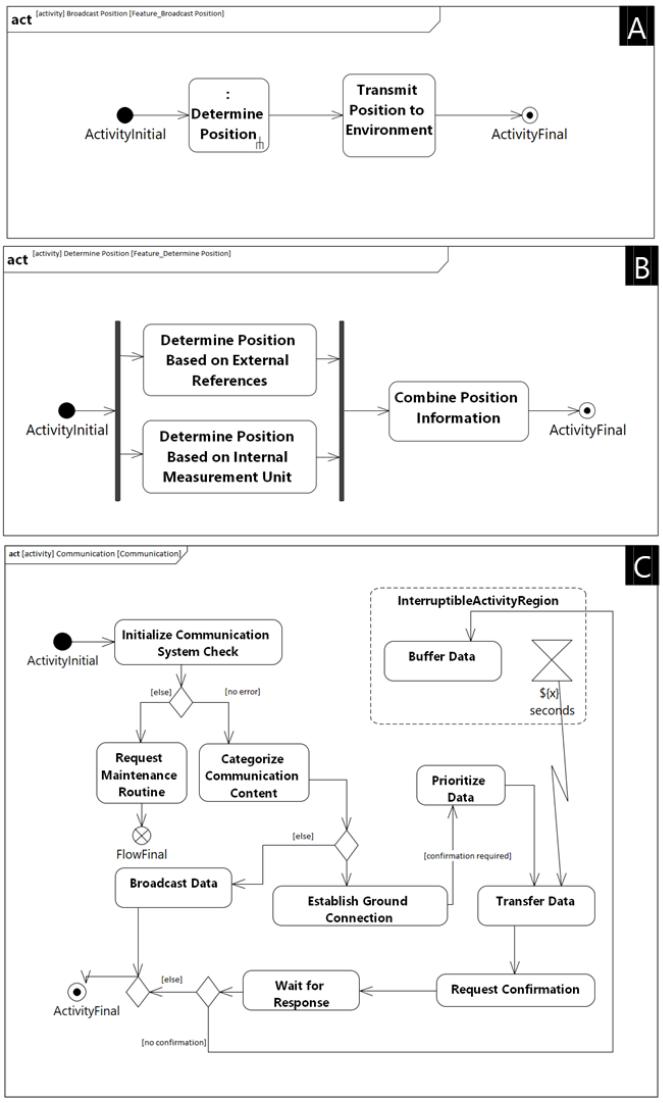

Figure 4. Activity diagrams containing activities and control flows for the three features a: 'Broadcast Position', b:'Determine Position', c: 'Communication'

As described in the previous section, the features described in this paper are: 'Determine Position', 'Broadcast Position', and 'Communicate Position'. To describe how these features are to be implemented, the 'Operating Principle', describing the solution-neutral way how these features are to be implemented, are specified as SysML activity diagrams.

As described in the requirements (Fig. 2), the VTOL is required to broadcast its position to its environment. The VTOL requires a feature to broadcast this information to its environment as concluded from the according use cases (Fig. 3). To describe how this feature operates on an abstract and solution neutral level, the following two steps are required. First, the VTOL must determine its position. Second, the VTOL must transmit this position to its environment. Using these abstract and solution-neutral actions, a SysML activity diagram can be modelled in which these required steps are the two main actions as shown in figure 4a. After the control flow is specified, there are primarily two possible ways to continue. On the one hand, one could specify the required object/information flows in addition to the control flows, as depicted in the appendix (Fig. A2). On the other hand, the control flows could be left underspecified to allow a maximum of reusability and flexibility for in the specification of the logical architecture. While both approaches have their advantages, both also have drawbacks. If object flows are defined as activity parameters and action pins in the activity diagram as a next step, the resulting activity diagram has the advantage that all required information flows are presented in one view. Additionally, the direct specification of activity parameters allows a good reuse on activity level, as required and provided interfaces are directly visible in the graphical representation. A major drawback of this approach is that the diagrams for complicated interactions with many inputs and outputs are quickly overloaded. In addition, the ruse of different variants becomes more difficult as interfaces of the functional system architecture are already included in this view. Especially, when textual interface requirements are written in parallel for each action or known from previous development projects, the introduction of additional object flows has also the drawback that these requirements are to be maintained redundantly to the action pins and activity parameters. As our running example presents a rather simple view on the actions with only a few interactions, we chose the first option in this work and present the resulting diagrams in figure 4.

Since this feature 'Operating Principle' uses a determine position action, the corresponding 'Determine Position' feature as SysML activity diagram is described in figure 4b. This feature describes a possible way to implement a position determination, that uses a combination of internal measurements and external references to determine the position. As this feature has no communication specific parts, it may also be used in the specification of navigation tasks that rely on a positioning functionality. By using this rather abstract description, the specification intentionally abstracts away from concrete implementations, such as satellite navigation systems (for external reference positioning) or inertial measurement units (for internal measurement positioning). By this it is possible to change and reuse the specification more easily, if a different system is to be developed or a later engineering analysis shows that the initial performance assumptions of the technical are not enough to fulfill the non-functional requirements. Of course, this

description is also not fully solution-neutral, as at least two additional possibilities exist to implement the positioning task. First by using internal measurements alone, and second by only relying on external references. As these problems often occur in systems engineering applications, the CUBE Method introduces design constraints at this point, which enable the systems engineer to express his reasons why this 'Operating Principle' was favorized over the other possibilities. For the positioning task for example, this constraint would be that experiences, benchmarks and technical analyses with the currently available technical solutions show that neither external references (as these signals can be easily distorted or modified [17]) nor internal measurements (due to measurement errors and drift effects [18]) alone are sufficient to precisely and reliably determine a moving objects position in a real-world environment.

To implement the second task, the communication feature, combines possibilities to transmit information from the VTOL system to its environment. The corresponding SysML activity diagram is presented in figure 4c. First the communication system is checked. If the communication system reports errors, a maintenance request action is triggered, and an information is returned. Thereafter, the information is categorized in two categories. In the first communication kind, which is also required to transmit the position data, the data is simply broadcasted to the environment, in the second communication kind, the communication feature requests a response and buffers all information it receives before a confirmation was received.

Actions \ Logical Components	Communication	External Reference Positioning	Internal Positioning	Position Coordinator
Broadcast Data	x			
Buffer Data				x
Categorize Communication	x			
Combine Position Information				x
Determine Position		x	x	x
Determine Position Based on External References		x		
Determine Position Based on Internal Measurement Unit			x	
Establish Ground Connection	x			
Initialize Communication System Check	x			
Prioritize Data				x
Request Acknowledgement	x			
Request Maintenance Routine	x			
Transfer Data	x			
Transmit Position to Environment	x			
Wait for Response	x			

Figure 5. Allocation overview between logical components and actions. The logical components need to fulfill the allocated actions.

After the actions for the features were specified, the actions are allocated to logical components that fulfill these tasks. As graphical representations are usually hard to read in industry size projects, the allocation from actions to logical components can also be presented in allocation tables as shown in figure 5. For the previously introduced features, four logical components were identified to perform the required tasks. The communication block handles everything related to the information communication and thus most of the actions presented in figure 6. For better reusability, the positioning tasks are split into components for 'Internal Positioning', 'External Reference Positioning', and 'Position Coordination'.

Logical Architecture Elements as VTOL Feature Realization

Based on the operating principle, a logical architecture for each feature is obtained, by allocating each action from the feature to a logical block that performs this functionality. This also implies that the functional decomposition on a decomposition level is only finished, if all actions can be unambiguously mapped to logical blocks, which serve as the next decomposition level. In case of our running example, this means that the VTOL system would have 'Internal Positioning', 'External Reference Positioning', 'Position Coordination', and 'Communication' as sub-systems. Therefore, the logical architecture not only significantly influences the overhead required in the systems engineering approach, but also influences the reusability of the system elements for other technical solutions. For example, if the 'Communication' was not modeled as a logical block, one would save a layer for communication functionality at the cost of having to model all communication-related functionality as functionality of the other systems.

Apart from its structural purposes on system level, the logical architecture also specifies the system interfaces on a logical information level. This means that in contrast to the technical layer, which e.g., specifies concrete radio frequencies or information encodings used for communication, the logical architecture only specifies which information is communicated between the different logical blocks. It is necessary to remain on a logical information level to enable reuse based on the solution-neutral description, since a concretization of the interfaces would rob degrees of freedom in the technical implementation. If the interfaces are directly added to the actions as action pins and activities as shown in the appendix (Figure A2), these interfaces can be directly derived from the information received or transmitted by the actions allocated to the logical block. If the specifying engineer decided not to specify the interface, she has to derive them in the creation of the logical architecture based on her knowledge of the in- and outputs of these interfaces and to achieve traceability by allocating the ports of the logical architecture to the activities that require these interfaces. Regardless of the modeling variant selected, all interface information must always be included in the requirements. Therefore, the information is always present and must be represented as a part of the Logical Architecture in both modeling variants of the Operating Principle. Since a logical block may implement more than one action, the logical block may also contain other ports as interfaces, that are not required by the feature, but it always requires all information the action receives or transmit to implement its functionality. Since an action should not be implemented by multiple logical components, as this would mean that the action itself is not decomposed enough, all interfaces follow from the feature specification.

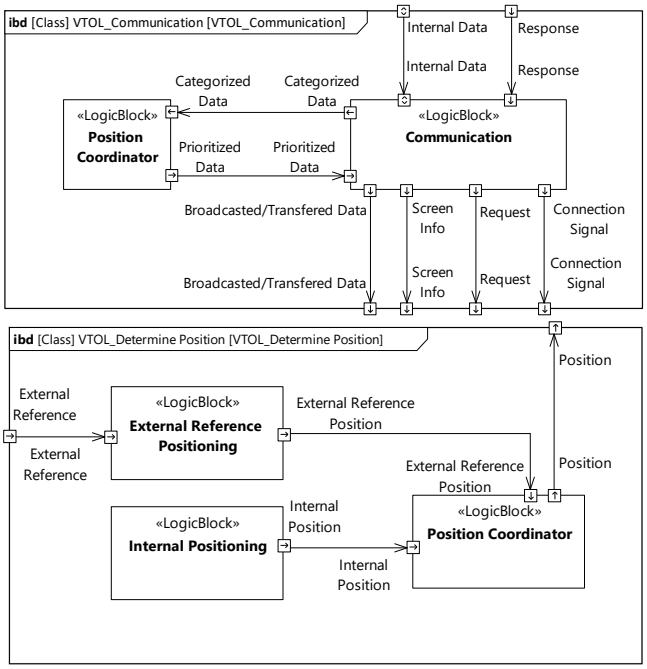

Figure 6. Internal Block Diagrams for the features a: 'Communication', and b: 'Positioning' describing the logical signal flows and the logical architecture.

For the 'Communication' (Fig. 6a) and 'Broadcast Position' (Fig. 6b) features, the feature specific logical blocks are shown as SysML internal block diagram in Figure 7. In these diagrams, a feature specific view is presented in which all logical blocks and interfaces are shown that are required to implement the feature. To indicate that logical blocks are considered, the blocks of the internal block diagram have the stereotype «LogicBlock». For the communication feature, in the image above, all communication is handled by the 'Communication' block, which receives e.g., position information from the 'Position Coordinator'. For the 'Determine Position' feature shown below, the 'External Reference' is received as input of the feature specific view and transformed by the 'External Reference Positioning' logical block to a 'External Reference Position', which is then combined in the 'Position Coordinator' with the 'Internal Position' from the 'Internal Positioning' bock to an overall 'Position' transmitted to the outside world of this feature specific view.

While the feature specific views have their advantages in the feature specification, on system level, often an overall view is needed, to specify how the feature realizations work together as system on a logical level. Thus, figure 7 presents an overall systems view on the logical blocks specified by all features presented in this paper. This view can be obtained by merging all feature specific views into an overall representation.

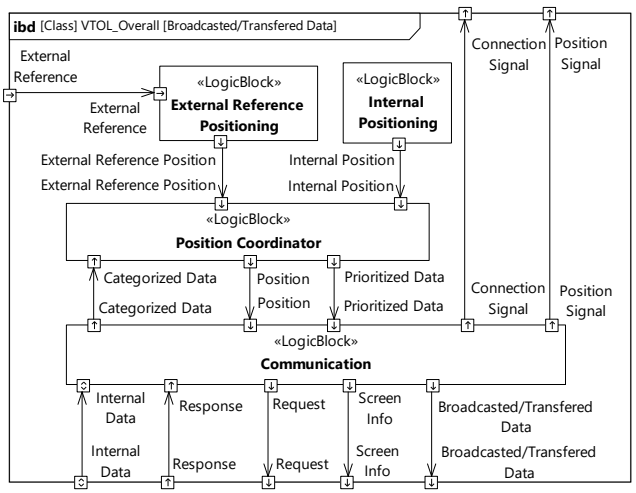

Figure 7. Internal Block Diagram of an system view containing all feature specific views, signals and architecture elements.

Because the logical view is not refined enough to build a VTOL, the logical blocks must be mapped to technical components in the product architecture in order to achieve a product specification. For the Logical Architecture presented in this section, the allocation is described in Figure 11 and the Product Architecture is provided in the next section.

Product Architecture Specification

The Product Architecture describes the concrete hardware or software implementations of the elements in the Logical Architecture and therefore describes the technical elements required to build the overall system. By distinguishing between the Logical and the Product Architecture, the systems engineer is enabled to specify a concrete realization of the product under development, without the need to commit to a concrete technical implementation at an early development stage. For that reason, a reuse of the logical elements is possible even when the technical constraints that lead to the product architecture change during the long development cycles of the aviation industry, or when new variants of the original product are developed and brought into the market or when components are outsourced to suppliers. The specification of the logical elements remains unaffected, no matter which realization and which producer have been selected. Especially, when complete features are reusable, the presented approach aids in reducing the development effort.

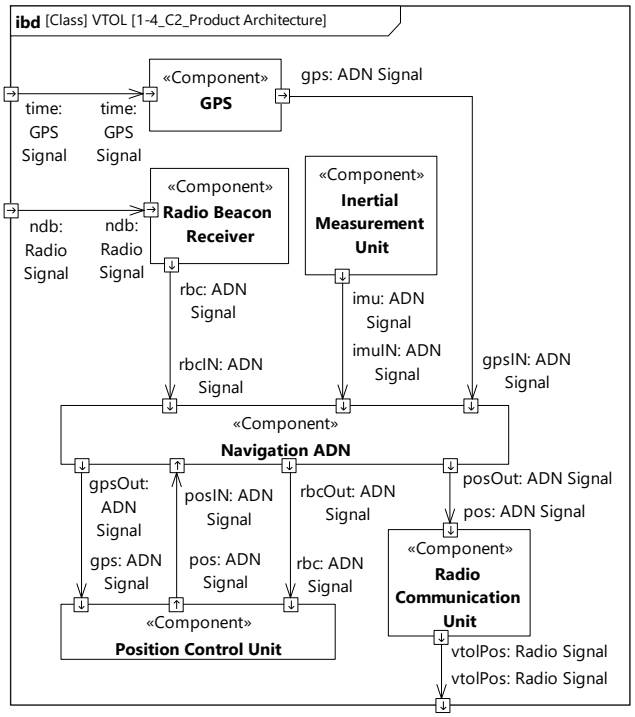

Figure 8. Internal Block Diagram of the Product Architecture describing the technical realization of components.

For the logical blocks of the position broadcasting functionality, the resulting Product Architecture is presented in figure 8. To distinguish the technical elements from the logical elements, all blocks in this internal block diagram have the have the stereotype «Component». In this architecture, the 'External Reference Positioning' is realized by two components, a GPS component and a radio beacon receiver to increase the accuracy of the GPS signal. In addition, the 'Internal Positioning' is performed by an 'Inertial Measurement Unit'. The position communication is then implemented by a 'Radio Communication Unit' while the combination of the now three position signals is handled by a 'Position Control Unit'. Finally, the internal communication is implemented by an Aircraft Data Network (ADN) in the diagram shown as the 'Navigation ADN'. Even though, the product architecture is rather small, it already shows some of the advantages of splitting the Logical and the Product Architecture in the Technical Architecture description (Fig. 9). By this, it is for example possible to also introduce region specific variants that use other satellite navigation systems such as the European Galileo System instead, or by changing the communication network more easily in later points of the product development.

Technical Components \ Logical Components	Communication	External Reference Positioning	Internal Positioning	Position Coordinator
GPS		x		
Inertial Measurement Unit			x	
Navigation ADN	x			
Position Control Unit				x
Radio Beacon Receiver		x		
Radio Communication Unit	x			

Figure 9. Allocation overview between Logical and Technical Components. This mapping shows which logical function/feature will run on which technical component.

For simplicity, the further mapping of these technical elements is not described in this paper, but it is possible to further parametrize the technical blocks in a model-driven systems engineering approach as described in [19], by deriving concrete instances of the technical blocks and connecting them to other elements of a CAx process chain in a virtual product design approach. To continue the system specification, either the specification of the subsystems can be continued on further decomposition levels, or a final specification of the realization (realization layer) of a system component can be performed. This is followed by the implementation. In addition, a test case must be defined for each requirement. By the presented approach it becomes therefore feasible to test purposefully on different levels (e.g. whole system or only individual components).

CONCLUSIONS

In this paper, the MBSE-based methodology 'Compositional Unified system-Based Engineering' (CUBE) helps to control the complexity of the problem of the integration of VTOLs in the already complex multimodal transportation systems of metropolises around the globe. The CUBE methodology combining established MBSE approaches with a feature-driven development helps to build the right product to solve such an exemplarily challenge.

The general usability of this approach is explained by describing the SoI (VTOL) in use-case diagrams regarding all major stakeholders. The requirements derived from those use-cases are used to develop major features of the system. Each feature is further described in the operating principle with activities. The logical and product architecture consequently developed to describe the technical solution for the required system.

This concept is one approach to create systematical and model-based requirements, architectures and specifications using the concept of abstraction and decomposition. The concept has the major advantage to reduce costs in the later development, since the frontloading is ensuring a high quality of specification artifacts. These can be directly traced to test cases in all abstraction layers. When design flaws occur, it is therefore easy to find all affected artifacts.

Author contact: N. Jäckel jaeckel@fev.com, C. Granrath granrath_c@vka.rwth-aachen.de, L. Wachtmeister wachtmeister@se-rwth.de, A. Karaduman karaduman@fev.com, B. Rumpe rumpe@se-rwth.de, J. Andert andert@mechatronics.rwth-aachen.de

APPENDIX

The appendix contains figure A1 with the full use-case diagram of the VTOL system and figure A2 displaying the activity diagram of the operating principle with control flows and signal flows.

REFERENCES

[1] K. I. Swartz, „Joby Transitions," *Vertiflite*, Bd. JAN/FEB, pp. 42-46, 2021.

[2] T. H. Ha, H. Lee und J. T. Hwang, „Large-scale design-economics optimization of eVTOL concepts for urban air mobility," in *AIAA Scitech Forum*, San Diego, California, 2019.

[3] T. Edwards und G. Price, „eVTOL Passenger Acceptance," NASA Ames Research Center, Moffett Field, CA, United States , 2020.

[4] K. A. Awadhi, A. Saleem, D. Heckmann, A. Nase und D. Moormann, „Paving the Way for Commercialization of Autonomous Aerial Taxis in Dubai: Key Lessons Learnt From First Demonstration Flight and Requirements Definition," in *26th ITS World Congress*, Singapore, 2019.

[5] M. Fu, R. Rothfeld und C. Antoniou, „Exploring Preferences for Transportation Modes in an Urban Air Mobility Environment: Munich Case Study," *Transportation Research Record*, Bd. 2673 (10), pp. 427-442, 2019.

[6] B. German, M. Daskilewicz, T. K. Hamilton und M. M. Warren, „Cargo Delivery in by Passenger eVTOL Aircraft: A Case Study in the San Francisco Bay Area," in *2018 AIAA Aerospace Sciences Meeting*, Kissimmee, Florida, 2018.

[7] K. R. Antcliff, M. D. Moore und K. H. Goodrich, „Silicon Valley as an Early Adopter for On-Demand Civil VTOL Operations," in *16th AIAA Aviation Technology, Integration, and Operations Conference*, Washington, D.C., 2016.

[8] S. Kriebel, J. Richenhagen, C. Granrath und C. Kugler, „Systems Engineering with SysML The Path to the Future?," *MTZ worldwide Vol. 79*, pp. 44-47, 2018.

[9] S. Hillemacher, N. Jäckel, C. Kugler, P. Orth, D. Schmalzing und L. Wachtmeister, „Artifact-Based Analysis for the Development of Collaborative Embedded Systems," in *Model-Based Engineering of Collaborative Embedded Systems*, Springer, 2021, pp. 315-331.

[10] C. Granrath, C. Kugler, P. Orth und J. M. Richenhagen, „Verfahren zur Spezifikation eines Produkts". Germany Patent DE102020004396A1, 10 09 2020.

[11] I. C. Kleinbekman, M. A. Mitici und P. Wei, „eVTOL Arrival Sequencing and Scheduling for On-Demand Urban Air Mobility," in *2018 IEEE/AIAA 37th Digital Avionics Systems Conference (DASC)*, London, UK, 2018.

[12] M. Shamiyeh, R. Rothfeld und M. Hornung, „A Performance Benchmark of Recent Personal Air Vehicle Concepts for Urban Air Mobility," in *31st Congress of the International Council of the Aeronautical Series*, Belo Horizonte, Brasil, 2018.

[13] G. Wilke, „Aerodynamic Performance of Two eVTOL Concepts," in *New Results in Numerical and Experimental Fluid Mechanics XII.*, Springer, Cham, 2018, pp. 392-402.

[14] N. Jäckel, C. Granrath, R. Schaller, M. Grothenrath, M. Fischer, P. Orth und J. Andert, „Integration of VTOL air-taxis into an existing infrastructure with the use of the Model-Based System Engineering (MBSE) concept CUBE," in *Vertical Flight Society 76th Annual Forum*, virtual, 2020.

[15] M. Kremer, C. Sébastien, C. Granrath und M. Meyer, „Scenario- and Model-based Systems Engineering for Highly Automated Driving," *ATZworldwide*, Bd. 122, pp. 16-21, 2020.

[16] C. e. a. Granrath, „A reference simulation model architecture and interface standard for modeling and test of electric vehicles,“ *eTransportation,* Bd. 4, p. 100060, 2020.

[17] D. Schmidt, K. Radke, S. Camtepe, E. Foo und M. Ren, „A Survey and Analysis of the GNSS Spoofing Threat and Countermeasures,“ *ACM Computing Surveys,* pp. 1-31, 2016.

[18] M. Kok, Hol, J. D. und S. T. B., „Using Inertial Sensors for Position andOrientation Estimation,“ *Foundations and Trends in Signal Processing,* pp. 1-153, 2017.

[19] M. Dalibor, N. Jansen, B. Rumpe, L. Wachtmeister und A. Wortmann, „Model-Driven Systems Engineering for Virtual Product Design,“ *Proceedings of MODELS 2019. Workshop MPM4CPS,* pp. 430-435, September 2019.