The Journal of Systems & Software 189 (2022) 111306

Contents lists available at ScienceDirect

The Journal of Systems & Software

journal homepage: www.elsevier.com/locate/jss

Towards a logical framework for ideal MBSE tool selection based on discipline specific requirements ☆

Azad Khandoker [a,*], Sabine Sint [b], Guido Gessl [a], Klaus Zeman [a], Franz Jungreitmayr [a], Helmut Wahl [a], Andreas Wenigwieser [a], Roland Kretschmer [c]

[a] Institute of Mechatronic Design and Production, Johannes Kepler University Linz, Altenberger Straße 69, 4040 Linz, Austria
[b] Christian Doppler Laboratory for Model-Integrated Smart Production (CDL-MINT), Department of Business Informatics - Software Engineering, Johannes Kepler University Linz, Altenberger Straße 69, 4040 Linz, Austria
[c] Institute for Software Systems Engineering, Johannes Kepler University Linz, Altenberger Straße 69, 4040 Linz, Austria

ARTICLE INFO

Article history:
Received 25 March 2021
Received in revised form 20 February 2022
Accepted 14 March 2022
Available online 21 March 2022

Keywords:
MBSE
Filtration method
Tool selection
Software QFD

ABSTRACT

Model-Based Systems Engineering (MBSE) has emerged with great potential to fulfill the non-linearly rising demand in interdisciplinary engineering, e.g., product development. However, the variety and complexity of MBSE tools pose difficulties in particular industrial applications. This paper tries to serve as a guideline to find the ideal tool for a specific industrial application as well as to highlight the key criteria that an industry might consider. For this purpose, we propose a logical framework for MBSE tool selection, which is based on market research, the approaches of Quality Function Deployment (QFD), and decision matrix. As customers are at the center of any product, accordingly the needs of MBSE tool users are addressed within this research as the fundamental starting point. Market research and extensive discussions with MBSE tool vendors and academia show the current situation of MBSE tools. To compare the performance of the considered tools, a set of user needs is defined. QFD is performed to analyze the user needs with respect to evaluable technical properties. Subsequently each tool performance is assessed using a decision matrix. Through this process, a well-defined functional structure of MBSE tools is sketched, and in order to identify the properties of an ideal tool, all the attributes of different MBSE tools are mapped to a common platform. For the purpose of evaluation, we apply our proposed logical framework to select an exemplary MBSE tool for interdisciplinary application.

1. Introduction

Model-Based Systems Engineering (MBSE) is defined by the International Council on Systems Engineering (INCOSE) as "the formalized application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases" (Incose, 2015). Hence, MBSE is the digital representation of traditional (document-based) Systems Engineering (SE). By this definition, an MBSE method is supposed to support and supervise all the phases of a Product Life Cycle (PLC) from product development until replacement/retirement, which is hard to achieve through traditional document-based SE. The life cycle phases are dominated by the time domain, where ensuring proper communication, consistency, and traceability are major challenges. To supplement those challenges, MBSE has come into play to ensure proper communication, traceability, consistency, design precision, re-use of artifacts and to master complexity of SE methods.

MBSE tools have to cover all phases of a PLC, which includes tasks such as modeling, simulation, system architecting, design, verification, validation, production, delivery, maintenance, or project management. Currently, most of the available tools (software systems) to support these tasks, typically CAx-systems for Computer-Aided Design (CAD), Computer-Aided Manufacturing (CAM), Computer-Aided Engineering (CAE), or multi-physics simulation, Enterprise Resource Planning (ERP) systems, Product Lifecycle Management (PLM) systems are arranged in widely distributed, mostly isolated, discipline- or task-specific silos of information. To the best of our knowledge, there is no single tool on the market that can ensure an integrated performance of all MBSE tasks. However, for the initial phase of the PLC, the System Development Life Cycle (SDLC), there are already some tools based on modeling languages such as Systems Modeling Language (SysML) that can be used for a successful realization of certain parts. They have the ability to provide interfaces to

☆ Editor: Doo-Hwan Bae.

* Corresponding author.
E-mail address: azad.khandoker@jku.at (A. Khandoker).

https://doi.org/10.1016/j.jss.2022.111306

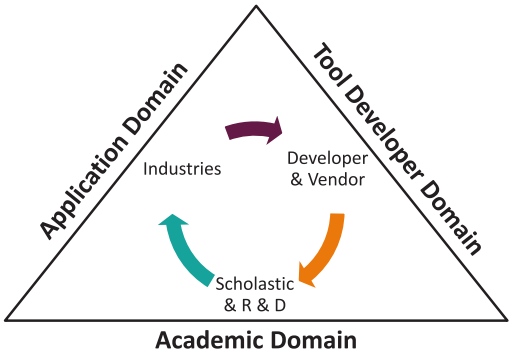


Fig. 1. Domains associated with MBSE.

other relevant tools that are used to support the remainder of the PLC. The available MBSE tools (System Modeling Tools), their capabilities as well as their selection process for the respective domain are presented in the following parts of this paper.

To understand the actual situation of MBSE, it is not enough to consider only the tools. Rather, it requires a holistic view of the associated domains that actually play a role in the emergence of SE and MBSE and manage their evolution through numerous changes over time. Through our research we have found the following three domains, respectively `Academic Domain`, `Tool Developer Domain` and `Application Domain`, which are inherently related to the above mentioned scenario of evolution and shown in Fig. 1. These three areas are causally interlinked and a change in any of them affects the others. The interesting side of that phenomenon is their integral relationship influencing the MBSE related technologies to further expansion. As a result, we see many leaps and bounds, for example, in the various versions of SysML, system architecture tools, FMIs (Functional Mock-up Interface) and platforms for tool interfaces such as Syndia, Smartfact, etc. In the context of Fig. 1, the `Academic Domain` represents academic and research institutions such as universities, the `Tool Developer Domain` incorporates companies, who develop MBSE base tools in relation to developers and vendors, and the `Application Domain` includes industries that are applying MBSE to support product development and other phases of the PLC. During our research, we frequently observed a lack of harmony in those domains that actually prevents MBSE from reaching its full potential. The identified shortcomings are presented below.

Application Domain Shortcomings

- Lack of awareness, consensus or standards of how to apply MBSE in a specific industry
- Difficulty in bringing all stakeholders together
- Organizations (e.g., product development) are not fully engaged with modeling culture.
- Non-interoperability and inconsistency of MBSE tools (communication challenges)
- Too high user expectations — MBSE tool vendors promise a lot (what is needed for engineering along PLC), but in reality users experience "different scenarios", according to Kautz et al. (2018, p. 233) *"Like many other new and innovative technologies, MBSE has promised a lot and has not kept all of its promises"*.
- Unclear purpose of MBSE (both as a method and as a tool)
  - Applications are strongly related to specific use cases.
  - Unclear scope of MBSE methods/ tools (specific use cases versus general approach)
- Insufficient management, preparation, persuasion, commitment for introduction of MBSE (both methods and tools)
- Lack of a dedicated "inter-discipline" that takes the responsibility for the integration of all involved specialized disciplines, e.g., as system architects, system analysts
- Lack of separate organization (roles, persons, functions, processes, tasks) with responsibility for the integration (system design, system integration) and unclear allocation of responsibility for the multi-faceted topic MBSE
  - Cultural change management is not yet practiced or recognized by all. There are cultures that resist change. This resistance is a form of risk aversion, but there are no changes without risks. Hence, changes have to be managed, guided and accompanied.
  - Various stakeholders and interests are involved in the application policy of MBSE.
- Many application domains lag behind the possibilities that are offered by the newest technologies and tools, due to technical, organizational, and educational issues.
- General modeling languages such as SysML, Unified Modeling Language (UML), Business Process Model and Notation (BPMN), etc. and their diagrams are still complex, confusing and too abstract for many users.
- Users of MBSE tools often understand MBSE as "SysML tool based application" although MBSE is a data centric approach, which goes far beyond SysML.
- Not enough trained work force to implement MBSE on industry level
- Real understanding gap if someone introduces MBSE, as SE is not so familiar in many industries and by its very nature depends on system theories, which also need to be addressed
- Many industrial users, who have tried to apply MBSE but failed to achieve the expected benefit from it, now consider MBSE as a hype.

Tool Developer Domain Shortcomings

- Individual tool development is based on SysML, UML, Lifecycle Modeling Language (LML), BPMN etc. languages and in-house codes that are foreign to many other engineering disciplines.
- There is no "single source of truth - tool" that could continuously and consistently integrate the entire PLC, which requires a bunch of tools with operating environments that ultimately confuse end users.
- Lack of tool developers instead of merely tool vendors.
- The advent of new names and marketing hypes (Model-Based Engineering (MBE), Model-Driven Engineering (MDE), Model-Driven Development (MDD), Model-Driven Architecture (MDA), Agile MBSE, etc.) confuses potential users.
- Tool vendors' marketing teams and development teams seem to be not in synergetic relation (marketing hype is being observed). An absence of interdisciplinary marketing teams is noticed.
- Co-simulation, interoperability and model exchange are among the greatest challenges (creating and maintaining code generators; design and development of Domain Specific Languages (DSLs); need of domain knowledge for modeling, etc.). Thus, different tool developers still cannot come to an agreement for a common solution.
- Lack of discussion about the MBSE evolution (Systems Theory → Systems Engineering (SE) → MBSE) leads to an education gap within the community and a missed opportunity to transfer knowledge to end users.
- To establish the purposeful use of MBSE, proper guidelines and tutorials (video or/and text-based materials) are required from tool developers, which in most cases are not freely available.

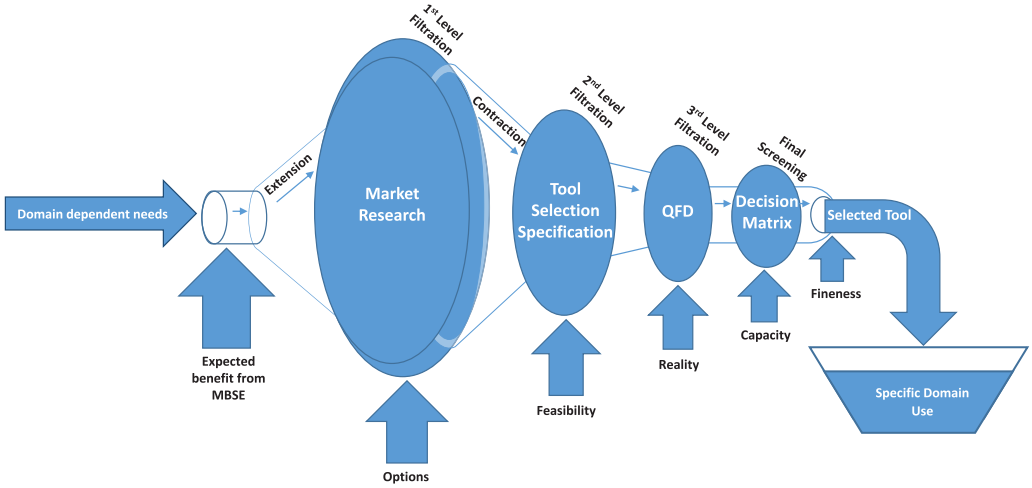


Fig. 2. MBSE filtration method.

Academic Domain Shortcomings

- Often book authors (for SE and MBSE) try to present examples and various discussions from the ICT/computer science domain to promote proper understanding. However, for learners from different disciplines, this may give the impression that MBSE is dedicated exclusively to the discipline of computer science.
- MBSE tools are more focused rather than methodologies that are actually the lifeblood of MBSE.
- A discipline, so-called inter-discipline, is needed to manage, relate and establish connections between different involved disciplines.
- Lack of dedicated "inter-discipline" personnel that takes the responsibility for the integration of all involved specialized disciplines, e.g. system architect, system analyst, etc.
- Lack of education, curricula, courses at universities or schools for such inter-disciplines (system design, system integration, system architect)
- Not enough MBSE related Research and Development (R&D) around the world, mostly economically developed countries are involved in R&D but only very few emerging economy and developing countries are participating.

Consequently, considering the above-mentioned shortcomings, it is necessary to bridge the gaps between these three domains, and their active participation needs to be ensured to reduce or root-out the shortcomings. However, it is wise to concentrate on the application domain at first as this domain includes the users, and innovations are mostly user centered. Therefore, we focus on this particular domain in this article for the ideal MBSE tool selection. For this purpose, a suitable specification of requirements needs to be established first. Additionally, as MBSE is a young field, the users require orientation about the proper use and implementation of tools for their existing product development process. The associated costs (licensing, workforce training, maintenance and third party supporting tools) are seemingly high and the interested organization should have a dedicated MBSE team to be developed in parallel to the existing product development team to deploy MBSE in their organization (Villhauer, 2016). It is necessary to balance out all the associated shortcomings and to make decisions for the most suitable MBSE tool on the basis of requirements and the actual state of the art with respect to methods and tools, which can bring potential benefits to the particular organization by saving time, effort and money.

The remainder of this paper is as follows. In Section 2, we propose a filtration method for selecting a particular tool for a specific domain application and describe in detail each filtration step such as market research, Quality Function Deployment (QFD), and decision matrix. Section 3 shows an exemplary application of the approach on the basis of tool capabilities to identify the most suitable MBSE tool for a domain specific application. Section 4 presents some work related to our research. Finally, in Section 5 we conclude our paper and provide a outlook.

2. MBSE filtration method

The selection process is a challenging task. To the best of our knowledge, there is no proven method to make the right selection for interdisciplinary applications since the requirements are dependent on the discipline as well as on the application. However, there are some tools such as decision matrix, Pareto analysis, T-chart, conjoint analysis, decision tree, etc., offering the opportunity to find a suitable or improved decision, although it might not be the optimal decision. Considering the mentioned facts, MBSE is a versatile discipline, which is complex to implement in an organization where the team members follow traditional methods (e.g., document-centric) and tools for engineering tasks such as product development. Furthermore, re-organizing the existing team for MBSE tool use is also difficult and very few companies are willing to impede or even change their winning team. Before companies can attempt such kind of change, they need to ensure that they have the necessary capabilities and expertise to deal with the involved technical, societal and financial factors. In the following subsections, we describe our proposed filtration method, the different parts of it, and finally explain some limitations of the approach.

2.1. The method's elements

There are many MBSE tools on the market. The systematic discovery of a suitable tool with respect to the customer requirements is the main objective of this research. Therefore, a filtration method (cf. Fig. 2) is developed for filtering out less-suitable tools. In this method, four different filtration membranes are designed, namely, `Market Research`, `Tool Selection Specifications`, `Quality Function Deployment (QFD)` and `Decision Matrix` to filter out the irrelevant requirements and tools.

In the beginning of the process, a set of discrete domain dependent needs (cf. Section 2.2) is pushed through initiation

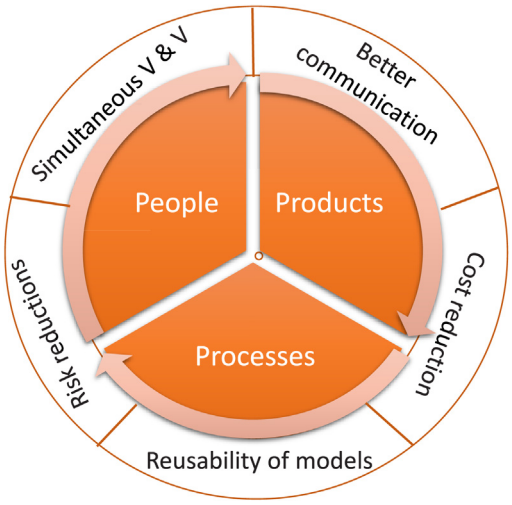


Fig. 3. Initial needs of MBSE in product development.

layers of the filtration process considering the expected benefit from MBSE. Based on those needs, the available MBSE tools are investigated through `Market Research` (first level filtration, cf. Section 2.3), which is actually the expansion of our initial needs since there are many new variables. Then `Tool Selection Specification` (second level filtration, cf. Section 2.4) is done considering the feasibility of the available MBSE tools. To perform the second level filtration, deeper knowledge on MBSE tools is required. Hence, a team from the IMDP (Institute of Mechatronic Design and Production) assumed the role of a user and a questionnaire was prepared based on their expectations towards an MBSE tool. The questionnaire (cf. Appendix) was then sent to certain MBSE tool developers to learn the unknowns of the considered tools. After receiving the answers, experts from MBSE fields (Institute of Business Informatics & Software Engineering, Institute for Software Systems Engineering, and another team from the IMDP at JKU Linz) formulated the `Tool Selection Specification` (cf. Section 2.4) considering the shortcomings described in the introduction section. To better understand and decide what is possible or not, `QFD` (third level filtration, cf. Section 2.5) is then performed. Previously found tool selection specifications are used as customer needs in the QFD, and to support user requirements, technical properties are allotted. Through this QFD, comparative assessments of different MBSE tools with respect to customer demands are also conducted. At the end of the method, `Decision Matrix` is applied as a final screening (cf. Section 2.6).

2.2. Domain dependent needs

As a first step of our filtration method (cf. Fig. 2), we have to identify the domain dependent needs that are relevant to MBSE. The development of SE towards MBSE is the reflection of an increasing demand in the application domain (cf. Fig. 1) for a unified support and control of the PLC phases. Fig. 3 describes the needs that are more or less inherently present in every kind of project, regardless of whether the project is dedicated to simple or complex system design and development. The transformation of SE towards MBSE is a transition from "document-centric" to "model-centric" approaches (Mitchell, 2014). It enables engineering with formal models, i.e. machine-readable and processable representations, and offers a number of advantages to drive the engineering process effectively and efficiently (Bézivin, 2005; Incose, 2015). Thus, the product development team would design models of the systems instead of documents. The models are then used along the entire product life cycle, which will lead to various benefits such as cost savings in the long run. One of the challenges to achieve this objective is to incorporate the necessary domain-specific needs in the MBSE tools. Therefore, an analysis of domain dependent needs is necessary to narrow down the application field and to specify the basis for the market research phase. Through market research (cf. Section 2.3), the further expansion and tailoring of the domain specific user needs will be initiated.

2.3. Market research

`Market Research` (cf. Fig. 2) is conducted for gathering information about the customer needs, financial factors and the MBSE tools' idiosyncrasies, which triggers the following questions

- What are the domain specific user needs?
- What is the financial ground to invest for implementing MBSE to a business?
- What are the key components and use cases of MBSE tools?

First, to answer these questions, domain specific user needs have to be analyzed. Historically, MBSE is mostly applied in the Aerospace and Software industries. At present, it catches the interest of many other industries due to its interdisciplinary applications, as systems are getting more and more interdisciplinary and complex. However, as a result, user needs are also becoming broader and more diverse and must be handled along with system complexity. On top of any user needs, interoperability and consistency of MBSE tools are main concerns. Therefore, on the one hand, market research is considered as the extension route to identify additional domain specific needs, as the initial needs were very basic and common. On the other hand, market research is also considered as the first filtration process to select MBSE tools common points in the MBSE arcade, which is necessary for the further refinement of the user needs.

Second, the financial investment involved in implementing MBSE have to be investigated. Cost is one of the most important parameters that is embedded with any kind of product design and development or application decision. Indeed, it is necessary to evaluate the economic importance beforehand to inaugurate any product or service to the market. The key challenges for implementing MBSE to the running business model are whether it is technically feasible and financially beneficial in the long run (Madni and Purohit, 2019). In addition, it is important to analyze whether such a transition is achievable within budgetary constraints (Madni and Purohit, 2019).

Third, the key components are identified. To acknowledge and realize MBSE, we need to understand the whole discipline, which covers the available types of MBSE tools, methodologies, multi-directional development pictures, fundamental components of the tools and alternative approaches.

The customer needs identified through market research then form the basis for fine-tuning in the tool selection specification phase in order to have more explicitly defined customer needs.

2.4. Tool selection specification

Since a comprehensive market research in this field usually reveals a large abundance of data (number of ... etc.), further filtering is necessary. In our logical framework, the `Tool Selection Specification` is the second level of the filtration process. All information found from the market research is assessed and re-evaluated to find a common understanding as well as a suitable specification that reflects the engineering specific customer needs and wishes. Despite that, the selection may depend on the users' view of their environments and their mode of use. Therefore, the end user needs will drive the requirements of a MBSE tool. It is not always possible to satisfy all user needs from different engineering fields. For instance, to perform this analysis, surveys, interviews or questionnaires can be conducted.

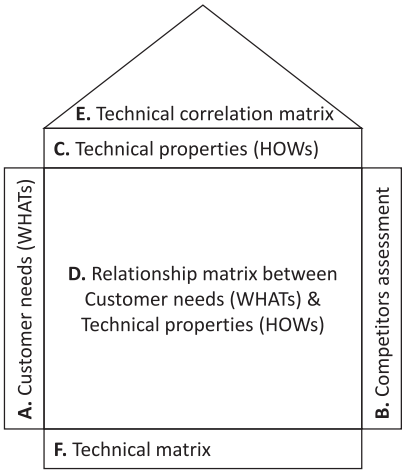


Fig. 4. Different active parts of the QFD.

2.5. Quality Function Deployment (QFD)

QFD is a means to assure design quality aiming at customer satisfaction (Alrabghi, 2013). It helps to analyze customer needs in relation to technical properties of a solution. Moreover, it helps to evaluate competing solutions and products. Since QFD analysis is a known method, we would not take the opportunity to describe the whole process and different attributes of the QFD in detail. Nevertheless, the dominant part of QFD is known as *House of Quality (HoQ 1)* matrix (cf. Fig. 4), which consists of six main features, they are respectively- `A. Customer needs` (WHATs), `B. Competitors assessment`, `C. Technical properties` (HOWs), `D. Relationship matrix between WHATs and HOWs`, `E. Technical correlation matrix`, and `F. Technical matrix` (Hauser and Clausing, 1988). The relationship matrix of WHATs and HOWs is used to identify the degree of relationship or linkage between each customer need and technical property. It is located in the center of the HoQ and considered the principal part of the QFD process, since the final analysis stage depends profoundly on the relationships of WHATs and HOWs. The technical correlation matrix indicates, whether there is positive/negative or no correlation between the different technical properties. The purpose of the technical matrix is to afford an initial ranking of the technical properties. The competitors assessment shows the status of, e.g., available tools (found from market research) considering the customer importance, which is also reflecting the comparison between tools.

2.6. Decision matrix

After the third level of filtration through QFD, the final screening of our proposed filtration method is executed through a decision matrix. A decision matrix is a list of values in rows and columns that allows identifying, analyzing, and rating the performance of relationships between sets of values (Chang, 2015). In general, by a decision matrix strengths and weaknesses of an investigated topic can be assessed more clearly on the basis of customer needs and technical properties. Thus, a user can make a clearer choice in relation to the needs.

2.7. Limitations

The aim of the presented method is a logical approach /framework to propose an ideal tool for a specific industrial application and to highlight key criteria for this purpose. The method should serve as a guide, although there are also a few limitations and delimitations, which are explained in more detail below.

Our method for the right tool selection touches different areas and phases of the PLC. A variety of requirements, functionalities, and application areas are analyzed to find out whether the selection fits or not. Each analyzed area can be very complex in detail and there are different approaches to describe them in more precise terms. For instance, within the methodology of requirements engineering (and its methodology) there are various approaches that analyze requirements with their structure in a more formalized way. To name an example, Goal-Oriented Requirements Engineering (GORE) such as Keep All Objectives Satisfied (KAOS) should be mentioned (van Lamsweerde and Letier, 2002; Werneck et al., 2009). Requirements and their dependencies are analyzed and weighted. In our approach, QFD includes implicitly such weightings in rough gradation, and for space reasons we therefore limit ourselves to them in this extensive area. The essential goal of considering a prioritization of individual Key Performance Indicators (KPIs) can be accomplished anyway by this means.

MBSE as method includes Requirements Engineering, Functional Architecture, Physical Architecture, V&V etc. If a user of a tool wants to perform certain actions, it must be analyzed to what extent tools can support these actions. Are there tools that offer support or not? With our method we check if the functionality/options are provided. We delineate our consideration from how good the support is for the option. To assess this more accurately, a serious evaluation would be required beyond the scope of the presented approach. Our goal is to provide a guideline as an auxiliary tool to clearly define and limit options based on measurable capabilities.

In this context, we want to state that our usage of the method (cf. Section 3) describes an approach for the new establishment of tools in a company. It does not deal with the analysis of existing tools, although the method itself is not limited to this. Based on experience in a company and user evaluations, the weighting in QFD or Decision Matrix may be different than in our example application. In this regard, when analyzing systems such as CAx systems, a distinction can be made between whether a system is being newly introduced or a migration is being performed. Thus, different weightings can be applied. Beside economic efficiency, the strategy and the benefits for a new introduction or migration must be evaluated first, which is a challenge in itself.

Of course, it can happen that the tool selection is performed several times in a company, which may lead to different results depending on meanwhile taken decisions. In order to integrate this successfully in the method, a benefit assessment of change in workflow, quality improvement, variant diversity, etc. would be necessary, where significant, measurable changes could arise. This is currently beyond the scope of the method presented, but will be considered as an extension in future work. The current process allows a better assessment of strengths and weaknesses of tools based on customer needs and technical properties.

3. Exemplary selection of MBSE tools for domain specific applications

In this section, we show an exemplary process on the basis of tool capabilities to identify the most suitable MBSE tool for domain specific application. The source files for the application of our method can be found at https://doi.org/10.6084/m9.figshare.16429974.v1.

3.1. Domain dependent needs

For the selection process of a suitable MBSE tool we start with the analysis of common needs for any project, namely costs, communication, re-usability of models, Verification & Validation (V&V), and risk reduction (cf. Fig. 3).

Indeed, cost is an important factor in any project. MBSE can be the tool to reduce the risk in project costs. In the traditional engineering approach or even in SE, often project costs overrun, schedules slip, performance reduces and, in the worst case, the whole project could be canceled (Harvey et al., 2012). To counteract these issues in terms of MBSE, cost reduction method derived from Model-Based Reviews (MBRs) could be an option. This method introduces transparent communications between the different disciplines involved in a project (Harvey et al., 2012). A route can be instantiated by the MBSE tool for internal document views, links and commenting functions to provide information to reviewers. Thus, they can immediately post questions and these can be answered directly by the developers. Since all documents no longer have to be printed out for review and the markups no longer have to be combined into a decision document, these time-consuming steps can be deducted and thus the costs of the review (cost of time) can be significantly reduced (Harvey et al., 2012). Additionally, from our research, we understand cost as an inherent property of the other needs such as communication, re-usability of models, V&V, and risk reduction.

As systems are getting more and more complex over time, it is important to master this rising complexity. Most of the traditional engineering approaches are inefficient in this regard. In a first step to tackle the issue of complexity, a proper communication has to be established by connecting different levels of stakeholders. Indeed, a communication medium has to ensure that information can be exchanged any time and that all involved stakeholders have access to actual information (no outdated data). Therefore, there is an urge to establish another level of direct communication between systems already at the component level. To enable this interaction between systems, it is crucial to consider the realization of communication by simulating and considering uncertain behaviors and ambiguous interactions already in the early stages of product development. For this purpose, models are essential and form the basis of the MBSE process. The advantages are that models are machine-readable, a kind of visualization, changeable, and dynamic. Thus, the communication process within the system models can be improved by creating feedback loops in the phase of product development. This communication process helps to bring all categories of stakeholders together by maintaining a constant exchange of information. This creates a symbiotic relationship and interaction within the system level as well as the stakeholder level and establishes a win–win situation (Zeman et al., 2020).

In-terms of engineering design, it is not enough to only model a system (product), but also to verify and validate it.. V&V is the process to evaluate the system requirements. Once the requirements of a particular system are verified in the early part of the PLC, the system can execute and visualize its designed actions (as simulation). Through this simulation the performance of the system is analyzed and evaluated before it is produced. This already allows validations and verification to be automated at an early stage and evaluation information is preserved for later phases. In fact, with regard to the MBSE application, the verification can be done through different model views, which, due to the dynamics of models, may lead to better results and performance in later phases of the PLC. In the so-called traditional engineering process, this automation is hard to realize at an early stage.

Risk is the built-in property of any kind of decision or choice. Where improvements are required, decisions are obligatory. Risk can be twofold: (*i*) Analyzed risk, that may lead to success; (*ii*) Embedded risk that may drive towards disaster due to its untraceable character. In MBSE, risk mitigation can be performed through the monolithic simulation possibilities as well as the use-case specific views. In MBSE approaches systems are built as models. Models can follow an abstract black-box approach and although they are not the real physical parts, they represent them in an abstract way and provide insights about them (which is known as an information model) (Brambilla et al., 2017). The potential use of these information models is endless and rational throughout the PLC.

Another important part of domain dependent needs is the question of how often information can be reused. In a project often different parts are exchanged, adapted, and used in other scenarios of the project or even copied for a new project as a starting point. The reusability of information is an important factor and is ensured by machine readability and dynamics of models.

It is a known fact that systems are inter-linked with products, people and processes. As a result, update or improvement requirements are difficult to implement in existing systems due to insufficient traceability and consistency. By applying MBSE, these issues can be solved more easily through engaging all teams under a common platform.

To fulfill the above-mentioned needs, MBSE plays a vital role in supporting this coordination by providing a means to capture all information from the different disciplines and to share it among the designers and other stakeholders (Harvey et al., 2012). However, based on the above discussions, the domain dependent needs are considered as the basis for the next phase, the market research.

3.2. Market research

In the market research we attempt to answer the questions about domain specific user needs, financial investment, and key components of MBSE tools by targeting and analyzing the `Application Domain`, `Tool Developer Domain`, and `Academic Domain` (cf. Section 1), which could potentially cover all the MBSE related aspects. To gain a deeper understanding on this particular topic various researchers of the JKU, who are working on MBSE topics, were contacted and interviewed in an open table discussion.

First, we analyzed the domain specific user needs. Thereby, we examined MBSE tools regarding how they support the PLC. The obtained customer needs from market research are as follows and in Section 3.3, they are further tailored to reflect more precise user requirements:

- Usability
- Design options
- Customization
- V&V
- Cloud based usability
- Simulations
- Model transformations
- Reusability and consistency
- Import/ export capability
- Security options
- Interoperability

Second, we want to contemplate the financial investment involved in implementing MBSE. The challenges to be tackled in this context are analyzed in our market research and the advantages of MBSE tools are substantiated by various publications. The

factors associated with MBSE investments and expected returns largely depend on the inherent characteristics of the system, such as complexity, characteristics of the operating environment, and system lifetime (Madni and Purohit, 2019). The complexity of the system can be considered as a function of the number of unique components in the system, the interactions between them, the amount of knowledge needed to develop the system, and the amount of information needed to describe the system (Madni and Purohit, 2019). Using MBSE, the time to market is reduced substantially and thereby a competitive advantage can be achieved (Schuler et al., 2015). Additionally, there is a strong correlation between better systems engineering and shorter delivery times (Boehm et al., 2008). In fact, according to one investigation, leading companies using MBSE methods reach their targets for quality, cost, time to market and sales in 84% of their projects (Schuler et al., 2015; Elm, 2011). A *Return On Investment (ROI)* of 3.5:1 was confirmed in a study of executed projects in terms of MBSE implementation (Schuler et al., 2015; Boehm et al., 2008; Honour, 2010). Indeed, MBSE investment covers a number of major costs such as infrastructure cost, model related cost (goal, purpose and scope of model as well as configuring the model for the intended number of users), training cost, etc. (Madni and Purohit, 2019). Furthermore, the obtained gains include early defect detection, reuse, product line definition, risk reduction, improved communication, usage in supply chain and standard conformance, etc. (Madni and Purohit, 2019), which are long-term benefits gained by a relatively high initial investment. Beside the above mentioned facts, the reasons for using MBSE in the industry could be manifold, (*i*) pressure from the supplier/vendor, (*ii*) to keep pace with collaborating companies, or (*iii*) feeling the need of modernization (etc.). These types of retrofitted needs are also important to consider to precisely evaluate the needs of MBSE. However, those aspects are beyond the scope of this paper, as they would lead this research in another direction. Therefore, we do not consider them further.

Third, we want to identify different key components for MBSE tools. From the market research, different open source and commercial tools were found. They have advantages one over the other. Open source tools are free of charge but for commercial uses, they have some restrictions such as (*i*) the tool requires a permission for commercial use or (*ii*) the user need to pay for the commercial use. In addition, add-on-tools (third party tools) for open source MBSE tools are not always free of charge. Despite these limitations, open source tools are a great step towards MBSE market expansion and educating users from different fields. Most of commercial tools are free to use with certain limitations. For instance, the user can try it for free only for one to six months or only a limited number of models can be created. Nevertheless, beginners could test the tool and at least they could understand what is possible or not with a certain type of MBSE tool.

From the `Market Research` (cf. Fig. 2) and the best of our knowledge, we found no open source or commercial MBSE (SysML) tool that can satisfy the needs of all PLC phases in terms of real-life applications. For this reason, the integration of third party tools (CAD, PLM, Application Lifecycle Management (ALM), Product Management (PM), Enterprise Resource Planning (ERP), simulation tools etc.) is required. However, third-party tools are usually less attractive to end users in terms of usability and financial facts. Alternatively, developers of Syndeia[1] developed a platform for integrated model-based engineering, where users can incorporate different domain specific tools such as ALM, PLM, CAD, simulations and database for MBSE use cases to satisfy all phases of PLC. Nevertheless, *Functional Mock-up Interfaces (FMI)* for model exchange and co-simulation of dynamic models show great potential of interoperability for different MBSE tools (Morkevicius, 2011). In addition, Smartfact[2] is a new MBSE online-based platform for tool integration. The tool can integrate three categories of tools, (*i*) modeling tools (Cat.-1), (*ii*) collaboration tools (Cat.2), and (*iii*) ALM tools (Cat. 3). They plan to integrate PLM tools as the fourth category to enable the support of the whole PLC of a product by one platform. In the above-mentioned multi-directional discussions only MBSE tools are considered, but the operation of a tool requires three key components as shown in Fig. 5: (*i*) `Language`, (*ii*) `Method`, and (*iii*) `Tool` (Roques, 2016). These components are necessary to understand the application of a tool. In most cases, the tool developers are using SysML or UML as the core language of their tool, but there are also solutions where DSLs are used. For instance, the tool Capella uses ARCADIA (ARChitecture Analysis and Design Integrated Approach) DSML language, which is inspired by UML/SysML and NATO Architecture Framework (NAF) standards (Roques, 2016). Users often come across the language (SysML, UML, BMPN, etc.), but methods are mostly overlooked, as they have no direct influence on the usability of a specific tool. Nevertheless, methods are essential since they guide the user to a proper usage of the language in order to create a particular system model. Therefore, it is not obvious to the tool user that methods have an impact on the requirements. Indeed, a good method can improve the usability of the environment as well as ensure higher productivity. To have a clear picture on method, language and tool, their definitions are embedded below:

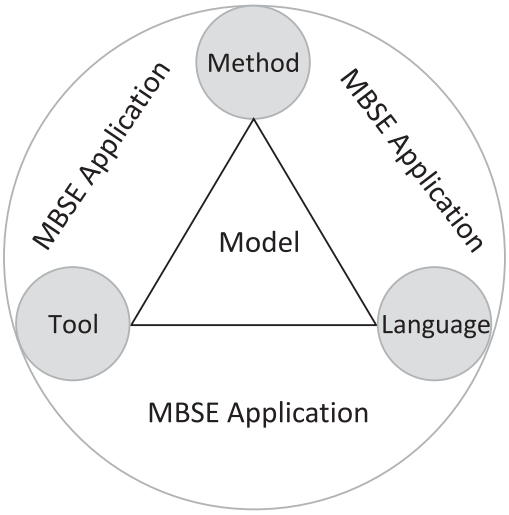


Fig. 5. Three pillars of MBSE tools.

`Language` formally describes the notations and elements that are required for modeling (e.g., SysML, UML, BPMN). A formal language, unlike a natural language, is one that operates with explicitly defined rules. This avoids ambiguity and ensures consistency (SysMLtools, 2018).

`Method` consists of techniques for performing a task, in other words, it defines the "HOW" of each task (Roques, 2016).

`Tool` creates the environment to apply the language and the method. It is an instrument to enhance the efficiency of tasks (SysMLtools, 2018).

Table 1 presents the key factors of different MBSE tools based on different criteria such as methodology, language, author, etc.

[1] http://intercax.com/products/syndeia/.

[2] https://smartfacts.com/.

Table 1
MBSE tools key factors based on market research.

Tools	Criteria								
	Methodology	Language	Author	Approach	Prime objective	Import files	Export files	Open source	Commercial
PTC Integrity Modeler	?	SysML, UML	PTC	Automatic Code	To provide single unambiguous system definition	.dwg, .idw, .CATPart, .prt	.xlsx, .xmi, .html, .docx, .pdf		X
Modelio	?	UML, BPMN	Modelio Soft	Model-Driven Approach	To support a balanced & efficient model/code development	.xmi, .bpmn, .uml	.xmi, .bpmn, .uml, .jpeg, .png	X	X
Cameo Systems Modeler	MagicGrid	SysML, UML, BPMN	No Magic	Based on Magic Draw Platform	To improve communication among the stakeholders	.emx, .epx, .efx, .mdl, .xmi, .reqif, .csv, .xlsx, .mof	.xlsx, .xmi, .xml, .html, .docx, .mof, various image format		X
Papyrus	?	SysML, UML, BPMN	LISE	Object Meta System Architecture developed	To be easily extensible as it is based on the principle of UML Profiles	.jar, .zip, .tar, .tar.gz, .tgz	.xmi, .xml, .project	X	
Capella	Arcadia	Arcadia DSML	Thales	Meta-Model Approach	To implement the Architecture Analysis & Design Integrated Approach	.jar, .zip, .tar, .tar.gz, .tgz, .reqif	.html, .reqif	X	
Enterprise Architect	Agile, FDD	SysML, UML, BPMN, etc.	SPARX	System Holistic Approach	Modeling through the entire ALM, PLM	.xmi, .bpmn-xml, .csv	.xmi, .bpmn-xml, .csv		X
Innoslate	LML	LML, SysML	SPEC Innovations	Implement the Life cycle Modelling Language (LML)	To support the full lifecycle with one tool LML	.docx, .csv, .pdf, .txt, .xmi (uml), .obj, .stl, .xml, .inno	.xlsx, .xmi, .docx, .csv, .mpp		X
IBM Rational Rhapsody	Telelogic Harmony SE	UML	IBM	Top-Down Hybrid Approach	To be vendor neutral and tool neutral	.xml, .reqif, .csv	.xml, .reqif		X
Core	Vitech	UML, SDL, SysML	Vitech Corp.	Incremental Layered Approach	To provide context free language for technical communication	.xml, .xlsx, .reqif, .csv	.xml, .docx, .csv		X
OBEO	Arcadia	ATL	Thales	Model-Driven Approach	To allow users for elaborate solutions by using business vocabulary	Can be managed by developing custom connectors	Can be managed by developing custom connectors	X	

The provided information is based on different websites of MBSE tool developers, blogs, phone conversations, YouTube videos, emails, publications and obtained questionnaire answers from the MBSE tools vendors (will be further discussed in Section 3.3). However, some criteria are not found in the mentioned sources and those fields are filled with question marks (?) in Table 1. It is important to note that some parts come from the tool developers and due to confidentiality agreements, references are not included. Indeed, this table (cf. Table 1) is used as a first filtration to restrict the number of suitable tools (which were found in our market research) with respect to key factors. To capture the scopes of MBSE tools (system modeling tools) in relation to their users, a use case diagram is shown in Fig. 6. In this particular case, the use case diagram is combined with a requirements diagram since some properties of an MBSE tool fit better into the requirements category. Use-case diagrams show communications among system transactions (use-cases) and external users (actors) in the context of system boundary (subject). Actors may represent wetware (persons, organizations, and facilities), software systems, or hardware systems (Weilkiens, 2007). Defining the relations between the system subject and the system actors is an effective informal method to define the system scope (Friedenthal et al., 2014). The ultimate goal of this presentation (cf. Fig. 6) is to assess the aspects of the MBSE users and to perform better analysis based on this.

3.3. Tool selection specification

To find a common understanding as well as a suitable tool specification, the information found from the market research is assessed and re-evaluated. To be more precise about the user needs, our research team contacted different MBSE tool developers with a questionnaire (cf. Appendix) to investigate deeper functionalities of a specific tool. Additionally, we wanted to find out if our understanding of customer needs and technical properties are the same as from tool developers. Moreover, the questionnaire helps to identify the comparative thoughts of MBSE tool developers and thus helps us to determine the fine-tuned specifications, which are explained below.

Usability: The extent to which a product can be used by specified users to achieve specified goals with respect to effectiveness,

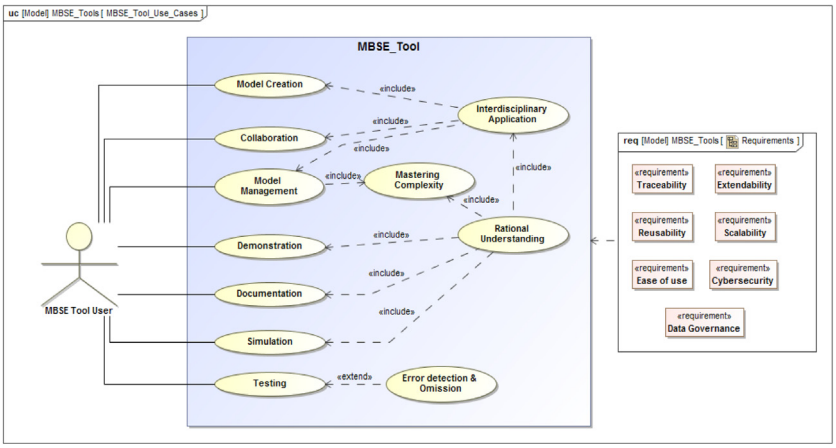


Fig. 6. SysML use case diagram for MBSE communication.

efficiency, and satisfaction in a specified context of use (Bevan et al., 2015).

Design Options: Design through different languages like SysML, LML, UML, BPMN etc.

Interdisciplinarity: The tool should enable integration of different models into a single repository while linking elements from other models conforming to different standards (Morkevicius, 2011). Specifically, the tool should be easily understandable by various stakeholders from different disciplines.

Customization: The tool should support multiple user roles with different sets of displayed/hidden features and customizable report templates, and provide an open API-enabling enhanced productivity through plugins and integration within the tool network.

Requirements Engineering: It organizes the stakeholders' requirements, performs seamless traceability and requirements validation in different stages of product development.

Functional Architecture: A system's functional model in relation to systems requirements that identifies system functions and their interactions (Space & Missile Systems Center, 2004).

Physical Architecture: It introduces further details and design decisions that include all the viewpoints and the way that a product will be designed. Links with requirements, functions, controls and operational scenarios are embedded. It is supposed to satisfy the functional architecture (Space & Missile Systems Center, 2004).

V&V: It is intended to check whether the system model is satisfactory and conforms to the designed purpose, taking into account the system requirements and owners' expectations (Sommerville, 2018). The V&V rules can also be customized by the users.

Simulation: It connects system architecture models with engineering and business analysis to calculate system performance (import engineering analyses into the model), check requirements, and perform design trade offs (No Magic, 2016). Additionally, simulations help to understand how tools are functioning by the use of FMI as well as to support abstraction and conceptual alignments between various parts of the system.

Life Cycle Competence: This describes that a tool has the fitness to comply or fit with any phase of the PLC.

Repository: This is a data administration and communication system to support the collaboration within large teams, where teams and models can be managed and accessed virtually. Team members can review and comment decisions through a virtual platform.

Communication & Support: How quickly do the tool vendors respond to the client via instant online messaging, email or by phone?

Training Resources: Availability of online training resources such as — YouTube videos, example models in product websites, white-papers, journals, and presentations.

Trial Version: The availability of a trial version and its duration are both chosen as criteria because these help users gaining knowledge about a specific tool in order to make their decision.

Version Complexity: How different is the previous version from the new one? How often do tool versions change? Is the old version model compatible with the new version's models? Is the usability hindered due to version changes?

Download & Installation: Compatibility with different operating systems should be guaranteed, or the tool should be offered as a web-based solution and thus be platform independent.

License VS Users: Per license, how many users can use the tool? What is the maximum number of users who can work in the same repository database system?

Stability of the Developer: Developer company strength in terms of number of customers, establishment, growth rate, etc. This stability is an important factor because the PLC may last for many years and the wind-up of a developer-organization during the PLC might heavily affect the user–company's business safety net.

Model Transformation: convert one model to another model or textual artifact by ensuring consistency.

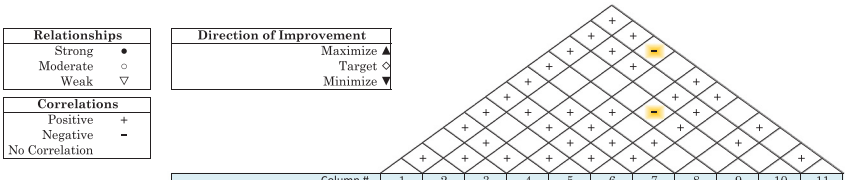
Human Readability: Humans can understand model representations in terms of tables/matrices, diagrams, graphs, etc.

Reusability & Scalability: Reusability allows the developed models and data to be utilized again by other modules or projects. Scalability enables the tool chain to handle a growing amount of engineering work or expand its capabilities to accommodate this growth (Jinzhi et al., 2018), for example supporting multiple databases with modifiable schema.

Costs: License initial costs, maintenance, training, and per year renewal costs.

3.4. QFD

Based on our research findings, the QFD analysis is performed in order to understand the reality of MBSE tools in the context



Relationships	
Strong	●
Moderate	○
Weak	▽

Correlations	
Positive	+
Negative	–
No Correlation	

Direction of Improvement	
Maximize	▲
Target	◇
Minimize	▼

Column #		1	2	3	4	5	6	7	8	9	10	11	Customer Competitive Assesment						
	Direction of Improvement	▲	▲	▲	▲	▲	▲	◇	▲	▲	▲	▲							
Row #	Customer Need (Explicit and Implicit) \ Technical Property	Methodology	Language	Symbiotic Aspects	Internal & External Interfaces	Traceability	Compliance & Interoperatibility	Regulations & Standards	Model validator	File Format	Ontology Support	Model Liberary	Enterprise Architect	Cameo	Innoslate	PTC Integrity Modeler	Modelio	Papyrus	Capella
1	Usability	▽	○	●	○	○	▽	●	○	○	●	●	5	5	4	3	3	3	5
2	Design Options	▽	●	▽	○	○	○	●	▽		●	●	5	4	4	4	3	3	3
3	Interdisciplinary Application	●		●	●	●	○	●	○	○	●	●	3	3	3	3	2	2	2
4	Custimazition	▽		●	○	●	○	○	○		▽		3	3	3	3	2	1	2
5	Requirement Engineering	○	▽	▽	○	●	▽	○	○			○	4	4	4	4	2	1	1
6	Functional Architecture	○	○		○	●	▽	▽	▽			●	3	3	2	2	1	1	4
7	Physical Architecture	▽	○		○	●	▽	▽	○		○	●	2	2	2	2	2	2	3
8	Verification & Validation (V & V)	○	○	●	○	○	○	○	▽			▽	3	3	4	4	1	2	1
9	Simulations	▽	○	▽	●	▽				●	○	▽	3	3	3	3	2	1	1
10	Lifecycle Competence	▽	○		○	○	▽	▽	▽		○	▽	3	4	5	3	2	1	1
11	Repository	▽		●	○	●	○	▽				●	4	4	5	5	3	1	1
12	Communication & Support			○				○					4	5	5	2	3	2	4
13	Training Resources			○								▽	3	4	4	2	3	4	4
14	Trail Version			○									2	5	4	2	5	5	5
15	Version Complexity			○	▽							▽	5	4	4	4	3	4	5
16	Download & Installation			○				▽					4	3	5	2	5	5	5
17	Licence VS User							▽					4	3	3	2	5	5	5
18	Stability of the Developer												5	5	5	5	5	5	5
19	Model Transformation	▽	○	○	○	○	○	○	○	●	●	●	4	4	3	3	3	3	4
20	Human Readability		▽	●	▽		▽	○		●	▽		4	4	4	4	3	2	2
21	Reusability & Scalability	▽	▽	○	●	●	▽	▽	○	○	▽	●	3	3	3	3	1	1	1
22	Costs			○	○		▽					●	4	2	2	2	5	5	5
	Max Relationship	9	9	9	9	9	3	9	3	9	9	9							
	Technical Importance Rating	126,9	162,4	382,8	287,1	372	128	255,9	119,4	167,7	235,5	434,4							
	Relative Weight	5%	6%	14%	11%	14%	5%	10%	4%	6%	9%	16%							

Fig. 7. QFD of MBSE tools.

of customer needs as well as to gain an improved filtration for the decision matrix. In this QFD, technical properties of a solution (HOW) are specified alongside the customer needs (WHAT) which are the user needs we identified in the previous subsection (cf. Section 3.3). The technical properties for the QFD were developed by our research team to be suitable for realizing the customer needs, based on their know-how and expertise in the field of MBSE. It is important to note that the technical properties here are not the end code solutions of the tool (MBSE software) as MBSE tools are evolved through software coding (Alrabghi, 2013).

We performed our QFD based on the HoQ and results are shown in Fig. 7. The relationship matrix of WHATs and HOWs was populated by our research team. The `relationship matrix` uses `weak`, `moderate` and `strong` relations with weights of (1, 3, 9) respectively. If there is no relation at all, the field stays empty. Based on our analysis in the field of engineering, the `technical correlation matrix` is set with a `positive` or `negative` mark for the correlation. If there is no correlation, the field stays empty. Based on the `Technical Importance Rating`, MBSE tool developer teams have the opportunity to map and propagate wise design solutions. The technical importance rating is calculated as shown in Eq. (1), where i describes one specific customer need out of all n customer needs.

$$\textit{Technical Importance Rating} = \sum_{i=1}^{n} (\textit{Percentage Of Customer Importance}_i \cdot \textit{WeightOf Relationship}_i) \quad (1)$$

For instance, the technical importance rating for the technical property methodology is computed as follows (relationship of WHATs and HOWs with no relation are ignored since the calculated value is 0): [Technical importance rating $= 5.38 * 1 + 5.38 * 1 + 4.30 * 9 + 4.30 * 1 + 5.38 * 3 + 4.30 * 3 + 4.30 * 1 + 5.38 * 3 + 4.30 * 1 + 4.30 * 1 + 5.38 * 1 + 5.38 * 1 + 4.30 * 1 = 126.90$].

The `relative weight` describes the percentage of each technical importance rating value regarding all technical importance rating values. It gives an idea about the importance of each technical property and its necessity in relation to the customer needs. In addition, the technical importance rating helps the MBSE software development team to prioritize the used `technical property` in the QFD based on the found rank order. For instance, the technical property `Model Library` has the highest score with 434.4 among all technical properties. Thus, during development of QFD software, the model library should get the highest priority and the other technical properties will be prioritized based on their found rank order in the QFD.

Furthermore, in Fig. 7 the `direction of improvement` row shows the current status of the technical properties of the considered solution by `maximize` (needs improvement to fulfill projected goal), `target` (satisfactory), and `minimize` (too sophisticated, need to be reduced for implementation).

The `customer competitive assessment` shows the status of the available MBSE tools (found from market research) considering the customer importance, which is also reflecting the comparison between tools. `Customer importance` is considered

from maximum of 5 to minimum of 1. The maximum and minimum values are varied based on the necessity of each customer need or how important the needs are for the customer. Mostly the tolerances of choice (customer importance) become more precise, when the involved QFD team gains experience over time and tracks the changes of QFD by comparing the new with the old version (QFD). Thus, in the customer competitive assessment the values vary from 5 to 1 depending on whether the tool meets the customer need or not. In the customer competitive assessment, not all listed tools of Table 1 are taken into account due to our lack of expertise in these tools. Based on the customer importance, it is possible to calculate the `percentage of importance` by applying Eq. (2).

$$Percentage\ Of\ Importance = \frac{Customer\ Importance \cdot 100}{Total\ Number\ Of\ Customer\ Importance} \quad (2)$$

The percentage of importance describes how important the customer need is compared to all the customer needs. As an example, considering the first customer need `Usability` we can calculate the percentage as follows: Customer importance is 5 and the sum of all customer importance values is 93. Thus, the percentage of importance is 5.38% (by applying Eq. (2)). Our main points of interest in this QFD are customer needs and technical properties relations as well as the competitiveness of different MBSE tools in relation to each other. However, the maximum relationship column presents the maximum weight of relationship in each row or column of customer needs and technical properties relation.

It is important to note that the performed QFD in this paper is only one example. If a QFD analysis is carried out in a larger context in MBSE software development by, e.g., a company, it might be in more detail, and other technical properties and customer needs might be considered.

It is necessary to point out that the process of QFD is not the same in software industry as in the classical field of manufacturing, although the purpose of QFD is universal. Customers of both areas have certain needs concerning the use of products and developers are supposed to satisfy customers by developing desirable solutions, where time, costs, and quality are met (Herzwurm and Schockert, 2003). Indeed, software QFD usually is focusing on the ability to prioritize the engineering activities and is paying less attention to the deployment of the software's last line of code (Herzwurm and Schockert, 2003). Therefore, for the technical properties of QFD, relevant higher-level technical aspects are prioritized. On the other hand, common technical measures, e.g. cut/copy/paste, activation of objects by mouse clicks, network features, data files, etc. are not considered for this QFD. To be explicit as well as to justify the considered technical properties, their definitions are presented below.

Methodology consists of techniques, processes and methods for performing MBSE related tasks (Estefan, 2008).

Language: MBSE tool language operates with explicitly defined rules. This avoids ambiguity and provides consistency (SysML-tools, 2018). It is important to consider the implementation process of different languages (e.g., SysML, BPMN, UML) in the same platform in order to support the different phases of a PLC.

Symbiotic Aspect: The tool not only supports the user but additionally learns from the user for future implementations by ensuring, e.g., better communication across different wings of a particular tool, which will deploy a win–win situation and enable adaptability along the PLC (Zeman et al., 2020).

Internal & External Interfaces: Internal interfaces define how different models (requirements, functional, physical, performance, or process models) are connected within a tool. To support that tool with integrated models, data can be exchanged, read or analyzed by other tools or platforms (i.e. the tool developers should also implement external interfaces for co-simulation).

Traceability: It helps to authenticate the history by means of recorded data. Inauguration to the data information defines the necessity of citation for the traced artifact that could provide the trajectory of transformation.

Compliance & Interoperability: Tools interoperability is the ability to use parts of another system (Oemig and Snelick, 2016). Thus, it is possible to exchange models and data in order to process them further. Among other things, it is based on interfaces. For ensuring interoperability and compliance between tools, as a minimum requirement, the tools and their models should be consistent (notation, syntax, semantics) to enable connectivity and processability.

Regulations & Standards: There exist various domain regulations (e.g., mechanical, software regulations) and standards (e.g., ISO 9126 & IEEE 830:1998). A tool should have the ability to implement and support them allowing a user to determine whether the regulations are met or not. In order to fulfill this requirement, the tool can for example extemporize the design safety boundary as well as fault tree generation to confirm data safety. Additionally, tools should guarantee functionality, reliability, safety, efficiency, maintainability, portability, database and documentation observing standards.

Model Validator: It supports the user to validate models if they are syntactically or semantically correct. Model validators can display corrections based on the characteristics of the models and a user can accept them or not, based on his/her knowledge. Tools can have integrated model validators or external validators should be integrated in order to offer this correction possibility to the user.

File Format: The tool providers develop tool solutions that support different file formats in order to import and export data by ensuring human and machine readability.

Ontology Support: The tool should provide a basic but comprehensive set of design elements that establish well-defined domain concepts in terms of the terminology, definitions, and relationships as needed to model real world applications (L. S. Committee, 2015; Graves, 2007).

Model Library: It supports storage and retrieval of models that may be configured with respect to the intended application. It is a place in the tool where different models are available to promote and increase the reuse and exchange of models.

From the QFD analysis (cf. Fig. 7), we found out that there is a great room of improvement for almost every technical property to satisfy the current needs in the field of MBSE. Most of the MBSE tools (in our QFD) do not meet the customer needs in comparison to designated customer importance. Indeed, the customer needs and their importance depend on the application domain. In the correlation matrix (the roof of the HoQ 1), mostly positive relations are observed. Thus, it is clear to the MBSE tool development team that changes in one technical property can affect other properties positively or negatively (Alrabghi, 2013). However, by observing the customer competitive assessment to customer importance, it could be assumed that the examined MBSE tools fulfill the initial purpose (SDLC), however certainly not support the entire PLC.

MBSE tool development is a complex process, as diverse stakeholders communities have to be considered. In addition, customer needs change over time. To accommodate such changes MBSE

Decision Model		MBSE Tools							
		Enterprise Architect		Cameo		Innoslate		PTC Int. Modeler	
Criteria	Weight	Rating	Score	Rating	Score	Rating	Score	Rating	Score
Usability	5	5	25	5	25	4	20	3	15
Design Option	5	5	25	4	20	4	20	4	20
Interdisciplinary Application	4	3	12	3	12	3	12	3	12
Customization	4	3	12	3	12	3	12	3	12
Requirement Engineering	5	4	20	4	20	4	20	4	20
Functional Architecture	4	3	12	3	12	2	8	2	8
Physical Architecture	4	2	8	2	8	2	8	2	8
Verification & Validation	5	3	15	3	15	4	20	4	20
Simulation	5	3	15	3	15	3	15	3	15
Lifecycle Competence	4	3	12	4	16	5	20	3	12
Repository	5	4	20	4	20	5	25	5	25
Model Transformation	5	4	20	4	20	3	15	3	15
Human Readability	4	4	16	4	16	4	16	4	16
Reusability & Scalability	4	3	12	3	12	3	12	3	12
Cost	5	4	20	2	10	2	10	2	10
Total	68	53	244	51	233	51	233	48	220

Description	Rating
Very low fit	1
Low fit	2
Fit	3
Good fit	4
Excellent fit	5

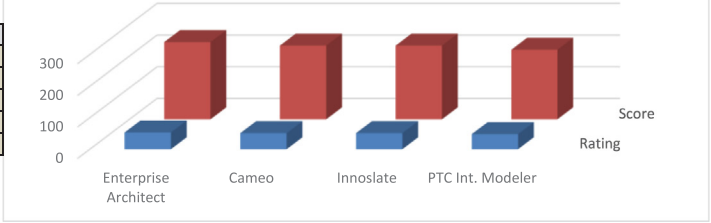


Fig. 8. Decision matrix of MBSE tools.

tool developers need to adopt new approaches with new methodologies. Hence, the progress in this field is a continuous process, since variables and boundary conditions change with time. Thus, this holds the engineering discipline in continuous challenge with customer needs. As a young discipline, MBSE has not yet reached the peak of its learning curve, whereas it is growing sharply.

3.5. Decision matrix

In our approach, the presented decision matrix supports the identification of suitable MBSE tools based on our previous analysis. Fig. 8 presents the results of our decision matrix for different MBSE tool developers. In contrast to the QFD, the open source tools are not listed anymore. The ratings of these tools in our QFD was lower than of the remaining tools. This may be related to their apparent focus on software development rather than supporting interdisciplinary needs in MBSE. We keep the same weightings of customer needs and customer importance from the QFD. But we dropped those customer needs from our `criteria` for which the relationships with technical properties in the QFD were weak, such as training resources, trial version, license, etc.

The rating for the various tools is used from the QFD for each criterion and then the scores of the tools are calculated by summing up the weighted rates (multiplying weight and rate). As it is shown in Fig. 8, the tool `Enterprise Architect` achieves the highest score with 242. `Cameo` and `Innoslate` have the same scores of 233. The score difference between these tools is quite low, but considering the comparatively strong customer requirements such as usability, design options and costs, Enterprise Architect is considerably better than its competitors. From our own analysis, `Enterprise Architect` offers a wide range of modeling and simulation possibilities and is quite intuitive to use, since it is structured similarly to other widespread programs (e.g., Microsoft Office). `Innoslate` is a modern tool for life cycle based MBSE approaches, since it applies the Life cycle Modelling Language (LML), which is considered as one of the future languages of MBSE. `Cameo Systems Modeller` uses an improved knowledge capture process for model/data reuse by capturing information in a more standardized way and leveraging built-in abstraction mechanisms of model-driven approaches (Pietrusewicz, 2019). In our opinion, this is an important step in the development of a symbiotic relationship between the tool and the user. The tool with the lowest score is `PTC Integrity Modeler`. Nevertheless, it can still be the best choice for a company. For instance, if a company has previously used PTC products (e.g., Creo/Windchill/Mathcad) and thus the users are familiar with the working principles and environments of the PTC tools. Thus, the `PTC Integrity Modeler` would be the best fit for their own team.

Furthermore, we compared the found results from the decision matrix with the user experience of the IMDP team to evaluate our exemplary approach. Their expertise in and notions on different MBSE tools confirm our results. However, the matrix results may vary, if another group of experts is applying it.

We can conclude that we cannot select the best tool out of this matrix, but strengths and weaknesses of tools with respect to customer needs and technical properties can be assessed more clearly. As a result, the overall picture of MBSE tools in relation to domain-specific requirements is covered, which is the main purpose of this paper.

However, some functionalities of different MBSE tools (commercial & open source) are not fully discovered by our approach. Thus, we could not figure out every detail of MBSE tools and deliver a pinpoint analysis, which is a limitation of our research. Nevertheless, our approach could help both parties, who are very ambitious about MBSE tools and disappointed about the potential use of MBSE tools, to find some answers with regard to MBSE tool selection. In addition, we believe our approach may serve as a general guideline for selection processes in various research domains.

4. Related work

Mastering the art of tool selection is quite laborious and complex. Depending on the engineering discipline, there are different needs that must be met, and therefore the selection is handled differently. During our literature search, we found no single method which fits all requirements for selecting appropriate

tools. However, we observed some common patterns in different publications.

Heindl et al. (2006) present a value-based tool selection approach for choosing an optimal tool for requirements engineering of project- or department-specific requirements. Due to the variety of departments and projects not a single tool is able to meet all requirements. Their tool selection approach is based on the value rating of the tool's features, where tools are evaluated and selected based on expert knowledge (features) (Heindl et al., 2006). Our approach is similar, but we rely on methods such as QFD and decision matrix for tool evaluation based on customer needs.

Powell et al. (1996) propose a working strategy for software tool selection, as there is a lack of acceptable tool selection methodology. In their process a generic issue checklist is used for the purpose of qualitative comparison of individual tools, ensuring traceability between the issues (tool requirements criteria and supporting evidence). The authors claim that their approach limits the risk of error in the tool evaluation. Their process consists of different stages, such as demonstration, assessment against criteria, and pilot study. After each stage a decision is made whether to proceed to the next one or to go directly to the pilot stage. We estimate that the early phases require fairly little investment. Only if the selection process reaches the pilot phase but then fails, significant time and effort will have been wasted. In our approach, we have no option to bypass any filtration steps. Compared to the approach in Powell et al. (1996), our approach may be slower, but it allows us not only to select the best available tool, but also to rate (in an abstract point score) how well that tool is suited for a specific application.

Another approach is presented by Kahani et al. (2019) where a qualitative framework for comparing and verifying metamodel based transformation tools is proposed. In their comparison process they assess the tools based on 45 facets, which are organized into six different categories namely, General, Model-level, Transformation, User Experience, Collaboration Support, and Runtime Requirements (Kahani et al., 2019). Based on the categories, the 60 different model transformation tools found are classified. The authors analyze in general what tools are available in the field of model transformations and do not try to make an optimal selection, leaving it open for the readers to evaluate which of these are best suited to their specific needs. As such their survey serves as a starting point in the search for model transformation tools to get a first insight into comparative aspects based on different attributes of the tools. Similarly, in Rashid et al. (2015), the authors analyze MBSE tools with regard to embedded systems using a systematic literature review, focusing on scientific research papers published between 2008 and 2014. They investigate several research questions and identify 39 tools used in current research of embedded systems which they categorize and analyze in several aspects. Due to their survey character, Kahani et al. (2019) and Rashid et al. (2015) both present information relevant to tool selection, but are not intended to make a specific recommendation for which tool to pick. In contrast, our paper not only examines scientific approaches, but also proposes a way to select MBSE tools for supporting the entire PLC.

In summary, it has been observed from different literature that many tools are available to support specific engineering domains. Mostly, depending on the purpose, specific requirements are set by researchers, and based on them, the search for the tools and the evaluation is carried out. However, the methodology differs in the different approaches, including selection based on (*i*) criteria, (*ii*) expert knowledge, and/or (*iii*) analysis through different defined stages. In our approach, we combine these types of approaches into an overall logical framework of a filtration process.

5. Conclusion and outlook

At present, one of the central questions in the MBSE application domain is "Which tools should be selected and how to ensure the effective use of the chosen tools?". If the wrong tool is chosen or used improperly, this can lead to considerable breakdowns in an organization (Morkevicius, 2011). Therefore, it is important to train the team early on, both in MBSE techniques and the use of the specific tool, to avoid unexpected costs for the company in the long run.

In this paper, we propose a logical framework of a filtration method for selecting a particular MBSE tool for a specific application domain. Within this framework, user needs are analyzed and prioritized, taking into account a set of domain specific requirements that is corrected through a filtration process. We were able to demonstrate the applicability of our method/framework by means of an exemplary study where it was possible to select one MBSE tool that fits best to the specific user needs. Companies/organizations can apply our approach in order to achieve a well-grounded MBSE tool selection for their own needs. They should plan and follow an implementation process for their team training, otherwise even a properly selected tool would fail to add any value in terms of application but might be a burden instead.

Regarding our presented approach, we propose the following next steps. First, we plan to apply our method in an industrial case study to check its applicability in a larger context. Second, we want to analyze if our requirements-driven tool selection framework is also applicable in the context of combining MBSE with cutting-edge digital engineering transformation, where hardware and software functions are integrated to design, simulate, and optimize complex products. Third, we want to extend our approach to integrate feedback from past choices in a company into improved analytics with (new) evaluable criteria such that the company can improve its decisions in the future. Finally, we want to use the presented approach in different application domains to check the generalization of our approach.

CRediT authorship contribution statement

Azad Khandoker: Conceptualization, Methodology, Investigation, Writing – original draft, Writing – review & editing, Formal analysis, Project administration, Data curation, Visualization. **Sabine Sint:** Writing – original draft, Formal analysis, Writing – review & editing, Data curation, Visualization. **Guido Gessl:** Writing – original draft, Formal analysis, Visualization. **Klaus Zeman:** Supervision. **Franz Jungreitmayr:** Validation. **Helmut Wahl:** Resources. **Andreas Wenigwieser:** Resources. **Roland Kretschmer:** Validation.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work has been supported by the COMET-K2 Center of the Linz Center of Mechatronics (LCM) funded by the Austrian federal government and the federal state of Upper Austria, the Austrian Federal Ministry for Digital and Economic Affairs and the National Foundation for Research, Technology and Development (CDG). We thank Andreas Nemetz for proof-reading.

Appendix. Questionnaire for MBSE tool

1. Is it possible to simulate the designed multidisciplinary product model with your MBSE tool? If yes, please name the kind of simulation/s.
2. Which Language/s is/are used by your MBSE tool? (UML, SysML, BPMN etc.)
3. How does your MBSE tool support multi-disciplinary product design?
4. How are the requirements managed in your MBSE tool?
5. How does the MBSE tool support the design-process of the Functional product architecture?
6. How does the MBSE tool support the design-process of the Physical product architecture?
7. Does your tool allow multi user access via data repository based system (Cloud base)?
8. Which types of file formats are supported by your MBSE tool? Please specify the import and export format separately.
9. Has your methodology been developed depending on a specific tool, if so which one?
10. What is the prime objective behind developing your tool?
11. How does the MBSE tool help to verify the designed systems or based on which criteria does your tool verify the designed systems?
12. Is the life cycle application integrated with the tool, or is another tool needed from 3rd part developer for life cycle application? Please provide the 3rd party life cycle tool name.
13. Is your tool compatible for information exchange with third party tools or software platforms?
14. How are the versions of the tool different from each other in terms of user interface?
15. How much is your company/tool annual average growth rate (in percentage %)? Please provide the name of your main clients (only company name).
16. How much does each license cost and for how many users?
17. Does your license have renewal fee for each year?
18. Do you charge for after sales service? If yes, then how much per hour (rough estimate)?
19. Do you think your tool is easily learnable through self-study?
20. What other important attributes of your tool are not mentioned above? (Optional)

References

Alrabghi, L., 2013. QFD In Software Engineering (Master's thesis). Kent State University.

Bevan, N., Carter, J., Harker, S., 2015. ISO 9241-11 revised: What have we learnt about usability since 1998? In: Kurosu, M. (Ed.), Proc. of the 17th International Conference on Human-Computer Interaction: Design and Evaluation, Part I. In: Lecture Notes in Computer Science, vol. 9169, Springer, pp. 143–151. http://dx.doi.org/10.1007/978-3-319-20901-2_13.

Bézivin, J., 2005. On the unification power of models. Softw. Syst. Model. 4 (2), 171–188. http://dx.doi.org/10.1007/s10270-005-0079-0.

Boehm, B.W., Valerdi, R., Honour, E.C., 2008. The ROI of systems engineering: Some quantitative results for software-intensive systems. Syst. Eng. 11 (3), 221–234. http://dx.doi.org/10.1002/sys.20096.

Brambilla, M., Cabot, J., Wimmer, M., 2017. Model-Driven Software Engineering in Practice, Second Edition. In: Synthesis Lectures on Software Engineering, Morgan & Claypool Publishers, http://dx.doi.org/10.2200/S00751ED2V01Y201701SWE004.

Chang, K.-H., 2015. Chapter 2 - Decisions in engineering design. In: Design Theory and Methods using CAD/CAE. Academic Press, Boston, pp. 39–101. http://dx.doi.org/10.1016/B978-0-12-398512-5.00002-5.

Elm, J., 2011. A study of systems engineering effectiveness: Building a business case for SE. In: INCOSE International Symposium, Vol. 21. pp. 248–262. http://dx.doi.org/10.1002/j.2334-5837.2011.tb01203.x.

Estefan, J., 2008. Survey of model-based systems engineering (MBSE) methodologies. INCOSE MBSE Focus Group 25, 1–70.

Friedenthal, S., Moore, A., Steiner, R., 2014. A Practical Guide to SysML: The Systems Modeling Language, third ed. Morgan Kaufmann.

Graves, H., 2007. Ontology engineering for product development. In: Proc. of the Workshop on OWL: Experiences and Directions. In: CEUR Workshop Proceedings, vol. 258, CEUR-WS.org, pp. 1–8, URL: http://ceur-ws.org/Vol-258/paper02.pdf.

Harvey, D., Logan, P., Waite, M., Liddy, T., 2012. Document the model, don't model the document. In: Proc. of the Systems Engineering/Test and Evaluation Conference and 6th Asia Pacific Conference on Systems Engineering. pp. 1–11.

Hauser, J.R., Clausing, D.P., 1988. The house of quality. Harv. Bus. Rev. 66, 63–73.

Heindl, M., Reinisch, F., Biffl, S., Egyed, A., 2006. Value-based selection of requirements engineering tool support. In: Proc. of the 32nd EUROMICRO Conference on Software Engineering and Advanced Applications. IEEE Computer Society, pp. 266–273. http://dx.doi.org/10.1109/EUROMICRO.2006.64.

Herzwurm, G., Schockert, S., 2003. The leading edge in QFD for software and electronic business. Int. J. Qual. Reliab. Manage. 20 (1), 36–55. http://dx.doi.org/10.1108/02656710310453809.

Honour, E., 2010. Systems engineering return on investment. In: INCOSE International Symposium, Vol. 20. pp. 1422–1439. http://dx.doi.org/10.1002/j.2334-5837.2010.tb01150.x.

Incose, 2015. INCOSE Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, fourth ed. Wiley.

Jinzhi, L., Wen, Y., Liu, Q., Gürdür, B.D., Törngren, M., 2018. MBSE applicability analysis in Chinese industry. In: INCOSE International Symposium, Vol. 28. pp. 1037–1051. http://dx.doi.org/10.1002/j.2334-5837.2018.00532.x.

Kahani, N., Bagherzadeh, M., Cordy, J.R., Dingel, J., Varró, D., 2019. Survey and classification of model transformation tools. Softw. Syst. Model. 18 (4), 2361–2397. http://dx.doi.org/10.1007/s10270-018-0665-6.

Kautz, O., Roth, A., Rumpe, B., 2018. Achievements, failures, and the future of model-based software engineering. In: Gruhn, V., Striemer, R. (Eds.), The Essence of Software Engineering. Springer, pp. 221–236. http://dx.doi.org/10.1007/978-3-319-73897-0_13.

L. S. Committee, 2015. Lifecycle modeling language (LML)specification. URL: http://www.lifecyclemodeling.org/spec/current. Accessed: 2018-12-03.

van Lamsweerde, A., Letier, E., 2002. From object orientation to goal orientation: A paradigm shift for requirements engineering. In: Proc. of the 9th International Workshop on Radical Innovations of Software and Systems Engineering in the Future, Revised Papers. In: Lecture Notes in Computer Science, vol. 2941, Springer, pp. 325–340. http://dx.doi.org/10.1007/978-3-540-24626-8_23.

Madni, A.M., Purohit, S., 2019. Economic analysis of model-based systems engineering. Syst. 7 (1), 12. http://dx.doi.org/10.3390/systems7010012.

Mitchell, S.W., 2014. Transitioning the SWFTS program combat system product family from traditional document-centric to model-based systems engineering. Syst. Eng. 17 (3), 313–329. http://dx.doi.org/10.1002/sys.21271.

Morkevicius, A., 2011. Making the Most of EA Modeling Tool. Technical Report, No Magic Inc..

No Magic, 2016. MBSE ecosystem: System and analytical models integration for critical systems simulation. URL: https://www.youtube.com/watch?v=mW1V5zETkZ4. Accessed: 2018-11-04.

Oemig, F., Snelick, R., 2016. Healthcare Interoperability Standards Compliance Handbook - Conformance and Testing of Healthcare Data Exchange Standards. Springer, http://dx.doi.org/10.1007/978-3-319-44839-8.

Pietrusewicz, K., 2019. Metamodelling for design of mechatronic and cyber-physical systems. Appl. Sci. 9 (3), 376–416. http://dx.doi.org/10.3390/app9030376.

Powell, A., Vickers, A., Williams, E., Cooke, B., 1996. A practical strategy for the evaluation of software tools. In: Brinkkemper, S., Lyytinen, K., Welke, R.J. (Eds.), Method Engineering: Principles of Method Construction and Tool Support. Springer US, pp. 165–185. http://dx.doi.org/10.1007/978-0-387-35080-6_11.

Rashid, M., Anwar, M.W., Khan, A.M., 2015. Toward the tools selection in model based system engineering for embedded systems - A systematic literature review. J. Syst. Softw. 106, 150–163. http://dx.doi.org/10.1016/j.jss.2015.04.089.

Roques, P., 2016. MBSE with the ARCADIA method and the capella tool. In: Proc. of the 8th European Congress on Embedded Real Time Software and Systems. HAL, pp. 1–11, URL: https://hal.archives-ouvertes.fr/hal-01258014.

Schuler, R., Kaufmann, U., Adam, A., Binder, B., Breetz, L., DiMaio, M., von Dungern, O., Hooshmand, Y., Muggeo, C., Munker, F., Pfenning, M., Priglinger, S., Pribernow, P., Scholl, A., Weilkiens, T., Woll, R., 2015. 10 Theses about MBSE and PLM - Challenges and Benefits of Model Based Engineering (MBE). Technical Report, Gesellschaft für Systems Engineering e.V., http://dx.doi.org/10.13140/RG.2.2.23703.16806.

Sommerville, I., 2018. Software Engineering, tenth ed. Pearson Studium.

Space & Missile Systems Center, 2004. SMC Systems Engineering Primer & Handbook, third ed. U.S. Air Force.

SysMLtools, 2018. How to define sysml tool evaluation criteria for MBSE. URL: https://sysmltools.com/define-sysml-tool-evaluation-criteria/. Accessed: 2018-12-05.

Villhauer, E., 2016. Why model-based systems engineering reduces costs. URL: https://blog.nomagic.com/why-model-based-systems-engineering-reduces-costs/. Accessed: 2018-11-26.

Weilkiens, T., 2007. Systems Engineering with SysML / UML - Modeling, Analysis, Design. Morgan Kaufmann.

Werneck, V.M.B., de Pádua Albuquerque Oliveira, A., do Prado Leite, J.C.S., 2009. Comparing GORE frameworks: i-star and KAOS. In: Anais do WER09 - Workshop em Engenharia de Requisitos, Valparaíso, Chile, Julho 16-17, 2009. pp. 1–12.

Zeman, K., Jungreitmayr, F., Azad, K., Scheidl, R., Wahl, H., Thomas, B., Haas, R., Hoffelner, J., Boschert, S., Rosen, R., Aschpurwis, C., Wanner, B., 2020. Symbiotic mechatronics: An alternative view on complex systems. In: Proc. of of TMCE 2020. pp. 569–582.

Azad Khandoker: MSc. Azad Khandoker graduated with Master of Science Aeronautical Engineering degree from University of Applied Science Joanneum, Austria in 2016. He worked with Bombardier Aerospace as a Maintenance Planner (2017). He worked with TU Graz as a Research Assistance for light weight design and multi-material joining technology for automotive industry (Mar.2014–Feb. 2015). Since Jul-2018 Azad is employed by JKU Linz, as a Senior Scientist, where he is working for the COMSYS project. He is also pursuing Ph.D. in Technical Science in the same university. His research interest includes MBSE, Systems Engineering, theory building, interdisciplinary product design & development, etc.

Sabine Sint: Dipl.Ing. Sabine Sint (former Wolny) studied Business Informatics at the TU Wien, where she received her master's degree in October 2013. From 2013 to February 2019, she worked as project assistant at the Research Unit of Building Physics with focus on project management and development of software solutions. From 2015 to February 2019, Sabine also worked in the Business Informatics Group at the Institute for Information Systems Engineering. Currently she is working as Ph.D. at the JKU Linz in the CDL-MINT and as project assistant at TU Wien. Her research interests include model driven engineering, reverse engineering, data integration, etc.

Guido Gessel: Guido Gessl, a student of Mechatronics at the JKU Linz, had been employed as a student assistant at the IMDP until June 2021.

Klaus Zeman: o.Univ.-Prof. DI Dr. Klaus Zeman is Head of the Institute for Mechatronic Product Development and Manufacturing at JKU Linz. He earned his doctorate degree in 1984 from TU Wien. He worked with the VOEST-Alpine since 1984 to 1994 and his last position was Deputy Head of the rolling mill technology division. He joined the JKU Linz (1996) as a professor in Mechatronics department. He is also the supervisory board of the LCM and active member of many scientific & engineering societies like WiGeP , MC, VDI, ÖWGP, etc. His research fields are computer-aided product development, machine dynamics modeling, simulation, etc.

Franz Jungreitmayr: DI(FH) Franz Jungreitmayr studied Automotive Engineering at FH Joanneum in Graz, where he got his degree 2011. Currently he is a Ph.D. student and works as university assistant at the Institute for Mechatronic Product Development and Manufacturing at JKU. Additionally, he is working on the COMSYS project.

Helmut Wahl: DI Helmut Wahl works as university assistant at the Institute for Mechatronic Product Development and Manufacturing at JKU. He is doing his Ph.D. and working on the COMSYS project. His research interests are systems engineering, simulation, effective design, etc.

Andreas Wenigwieser: Andreas Wenigwieser graduated as BSc. in Mechatronics at the JKU Linz in 2018. From 2015 to June 2021, he worked as Tutor for Introduction to Mechanical Engineering at the Institute of Mechatronic Design and Production at the JKU Linz. Since 2017 he is employed at the institute as project assistant, working on the COMSYS project. Currently he is also finishing his master's degree in Mechatronics. His master thesis is about the modeling of a conveyor belt system for the gap correction in a parcel flow. His research interests includes MBSE, systems engineering, mechatronic product development, etc.

Roland Kretschmer: Roland Kretschmer has finished his Ph.D. at JKU Linz 2020 under supervision of Prof. Alexander Egyed. He was working as a Senior Scientist at the Institute of Software Systems Engineering until end of 2020.