A SYSML-BASED APPROACH TO EXPLORING INNOVATIVE SYSTEM IDEAS FOR AERONAUTICAL APPLICATIONS

Jutta Abulawi*
***Hamburg University of Applied Sciences, Berliner Tor 9, D-20099 Hamburg, Germany**

Keywords: *systems modeling language, risk awareness in system design, graduate training*

Abstract

This paper presents an approach for exploring innovative system ideas using the Systems Modeling Language (SysML) and combining technology-oriented perspectives with risk- and user-oriented perspectives. It was developed to introduce senior aeronautical engineering students to methods commonly used in software-centric system design projects. The approach is also applicable for familiarizing professional aeronautical engineers with the SysML. Its objective is to foster awareness on risks caused by misuses and unintended interaction of a system with its environment and to support a user-centric perspective in system design. The approach uses abstract, discipline-independent graphical models and explanatory tables for documenting and communicating initial considerations and decisions, considering all phases of the entire life-cycle of a planned system. The approach is well-suited for use in early stages of multi-disciplinary development projects for innovative systems.

1 General Introduction

The Systems Modeling Language (SysML) [1] was developed to support the specification, design and verification of complex systems. Derived from version 2 of the Unified Modeling Language (UML 2), it reuses a considerable subset of language elements and extends them to meet the needs of systems engineers [2]. The SysML is a semi-formal, graphical language, supporting an object-oriented approach to the discipline-neutral structured analysis of complex systems. It depicts real and abstract objects as graphical nodes, e.g. named blocks, and models relations as graphical paths. All model elements and attributes are stored in a computer-interpretable format.

Despite its potential, there are still not many people who know the language well enough to use it professionally [3]. In particular, engineers from the mechanical domain are still reluctant to learn and use the SysML because the available literature describes language applications from a software-centric perspective or focuses on electrical/electronic systems. To make the language more attractive to engineers with a mechanical background, a simplified approach to using the SysML in early stages of system design projects has been developed at Hamburg University of Applied Sciences (HAW Hamburg).

Another objective of the approach is to make technically-minded engineers with a mechanical background more aware of possible unwanted effects resulting from the interaction of a system with its environment including all users and other actors. The approach will be explained in the following sections.

2 Starting Point

The need for developing a simplified approach to using the SysML was identified during the conception of a postgraduate course which introduces aeronautical engineering students to systems engineering. The students usually have a strong mechanical background, and only few have been familiarised with systems thinking in their undergraduate career.

To increase the sustainability of teaching and learning, a competency-based course design was used, without a written exam at the end of the course. Instead, a project-oriented method of teaching and assessing the students was chosen with the following learning outcome:

"*In small teams, students explore an innovative, complex system idea using systems engineering analysis and specification methods including SysML diagrams*."

In their projects, students practice systems thinking and analysis methods by exploring an innovative aeronautical system of their choice. As project deliverables, they prepare a set of SysML diagrams with accompanying tables, an initial system specification, a short report, and a final presentation. Where suitable, they support their work with well-established analyses like quality function deployment (QFD), design structure matrix (DSM), fault tree analysis (FTA), failure mode and effects analysis (FMEA) and other suitable methods.

To introduce students to the fundamentals of systems engineering and various aspects pertaining to its application in the aeronautical industry, 16 interactive, weekly lectures (three hours each) are offered, with voluntary participation. As background literature, [4] and [5] are recommended.

Students are encouraged to take a research-based approach to learning which actively engages them in inquiry and research. The lecturer assumes a mentoring role in supervising the projects [6]. The students are asked to experiment with analysis and documentation methods and assess their applicability with respect to the special nature of their system idea. They shall identify challenges and shortcomings and suggest improvements to the methods where suitable.

To assess the students' performance in the course, the deliverables are submitted as a portfolio. It includes a review of the project with lessons learned and suggested improvements to the course and the use of the SysML. Grading is performed with a set of pre-defined criteria and weighting factors which are explained to the students during the course.

3 Theoretical Framework for the Approach

The Vee model from [7] can be considered as an accepted way of modelling systems engineering projects, using a top down approach (cf. [4]) for system analysis and design, followed by system synthesis including the integration of the system into its environment. Figure 1 shows that it starts with requirements elicitation and analysis and proceeds with defining an operational concept and baseline architecture. The system is hierarchically decomposed into subsystems and components.

The beginning of the process requires a good openness for all technical disciplines to avoid unwanted bias at an early stage. Once the decomposition has reached the level where functions can clearly be allocated to technical solutions from specific disciplines, expert teams from these disciplines design their subsystems and components. Finally, the system is assembled in a bottom-up process until it is completely implemented and integrated. Design and synthesis are accompanied by continuous verification and validation activities at all levels of decomposition.

The first activities on the left hand branch of the Vee model need an unbiased perspective considering the system as a whole, without differentiating between software and hardware, i.e. mechanical and electrical components. This requires a discipline-independent approach to analysis and modelling.

As the SysML was developed to address this need [2], it is best to start its application in the early stage of system analysis and definition.

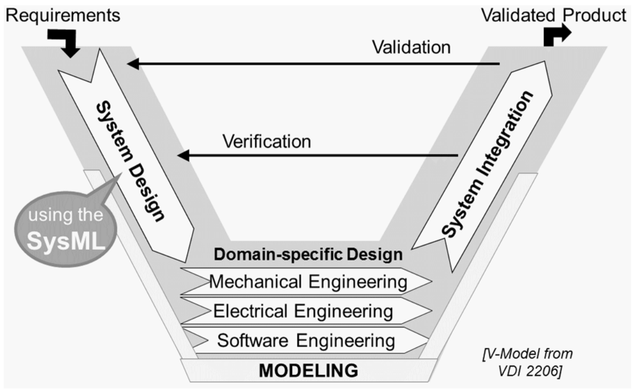


Fig. 1. The Vee Model depicting main stages in mechatronic system development processes

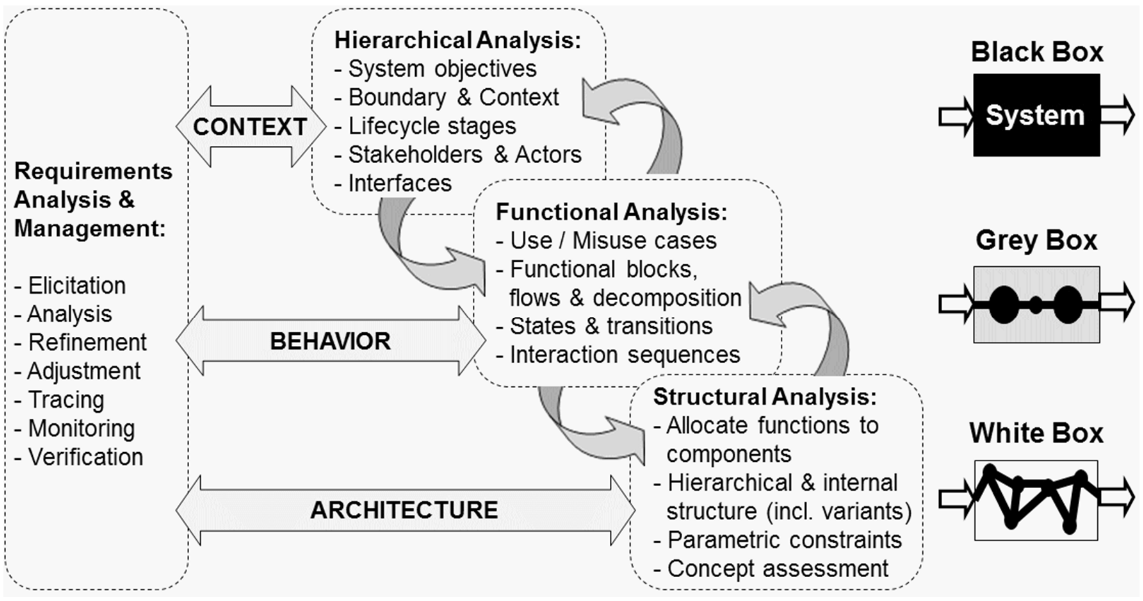


Fig. 2. The Conceptual Framework for the SysML-based System Exploration and Definition Approach

So far, there is no established standard methodology for using the SysML in model-based systems engineering [8]. Based on diagram examples and explanations given in [1], [9], [10] and [11], an elementary set of SysML language elements and diagrams was identified to be used by the students in their projects. To facilitate teaching and learning, a logical sequence of analysis and decision steps needed to be defined which was suitable to guide the students in the exploration of their system idea.

For holistic system analyses, Ropohl suggests three perspectives:

- functional concept,
- structural concept,
- hierarchical concept [12].

A slightly different but also three-fold categorization is defined by Haberfellner et al.:

- the context and its relation to the system,
- system effects => functions & reactions,
- system structure [4].

Both classifications address the analysis and exploration of systems but not the specification. The latter is the outcome of a requirements engineering process which needs input from all three analysis perspectives. To describe this, a new process model was defined for the initial exploration of system ideas, as shown in Figure 2. It uses a four-fold classification to guide engineers in elaborating an innovative, complex system idea into a system concept:

- Hierarchical Analysis,
- Functional Analysis,
- Structural Analysis,
- Requirements Analysis & Management.

4 Suggested Analysis Steps using the SysML

This section of the paper briefly describes the system exploration and definition approach shown in Figure 2. It explains which special perspectives facilitate initial risk assessment and the definition of suitable mitigation measures in early stages of system definition.

4.1 Hierarchical Analysis

All analyses from the hierarchical perspective treat the system as a ‘black box’. In the student projects, the teams generate their system ideas by trying to match a technology push (i.e. the application of an innovative technology) with a market pull (i.e. a group of potential customers). This principle is shown in Figure 3.

4.1.1 System Objectives

System objectives are defined in a non-formal, creative process using brainstorming, research into the field of interest and benchmarking. The objectives are qualitative functional statements describing a preliminary operational concept and its advantages to potential users and system owners. They are documented in a pager and presented in an elevator pitch session.

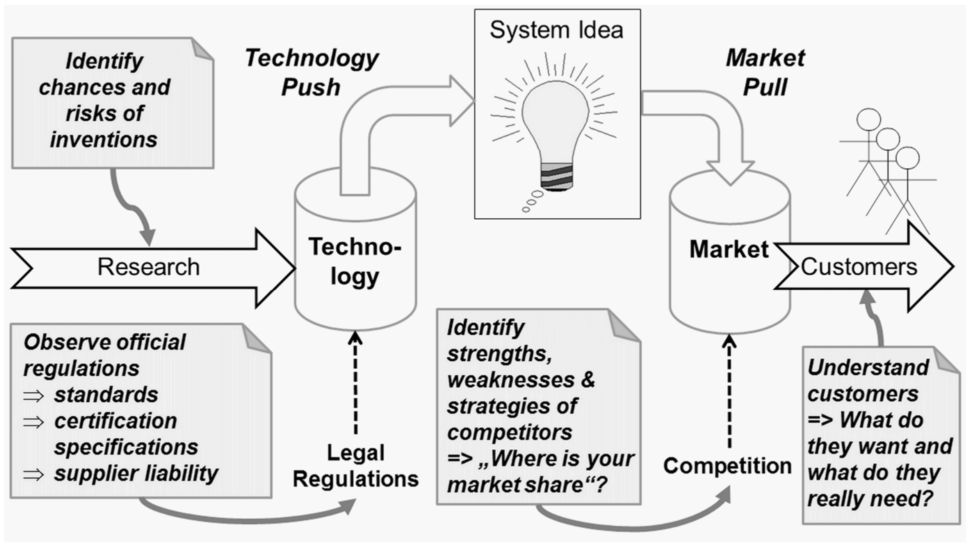

Identify chances and risks of inventions
Technology Push
System Idea
Market Pull
Research
Techno-logy
Market
Customers
Observe official regulations ⇒ standards ⇒ certification specifications ⇒ supplier liability
Legal Regulations
Identify strengths, weaknesses & strategies of competitors => „Where is your market share“?
Competition
Understand customers => What do they want and what do they really need?

Fig. 3. The System Idea and its Enablers and Threats

4.1.2 System Boundary and Context

It is important to clearly define the system boundary at the beginning. The boundary differentiates between the system of interest and its environment. The context analysis explores the environment in which the system shall be used. Both are based on the preliminary operational concept and documented in a SysML context diagram, which is a special type of internal block diagram shown in Figure 4.

4.1.4 System Lifecycle Stages

It is the nature of systems engineering, that the system of interest is planned from 'cradle to grave'. Consequently, the entire lifecycle of the system needs to be anti cipated. This is best done by considering each stage in the lifecycle as a specific state. The lifecycle analysis is therefore documented in a SysML state machine.

4.1.3 System Stakeholders and Actors

The 'stakeholder needs and requirements definition process' as defined in [5] is a natural starting point for exploring a new system idea, once system objectives have been set.

Stakeholder identification is facilitated by creating a SysML block definition diagram where stakeholders are depicted by manikins arranged around a block representing the system [9]. The subsequent analysis identifies needs and requirements of each stakeholder, and the attitude and potential influence on the system. This information is documented in a table linked to the stakeholder diagram.

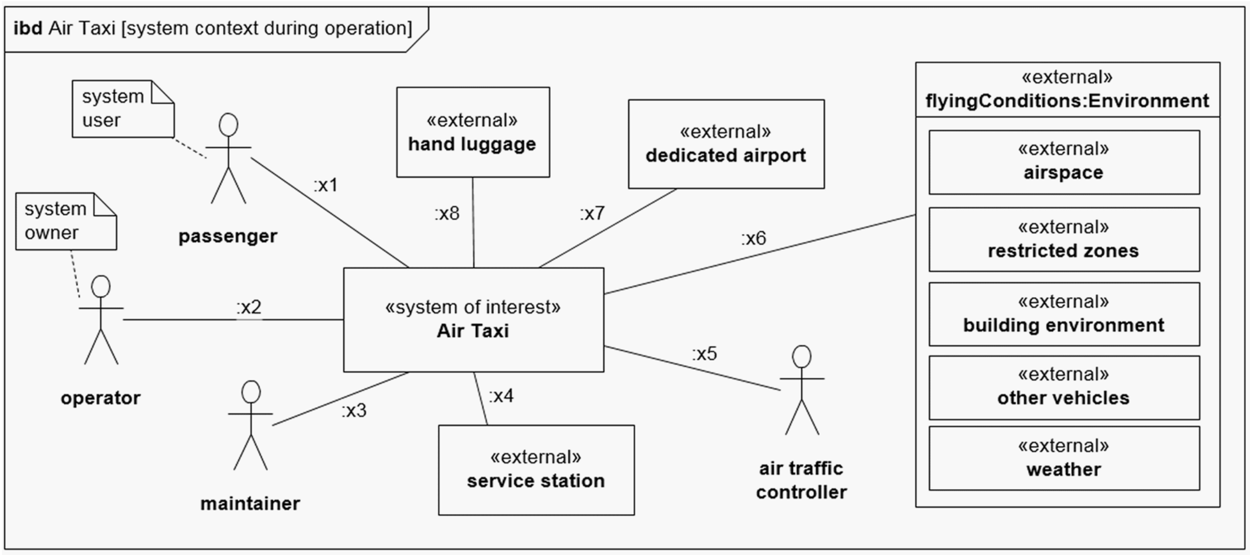

ibd Air Taxi [system context during operation]
system user
passenger
system owner
operator
maintainer
«external» hand luggage
«external» dedicated airport
«system of interest» Air Taxi
«external» service station
air traffic controller
«external» flyingConditions:Environment
«external» airspace
«external» restricted zones
«external» building environment
«external» other vehicles
«external» weather
:x1
:x2
:x3
:x4
:x5
:x6
:x7
:x8

Fig. 4. An Example of a Context Diagram Identifying Actors and External Elements

Human actors identified in the context analysis are the most important stakeholders of the system. Other relevant stakeholders from the operational stage of the system lifecycle can be owners of external system elements interacting with the system or neighbours of the system. To identify other stakeholders with important needs and requirements, other stages of the system lifecycle should be analysed.

The relative significance of individual stakeholders can be used to solve conflicts of interest by overriding or prioritizing individual stakeholder requirements or defining acceptable compromises. It is important to identify and eliminate contradictions in the initial set of requirements before proceeding.

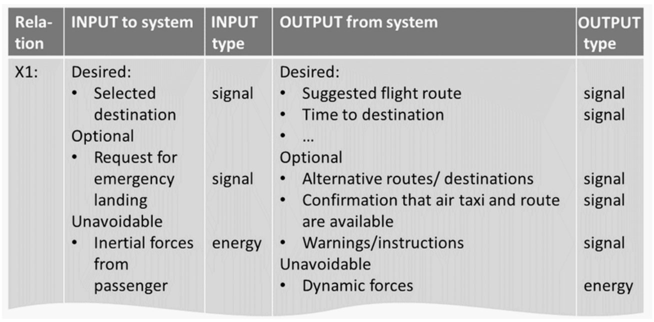
4.1.4 System Interfaces

The system will require input signals, energy and/or material/objects to produce the desired output, which may also take the form of signal, energy and/or material/objects. To identify all interfaces the system needs to fulfil its mission, the context diagram can be used.

It depicts interfaces as graphical paths connecting external elements or actors to the black box representation of the system. The nature of each interface is described in a separate table, as shown in Figure 5.

As long as the system is treated as a black box, the identification of required inputs to the system focuses on the system main function as defined in the system objectives, and on the preliminary operational concept. The analysis needs to be refined in later stages of system design for auxiliary functions which may need additional input from external elements and may generate unavoidable output.

Already at this early stage, the system analysis should be used to embark into an initial risk analysis. This is done by differentiating between desired and unavoidable (or accidental) input and output.



Rela-tion	INPUT to system	INPUT type	OUTPUT from system	OUTPUT type
X1:	Desired:		Desired:	
	• Selected destination	signal	• Suggested flight route	signal
			• Time to destination	signal
			• …	
	Optional		Optional	
	• Request for emergency landing	signal	• Alternative routes/ destinations	signal
			• Confirmation that air taxi and route are available	signal
	Unavoidable			
	• Inertial forces from passenger	energy	• Warnings/instructions	signal
			Unavoidable	
			• Dynamic forces	energy

Fig. 5. Extract from the Input/Output Analysis Table

Unavoidable input includes disturbances from the system environment. They may result from moving or heat-releasing components, electromagnetic waves, weather conditions like rain, snow, wind, sunshine, spillages etc.

Unavoidable output can be produced by chemical processes inside the system, e.g. exhausts from combustion. The system can also emit heat, electromagnetic or sound waves, and vibrations etc. It will be very difficult to identify all these effects without knowing anything about the internal composition of the system, so this part of the analysis will be based on assumptions and needs to be revisited in later stages of the design when knowledge about the system and its characteristics is building up.

It may be helpful to draw a dedicated diagram just depicting the inputs and outputs of the system without attributing them to specific external actors or elements as a backup and for later reference. The general sensitivity of the environment to this output must be assessed to derive requirements specifying permissible threshold values etc..

4.2 Functional Analysis

The functional analysis takes an abstract perspective of the system structure which can be considered as a ‘grey box’ view. It explores what the system shall do to fulfil its mission, and how it is supposed to behave and react under certain circumstances. Again, the analysis of the intended purpose of the system is accompanied by an initial risk analysis.

4.2.1 Important System Use Cases

The natural starting point for defining system functions is the intended use of the system which often involves human users. These actors should have been defined in the context diagram. The scenario-based use case analysis is documented in SysML use case diagrams. They describe what users and other actors do with the system when operating or otherwise handling it, and help with the breakdown into subfunctions and associated activities, see Figure 6.

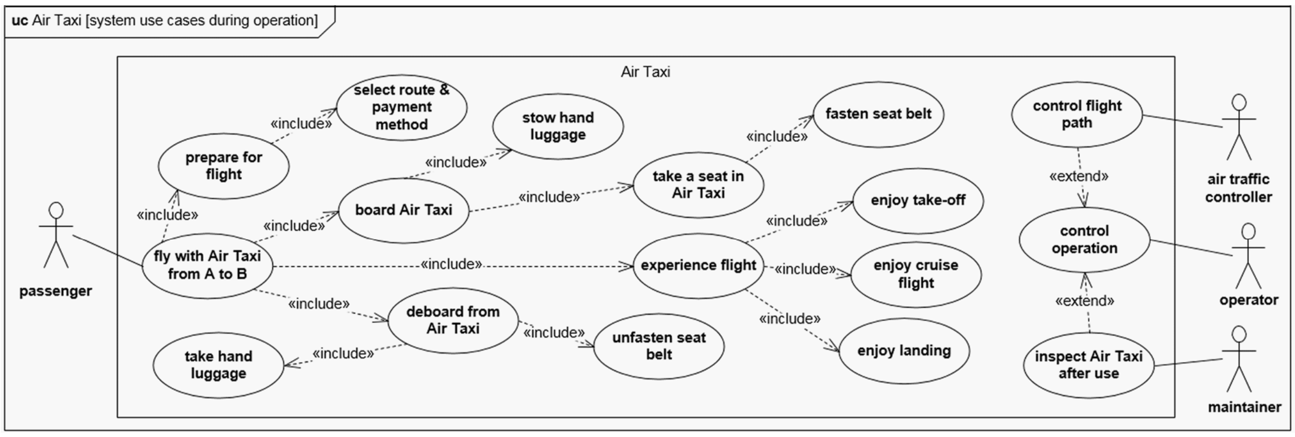


Fig. 6. Example of a Use Case Diagram

It is advisable to analyse use case scenarios from other lifecycle stages to identify additional system requirements. For example, large sub-systems (e.g. fuselage sections) or systems containing explosives (e.g. airbags) need special design features for safely shipping them to the customer. To avoid confusion, use cases from different lifecycle stages should be described in separate use case diagrams.

4.2.2 Conceivable Misuse Case

After exploring intended use cases in ‘sunshine scenarios’, it is advisable to imagine what ideas actors might develop for misusing the system, i.e. doing things with it which were not intended by the supplier of the system.Other misuse cases worth considering are scenarios where users intend to use the system for its original purpose but are ignorant or negligant and do not interact with the system as planned. Figure 7 shows an example of how misuse cases are documented in separate use case diagrams.

Again, lifecycle stages other than the operation may need to be considered in this risk analysis, to identify risks for which the supplier of the system might be liable.

All misuse cases should be assessed to classify them according to their consequences:

A: The system can and must be designed such that the misuse is safely prevented.

B: There is no affordable technology or design principle available which could safely prevent the misuse, and the risk is of a kind for which the supplier of the system will not be liable as long as suitable instructions, warnings, placards etc. are provided. Effects of the misuse must be mitigated by the design if possible and affordable.

C: There is no necessity to prevent the misuse because it is the user’s own risk without liability of the system supplier. However, it may be advisable to protect the system from negative consequences of this misuse case.

X: The misuse has severe consequences and a high probability causing an intolerable risk. It must be prevented by all means, but until now, no affordable technology is known which could safely prevent it.

From misuse cases categorized as ‘A’, requirements are derived to make sure that the design safely prevents the misuse. Categories ‘B’ and ‘C’ are checked for requirements pertaining to instructions and warnings etc. and for technical solutions to mitigate consequences for the system and its environment. Misuse cases of Category X bring the project to a halt because the system cannot be realized unless a technology becomes available which safely prevents the misuse.

4.2.3 Functional Blocks and Decomposition

It can be helpful to refine use cases in a functional analysis with SysML block definition diagrams as suggested in [13]. In a top-down approach, all necessary sub-functions are determined for a specific use case.

The functional decomposition is depicted in a tree-shaped diagram with the associated use case linked to the root node. Each sub-function is represented by an activity block and given a uniqe name consisting of an active verb describing the action and a noun denoting the

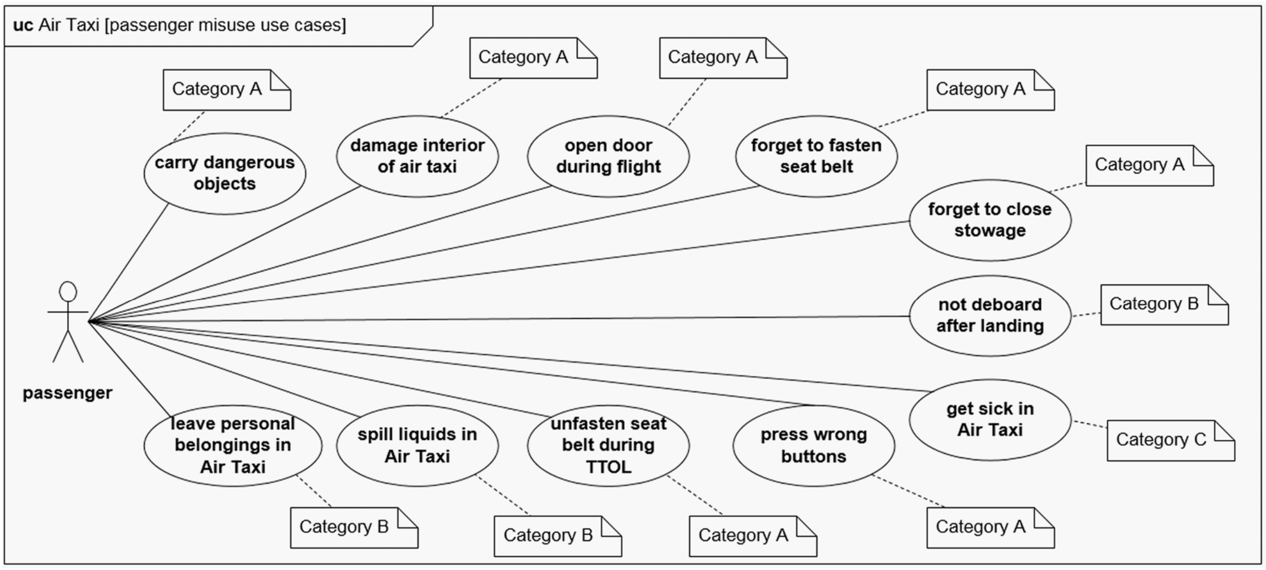

uc Air Taxi [passenger misuse use cases]
passenger
carry dangerous objects
damage interior of air taxi
open door during flight
forget to fasten seat belt
forget to close stowage
not deboard after landing
get sick in Air Taxi
press wrong buttons
unfasten seat belt during TTOL
spill liquids in Air Taxi
leave personal belongings in Air Taxi
Category A
Category B
Category C

Fig. 7. Example of a Diagram with Categorized Misuse Cases

object to which the action refers, both of which may be specified by attributes.

If misuse cases of categories A to C might occur in combination with the use case of interest, they must be integrated into the analysis to identify subfunctions required for their prevention and/or mitigation.

The functional decomposition is suited for reusing functional breakdown patterns when use cases are similar. The tree-shaped diagram supports a brainstorming approach. Engineers with a background in mechanical design are usually familiar with this kind of analysis.

4.2.4 Functional Flows

To make sure that all actions needed for a specific activity have been identified in the functional decomposition, functional flows can developed. They describe use case scenarios as complex sets of actions linked by control and object flows.

They are documented in SysML activity diagrams which offer a suitable variety of graphical nodes for decision-based and event-triggered system behaviour with sequential and parallel actions. Complex actions may be refined in subordinate diagrams to keep the diagrams readable and prepare encapsulation of action sets.

This analysis is particularly well suited for using the scenario technique and can be combined with user-centric design methods. Engineers with a background in programming are familiar with this kind of analysis and may find it easier than the functional decomposition.

It is possible to model the allocation of activities to specific components or subsystems with ‘partitions’ [1]. However, this modelling concept should not not be used too early in the design process because it anticipates design decisions which belong to the structural analsysis, cf. section 4.3 of this paper.

4.2.5 System States and Transitions

The analysis of system states and transitions takes a completely different perspective on the behaviour of the system. It is documented in SysML state machines. Teaching experience has shown that most mechanical engineering studens find it difficult to distinguish between activities and states, and to identfy triggers, guards and actions associated with state transitions.

Basic operational modes of a system like ‘off’, ‘idle’ and ‘working’ are not so difficult to identify, but many systems have more states. States may even be composite, e.g. a jet engine engine can be working with or without thrust reversers in action.

Quite often, transitions between states need to be controlled by so-called ‘guards’ to ensure they only occur when they are safe. For example, activation of thrust reversers must be prevented unless the aircraft has successfully touched down on the tarmac after completing its landing approach. The state-oriented analysis is therefore extremely important for safety-critical systems.

4.2.6 Critical Interaction Sequences

The third perspective on system behaviour focuses on interaction sequences between system components and external blocks or actors. The SysML sequence diagram is well suited for this, but for the students' projects it not compulsory to use analyse third perspective.

Sequence diagrams are only used in the exploration of system ideas when the core of the system requires specific sequences of signal processing and exchange between various components and actors.

4.3 Structural Analysis

This third analysis category leads to the definition of a baseline version of the system architecture, i.e. a technical solution. Ideally, architectural concepts are varied making use of the entire solution space.

Alternative concepts are analysed in trade studies and assessed with pre-determined criteria to identify an overall optimum solution.

4.3.1 Allocation of Function to Components

Following the logic of making a functional analysis prior to defining the technical solution, the functional breakdown is the starting point for the structural analysis. A simple table can be used for the allocation of functions to system components if no alternative solutions shall be developd [13]. In mechanical engineering projects, a morphological matrix is used instead to gather ideas from creativity and support the definition of design alternatives.

4.3.2 Hierarchical Structure of the System

The hierarchical breakdown of the technical solution into subsystems, components and parts is described in SysML block defintion diagrams. In this approach, they are used to graphically model the preliminary bill of materials.

In addition, block definition diagrams can be used to define which external components a system uses during its operation. For example, the engine system uses the electrical system of the aircraft. This dependency can be represented by a SysML graphical path called 'aggregation'. It cannot be described in a bill of materials.

In block definition diagrams, component-specific parameters can be defined which are needed to dimension, simulate or quantitatively assess a specific design solution.

4.3.3. Internal Structure of the System

The most important step of the structural analysis is the physical architecture definition in SysML internal block diagrams. Here, the system elements defined in the block definition diagram are arranged in a schematic layout depicting their physical interfaces as ports and their connections as flows of signals, energy and/or material/matter.

One drawback of the SysML is that structural variants are difficult to describe. The definition of alternative architectural concepts is currently best done in different diagrams.

4.3.4. Important Parametric Constraints

Mathematical relationships between quantitative parameters defined in block definition diagrams are described in SysML parametric diagrams. Unfortunately there is no SysML modeling software which can process parametric diagrams to perform calculations.

Nevertheless students are asked to use this diagram kind for modelling quantitative assessments which they consider relevant for decision-making in the early stage of their system design project. The mathematical evaluation can be used for dimensioning parts of the system or for comparing concept variations in trade studies.

4.3.5. Concept Assessment

In the students' projects, time is usually too short to define competing concepts and compare them in an analytical trade study. As all students are familiar with methods for systematically assessing concept variants, this part of the analysis is not requested for submission.

The SysML offers no specific language elements for concept assesment. Further research is needed to develop a simple method for performing SysML-based trade studies. Alternatively, a method could be developed to extract relevant information from a SysML model and import it to other model-based approaches for trade studies, e.g. the PARADIGMshift method suggested in [14].

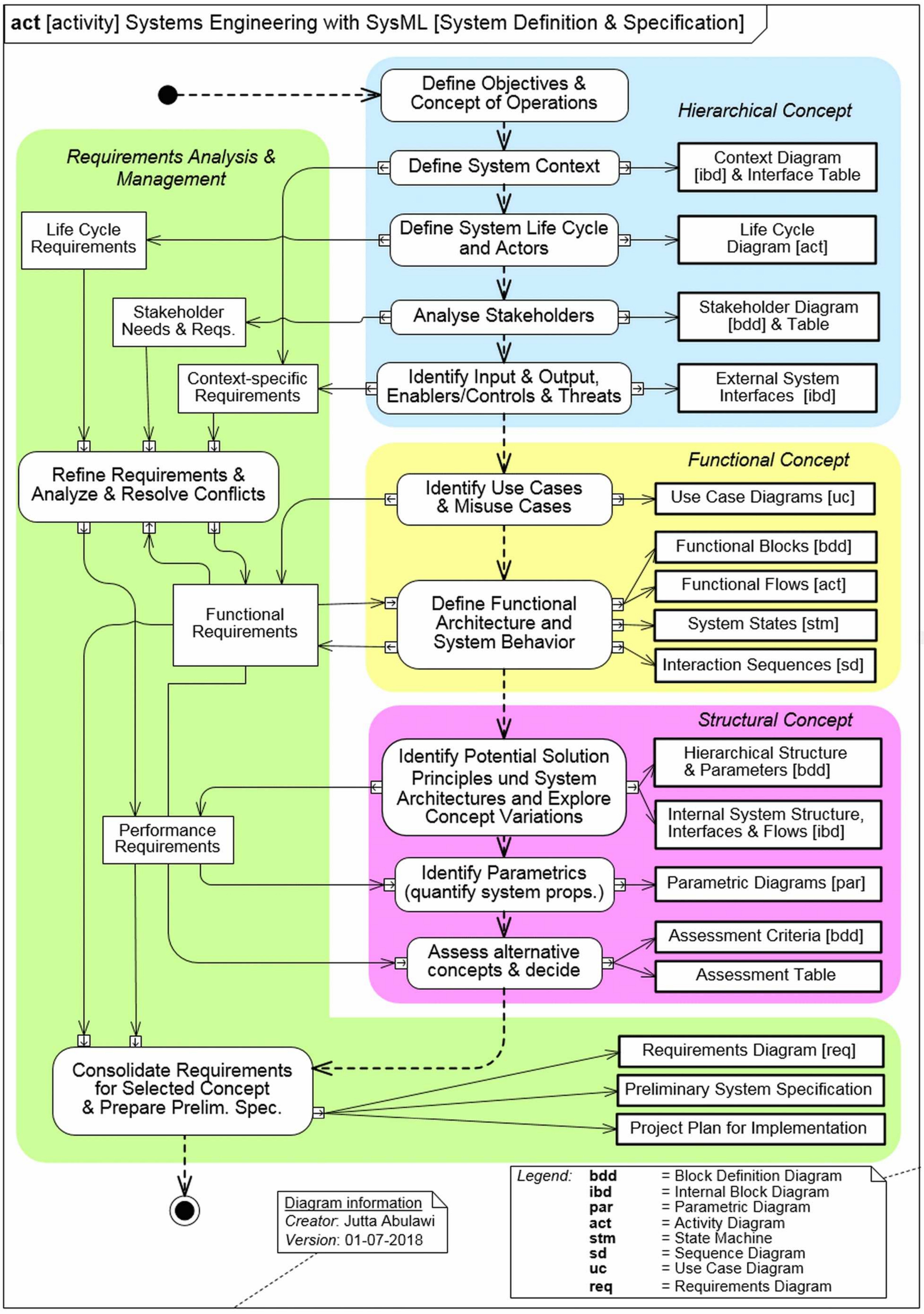

act [activity] Systems Engineering with SysML [System Definition & Specification]
Hierarchical Concept
Define Objectives & Concept of Operations
Define System Context
Context Diagram [ibd] & Interface Table
Define System Life Cycle and Actors
Life Cycle Diagram [act]
Analyse Stakeholders
Stakeholder Diagram [bdd] & Table
Identify Input & Output, Enablers/Controls & Threats
External System Interfaces [ibd]
Requirements Analysis & Management
Life Cycle Requirements
Stakeholder Needs & Reqs.
Context-specific Requirements
Refine Requirements & Analyze & Resolve Conflicts
Functional Concept
Identify Use Cases & Misuse Cases
Use Case Diagrams [uc]
Functional Blocks [bdd]
Functional Flows [act]
System States [stm]
Interaction Sequences [sd]
Define Functional Architecture and System Behavior
Functional Requirements
Structural Concept
Identify Potential Solution Principles und System Architectures and Explore Concept Variations
Hierarchical Structure & Parameters [bdd]
Internal System Structure, Interfaces & Flows [ibd]
Performance Requirements
Identify Parametrics (quantify system props.)
Parametric Diagrams [par]
Assess alternative concepts & decide
Assessment Criteria [bdd]
Assessment Table
Consolidate Requirements for Selected Concept & Prepare Prelim. Spec.
Requirements Diagram [req]
Preliminary System Specification
Project Plan for Implementation
Diagram information
Creator: Jutta Abulawi
Version: 01-07-2018
Legend:
bdd = Block Definition Diagram
ibd = Internal Block Diagram
par = Parametric Diagram
act = Activity Diagram
stm = State Machine
sd = Sequence Diagram
uc = Use Case Diagram
req = Requirements Diagram

Fig. 8. Suggested activities for the SysML-based exploration of innovative system ideas

4.4 Requirements Analysis

The SysML supports requirements modelling with a special requirements diagram. It permits the definition of hierarchical and other dependencies between individual requirements and other objects of the system model. For example a requirement can be linked to a block representing a component which is part of the system to satisfy the requirement. Additionally, a requirement can be traced to the test case defined for its verification.

However, industry is still reluctant to use the SysML for requirements engineering and management. Consequently, a conventional approach to documenting requirements is used in the students' projects which shall not be described here as it does not use the SysML.

5 Conclusion

Since 2016, the approach outlined in Figure 8 has been used and refined at HAW Hamburg for teaching systems engineering in the Master of Science Programme for Aeronautical Engineers. Before, students were trained to use the SysML by describing existing complex systems in a reverse engineering project. Although they found it easier to use all diagram kinds in their system models, they did not experience the real purpose of the SysML.

In the recent semesters, when students were asked to identify a new system idea and explore it with SysML-based analysis and other methods, the motivation among the students has increased considerably. They did not only enjoy exploring their own ideas, but were much less reluctant to experiment with analysis methods to adapt them to their needs.

Many graduates have chosen to use some of the SysML-based analysis in their master theses. Some have even started a professional career in systems engineering.

References

[1] *OMG Systems Modeling Language™*. Version 1.5. May 2017. http://www.omg.org/spec/SysML/1.5/

[2] Hause, M. The SysML Modeling Language. *Fifth European Systems Engineering Conference*, Edinburgh, 18-20 Sept. 2006.

[3] Petersen, T.: Exploring System Space with Graph Theory. *INCOSE International Symposium.* Volume 26, Issue 1, pp 1120-1128, July 2016.

[4] Haberfellner, R. et al. *Systems Engineering – Grundlagen und Anwendungen.* 13th edition, orell füssli Verlag Zürich, 2015.

[5] Walden, D. et al. *INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities.* 4th edition, John Wiley & Sons, 2015.

[6] O'Mahony, C. and Woodcock, L. Curricular and Strategic Perspectives on Research-based Learning. *EUA Learning and Teaching Forum*, Paris, 2017.

[7] Forsberg, K. and Mooz, H. The Relationship of System Engineering to the Project Cycle. *Joint conf. of NCOSE and ASEM*, Chattanooga, 1991.

[8] Gausemeier, J. et al. *Systems Engineering in industrial practice.* Heinz Nixdorf Institute, Paderborn, 2015.

[9] Weilkiens, T. *Systems Engineering with SysML/UML modeling, analysis, design.* Morgan Kaufman Publishers, 2008.

[10] Friedenthal, S. and Moore, A. and Steiner, R.: *A Practical Guide to SysML.* Elsevier, Amsterdam, 2009.

[11] Rupp, Chr.; Queins, S.; Zengler, B.: *UML 2 glasklar. Praxiswissen für die UMLModellierung.* 3rd edition. Hanser, München/Wien, 2007.

[12] Ropohl, G. *Allgemeine Technologie. Eine System-theorie der Technik.* 3rd edition. Karlsruhe, Universitätsverlag, 2009

[13] Lamm, J. and Weilkiens, T. Method for Deriving Functional Architectures from Use Cases. *Systems Engineering*, Vol. 17, Issue 2, pp. 225-236, 2014.

[14] Schumann, H. et al. PARADIGMshift: A Method for Feasibility Studies of New Systems. *Procedia Computer Science 44*, pp. 578 – 587, 2015.

Contact Author Email Address

mailto:jutta.abulawi@haw-hamburg.de

Copyright Statement