INTERNATIONAL DESIGN CONFERENCE – DESIGN 2020
https://doi.org/10.1017/dsd.2020.170

APPLYING MODEL-BASED SYSTEMS ENGINEERING TO THE DEVELOPMENT OF AUTONOMOUS VESSEL FUNCTIONS

O. Bleisinger [1,✉], S. Forte [2], C. Apostolov [3] and M. Schmitt [4]

[1] Fraunhofer IESE, Germany, [2] Technische Universität Kaiserslautern, Germany, [3] CONTACT Software GmbH, Germany, [4] blackned GmbH, Germany

✉ oliver.bleisinger@iese.fraunhofer.de

Abstract

Developing autonomous functions for complex systems leads to high demands on the consideration of dependencies to external actors in the usage phase. In Model-Based Systems Engineering (MBSE), this can be achieved by modelling operational aspects. Operational aspects are model elements and their relationships to each other. In this contribution, modelling of operational aspects with a MBSE-approach will be demonstrated exemplary on a case study related to the development of a yacht with an autonomous docking assistant. Currently modelling operational aspects is not common in the civil sector.

Keywords: systems engineering (SE), model-based engineering, modelling, model-based systems engineering, autonomous systems

1. Introduction

Developing autonomous functions for complex systems leads to high demands on the consideration of dependencies and interconnections with external actors in the usage phase. Through the interaction of the external actors with the considered system in development, also called System of Interest (SoI), a great influence on its intended behaviour exists. Because of this, it is helpful to consider the usage phase of a complex system with autonomous functions in early phases of the system development and possible reconfiguration of a single system in the System of Systems ecosystem (Abramovici et al., 2016). In Model-Based Systems Engineering (MBSE), this is achieved by modelling so-called operational aspects as part of the system architecture of the SoI. Operational aspects are model elements and their relationships to each other in specially tailored views on the operational architecture of the System of Systems (SoS) in which the SoI and all relevant external actors are sketched. In this contribution, the operational aspects can be considered as parts of the architecture of a SoI, whereas the operational architecture belongs to the holistic model of a SoS.

By depicting an autonomous docking scenario of a vessel, the following figure illustrates the distinction between system architecture of the SoI and the operational architecture of the SoS using a layer model. In the figure, it is also indicated, that the operational architecture can serve as a link between the developments of different individual systems involved in the SoS. In this case, the individual SoI can be in different phases of development, e.g. according to a V-model (Eigner et al., 2018).

Modelling operational views is typically and initially done in early development phases and the resulting views are detailed and adapted in the further steps of a systems development. In addition,

https://doi.org/10.1017/dsd.2020.170 Published online by Cambridge University Press

operational views of SoIs and operational architectures of SoS have a high potential for reuse in future system generations. The impact of the architecture of a newly developed system on the operational architecture can be interpreted as a measure of the disruptiveness of a technology introduced with the novel system.

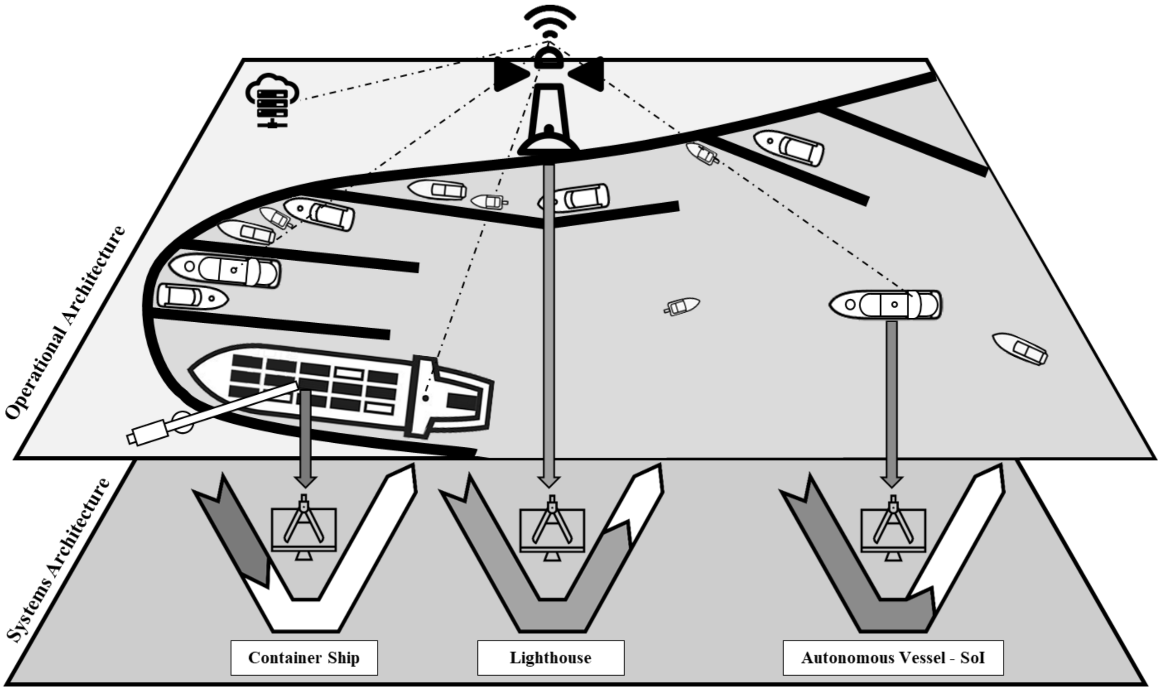


Figure 1. Depiction of the differentiation between the SoI architecture and the operational architecture of a SoS on the example of an autonomous docking scenario

In general, following exemplary aspects are contained in operational views. In the application example in section 5, these aspects are demonstrated more concretely:

- Concrete usage scenarios for the SoI
- Partitioning of the SoI functionalities
- Assignment of the SoI functionalities to the usage scenarios
- Operational context and related operational constraints of the SoI
- Interfaces of the considered SoI
- Interaction with external actors, e.g. communication with external systems

Considering operational aspects with suitable views supports the phase of requirements elicitation in the development of autonomous vessel functions. It especially enables a reliable way to capture non-functional requirements, which are often regarded as drivers of system architectures (Dörr et al., 2005). In addition, operational views in very early project phases, e.g. the initiation of a system development project, can serve as a way to communicate with stakeholders beyond classical requirements. It allows an easy way for differentiation of the SoI and SoS - e.g. for management presentations regarding the scope of development projects. A quite special application also consists of carrying out a first simulation-based partial validation of the entire SoS design with the help of an operational architecture during the phase of requirements elicitation (Antonino et al., 2018). This can be especially helpful when checking the compatibility of interfaces between different systems.

2. State of the art of modelling operational aspects

Model-Based Systems Engineering (MBSE) offers processes and methods that support holistic systems analysis and design (ISO15288, 2015). However, in early phases of system development, the focus is on the analysis of the business area or mission (Business or Mission Analysis) and on the

https://doi.org/10.1017/dsd.2020.170 Published online by Cambridge University Press

definition of stakeholder needs and requirements (Walden et al., 2015). Possible usage scenarios as well as operational constraints of the SoI remain incompletely defined or imprecisely described, although they have great influence on the definition of the later system requirements.

In contrast, aspects of operational architectures are addressed in corresponding views of the most common enterprise architecture frameworks. Widely used frameworks in the defence sector include the Department of Defense Architecture Framework (DoDAF, USA), the Ministry of Defence Architecture Framework (MoDAF, UK) and the NATO Architecture Framework (NAF). The architectures created in the defence sector, in particular the operational views, are usually an essential component of capability management and support the needs analysis as well as the derivation of reference and target architectures in the procurement of military equipment. Originally, they are intended to create a common understanding of multinational operational principles and interfaces and thus ensure interoperability between systems of alliance partners. On the example of NAF, the embedding of operational views into an overall architecture can be illustrated as a representative example of military Enterprise Architecture Frameworks. NAF combines operational aspects in its own viewpoint, the NATO Operational View (NOV). The following exemplary aspects are detailed in corresponding sub views of the NOV:

- Roles and actors of the organizational structure
- Operational constraints (e.g. weather, mobility and spatial distribution of actors)
- Information exchange relations within the organizational structure and with external actors
- Operational activities under consideration of internal control and information flows
- Temporal dependencies of different operational activities

Through transition to the strategic view in NAF (NATO Capability View), the contribution of operational processes and operational actors to the fulfilment of corporate capabilities is presented. The aim is usually to describe operational aspects without restricting a technical solution too much or even anticipating it. As a result, the SoIs are often not at all the subject of the operational views or are only depicted as black box. A technical elaboration of the solution components then takes place in system or service views (NATO System View, Service-Oriented View). Comparable conclusions can also be drawn from the consideration of Enterprise Architecture Frameworks in civil applications, such as The Open Group Architecture Framework (TOGAF) or ArchiMate. In particular, the levels of the ArchiMate framework, which are loosely based on the concepts of the IEEE 1471 standard, can be clearly mapped to the NAF viewpoints, showing the similarity of the concepts of NAF and ArchiMate and the difference to the authors approach demonstrated on the application example in section 5 (see Table 1).

Table 1. NAF views mapped to the ArchiMate layers and included aspects of NAF/ArchiMate in the authors approach in respect to the application example

NAF main view	*ArchiMate layer*	*Figure 7*	*Figure 8*
Capability view	*Strategy layer*	-	-
Operational view	*Business layer*	*partially included*	*partially included*
Service-oriented view	*Application layer*	-	*partially included*
System view	*Technology layer*	*partially included*	*partially included*

The following is common to operational views in both application areas: Not technical systems, but organizational units and business processes are considered. Therefore, the concepts do not focus on controlling the system development process. The established architecture frameworks rather support the description of organizational transformation processes and the associated contribution to value creation (products) or the creation and development of corporate capabilities. Primary stakeholders are strategic management, product portfolio management, sales or end users.

The emergence of highly automated and autonomous SoSs and related autonomous functions of SoIs implies a close coupling of operational and technical-systemic aspects (Eigner et al., 2015). In particular, increasing interconnectivity and dynamic changeability are the causes of increasing

https://doi.org/10.1017/dsd.2020.170 Published online by Cambridge University Press

complexity (Haberfellner et al., 2019), which also affect the individual SoIs and their development process. To cope the increasing complexity in the development of autonomous functions for SoIs, a comprehensive presentation of architecture-relevant aspects from the usage phase of the related SoS is helpful. By doing so, it is possible to derive necessary requirements from specific usage scenarios during the development of participating individual systems or SoIs in a systematically way. These aspects are not yet covered in graphical modelling languages such as the Unified Modeling Language (UML) or the Systems Modeling Language (SysML). Right now in these languages only use case diagrams exist for the abstract description of functional requirements. Possibly the System Architecture Framework (SAF) (Andres et al., 2019) addresses corresponding views for mapping operational aspects. However, the corresponding views have not yet been fully implemented.

In automotive, modelling operational aspects for the design of autonomous systems is also discussed. For example, the Automotive Design Framework (ADF) used by RENAULT (Góngora et al., 2013) does not include all operational aspects considered necessary in the context of this contribution. The Automotive Architecture Framework (AAF) used by VOLVO (Pelliccione et al., 2017) also enables the documentation of operational aspects to tackle the complexity when developing autonomous functions. Both examples emphasize the general interest of the automotive industry in the description of operational aspects, but are not well established and currently rather experimental. In contrast, the modelling of operational aspects in the defence sector is mature and therefore mainly regarded as input for this contribution.

3. Current challenges for autonomous function development

To record the above-mentioned architecture-relevant aspects as comprehensively and precisely as possible and elicit the associated system requirements in a clear and coherent way for a specific SoI, is the task of the systems architect. The necessary requirements arise on the one hand from stakeholders who make demands on the SoIs functionality or specific qualities. On the other hand, they also result from the context in which the system will be used and the associated scenarios that can occur during the system usage phase. The system usage context and the various scenarios also have a considerable influence on the system requirements of stakeholders and developers (Figure 2).

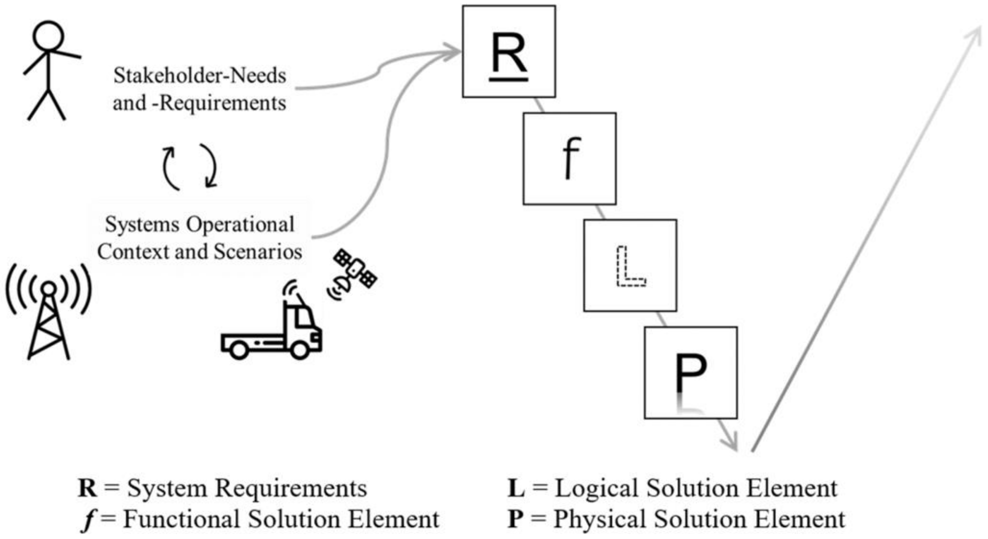


Figure 2. Influence of Stakeholder-Needs and-Requirements as well as operational constraints on the systems development and architecture

Especially in the case of SoS and embedded SoIs with autonomous functions, the usage of the system in its operational context must be communicated clearly and comprehensibly to the stakeholders in early development phases. If this is not the case, difficulties can already arise during the possible initiation/acquisition of system development projects, since a clear, coherent picture of the usage of the system is already lacking in the very early project phases and during the requirements elicitation.

https://doi.org/10.1017/dsd.2020.170 Published online by Cambridge University Press

In addition, the following general risks are associated with this, which can also lead to a budget and time overrun MBSE-driven system development projects:

- Legal risk due to not documented, but legally required, operational constraints
- Economic risk due to high potential of undesirable developments based on misunderstandings
- Technical risk in relation to autonomous functions, due to degrees of freedom in the implementation or solution/design space during the development

Although the risks mentioned can be countered by classical requirements engineering (i.e. without explicit modelling of operational aspects), operational views support the minimisation of these risks. For example, legally binding operational constraints can be clearly and precisely defined by means of operational views when considering an autonomous docking scenario of a vessel. If these operational constraints are not defined in a model, the incomplete system descriptions must be supplemented by further engineering artefacts (e.g. classical documents). However, in this case, the traceability between the engineering artefacts is affected and the formality of the system specification is reduced. An automated or semi-automated validation of the systems design by simulation or execution of a system model in early project phases is either not possible or the confidence of the simulation results is significantly reduced, since relevant aspects are not contained in the formal models. In order to counteract these problems, modelling operational aspects in early phases of the development of autonomous functions for civil applications is proposed. This is already a standard practice in the development or procurement of military systems. In the following, a approach for elicitation and modelling of such views is roughly sketched and demonstrated using the example of a vessel with the function of autonomous docking. The example should show the general benefits of Model-Based Systems Engineering with a focus on operational aspects for the development of autonomous vessel functions.

4. Modelling approach

Since the elicitation and modelling of operational aspects is helpful for the development of autonomous functions (e.g. for vessels), a brief but practical approach is presented. For this, it is assumed, that a specific SoI is developed based on a partially already existing SoS. This means that certain operational constraints are already given or must be derived from concrete usage scenarios. For the described approach it is furthermore assumed, that the operational context of the SoI is highly dynamic. Therefore, only exemplary usage scenarios can be considered. Based on the experiences from different projects, especially from the automotive and defence sector, the following steps can be proposed as a pragmatic, lightweight approach for elicitation and modelling of operational aspects for the development of autonomous vessel functions:

- Identification and documentation of operational constraints by interviews with stakeholders for the selected autonomous function of the regarded SoI
- Evaluation of the interviews with the focus on initial definition of operational constraints and operational actors in the related SoS
  → mainly input for modelling the operational constraints and operational actors exemplary shown in Figure 4, 6, 7
- Identification and documentation of the most relevant usage scenarios and derivation of a base scenario from the usage phase of the SoI within the SoS
  → mainly input for modelling of base scenarios and sub scenarios according to Figure 4, 5, 6
- Verbal, informal identification and documentation of further operational aspects for the specified usage scenarios - e.g. while brainstorming with stakeholders
  → used to refine the relevant operational aspects in the previous steps
- Definition of the required information exchange relations
  → mainly input for Figure 8
- Derivation of the operational requirements as well as identification of the operational drivers of the system architecture of the SoI
- Modelling of the operational drivers/aspects of the SoI system architecture

https://doi.org/10.1017/dsd.2020.170 Published online by Cambridge University Press

- Modelling the base scenario in its system context (differentiation between the SoI and its external actors/systems of the SoS
- Derivation and modelling of the various concrete usage scenarios associated to the base scenario in which the SoI must fulfil certain autonomous functions
- Graphical or illustrative representation of the relevant concrete usage scenarios of the SoI including a textual description
- Detailed modelling of the system context with operational constraints of the usage scenarios and typification of SoS Entities
- Modelling the communication and information exchange relationships between entities with the relevant interfaces

The approach does not claim to be exhaustive and merely serves as a guideline in order to demonstrate the previously discussed benefits of modelling operational aspects for the development of autonomous functions on the example of a vessel and an autonomous docking scenario.

5. Application example

In the following application example, the modelling of operational views of a SoI is shown to demonstrate how this can support the development of autonomous vessel functions. The focus will be on the autonomous docking assistant as exemplary function. The modelling is done prototypically in the OMG Systems Modeling Language (SysML), whereby a fictitious scenario with defined operational constraints serves as a basis for this case study. In the application example, based on the AAWA Position Paper (Jokioinen, 2016), only the part of the approach in section 4 related to the modelling of operational drivers/aspects will be demonstrated.

Scenario of the case study: A yacht enters a harbour in search of a suitable dock for anchoring. To start an autonomous docking process the yacht activates its autonomous docking assistant. Thereafter, the yacht initiates a query for available pier spaces in the port by communicating with the external systems. The communication is centralized by the transmission tower, which is connected with a GPS system, the pier management system and the authentication server. After identification of a free space, the port system starts a course inquiry of the vessels already located in the harbour, calculates an ideal mooring course for the entering yacht and transmits it accordingly to the navigation system of the yacht. The yacht takes over the course proposal of the port system and communicates with the pier system to start the final docking process as well as the detailed course calculation including tide (i.e. water level), current, wind, non-autonomous boats and the available sensor data. When developing the autonomous vessel function the following operational constraints and aspects should be considered:

- The signal from the autonomous docking assistant must not be transmitted directly (point-to-point) to any external entities. It is first transmitted to the transmission tower to start an authentication process by using a connected authentication server. The query of the yacht is only processed if there is an appropriate authorisation.
- The docking process should be performed quickly, so that other vessels are not interrupted more than absolutely necessary.
- The external systems send a message back to the yacht at the end of the docking process, i.e. as soon as the docking process is finished.
- Despite possible external disturbance (wind, slightly changing water level etc.) the docking process must be performed safely.

As depicted in Figure 3 (right side) the docking process should be performed frontal, so that following operational aspects of the case study are relevant for the operational views of the SoI, which must be mapped during the modelling:

- The entity of a yacht with an autonomous docking assistant exists.
- There are passengers as entities who are transported in the yacht.
- In addition, the authentication server and the transmission tower are relevant entities.

https://doi.org/10.1017/dsd.2020.170 Published online by Cambridge University Press

- The SoI’s context also includes a GPS satellite, a pier management system and other vessels.
- There is a captain on the yacht, who activates the autonomous docking assistant.

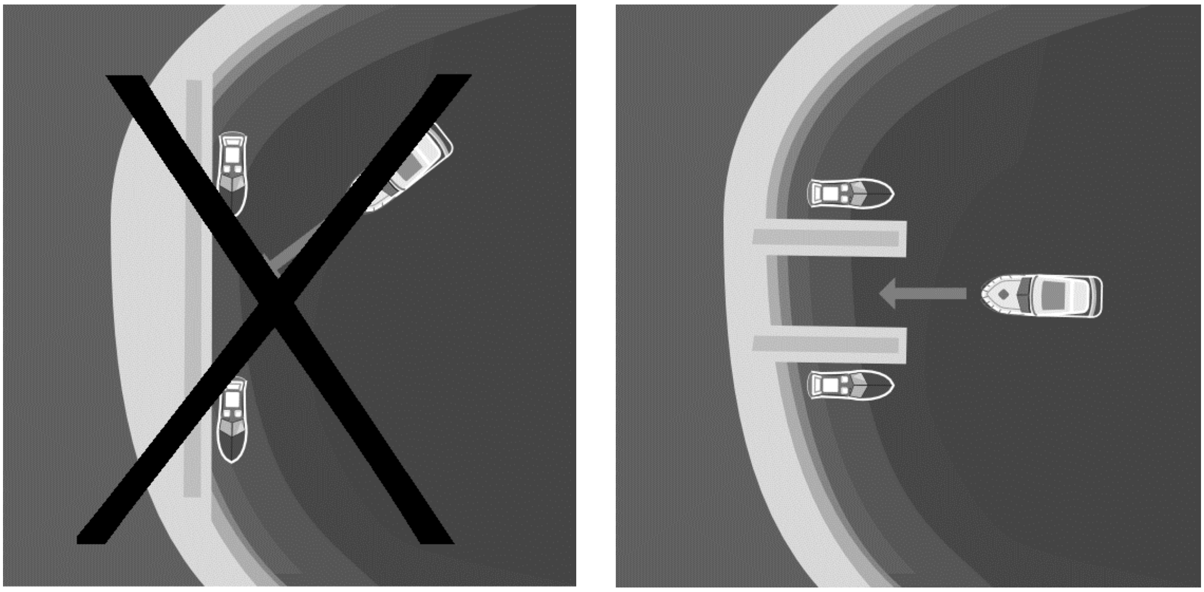


Figure 3. Sub scenario for the case study regarding the autonomous docking process of a yacht

Figure 4 shows the entity of the autonomous yacht (SoI) with the relevant base scenario “docking manoeuvre”. The entities of the system usage context are furthermore mapped as actors, who have or could have an influence on the usage of the system. The goal of this first step is to ensure the coverage of as many SoI-relevant and docking-manoeuvre-focused entities as possible.

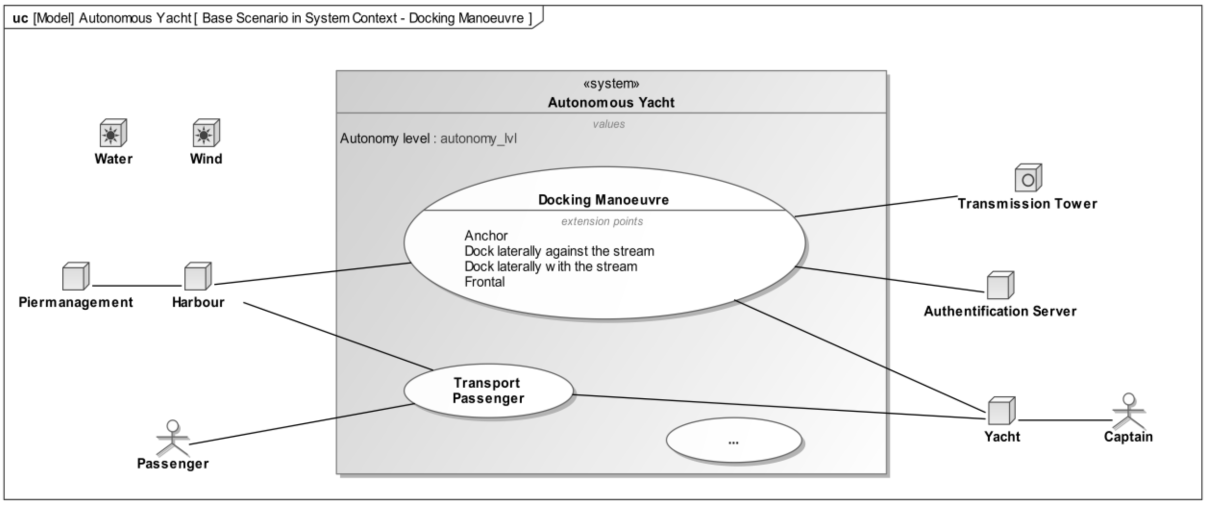


Figure 4. Base scenario of the System of Interest in its system context

In a next step, the possible sub-scenarios of the base scenario are analysed. For this purpose, one must consider the variations of the base scenario that can occur due to varying operational conditions or necessary degrees of freedom of the function. The possible sub-scenarios are modelled as extensions of the base scenario in a further use case diagram as shown in Figure 5. In the considered case study, four sub-scenarios (e.g. “frontal docking” and “dock laterally against the stream”) were examined, which could have a significant impact on the system architecture of the SoI. Those extend the basic scenario described above by further relevant operational aspects.

In order to illustrate the individual sub-scenarios, they are detailed with a graphical representation and an optional textual description according to Figure 6. Although this step leads to a non-formal, non-

https://doi.org/10.1017/dsd.2020.170 Published online by Cambridge University Press

machine-interpretable artefact in the system model, it can significantly improve the stakeholders' understanding of the SoI in its operational context. This significantly reduces the risk of undesirable developments and other risks, as described in section 3.

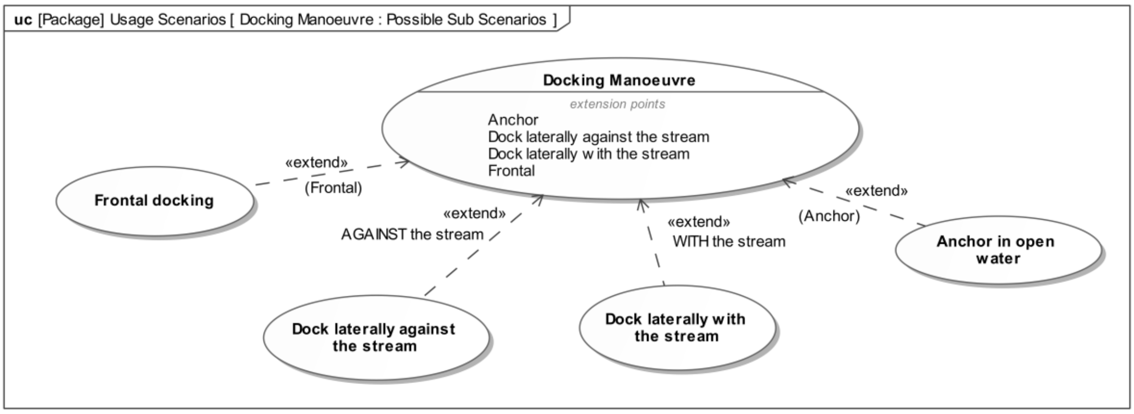
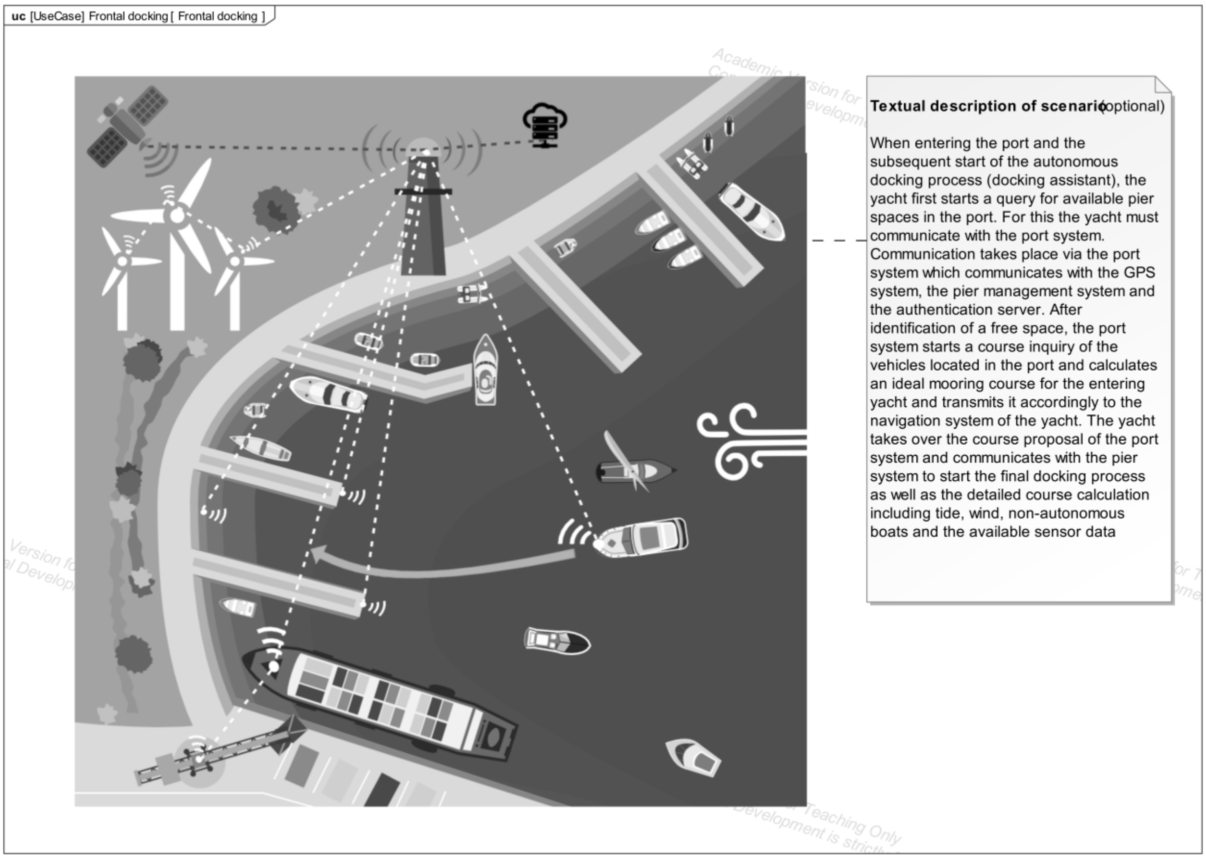

uc [Package] Usage Scenarios [Docking Manoeuvre : Possible Sub Scenarios]
Docking Manoeuvre
extension points
Anchor
Dock laterally against the stream
Dock laterally w ith the stream
Frontal
«extend»
(Frontal)
Frontal docking
«extend»
AGAINST the stream
«extend»
WITH the stream
«extend»
(Anchor)
Anchor in open water
Dock laterally against the stream
Dock laterally with the stream

Figure 5. Possible sub scenarios on the usage phase of the base scenario "Docking Manoeuvre"



Textual description of scenario (optional)

When entering the port and the subsequent start of the autonomous docking process (docking assistant), the yacht first starts a query for available pier spaces in the port. For this the yacht must communicate with the port system. Communication takes place via the port system which communicates with the GPS system, the pier management system and the authentication server. After identification of a free space, the port system starts a course inquiry of the vehicles located in the port and calculates an ideal mooring course for the entering yacht and transmits it accordingly to the navigation system of the yacht. The yacht takes over the course proposal of the port system and communicates with the pier system to start the final docking process as well as the detailed course calculation including tide, wind, non-autonomous boats and the available sensor data

Figure 6. Graphical representation of sub scenario "Frontal docking" and textual explanation

Following the analysis of the usage scenarios of the SoI, its operational context can be refined. Individual SoS entities are detailed and the operational constraints imposed by them are modelled as properties. In the concrete case, this is done by allocating the SysML actors (SoS entities) to blocks containing the relevant operational properties (see Figure 7, blocks "Yacht", "Transmission Tower", "Pier Management" and "Harbour"). The operational properties represent specific constraints which must be considered in the autonomous docking scenario to enable a proper docking. In addition, an initial functional partitioning of the SoI can also be undertaken, if this is possible and reasonable at the current stage of the development of the autonomous function or rather autonomous yacht. For

https://doi.org/10.1017/dsd.2020.170 Published online by Cambridge University Press

example, the autonomous yacht can be subdivided into the functional components “Docking assistant” and “Navigation system”, which the system will necessarily provide, and which must interact with each other in the context of the scenario (see Figure 7).

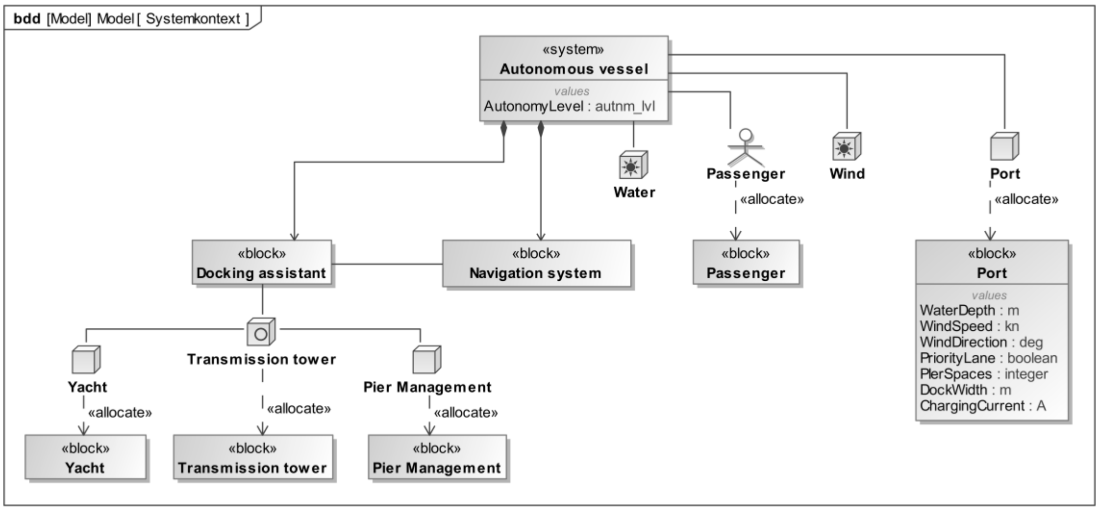


Figure 7. System context definition including relevant operational constraints

In a further step of the elicitation and modelling of the operational drivers/aspects of the system architecture, the communication and the information exchange relationships between the various entities of the SoS, defined in the previous step, are modelled as shown in Figure 8. The interfaces of the different SoS entities and the connections among them, through which the information flow takes place, are defined. Further, the transmitted information artefacts transmitted through the connections are specified. In addition, the dependency of the autonomous function from SoS entities, which do not result from information exchange between the SoI and the external actors/systems of the SoS, are specified. The relations between the autonomous yacht and the actors of the type “Water” and “Wind” in Figure 8 demonstrates this exemplarily. In the usage scenario, both mentioned actors are environmental influences, which can prevent the intended function of the autonomous docking assistant of working properly. For this reason, these external factors must be considered in the development of the autonomous yacht or rather the autonomous docking assistant.

As shown in Figure 8, the information exchange among the SoS entities can be specified with the help of a regular SysML internal block diagram.

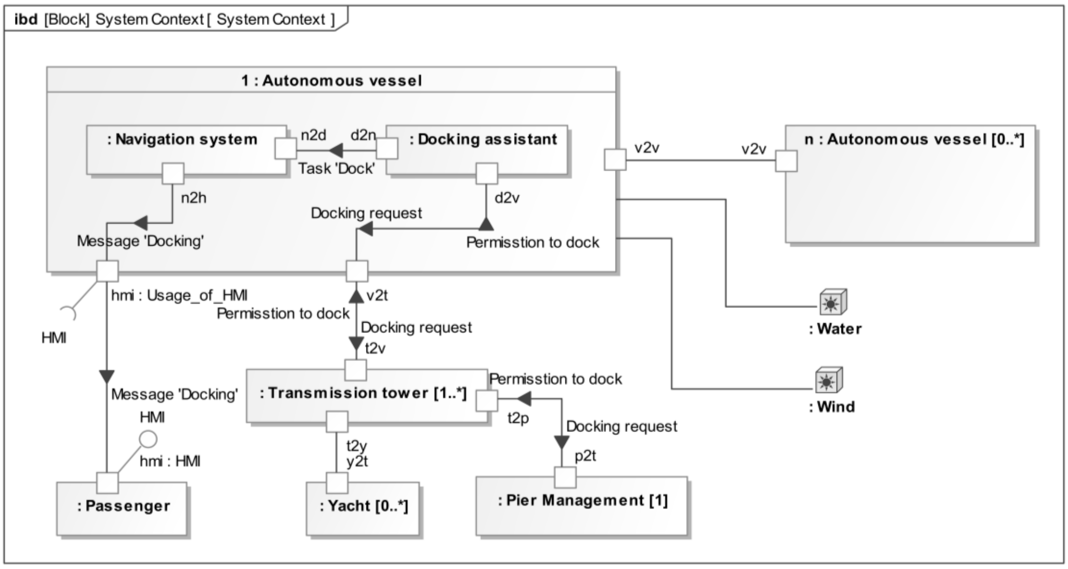


Figure 8. Information exchange relations between the SoI and its system context

https://doi.org/10.1017/dsd.2020.170 Published online by Cambridge University Press

6. Conclusion and outlook

Until now, MBSE methods have hardly been applied in the development of autonomous vessel functions, especially not with focus on modelling operational aspects in the civil sector – although there are high requirements on the consideration of the usage phase of the system when developing autonomous functions. By using MBSE approaches for the development of autonomous functions, it is possible to model operational aspects in established systems modelling languages like SysML, which enables the combination of the presented approach with already existing MBSE methods. To do so, further work is required to integrate operational approaches seamlessly into established MBSE methods. In addition to the elaboration of the roughly sketched elicitation and modelling approach described in section 4, an expansion of SysML-based modelling tools in respect to additional model elements and relations can also be considered – e.g. with a specific profile for modelling operational aspects. This would enable users to model operational aspects more efficiently and precisely. Furthermore, the simplified application example of an autonomous docking assistant presented in section 5 could be adapted to match the real world application more precisely. In this contribution, the simplified example was used to support the understanding of the benefits of modelling operational aspects without focusing on too many details of the application. Furthermore, aspects of SOTIF (2019) are not addressed in this contribution, since malfunctioning of system components and its impact on the functional safety is not the focus of the sketched modelling approach. In future, the authors aim at further testing the sketched elicitation and modelling approach in the area of industry 4.0 applications and autonomous driving.

References

ISO/IEC/IEEE (2015), *ISO 15288:2015: Systems and software engineering - System life cycle processes*, International Organization for Standardization, Geneva.

Abramovici, M., Göbel, J.C. and Dang, H.B. (2016), “Semantic data management for the development and continuous reconfiguration of smart products and systems”, *CIRP Annals - Manufacturing Technology*, Vol. 65 No. 1, pp. 185-188. https://doi.org/10.1016/j.cirp.2016.04.051

Andres, M. et al. (2019), “Technical Application of the System Architecture Framework (SAF)”, *Tag des Systems Engineering*, Vol. 2019

Antonino, P.O. et al. (2018), “Bridging the Gap between Architecture Specifications and Simulation Models”, *IEEE International Conference on Software Architecture Companion (ICSA-C).* https://doi.org/10.1109/icsa-c.2018.00029

Doerr, J. et al. (2005), “Non-functional Requirements in Industry - Three Case Studies Adopting an Experience-based NFR Method”, *13th IEEE International Conference on Requirements Engineering (RE05)*. https://doi.org/10.1109/re.2005.47

Eigner, M. et al. (2015), “mecPro²- Entwurf einer Beschreibungssystematik zur Entwicklung cybertronischer Systeme mit SysML”, *Tag des Systems Engineering*, Vol. 2015. https://doi.org/10.3139/9783446447288.017

Eigner, M., Dickopf, T. and Apostolov, H. (2018), “The KSCM as Part of a Holistic Methodology for the Development of cybertronic Systems”, *The Context of Engineering 4.0: Proceedings of the DESIGN 2018 15th International Design Conference*. https://doi.org/10.21278/idc.2018.0138

Góngora, H.G.C., Gaudré, T. and Tucci-Piergiovanni, S. (2013), “Towards an Architectural Design Framework for Automotive Systems Development”, In: Aiguier, M., Caseau, Y., Krob, D. and Rauzy, A. (Eds.), *Complex Systems Design & Management*, Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-34404-6_16

Haberfellner, R. et al. (2019). Systems Engineering Fundamentals and Application, ISBN 978-3-030-13431-0 (eBook). https://doi.org/10.1007/978-3-030-13431-0

Jokioinen, E. (2016). “Remote and autonomous ships - the next steps” [online], Available at: http://www.rolls-royce.com/~/media/Files/R/Rolls-Royce/documents/customers/marine/ship-intel/aawa-whitepaper-210616.pdf (accessed 10.11.2019).

Pelliccione, P. et al. (2017), “Automotive Architecture Framework: The experience of Volvo Cars”, *Journal of Systems Architecture*, Vol. 2017. https://doi.org/10.1016/j.sysarc.2017.02.005

Walden, D. et al. (2015), *INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities*, Wiley, Hoboken, ISBN: 978-1-118-99940-0

https://doi.org/10.1017/dsd.2020.170 Published online by Cambridge University Press