Chapter 9
Systems Engineering

Alain Biahmou

Abstract Unlike the first cars, which essentially have been mechanical systems, nowadays cars have become very complex mechatronic systems that integrate subsystems created in a synergy between people from different domains such as mechanical engineering, software engineering and electric and electronics (E/E). This fact has increased product complexity in the last decades and therefore the product development complexity. Complexity is multidimensional and consists of product, process, organizational, market as well as use complexity. A methodology for mastering complexity is Systems Engineering, which actually means applying systems thinking to tackle the challenges of creating complex products. The focus of this chapter is providing a deep understanding of systems engineering (SE) as well as a rough recommendation for companies that might be interested in implementing SE. Thus concepts for implementation are proposed. As an entry point, the context of product creation is presented with the challenges that are linked to. The need of appropriate methods is emphasized and the application of SE is motivated. In order to present SE as it is applied in the practice, SE processes are described in detail and the artifacts of the different steps are highlighted. For performing the processes described, SE tools and methods are presented. The important role that the company organization and the project management both play for SE projects as well as SE success factors are highlighted. Additionally, a proposal for an introduction process for SE is elaborated. A selection of functional features that can provide a cutting-edge advantage when practicing SE are presented and discussed. Two case studies are illustrated in order to provide real applications of SE and therefore an additional orientation for SE implementation. The relation between SE and Concurrent Engineering is addressed and some future challenges of SE are identified.

Keywords Systems engineering · Systems thinking · Mechatronics · Complexity management · V model · RFLP approach

A. Biahmou (✉)
EDAG Engineering AG, Reesbergstr. 1, 36039 Fulda, Germany
e-mail: alain.biahmou@edag.de

J. Stjepandić et al. (eds.), *Concurrent Engineering in the 21st Century*,
DOI 10.1007/978-3-319-13776-6_9

9.1 Introduction

The challenges regarding complexity have increased in the automotive industry, in the aerospace industry and in the most relevant areas of engineering during the last decades. The complexity can be observed regarding the product (technical complexity) as well as the processes of its creation (organisational complexity).

Product complexity is characterized by factors such as increasing of functions and components, but also component complexity, when a component integrates more functions and subcomponents. For instance, the number of electric components has increased in cars and therefore, the total length of the electric cables used in today's generation of well-known cars has reached a multiple of their initial values. Furthermore, components such as headlamps have been improved to include more functions, but also more sub-components such as sensors which are connected with the CAN-bus. These very small examples can give a brief impression of the real challenge that is to be tackled in product development nowadays and in particular, in the automotive industry.

Besides, complexity also is enhanced by the dependencies that must be covered during the product life. These dependencies may be existing between the components of the products as well as between these components and external actors such as project members, external documents and so on.

Furthermore, considering and integrating new technologies in existing products also is a source of complexity. A closer look on the consumer market reveals what can happen when a company producing cameras ignores advances in digital technologies, or when a company designing cell phones does not pay attention to the technologies that led to next generation devices, the smartphones.

Process complexity includes among others the methodical and collaborative procedures which are performed for product creation, for instance component integration. Complex products are developed by diverse disciplines—e.g. mechanical engineers, electric/electronic engineers, software engineers—using different processes, methods, tools and especially vocabularies. It is important for all these disciplines to understand the system as well as each other in order to conduct the project to success. This requires a meta-model of the system as basis for the work. Based on the system meta-model, a product model that integrates the partial models can be created. A partial model represents the perspective of an arbitrary discipline on the product model.

In practice, a challenge consists of creating such a product model, identifying explicit and implicit relationships resp. dependencies between the existing different partial models (see Fig. 9.1). Generating the partial models and updating the product model in the right sequence after modifications have occurred, is an additional challenge. In this case, investigating change propagations and taking them into account may be essential for being able to generate the right models.

Experience has shown that communication in projects as well as accessing data of other disciplines are important tasks that need more efficiency. This is due to the fact that the disciplines that own the data commonly use different data formats as

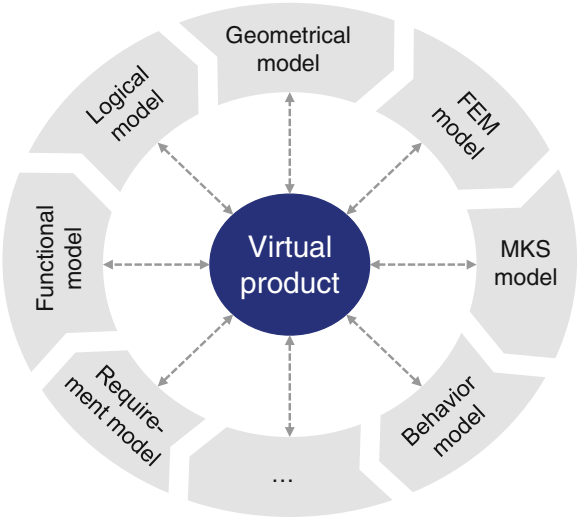

Fig. 9.1 Different views of a product

well as data management tools, for which access is not granted for data consumer (s). A PLM study has confirmed communication and data search as a very time consuming activity during product development (see Fig. 9.2) [1].

Another factor in car development is the extended enterprise. A high amount of companies of different types (see Fig. 18.2) are involved in the development of a vehicle. These companies generally are located in different countries and continents, since most of OEMs are global players, that is, modules or derivatives of vehicles often are developed all over the world (see Chap. 7). Thus, consulting services and engineering services providers as well as module suppliers that are

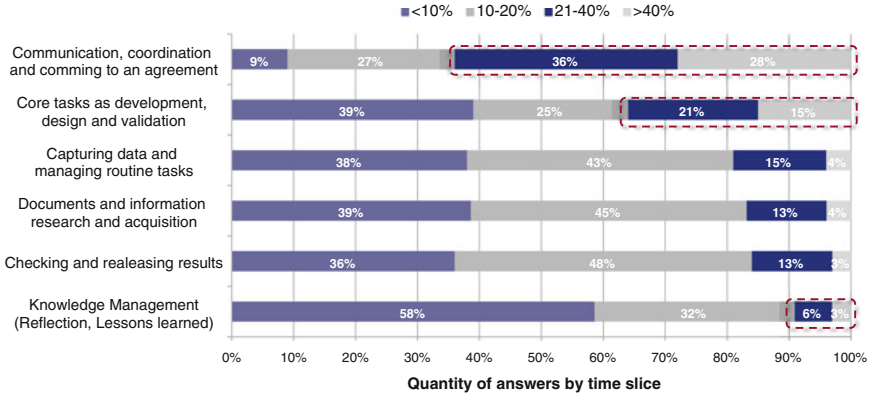

Fig. 9.2 Complexity in product development/time consuming tasks [1]

involved in the vehicle development process need product data in order to perform diverse tasks, e.g. Digital Mock-Up analyses [2]. These actors need also to actively participate to the product creation by interacting with project teams sometimes using different processes, methods and tools.

The interaction of the companies involved in the development of cars enforces therefore an exchange of data (e.g. CAD data) between different parties. The data exchange includes not only a file transfer between the OEMs (original equipment manufacturers) and the first tier suppliers, but also the suppliers with their own subcontractors. Figure 18.2 shows a sample data flow within a network of companies that are collaborating for a project. Managing a project which is conducted in such a widespread network leads to answering questions about data security, once more enhancing the technical and organizational complexity [3].

Apart from engineering issues, Winzer presents globalization as an additional complexity factor, since it leads to an explosion of the number of stakeholders, the number of laws and homologation as well as country specific customer interests to be considered. Globalization can lead to a higher number of suppliers [4]. Mass customization also is a further factor which increases complexity as it leads to an increasing of individual functions [4]. Mass customization helps providing individual products to customers (see Chap. 14). This yields the creation of a higher number of design alternatives, that complicates not only the product creation in term of simulations, data management and configuration management, but also the after-sale phase when it comes to deliver services, e.g. providing spare parts. A further complexity factor is the miniaturization, due to the fact that a general trend is towards smaller and compacts products [4]. The miniaturization of systems yields a new adjustment of system components, but may lead also to adopting new and complex manufacturing processes, e.g. micro machining. To sum up, aspects of complexity are not limited to market, product complexity, organizational complexity and process complexity (see Fig. 9.3) [5].

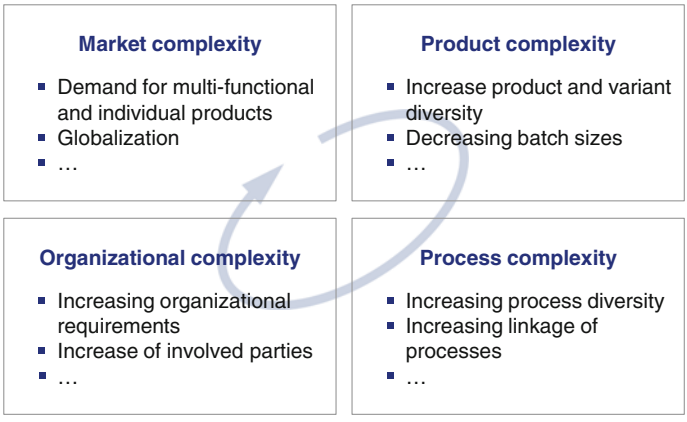

Fig. 9.3 Types of complexity [5]

The description of the complexity which has been presented above emphasizes the need of methods for mastering complexity in product development. The content of this chapter follows this premise. In Sect. 9.2 the motivation for systems engineering (SE) is highlighted. Section 9.3 describes SE in practical use (tools and methods, organization and project management, architecture). Section 9.4 draws a concept for Functional Blocks to support SE. In Sect. 9.5 introduction of SE in a company is discussed. Section 9.6 shows case studies, followed by discussion in Sect. 9.7.

9.2 Motivation for Systems Engineering

A proposal for complexity management has been presented by Schuh and Schwenk [6] who proposed a reduction of variants, but this may be problematic since the carmaker should decide in that case not to fulfil certain customer requirements. This could be interpreted as a lack of flexibility, especially if competitors provide appropriate products.

An established methodology for mastering complexity is SE, which actually means applying systems thinking to tackle the challenges of product creation. Thus what do system and systems thinking mean? Many definitions of the term system already have been provided by research works. A system can be apprehended as a set of components which are linked by relations, forming a whole. Hitchins defines a system as an open set of complementary, interacting parts with properties, capabilities, and behaviors emerging both from the parts and from their interactions [7]. Further authors make a difference between systems, systems of systems (which are built by components that are large-scale systems), mega systems and intelligence-based systems which are able to comprehend, understand and profit from experience in order to adapt to changes of their environment [8]. A system is made up by the complex networking of resources such as manpower, equipment, facility, material, software, hardware and so on. Resources are to be considered as subsystems which interact with each other within or beyond the surrounding system. A system is characterized by inputs, outputs, internal processes and mechanisms as well as constraints [9].

Systems thinking is a pre-requisite for applying SE since the multidisciplinary teams involved in product creation have to understand the system-of-interest as a whole. Collaborative systems thinking is an emergent behaviour of teams resulting from the interactions of team members and utilising a variety of thinking styles, design processes, tools, and communication media to consider systems attributes, interrelationships, context and dynamics towards executing systems design [10]. System thinking is not new in the product creation area. Design methodologies that have been presented in the past that can be applied for different areas beyond mechanical engineering have highlighted systems thinking [11–13].

Lindemann also has presented an approach to complexity management, which highlights the connectivity of elements—product components, people, documents-involved in product design. Further methods for structural complexity management mentioned are systems dynamics, operational research etc. [5]. Although diverse methods address the complexity management, that topic still remains a main challenge in the industry: this fact is a matter of evidence when thinking about some car makers who are recalling their cars in order to fix failures [14].

According to the International Council on Systems Engineering (INCOSE), "SE is an interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem. It integrates all the disciplines and specialty groups into a team effort forming a structured development process that proceeds from concept over production to operation. SE considers both the business and the technical needs of all customers with the goal of providing a quality product that meets the user needs" [15].

SE focusses on the lifecycle of technical systems, that lasts from the first idea to the recycling. It apprehends the system to be developed as a whole artifact, which interacts with its environment. Moreover, SE takes the relations between the system and its components into account and its main purpose is coordinating the disciplines involved in product development [4]. This apprehension is very close to the approach of Hitchins, that emphasizes the fact that a system is active, interactive and adaptable to its environment. Further key aspects of Hitchins' approach are the human factor on the system, its dynamic context, its synthesis, holistic character as well as its analogy to an organism [16].

SE includes management and engineering and considers that a system is more than the sum of its elements. It is a continuous iterative process which includes multidisciplinary teams and is applied throughout the product lifecycle [17]. Although SE has developed itself very quickly in the last decades, there are approaches to improve the traditional methods and, therefore, the quality of systems. Tolk et al. [8] have characterized intelligence-based systems while emphasizing the role of semantics, simulation and agents. Works presenting computational intelligence in the meaning of automated or semi-automated processes for SE have been presented in the past [18–21].

Intelligence-based SE can be applied for developing different complex products such as nuclear plants, airplanes, cars, space shuttles, machines. It helps conducting simulation already in early development phases in order to reduce costs and improve quality. It can be applied also to automatically generate hardware or software specification. Especially in the automotive sector, it has contributed to tackle an important number of challenges going from monitoring driver inattention, enhancing pedestrian safety to the control of mobility systems [22–27].

SE also has emerged from a general approach to a more and more specific approach for a couple of domains such as automotive, aerospace, medical and manufacturing. A further SE area that is being established is Product-Service-SE [28, 29]. The motivation of this evolution is explained by specific branches of SE

being more efficient to solve domain relevant problems. While the most known publications about SE deal with the development of a specific product (e.g. a airplane, car) and therefore, are focused on technical aspects, the future approaches will be broader.

Some perspectives may be the application of SE in order to determine how engineering systems can interact with other systems taking social, economic and environmental factors into account. SE could be important for tackling challenges in the areas of critical infrastructure, health care, energy, environment, information security and other global issues [30]. Upcoming challenges of SE will be discussed in Sect. 9.7.

To sum up, complexity management as well as lacks in today's product development methods (e.g. requirement engineering) make SE necessary. SE targets a cost reduction by implementing methods to ensure a good quality design, a traceability between components and processes, that can help getting a better understanding of the system and therefore managing the systems development. In order to apply the SE approach, systems thinking capability, tools as well as methods are necessary. It is crucial to define system boundaries, interfaces as well as inputs and outputs of the system.

9.3 Systems Engineering in Practice

The main objective during systems development is the achievement of stakeholders concerns, who may be customers, owners, vendors or any person being related to that system. This is done by designing and integrating methods and models within the system, but also with other systems. Integrating means to network the break down structure of the different sub-systems, components and processes involved. Based on this, verification and validation which are two major tasks are to be performed in order to realize the quality standards [9].

Figure 9.4 shows two simplified schemas of the SE process, which have been proposed by Dikerson and Mavris (right) and Pineda and Smith (left) [31, 32]. This shows that even in the SE community, standardization remains a challenge. In an attempt to reach a common understanding, Ryschkewitsch et al. [33] has presented a set of definitions related to SE and its actors.

Many modified schemas already have been presented based on the V model in different variations. It is about an iterative and holistic process. For sake of brevity, the different phases will not be detailed. Practical recommendation to requirement engineering and management is provided by appropriate literature [34, 35], functional analysis in SE addresses the transformations to be performed in order to obtained required systems outputs from available inputs [36]. Elements of functional analysis have been presented among others by Buede [36] and systems integration [37, 38], verification and validation as well as testing also have been addressed by a multitude of scientific works [39].

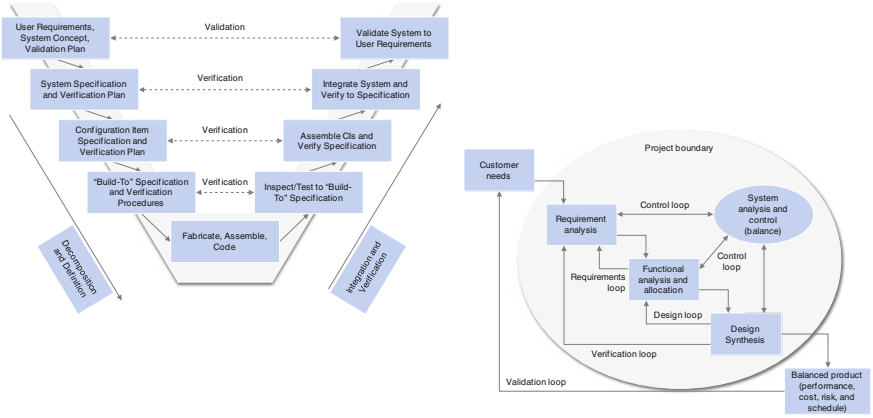

Fig. 9.4 Systems engineering process [31, 32]

From a SE point of view, requirement tracing is a premise, that means it should be possible to trace the impact of each requirement on the functional, logical, physical model as well as on verification and validation. Therefore, analyses can be performed in order to investigate to which degree a requirement has been fulfilled. Requirements engineering includes the generation, formalization, decomposition, analysis and management of product and service requirements with the objective to verify and validate them (see Chap. 5). Decomposing requirements is important since they often are formulated by persons (e.g. customers) who are not familiar with requirements engineering.

Based on requirements engineering, system functions are derived in order to create a functional model. Functional analysis consists of formalizing and decomposing system functions that are to be realized later by the design [32]. The next step consists of identifying functions that belong to the same sub-system, it is a phase during which subsystems are identified from the designed functions. Therefore, the output of this activities are groups of functions. Moreover, it is important to notice that the identified sub-systems can complement the use-cases that have been defined at project start, since the initial use cases are merely focused on fulfilling stake holder requirements. However, supporting functions and use-cases are necessary to get the whole system running.

Thus, test cases are to be defined in order to identify further inputs for the creation of the whole logical system, that enables all system components to operate together logically. Based on the groups of functions resp. sub-systems identified, a logical model of the whole system is created and behavioral models are created or generated and attached to its components. The simulation of the logical system is the basis for the creation of the overall system concept, that provides inputs for the physical design of the system in the different disciplines involved. Therefore, the disciplines can perform a concurrent development process based on system-oriented-concepts as well as a common model (see Sect. 9.3.1).

It is important to remark upon the requirements loop between the Requirements analysis and Functional analysis as well as the Design loop between Functional analysis and Design synthesis. Both ensure that the requirements analysis, the functional analysis as well as the design are continuously examined and updated. A verification loop insures that the tasks are done right, whereby a validation loop has the objective to check that the right things have been done, that is, an end product in which performance, costs, risks and scheduled characteristics are successfully balanced to satisfy customer expectations. The verification includes semantic, syntactic and plausibility checks.

The validation has to check how far the systems model corresponds to the real world. The tasks regarding systems analysis and control consist of overseeing and controlling all phases and activities of the System Engineering process [32]. Validation is crucial to ensure that the main goal has been achieved. Kolonay has introduced the concept of physics-based models to validate the conclusions made for concepts and system specification [40]. A further interesting approach consists in defining value ranges to reduce eventual gaps between the reference values and the actual values of component interfaces as well as tolerances between energy, material and signal flow [41].

In other articles, the so called RFLP approach, that stands for Requirements, Functional, Logical and Physical modeling, is presented as a procedural model for Systems engineering. In fact, the RFLP approach is approximately equivalent to the descending side of the V model, which it includes. It highlights the fact of creating a functional model out of requirements engineering, whereby the main function is decomposed into sub-functions that can spread over many hierarchical levels.

The logical model describes interdependencies between the system components and therefore enables the right assembly of components to a final construct that can be simulated resp. verified. In short, the logical model connects the behavior models of the single sub-systems and system components, so they can interact with each other.

The physical model is represented by the representation (e.g. geometrical) of sub-systems and system components in the involved disciplines. Although the logical model is to be created before the physical modeling, the practice shows that behavior models can be generated from a physical model (e.g. geometrical model) in order to be attached to its corresponding component in the logical model. This approach can occur automatically when an integrated environment is used for SE (see Chap. 27). Otherwise, single behavior models can be create with own tools and assembled within a logical models, assuming that the relevant interfaces are implemented by the software that is used to design the logical model.

Biahmou et al. [42] have developed an approach for deriving behavior models from 3D models in MATLAB/SimMechanics that have been created with CATIA V5. After updating the geometrical model, the behavioral model is to be updated by the user. In order to free the user from this task and achieve a structured and right synchronization of changes in partial models of a system, the application of ontologies has been addressed to identify the update sequence of models [43].

Why has the RFLP-approach been necessary although the V model has been existing? Even though the V model has been provided for the development of mechatronics products involving mechanical, software and electrical/electronical engineering, it does not explicitly provide details on creating models as well as model boundaries in its initial form. However, an amelioration of Ott emphasizes systems thinking and traceability [44].

Furthermore, the V Model is compatible with well-known design methodologies such as the approach described by Pahl and Beitz, which recommend the identification of the main function and its fragmentation on many levels in less complex and manageable functions. Since the design is an iterative task, the many steps of the V Model or the whole V Model can be performed many times [11, 45].

The general processes of SE can be adopted by some business branches, companies or institutions. NASA has defined a SE process based on tasks integration and control as well as interfaces management, where both are to be performed during all the phases of implementation of a space mission. Other activities consisting of requirements engineering, system analysis, design and configuration definition and finally the verification, can be achieved sequentially [46].

At this stage, questions of importance are, what is needed except to the processes, to practice systems engineering? Which tools and methods are relevant, which organization structure and project management form are needed, how can SE be introduced in a company and which services should provide a platform for applying systems engineering?

9.3.1 Tools and Methods of Systems Engineering

The different disciplines that are involved in SE (e.g. mechanical engineering, software, Electronic/Electric) have different vocabularies and use different tools as mentioned above. Therefore, they need a common language as well as a common reference, that is, a common product model to tackle the aforementioned challenge of communication and coordination. The systems modelling language (SysML) is the most widespread solution for these issues. SysML is derived from unified modeling language (UML) and the relevant sub-set of UML (UML4SyML) has been enhanced in order to take the characteristics of general-purpose systems into account. SysML can be used for representing systems, which may include combinations of hardware, software, data, people, facilities, and natural objects [47, 48]. Models created with SysML can be exchanged using the XML metadata interchange (XMI) format, enabling also developers who use different tools to exchange model information [48]. Due to its organizational concepts of package, models and views, SyML enables the achievement of a paradigm shift in the modeling of complex systems of systems, from a document-based approach to a model-based approach.

The document-based SE Approach consists of keeping all relevant information in documents of different disciplines, depicting them in a hierarchical tree and defining how the system is to be used. One of the limitations of document-based SE

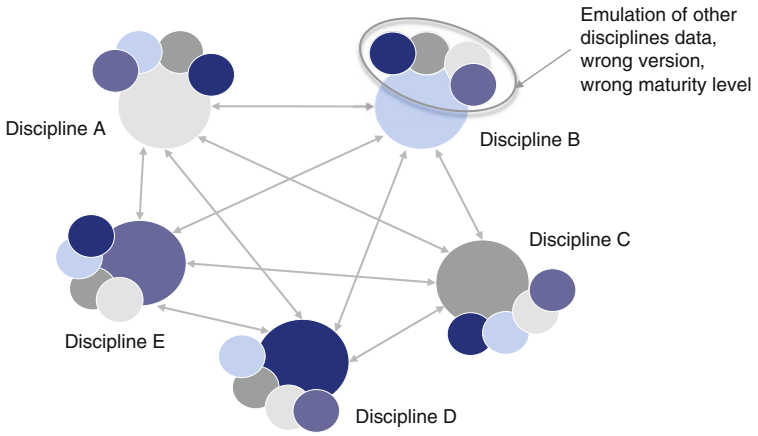

Fig. 9.5 Model creation in disciplines

is the difficulty to realize traceability and, therefore, to ensure the completeness of requirements. The link between requirements, design, engineering analysis and test information is often missing, leading, therefore, to a limited capability to manage change impacts.

Furthermore, document-based SE can lead to a significant lack of information across the disciplines involved in product development and to the use of inconsistent models, especially when each discipline is trying to create the missing model resp. information (see Fig. 9.5).

In contrary, model-based systems engineering (MBSE) is characterized by using a common model of the targeted system (see Fig. 9.6) to perform the tasks described on Fig. 27.3. That model may have references to distributed partial models. Alternatively, it can be associated with partial models that are used to represent the different artifacts (e.g. requirement model, functional model, logical model, geometrical model) and their relationships, in the same repository [49].

An important factor to be considered in the practice of MBSE is the representation of the variability, that for instance may be functional or physical. Since many details are not know or available in early development stages, it is important to agree on the right granularity of the common system model to support capturing of all relevant features as well as their relationships and therefore ensure traceability very early. Dumitrescu et al. [50] addressed this with a concept for introducing variability management on an intermediary level between vehicle features and component specifications.

Particular aspects of the global resp. common system model are called views or partial models, they can be generated from the common model by synchronizing the common parameters.

This means that a parameter model also is needed, in which implicit as well as explicit relationships of all involved partial models are represented. In the practice, a multidisciplinary team of well experienced specialists (e.g. designers) can define

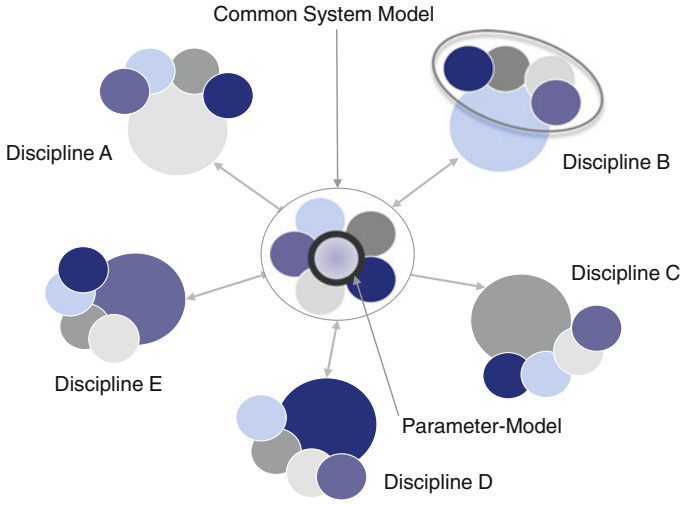

Fig. 9.6 Common system model

the parameter model. Even though a company is not applying systems engineering, each discipline generally use to work according to input-processing-output-principle (IPO). That is, a discipline receives data from another one, processes it to reach its own objectives, for instance to complete the geometrical design and at the end, provides their result—output—to other disciplines that are considered as data consumer in this case.

Further relevant methods of SE regard quality analysis and decision making support. Among others, conventional approaches such as Hazard Analysis and failure mode and effects analysis (FMEA), total quality management (TQM), Six Sigma, ISO/QS900x as well as software development quality systems based on capability maturity model integration (CMMI) or software process improvement and capability determination (SPICE), are used for quality management [51].

Equally, a multitude of approaches for technical decision making during early development phases are available [52]. However, the experience has shown that the partial models mostly are existing in proprietary formats with different modelling paradigms and therefore cannot be easily assembled to a global system model.

In order to apply SE without restrictions, standards are necessary for instance for model exchange or for enabling reuse or integration of components in third parties architectures. Therefore, the automotive open system architecture (AUTOSAR) specification has been elaborated for appliance by some companies that develop complex products (e.g. car makers, developers of engine control units, car software, development tools and microcontrollers) [53].

Hence the functional mock-up interface (FMI) has been developed as standardized interface to support model exchange between different tools for systems simulation, co-simulation, applications as well as the integration of models and their

associated data into product lifecycle management (PLM). Software tools which have implemented the FMI standard can create an FMI-conform model library, called functional mock-up unit (FMU) (see Sect. 13.5) [54]. A further methodological approach for complexity management, especially for reducing the risk of failures of safety relevant components is the norm ISO 26262, which describes a procedure model to be applied [55].

Although more methods might be mentioned in such a chapter, for sake of brevity, only the SysML as well as interfaces are emphasized in this work. Reason for this orientation are the importance of the right modeling of the whole system in terms of representing the reality and impact of the communication in the project, as both represent success factors for SE. A valuable taxonomy of SE process standards as well as methods which describe how to achieve the tasks described in the SE process is given [48]. The taxonomy also includes architecture frameworks, systems modeling methods, modeling and simulation standards as well as interchange and meta modeling standards.

The tools necessary for SE have to support the realization of a major objective of product development, which is reducing cost through the reduction of physical Mock-ups. Visualization techniques and simulations of 3D models help for creating virtual mock-ups [56].

SE tools can be classified on two subjective levels: the early phase of product development corresponding to the descending branch of the V model, with the horizontal line of the V model corresponding to the detail design resp. discipline specific design. The second category of tools can be associated to the ascending phase of the V model. The core tools belonging to the first category are tools for creating a common system model, that integrates all partial models and their relationships (e.g. tools for SysML modeling). Hence tools are necessary for creating requirements models, elaborating functional models and common parameter models as well. A tool to model and simulate logical systems also is required.

Moreover, a platform or a tool has to ensure the traceability from the requirements over the specific design to the product recycling. That platform or tool should be equipped with analysis capabilities in order to provide quick answers to queries of a project leader. For instance, such an analysis tool should be able to show whether all requirements are fulfilled or not. Therefore, a project leader could check the impact of changes on requirements. Queries of project leaders could include risk analysis as well as cost calculation after design changes.

The discipline-specific tools that are used in the horizontal line of the V model already are part of today's tools landscape in the companies and can be considered as provided. However, the communication of these stand-alone tools is to be ensured though the use or implementation of appropriate connectors and enterprise architecture. The most aforementioned functions are available in tools that are on the market as stand-alone, but there are integrative platforms that propose environments that comprise many of the relevant functionalities. They are often equipped with interfaces that enable the integration of selected third party tools. The decision, whether an integrated environment or many stand-alone tools are to be connected depends on the company's SE strategy and the importance of the

flexibility. Viewing from this perspective, SE can be applied without using an integrative development environment. A company can implement a specific architecture resp. environment with best-of-class tools of each category. For the ascending phase of the V model, manifold architectures, vehicle simulation and testing tools are commonly used for SE [26].

9.3.2 *Organisation Structure and Project Management for SE*

SE is applied by many companies, but the question is to know to which extent, as well as the impact of the existing organization structure on the practice of SE. The organization structures of carmakers have been component-oriented for a long time. Thus, the technical departments mostly spreads across the departments chassis, powertrain, interieur/exterieur, electric/electronics etc. [51]. Additionally to the component oriented structure, the experience has shown that a program resp. product oriented structure is implemented. Thus, a matrix organization is established in most cases. Project teams are made up of specialists from the different components departments (e.g. powertrain).

The success of SE project organized in that manner generally is strong dependent on the experience as well as the personality of the project leader. Success in this case is not limited to the question whether the result obtained by the project group is acceptable or not. The question is to know what would have been possible due to the skills of project members. Thus, the focal point is efficiency of the project team, adherence to delivery dates, product quality as well as stakeholder satisfaction. In order to take benefit of SE methods, it can be of advantage to move the location of the systems engineer according to Fig. 9.7. Doing so can help taking the focus away from the historical component-based development to the systems-based development.

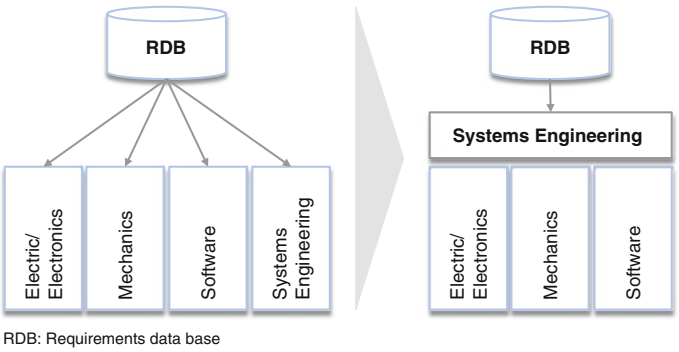

Fig. 9.7 Project organization for systems engineering

Applying SE requires therefore appropriate organization structures and corporate cultures. SE is not to be focused only on the product to be developed, instead the whole product creation landscape and processes including the manufacturing and the supply chain also are to be considered. This requirement is especially important when it is taken into account that many projects in aerospace or defense have been characterized by delays in delivering as well as increasing of initial unit costs. Sanders and Klein proposed an industrial V model that mirrors the conventional product V model in order to integrate manufacturing and supply chain considerations into the SE process [57].

It can be expected that similar approaches that consider further disciplines could improve collaboration and efficiency. For instance the development of Product-Service-Systems could be coupled to product SE on the same manner. Van Ruijven addresses the availability of a common framework that should include an ontology based on information models to support systems engineering. That framework should improve the communication within the project by defining SE processes based on the top level of an ontology that the author proposes. Additionally, information models of the breakdown of the process side of a system, the breakdown of the physical side of a system as well as of the work breakdown of a system are addressed to support that objective [58].

These proposals correspond partly to the need of a meta-model, that already has been mentioned by Winzer [4], to bring all involved parties to understand themselves without any ambiguities and emphasizes the importance of MBSE.

Moreover, merely the clear elaboration of SE processes surely will not be sufficient. The project management will have to emphasize the systems thinking at all stages of the project and should be supported by business services that orchestrate workflows in background and that process data from the relevant disciplines in order to provide up-to-date, integrative view data to the project management, but also to the disciplines. Such an overview is of great importance for project management in order to support decision making.

From a general point of view, Boehm et al. [59] have identified four key principles that are necessary to apply SE successfully: Stakeholder Value-Based System Definition and Evolution, Incremental Commitment and Accountability, Concurrent Multidiscipline System Definition and Development, and Evidence-Based and Risk-based Decision making. In order to emphasize the suitability of these principles, they are compared with lean SE principles as well as Hitchins' principles for successful SE.

As aforementioned, one could wonder why spectacular failures leading often to recall campaigns [14] are still been made in projects that have been driven by SE. One of the causes has been identified by Boy and McGovern as the lack of a human-centered focus in SE, that instead is claimed to be technology-centered and finance-driven. Thus, they propose to operate changes on rigid SE processes in order to provide the engineers with more flexibility, by applying Human-Centered-Design principles to achieve a Human-Systems Integration [60]. This calls for socio-technical leadership as an important skill for SE project leaders.

The idea of improving organizational aspects also has been identified by Winner [51], who pointed out both the staff qualifying for effective and efficient SE as well as subsequent modification of the management culture as part of challenges for SE in the automotive and supply industry.

Apart from the organizational structure of a company that can impact the success of systems engineering, the tools landscape is of great importance.

9.3.3 Architecture for Systems Engineering

Enterprises that decide to apply SE principles have to tackle the challenge of elaborating the architecture to be implemented in order to perform the processes mentioned above. An interesting approach for a SE platform consists of a tool federation instead of integration, whereby necessary information is shared between loosely coupled models (see Sect. 9.4).

A similar approach has been addressed by Bartelt et al. who proposed a software architecture resp. middleware for the configuration of simulation scenarios. The simulation scenarios can be integrated in simulation modules. The simulators communicate through the SimBus, that is an architecture on the basis of a CORBA-like platform. SimBus connectors allow the communication of applications [61]. The solution is specialized on simulation purposes, however SE addresses a broader spectrum.

A system framework for conducting SE is based on a complex architecture, since it is in turn a system of systems. Therefore, elaborating them requires the consideration of: Autonomy, Complexity, Diversity, Integration strategy, Data architecture and System protection [62]. Among other critical factors, robustness and alignment to business processes and technology are to be taken into account.

Explaining all processes and principles of architecture design would go beyond the scope of this chapter. However, apart from ISO/IEC 42010 (2011) [63] that provides standards for architecture description for systems and software, there are well-known commercial, defense and government frameworks [64] that can be used as guidance when elaborating enterprise architecture:

- the Zachman Framework for Enterprise Architecture [65],
- the Department of Defense Architecture Framework [65],
- the Federal Enterprise Architecture Framework [65],
- the Treasury Enterprise Architecture Framework [65],
- the Open Group Architectural Framework [65], and
- the NATO architecture Framework [66].
- An assessment of the five first frameworks by Urbaczewski and Mrdalj [65].

9.3.4 Summary Evaluation

Even though SysML at this moment is the language that is established to create common understanding models for SE, this term is still ambiguous among experts (e.g. SE from academia and industry). This calls for the necessity of harmonization for instance regarding basic terms such as function and behavior [67, 68].

Many companies still take the risk of considering the different phases of complex products creation as sequential tasks. They assume that the periodical information exchange between the disciplines is sufficient. This can lead in most cases to inconsistencies. Instead, a concurrent engineering approach is to be followed, ensuring a traceability between all the models and phases involved [69].

Moreover, many disciplines still provide other collaborating disciplines with data without an explicit and documented procedure. Preparing and providing data to others is generally a task on top of the daily job that consists of performing a progress in the original discipline, e.g. the design. Thus, requests from another discipline for receiving data can take a long time before being processed. It can cause the tendency to use obsolete data or to try creating data for own analyses.

9.4 Concept for Functional Blocks to Support SE

To bridge the disadvantages mentioned above, an alternative concept can be implemented, in which data and services sharing as well as the domain authority are highlighted. Thus, each discipline involved in product creation can access data of another discipline through sharing mechanisms, whereby re-creation of data is useless. A proposal of functional features to support this way of working is presented for companies as guidance for enabling Systems Engineering.

9.4.1 Selected Requirements

The three most important problems originating from collaborative work have been identified by Königs et al. [70] as follows: inconsistent, hard-to-retrieve or outdated data across the engineering departments, low transparency about changes and decisions due to non-existing or non-available information and low transparency about the impacts of changes due to missing documentation of dependencies. This traceability issue also has been addressed by many works [21].

Further requirements for a successful SE architecture are shown in Fig. 9.8 from an industrial study [1]. It reveals among others, that the extended enterprise as well as customers are to be taken into account and that the visualization of changes as well as using common data format are needed [71]. Thus, innovative techniques that include knowledge management can be applied to optimize the collaboration with the extended enterprise [72].

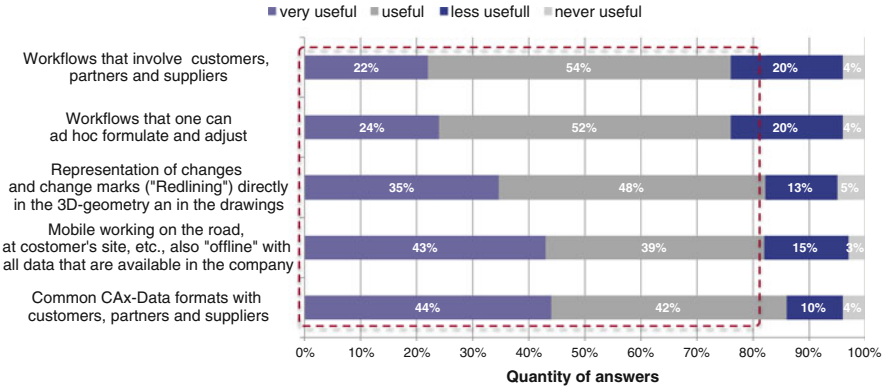

Fig. 9.8 Requirements for a successful development [1]

In fact, suitable interfaces are necessary for software communication, since the disciplines involved use different applications. Besides, neutral data formats such as STEP AP 214 (ISO 10303) are required due to the fact that the different software components involved often create proprietary file formats. A neutral file format can ease information exchange between many applications, because it helps avoiding bidirectional conversions.

Data exchange however, is not sufficient for an effective Systems Engineering. One of the challenges consists of applying data sharing; a use case could be a control system engineer generating a geometrical model from the global product in the suitable level of maturity in order to run a distributed simulation, in which both the geometrical modeler and the control system modeller are communicating.

Moreover, a SE platform should implement the most widespread software interfaces that are necessary to integrate further, third party tools.

9.4.2 Concept

A real prerequisite for SE is the alignment of methods and models of the disciplines involved in product development. Investigations that are performed by an arbitrary discipline do not need only a data set of another discipline as input, instead additional meta-data such as the maturity level are necessary.

Thus, a basic methodological work to be performed consists in gathering the relations between the disciplines models and to create additional stage gates that will be used to characterized data needed by other disciplines. Doing so enables the definition of fix or ad hoc workflows, since the stage gates can be used as attribute to identified the data to be retrieved. Furthermore, consistency also can be assured by synchronizing partial models of corresponding maturity levels.

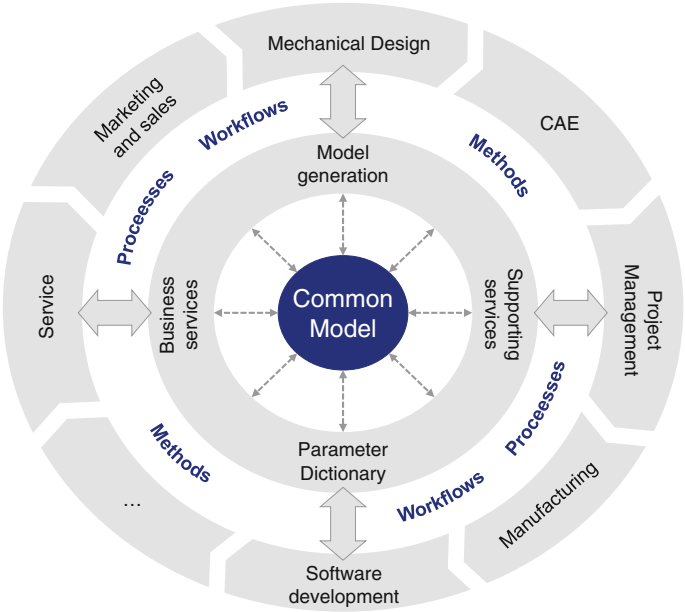

Fig. 9.9 Types of services needed for systems engineering

In order to fulfil the requirements mentioned above, a service-oriented architecture can be implemented with the functionality presented on Fig. 9.9. Thus, the disciplines can apply collaborative processes and methods in order to optimize the way of working according to the global product and not to a single aspect.

Model generation services can be used to generate an up-to-date, contextual partial model. They can be crucial when it comes to configure models for studies, especially when design of experiments are to be realized. A very simple example consists of a suitable geometrical model being automatically generated from the common system model in order to perform a finite element analysis (FEA), whereby the relevant specific attributes such as the level of maturity or the release status would have been taken into account.

Since an SE platform is supposed to be opened to external enterprises, encapsulating of the company's know how is an important criteria. Business services are to be identified, supported, and aligned with the products to be created as well as the processes. Thus, the services shall be defined on the basis of formulated use cases. The business services shall provide specific functions that are based on a service-oriented architecture, whereby many systems that are transparent for the user can be triggered in the background.

An example of business services is realized by lifecycle management services that shall provide all the generic functions of a product data management system (PDM) such as enabling configuration management, versioning and change management. Lifecycle management services shall ensure not only the update of the

common model in the correct sequence, but also its integrity. Thus, modified partial models shall be synchronized with the common model only when the modifications have been performed by specialists.

However, the non-specialists who are using and might modify an arbitrary partial model for studies shall be enabled to store and share the results of these studies as well as to record their decisions and rationales. To ensure the quality of studies results, each discipline will have to define a range of admitted models as well as attributes modifications, which a data consumer is allowed to perform.

Additional business services may be among others workflow services, right management services, intellectual property protection services and assessment services. The latter can help to examine important criteria of the product to be created, for instance cost-effectiveness or even fuel consumption of a car. Assessment services can, therefore, support the project manager in decision making.

A parameter dictionary is also required to represent the relationships between the different partial models as well as their explanations. Such a dictionary should include a parameter repository based on the parameter model described in Sect. 9.3.1. Depending on the implementation, the parameter model can be indispensable for updating the different partial models after a change has occurred in one model. The explanations provided by such a dictionary can help improving the understanding of terms in the different disciplines involved. For this purpose, the team in charge of defining the work methodology shall provide an ontology as a formal representation of a set of concepts to create a common meta model for the relevant domain, or to adopt an existing meta model [58]. The meta-model shall help all disciplines involved to have a common language and a common reference.

9.5 Introduction of SE in a Company

The introduction of SE in a company can be organized in many phases depending among others on the company's internal culture and organization. Figure 9.10 depicts a recommendation based on own experiences.

In the preparation phase, the analysis consists in making a self-assessment in order to document what are the strengths and the weak points of the actual way of working. The real work processes are to be considered instead of an eventual process description that might not be applied in daily work.

Based on the results of the process analysis, a SE vision resp. target can be formulated. Assuming that the SE vision is to be realized, the SE analysis phase can start. In this phase, SE preliminary processes are to be performed. This consists in organizational tasks, e.g. installing a project steering committee and providing a broader circle of relevant employees with general information based on preliminary analysis results and the consequential need for introducing SE.

The steering committee is mainly in charge of defining a SE strategy to reach the vision defined in the preparation phase. The strategy has to provide a clear path to be followed, e.g. defining how the extended enterprise is to be taken into account,

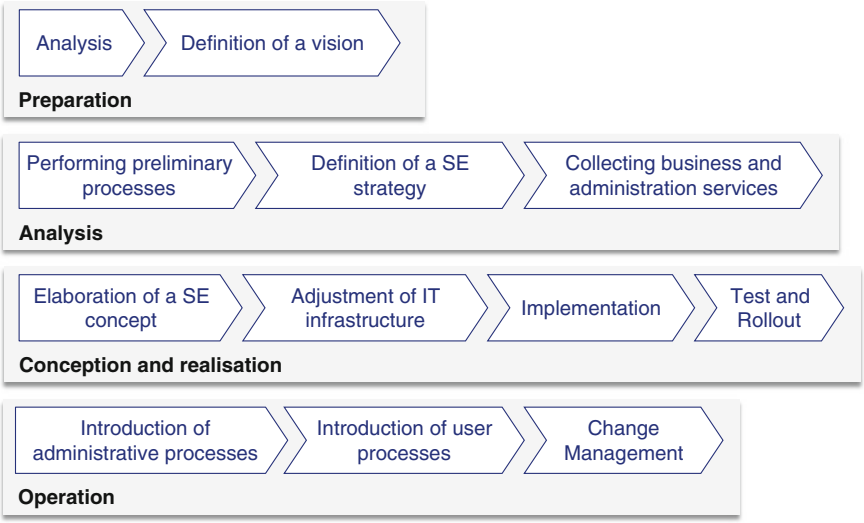

Fig. 9.10 Recommended phases for SE introduction

whether internal workflows and services are to be made available for extended enterprise. Defining the eventual SE levels to be realized is a strategic decision; SE could be realized first to an intermediary level with less functionality. Afterwards, a second or later level with more or full functionality could be implemented based on the experience gained at first levels. Strategic decisions in this context also consist of defining e.g. which types of systems to be used: opened systems, commercial-off-the-shelf-tools (COTS), integrated environments that provide the most functionality needed or taking best-of-class tools and connecting them. Outsourcing strategy and make-or-buy-strategy also shall be considered.

A further task that could be assigned to the analysis phase is the collection of use cases from the specific disciplines involved in the product lifecycle and complementing them with administration use cases.

During the concept and realization phase, requirements are derived from the collected use-cases and business as well as administration services are formulated. A functional architecture of the SE environment is to be defined, in which supporting services are considered. The methods of the different disciplines involved are to be aligned in order to take a higher benefit of SE. Thus, the concept considers not only the necessary software, but also the methodological way of working.

Depending on the SE architecture that has been defined, adjustment of the IT architecture might be necessary, e.g. additional leased lines could be necessary to guarantee some business services, especially when it comes to providing the extended enterprise with internal workflows. Further tasks after the concept and realization phase are the SE implementation and the test and rollout of the solution.

In the operation phase of the system administrative processes are introduced, system settings in tools can be adjusted and necessary accounts can be created in

order to enable user processes. Latter consist in using the business services that are implemented in order to perform specialist tasks. The change management is a continuous process that takes into account the fact that the SE environment can be improved based on change requests that could be introduced by stakeholders.

9.6 Case Studies

The practical value of SE is illustrated with the following two use cases.

9.6.1 SE in Aerospace—Investigating Force Fighting on an Aileron

Vuillemin et al. describe an application of model based SE for a force fighting issue in the aerospace domain. The challenge to be tackled consists of investigating the system behaviour when two forces acting in the same direction or in opposite directions are applied on a surface. This can be caused among others by errors such as signal conversion errors or by adjustment tolerances [73]. Figure 9.11 depicts a schematic presentation of the system. In order to solve the issue aforementioned, the RFLP approach has been applied.

Thus, the systems requirements have been formulated by the authors in textual form. The next step has been the functional analysis. The mission of the system as well as the functions and corresponding I/O interfaces have been determined and the connections of functions as well as their sequences have been modeled [73].

A logical architecture of the aileron has been created as a block diagram on which all relevant components of the system are represented. The components of

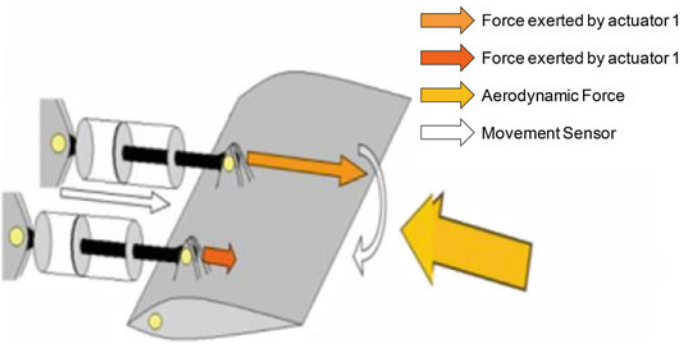

Fig. 9.11 Two forces acting on a surface [73]

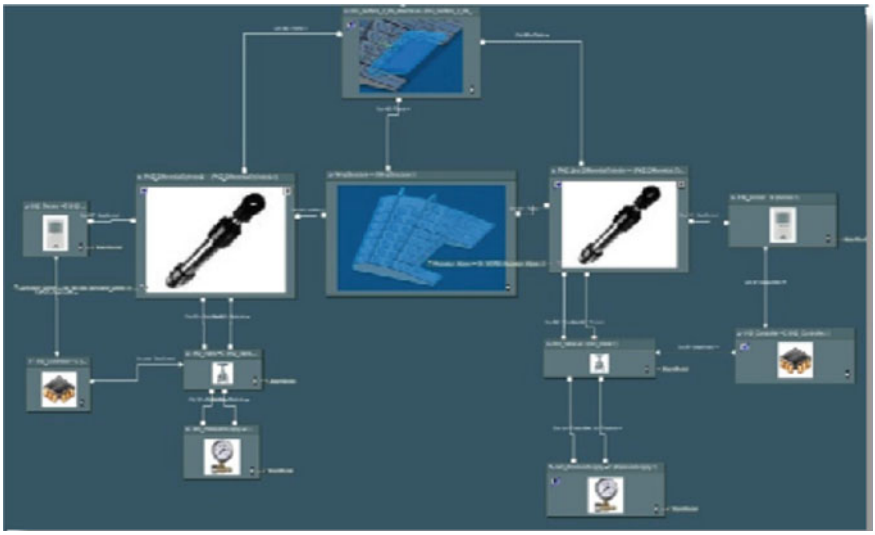

Fig. 9.12 Logical model of the aileron system [73]

the logical architecture (e.g. physical model, control system) are to be attached to behavioral models in order to enable a whole system simulation.

Since the global simulation is using a Modelica platform in background, the control component that has been created in MATLAB/Simulink is used for generating a functional mockup unit (FMU), as neutral format. Latter is attached to the logical model (Fig. 9.12) [73].

In order to consider additional models such as FEA models in the simulation, they have been simplified, then represented in the SID format, that can be imported by Modellica libraries [73]. In fact, SID files can be generated from the most commercial FEA programms with order reduction methods and animation information which is used by the Flexible Bodies library of Modelica. A multi-physic simulation of the whole aileron can be started, in which the deformations of the aileron are visualized. When movements of the aileron are required, the actuators receive hydraulic power from the hydraulic components they are connected with and therefore, the aileron is moving according to the impact of both fighting forces.

Several parameters that are important for the system, for instance the pressure, the position and the angle are represented in plots and therefore can be assessed by engineers. The system validation is performed through checking the links existing between the formulated requirements and each of the models created, that are the functional, the logical and the physical models. The objective is to determine potential discrepancies.

9.6.2 Hubble Space Telescope Systems Engineering

The Hubble Space Telescope (HST) is an observing system of systems that produces imaging, spectrographic, astrometric and photometric data. The HST has been developed based on SE principles, therefore its facet as a valuable result of SE has been highlighted in a case study originally provided by Mattice [74].

The HST case study has been guided by the Friedman-Sage Framework that enables the presentation of contractor, shared and government responsibilities for nine concept domains. These domains are spanning from the requirement definition and management over the systems architecting and conceptual design up to system and program management [74, 75]. For sake of brevity, this structure is not followed in this document.

Developing the HST has been necessary because the composition of the atmosphere limits the resolution of telescopes that are located on earth. Therefore, an alternative system was necessary, that should be located in space, outside the atmosphere.

The HST has been deployed 1990 in low-Earth orbit (600 km). It has been designed to observe the space permanently and to be maintainable, therefore adjustments can be performed during regular servicing missions. For this purpose, it has been equipped with necessary components such as grapples and handholds for control [74], but also with diverse components that enable the communication as well as an external control. Figure 9.13 depicts the major components of HST.

9.6.2.1 Requirement Definition and System Specification

The main requirement to the HST has been to provide relevant data to astronomers in order to help them conducting research in their scientific domain. Astronomer-scientists who took the role of a customer defined the requirements regarding the capability of HST. Among others, they described the observations that the HST should enable, when and by whom observing operations were to be performed [74]. Requirements also regarded the external controlling of the HST, design, development, in-orbit operations, maintenance and so on.

Due to conflicts between the astronomer scientists and the NASA in this phase, the HST-linked institute was created as neutral instance for managing HST project. It was in charge of defining the location, the research agenda, the scientific instrument requirements as well as playing a key role in HST ground and space operations [74]. An important lesson learned from this phase is the fact that the customer or user should be involved from the beginning throughout the project in order to get success.

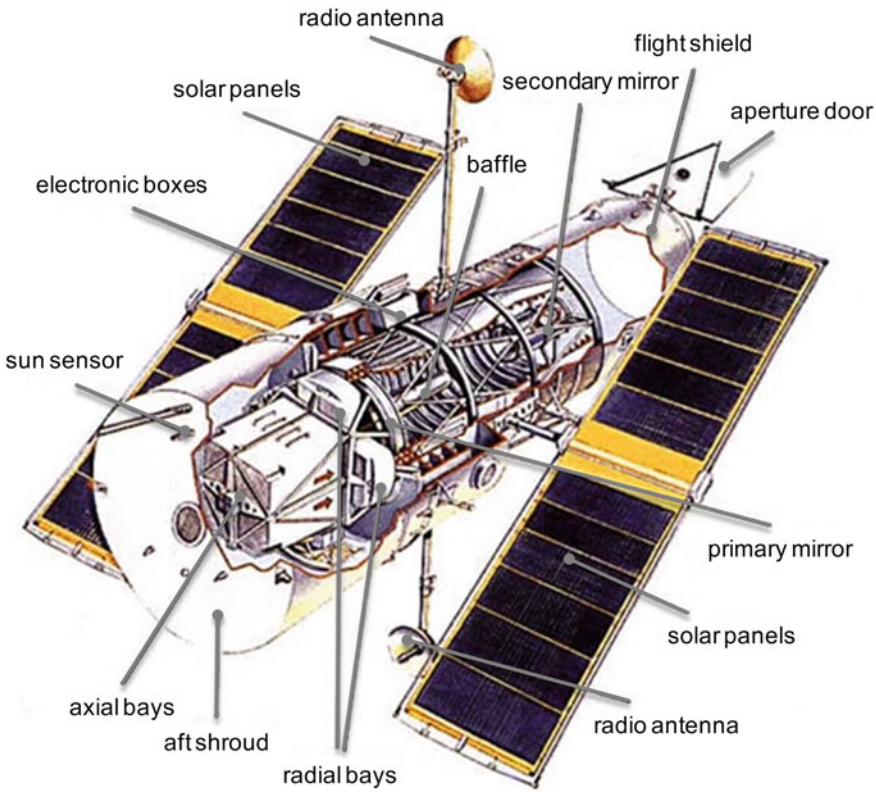

Fig. 9.13 Main components of HST [74]

9.6.2.2 Systems Architecture and Conceptual Design

Pre-program trade studies that were provided by contractors and NASA centers offered important inputs for HST project, since the submitted technical concepts addressed critical requirements and feasibility issues.

However, it is important to find the right balance between the different requirements such as costs and technical functions. For instance, some trade studies claimed the presumed advantages of reducing the HST mirror from 3 to 2.4 m in order to reduce costs. Doing so was assumed to positively impact manufacturing, test and assembly whereby the transport of the HST in the space still would not be impeded. The reduced size also should be suitable for fulfilling the requirements regarding light gathering, optical accuracy, pointing as well as stability control [74]. Analysis however had established some drawbacks related to that proposal, such as the reduction of the light collecting capability to the third, the reduction of the resolution capability and even more, although a weight reduction would have been achieved.

Furthermore, additional concepts including operation for launch, deployment and servicing was elaborated with cost trade-offs. The conceptual design was refined with costs and requirements for detailed design, development and construction were identified. Finally, the implementation of the system architecture has been more impacted by technical requirements rather than the cost projections [74]. This phase was characterized among others by the following activities:

- Risk, cost, schedule and configuration management
- Independent review and payload specification groups
- Case-dependent simulation, laboratory and ground testing prior to initial flight and on-orbit repair
- Definition of relative roles and contributions of involved stake holders.

9.6.2.3 Design

The high requirements (e.g. tolerance requirements) to the mirror that contains the primary mirror (Fig. 9.13) have been important factors to be considered in this phase. Innovative solution approaches were introduced in order to counter the weight of the mirror and therefore achieve the goal of zero gravity of the mirror for testing purposes [74].

Further critical points have been among others, the engineering and assembly of main sub-systems and components as well as guidance sensors.

9.6.2.4 System Integration

Reducing the mirror size from 3 m to 2.4 revealed itself to be a substantial modification, since other main components were impacted. The envelope of the optical components of the HST had to be redesigned to meet the new dimensions. This led to many component pairs not anymore linkable with each other. This challenge has been tackled using kinematic joints to dynamically isolate the components [74].

In order to perform the system integration of the HST physical, structural, electrical, optical, electronic, thermal control, power as well as operational software/hardware domains were involved. The phases of pre-ascension, ascension and post-ascension were distinguished [74].

Furthermore, instrument (e.g. mirror) specific requirements, e.g. for packaging, power, thermal control and orientation as well as additional functional requirements were to be taken into account and considered from the instrument level down to the component level.

A high amount of functions were simultaneously monitored and were able to be controlled not only on Earth but also from the shuttle that would take the HST on orbit, after it has been deployed [74].

Another important issue regarding system integration has been the weight allocation and management. The weight of subsystems was to be tracked in order to organise launching as well as maintenance missions [74].

In order to fulfil the weight requirements, an important effort (e.g. creation of weight reference plans, documentation) has been necessary to preserve flexibility from the development, fabrication, integration, deployment up to the maintenance phases [74].

The system integration included more than the weight dimension and had required a high degree of discipline, documentation and communication involving not only humans, but also machines. The importance of system integration is addressed by Langford [38] in several facet, whereby seven integrations principles are elaborated as recommendation: alignment, partitioning, induction, limitation, forethought, planning and loss. It may vary depending on the type of acquisition, the amount of detail in the system design, and the degree of specificity for the concept of operations.

9.6.2.5 Validation and Verification

Many options have been discussed for HST validation and verification. For instance, the alternatives of performing verification on ground or in the real operation conditions, that is, in the space, whereby costs and risks were to be considered. Performing also incremental tests were compared with conducting all-up tests that the NASA already had successfully experienced in former projects. The higher risks associated with HST led to thinking about conducting a complete system vacuum thermal test in a chamber, thus providing a realistic test environment [74].

Although this type of test only could generate new issues in term of costs, it was decided to adopt them in order to save long term costs since a system that is not tested enough might have generate more maintenance efforts and therefore more costs.

An up-and-running test program was performed during 30 days and showed the weaks of the system, such as the unreliability of power supply. Due to the tests results that were not satisfying, it also was discussed whether it would have been better to conduct tests cycles, in which incremental tests of the design would help reaching a final status that would satisfy the requirements [74].

These discussions highlight the dilemma in which the project members have been in, but also the key role of an early system integration and validation, which are supported by a consistent application of SE.

9.7 Discussion

SE can be apprehended as a means for mastering complexity while developing systems of systems. It defines processes as well as methods to be performed, that might suggest a certain rigidity. However, SE grants enough freedom in the detailed

realization of the high level processes. For instance, parallelization can be performed during requirement management, functional model development and so on. Besides, templates can and should be used in the different stages of SE in order to enable the parallelization of tasks across the activities described by the SE process. For instance, a logical system model can be elaborated using templates and therefore would be available for early studies, before the functional model has been detailed.

Furthermore, SE specifies that the inputs for the creation of physical models are to be provided by the logical model that represents the systems view (Fig. 9.7). This enables therefore the parallel creation of domain specific models instead of a sequential development of the physical models, that would be driven by the mechanical design. Since concurrent engineering addresses the parallelization of diverse phases and processes of product development, it can be considered as a complementary for systems engineering.

A further aspect that deserves to be discussed is the procedure of enabling SE in a company. Although the process of SE as well as tools and methods have been presented and referenced, an important factor remains the economic considerations, since the leading managers often would ask about the return of investment (ROI) when it comes to introduce new methods and systems. SE surely would not make an exception. Therefore, the SE strategy should emphasize the needed SE capabilities and if necessary, implement SE following an incremental approach. Especially the elaboration of an SE architecture should be approached with this perception in mind. While some software vendors are proposing integrated environment, a dedicated SE architecture that takes legacy applications into account could be more advantageous from a financial and technical point of view.

Integrated environments certainly enable a quick start-up, however, they can be characterized with limitations. This often is the case when a data set to be processed is existing in a proprietary format of competitors. Therefore, the support of standards is crucial (e.g., FMI, Spice, CMMI, ISO61508, ISO26262, Spice, DO178C, DO254, FDA, GAMP etc.). Even though some initiatives are working for boosting the openness of systems (e.g. Code of PLM Openness Initiative [76]), there is still a lot to do in order to integrate different systems.

Furthermore, while most scientists are agreeing in the meantime that MBSE is one of the core elements of SE, questions still remain about its formal creation and the language to described it. SysML certainly is widespread, however, investigations have shown that all the engineers involved in the product development are not experienced with it. Besides, further languages are existing. This calls for a stronger integration of SE in academia, not only for special programs, but also for all technical studies.

Considering the background of zero error products, manufacturing as well as other disciplines related to product creation, should be handled with the same engagement as product SE, since even an excellent design does not exclude making failures during manufacturing. This has been observed for instance in the HST case study, that identified the aberation failure [74] of HST after 1990 launch as a failure arising from polishing operations during manufacturing. Product-Service Engineering also should be considered from the first product idea onward.

Looking somewhat further ahead, the complexity will increase further and the question whether a general or a specific SE-approach should be used will remain. Next generations of systems of systems are planned to be self-optimizing, adaptive and even autonomous. Many carmakers as well as well-known software companies already have presented early prototypes of autonomous cars. This will bring challenges to be tackled by SE. Furthermore, the advances that might be expected in the development of electric vehicles, such as the communication of cars with power supply infrastructure as well as with diverse further systems (e.g. other cars) and the necessity to protect such systems of systems against hackers are some examples of complex challenges for SE. Sustainability calls for new materials and concepts, more and appropriate simulations will be necessary.

Not only the systems to be created using SE are to be considered, but also the enterprise processes for performing SE. In this case, the tends for bring-your-own-device, the integration of social media in the product development, whereby customers experience and requirements are captured, surely necessitates appropriate SE approaches. It can be expected that customer will not influence only the product development, but also its manufacturing, in regards of rapid prototyping technologies being more and more available for individuals [71, 77–79]. Thus, a manufacture-it-yourself mentality could call for new ways of developing products.

All the factors mentioned above as well as the expected influence of customers require efficient decision making tools and processes. The aforementioned challenge of human-centered design, the standardization of the vocabulary in SE as well as organizational changes in companies to support SE also are to influence the way of working [80]. In order to conform to SE principles such as the elaboration of a functional model, going back from existing physical systems to their functional models is supposed to be necessary, since these models are to be integrated in the functional model of their advisory system of systems [81].

The trend to shorten product lifecycles as well as personal proposals or recommendation to customers due to increasing capture of customer behaviour, for instance with cookies or game consoles, creates more challenges for systems integration, knowledge management and variability management [82]. Since companies are working more and more in cooperation (e.g., cooperation for developing car batteries or composite materials) and using clouds services, regarding the growing tend for mobile offices, intellectual property protection, especially enterprise rights management will influence today's conventional way of working.

9.8 Conclusions and Outlook

Product complexity is multidimensional and consists of product as well as process complexity.

Regarding product complexity, the number of functions as well as components has increased in the last decades. New functions have been created and assigned to

components and new components (e.g., electronic components of a passenger car) have been created, for instance, for safety purposes. Therefore, additional interfaces are to be considered on the one hand between the sub-systems of the car and, on the other hand, between the car and its environment, which includes passengers, especially when thinking of topics such as smart car, connectivity, and so on.

Product complexity is accompanied by process complexity, which is characterized by the use of more and more complex tools, interfaces and specifications when thinking in term of compliance [83]. In some cases, process discontinuities have led to important delays of the start of production (SOP) of well-known airplanes and cars models.

Process complexity is not limited only to development processes, since industrial processes also are impacted by the requirements that are to be fulfilled by products as well as company internal objectives such as target zero defect, lean, green and compliant manufacturing [84]. Besides, innovation leads, for instance, to applying new processes and materials, which may imply a higher degree of complexity like in smart factories.

In order to manage complexity while developing complex industrial products such as cars and planes, which of course are systems of systems, the techniques of SE can be applied. They are appropriate means that help the companies either to cope with complexity, to manage it or to reduce and eliminate it [79].

SE relies on models and methods that are to be elaborated for solving a problem and to reach a target of the system, which is ultimately satisfaction of stakeholders [85]. Verification and validation are necessary to ensure that requirements have been fulfilled in a manner that satisfies the stakeholders [59].

Since many disciplines are involved in the creation of complex systems, determining a meta-model of the system-of-interest as well as sharing knowledge are essential. Clear and defined interfaces between disciplines are necessary in order to enable information sharing. That common understanding is fundamental for systems thinking.

This chapter is an attempt to provide a deep understanding of Systems Engineering. The SE process as well as relevant tools and methods have been presented. Besides, proposals have been described for implementing a SE platform and for introducing SE in a company.

References

1. Müller P, Pasch F, Drewinski R, Hayka H (2013) Kollaborative Produktentwicklung und digitale Werkzeuge: Defizite heute – Potenziale morgen. Fraunhofer-Institut Produktionsanlagen und Konstruktionstechnik IPK
2. Stevenson M (2013) The role of services in flexible supply chains: an exploratory study. Int J Agile Syst Manage 6(4):307–323
3. Blessing Mavengere N (2013) Information technology role in supply chain's strategic agility. Int J Agile Syst Manage 6(1):7–24
4. Winzer P (2013) Generic systems engineering. Springer, Berlin

5. Lindemann U, Maurer M, Braun T (2009) Structural complexity management. An approach for the field of product design. Springer, Berlin
6. Schuh G, Schwenk U (2001) Produktkomplexität managen. Hanser, München
7. Hitchins DK (2003) Advanced systems thinking, engineering and management. Artech House, Boston
8. Tolk A, Adams KM, Keating CB (2011) Towards intelligence-based systems engineering and system of systems engineering. In: Tolk A, Jain LA (eds) Intelligence-based systems engineering. Springer, Berlin
9. Frezzini FR, Sachan R, Azimi M (2011) Review of systems engineering scope and processes. In: Kamrani AK, Azimi M (eds) Systems engineering tools and methods. CRC Press, Boca Raton
10. Lamb CMT (2009) Collaborative systems thinking. An exploration of the mechanisms enabling systems thinking. PhD thesis, Massachusetts Institute of Technology, Cambridge
11. Pahl G, Beitz W, Feldhusen J, Grote KH (2007) Konstruktionslehre – Grundlagen erfolgreicher Produktentwicklung. Springer, Berlin
12. Lindemann U (2005) Methodische Entwicklung technischer Produkte: Methoden flexibel und situationsgerechtanwenden. Springer, Berlin
13. Ehrlenspiel K (2003) Integrierte Produktentwicklung: Denkabläufe, Methodeneinsatz, Zusammenarbeit. Hanser, München
14. NN (2014) Chrysler ruft rund 870.000 Geländewagen zurück. Spiegel-Online. http://www.spiegel.de/auto/aktuell/jeep-cherokee-chrysler-ruft-rund-870-000-gelaendewagen-zuruecka-962246.html. Accessed 2 Apr 2014
15. Haskins C (2011) International council on systems engineering (INCOSE) systems engineering handbook. A guide for system life cycle processes and activities. V. 3.2.1, INCOSE-TP-2003-002-03.2.1
16. Hitchins D (2007) Systems engineering: a 21st century systems methodology. Wiley, Chichester
17. Moser AH (2014) Understanding complex systems. A case study in space industry. Springer International Publishing, Switzerland
18. Szabo C, Diallo SY (2011) Defining and validating semantic machine to machine interoperability. In: Tolk A, Jain LA (eds) Intelligence-based systems engineering. Springer, Berlin
19. Fumarola M, Seck M, Verbraeck D (2011) A simulation-based systems design in multi-actor environments. In: Tolk A, Jain LA (eds) Intelligence-based systems engineering. Springer, Berlin
20. O'Shea J, Zuhair B, Keeley C (2011) Systems engineering and conversational agents. In: Tolk A, Jain LA (eds) Intelligence-based systems engineering. Springer, Berlin
21. Elgh F (2014) Automated engineer-to-order systems a task oriented approach to enable traceability of design rationale. Int J Agile Syst Manage 7(3/4):324–347
22. Albus J et al (2008) Intelligent control of mobility systems. In: Prokhorov D (ed) Computational intelligence in automotive applications. Springer, Berlin
23. Prokhorov D (2008) Neural networks in automotive applications. In: Prokhorov D (ed) Computational intelligence in automotive applications. Springer, Berlin
24. Gandhi T, Manubhai Trivedi M (2008) Computer vision and machine learning for enhancing pedestrian safety. In: Prokhorov D (ed) Computational intelligence in automotive applications. Springer, Berlin
25. Bergasa LM, Nuevo J, Sotelo MA, Barea R, Lopez E (2008) Visual monitoring of driver inattention. In: Prokhorov D (ed) Computational intelligence in automotive applications. Springer, Berlin
26. Kamrani AK (2011) Genetic-algorithm-based solution for combinatorial optimization problems. In: Kamrani AK, Azimi M (eds) Systems engineering tools and methods. CRC Press, Boca Raton
27. Garlan CM, Colombi J (2011) Systems engineering case studies. In: Kamrani AK, Azimi M (eds) Systems engineering tools and methods. CRC Press, Boca Raton

28. Cavalieri S, Pezzotta G (2012) Product-service systems engineering: state of the art and research challenges. Comput Ind 63:278–288
29. Peruzzini M, Germani M (2014) Design for sustainability of product-service systems. Int J Agile Syst Manage 7(3/4):206–219
30. Kossiakof A, Sweet WN, Seymour SJ, Biemer SM (2011) Systems engineering principles and practice, 2nd edn. Willey, Hoboken
31. Dikerson CE, Mavris DN (2008) Architecture and principles of systems engineering. CRC Press, Boca Raton
32. Pineda RL, Smith ED (2011) Functional analysis and architecture. In: Kamrani AK, Azimi M (eds) Systems engineering tools and methods. CRC Press, Boca Raton
33. Ryschkewitsch M, Schaible D, Larson W (2009) The art and science of systems engineering. Syst Res Forum 03(02):81–100
34. Rupp C (2009) Requirementsengineering und –management; Professionelle, Iterative Anforderungsanalyse für die Praxis, 5th edn. Carl Hanser Verlag, München Wien
35. Gilb T (2005) Competitive engineering—a handbook for systems engineering requirements engineering, and software engineering using planguage. Elsevier, Oxford
36. Buede DM (1999) Functional analysis. In: Sage AP, Rouse WB (eds) Handbook of systems engineering and management. Wiley Inc, New York
37. Palmer JD (1999) Systems integration. In: Sage AP, Rouse WB (eds) Handbook of systems engineering and management. Wiley Inc, New York
38. Langford GO (2012) Engineering systems integration theory, metrics, and methods. CRC Press, Boca Raton
39. Shabi J, Reich Y (2012) Developing an analytical model for planning systems verification, validation and testing processes. Adv Eng Inform 26(2):429–438
40. Kolonay RM (2014) A physics-based distributed collaborative design process for military aerospace vehicle development and technology assessment. Int J Agile Syst Manage 7(3/4): 242–260
41. Sop Njindam T, Platen E, Paetzold K (2012) Modellbasiertes systems engineering Zur Frühzeitigen Absicherung Komplexer Multidisziplinärer System. Tag des Syst Eng, pp 271–282
42. Biahmou A, Fröhlich A, Stjepandić J (2010) Improving interoperability in mechatronic product development. In: Thoben KD et al (eds) Collaborative value creation throughout the whole lifecycle. Proceedings of PLM10 international conference, Inderscience, Geneve
43. Kuhn O, Liese H, Stjepandić J (2011) Methodology for knowledge-based engineering template update. In: Cavallucci D, Guio R, Cascini G (eds) Building innovation pipelines through computer-aided innovation. Springer, Berlin, pp 178–191
44. Ott S (2009) Konzept zur methodischen Systemmodellierung in der anforderungsgerechten Produktentwicklung. PhD thesis, Universität Wuppertal
45. Maurer M (2013) Automotive systems engineering: a personal perspective. In: Maurer M, Winner H (eds) Automotive systems engineering. Springer, Berlin
46. Aguirre MA (2013) Introduction to space systems: design and synthesis. Springer Science +Business Media, New York
47. Weilkiens T (2008) Systems engineering with SysML/UML: modeling, analysis, design, 2nd edn. Dpunkt Verlag, Heidelberg
48. Friedenthal S, Moore A, Steiner R (2012) A practical guide to SysML: the systems modeling language, 2nd edn. Morgan Kaufmann, Waltham
49. Brown B (2011) Model-based systems engineering: revolution or evolution? Thought Leadership White Paper, IBM Rational
50. Dumitrescu C, Tessier P, Salinesi C, Gerard S, Dauron A, Mazo R (2014) Capturing variability in model based systems engineering. In: Aiguier M, Bretaudeau F, Krob D (eds) Complex systems design and management. Springer, Berlin, pp 125–139
51. Winner H (2013) Challenges of automotive systems engineering for industry and academia. In: Maurer M, Winner H (eds) Automotive systems engineering. Springer, Berlin

52. Levandowski C, Raudberget D, Johannesson H (2014) Set-based concurrent engineering for early phases in platform development. In: Cha J et al (eds) Proceedings of 21th ISPE international conference on concurrent engineering. IOS Press, Amsterdam, pp 521–530
53. Kindel O, Friedrich M (2009) Softwareentwicklung mit AUTOSAR. Grundlagen, Engineering, Management für die Praxis. dpunktVerlag, Heidelberg
54. NN (2013) Functional mockup interface (FMI)—version 1.0. https://www.fmi-standard.org/downloads. Accessed 15 May 2014
55. NN (2013) ISO26262, ISO. http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=54591
56. Dineva E, Bachmann A, Moerland E, Nagel B, Gollnick V (2014) New methodology to explore the role of visualisation in aircraft design tasks: an empirical study. Int J Agile Syst Manage 7(3/4):220–241
57. Sanders A (2012) Klein J (2012) Systems engineering framework for integrated product and industrial design including trade study optimization. In: Dagli CH (ed) New challenges in systems engineering and architecting conference on systems engineering research (CSER). Elsevier, Amsterdam, pp 413–429
58. Van Ruijven LC (2012) Ontology and model-based systems engineering. In: Dagli CH (ed) New challenges in systems engineering and architecting conference on systems engineering research (CSER). Elsevier, Amsterdam, pp 194–200
59. Boehm B, Koolmanojwong S, Lane JA, Turner R (2012) Principles for successful systems engineering. In: Dagli CH (ed) New challenges in systems engineering and architecting conference on systems engineering research (CSER). Elsevier, Amsterdam, pp 297–302
60. Boy GA, Narkevicius JMG (2014) Unifying human centered design and systems engineering for human systems integration. In: Aiguier M et al (eds) Complex systems design and management 2013. Springer International Publishing, Switzerland
61. Bartelt C, Böß V, Brüning J, Rausch A, Denkena B, Tatou JP (2013) A software architecture to synchronize interactivity of concurrent simulations in systems engineering. In: Bil C et al (eds) Proceedings of 20th ISPE international conference on concurrent engineering. IOS Press, Amsterdam, pp 19–29
62. Cole R (2009) System of systems architecture. In: Jamshidi M (ed) System of systems engineering: principles and applications. CRC Press, Boca Raton, pp 37–70
63. NN (2011) ISO/IEC/IEEE 42010:2011—systems and software engineering—architecture description. Iso.org. 2011-11-24. Accessed 15 Feb 2014
64. NN (2012) Enterprise architecture. An overwiew. http://isa.unomaha.edu/wp-content/uploads/2012/08/Enterprise-Architecture.pdf. Accessed 3 Aug 2014
65. Urbaczewski L, Mrdalj S (2006) A comparison of enterprise architecture frameworks. Issues Inf Syst 7(2):18–26
66. Reich C, Burghard O (2009) Architekturentwicklung in der wehrtechnischen Industrie. http://www.bitkom.org/files/documents/Leitfaden_ArchitekturentwicklungInDerWtIndustrie.pdf. Accessed 3 Aug 2014
67. Albers A, Zingel C (2013) Challenges of model-based systems engineering: a study towards unified term understanding and the state of usage of SysML. In: Abramovici M, Stark R (eds) Smart product engineering. Springer, Berlin, pp 83–92
68. Rodriguez-Priego E, García-Izquierdo FJ, Rubio AL (2010) Modeling issues: a survival guide for a non-expert modeler. In Petriu DC, Rouquette N, Haugen, Ø (eds) MODELS 2010, Part II, LNCS 6395. Springer, Berlin 2010, pp 361–375
69. Sun J, Hiekata K, Yamato H, Nakagaki N, Sugawara A (2014) Virtualization and automation of curved shell plates manufacturing plan design process for knowledge elicitation. Int J Agile Syst Manage 7(3/4):282–303
70. Königs SF, Beier G, Figge A, Stark R (2012) Traceability in systems engineering—review of industrial practices, state-of-the-art technologies and new research solutions. Adv Eng Inform 26(2012):924–940
71. Chang D, Chen CH (2014) Understanding the influence of customers on product innovation. Int J Agile Syst Manage 7(3/4):348–364

72. Alguezaui S, Filieri R (2014) A knowledge-based view of the extending enterprise for enhancing a collaborative innovation advantage. Int J Agile Syst Manage 7(2):116–131
73. Vuillemin B, Croue N, Loembe S (2012) MBSE applied to an aerospace "force fighting" application, ERTS2 2012—embedded real time software and systems, Toulouse, 1–3 Feb 2012. http://www.erts2012.org/Site/0P2RUC89/TA-2.pdf. Accessed 4 Aug 2014
74. Mattice JJ (2005) Hubble space telescope systems engineering case study. Defense Acquisition University. https://acc.dau.mil/adl/en-US/37600/file/9105/Hubble%20Space%20Telescope%20SE%20Case%20Study%20-%20JJ%20Mattice.pdf. Accessed 4 Aug 2014
75. Friedman G, Sage AP (2004) Case studies of systems engineering and management in systems acquisition. Syst Eng 7(1):84–96
76. NN (2010) Code of PLM openness, ProSTEP iVip association, Darmstadt. http://www.prostep.org/en/cpo.html. Acessed 4 Aug 2014
77. Nicholds BA, Mo J (2014) Determining an action plan for manufacturing system improvement: the theory. Int J Agile Syst Manage 6(4):324–344
78. Nicholds BA, Mo J, Bridger S (2014) Determining an action plan for manufacturing system improvement: a case study. Int J Agile Syst Manage 7(1):1–25
79. ElMaraghy W, ElMaraghy H, Tomiyama T, Monostori L (2012) Complexity in engineering design and manufacturing. CIRP Ann Manufact Technol 61:793–814
80. Ito T (2014) A proposal of body movement-based interaction towards remote collaboration for concurrent engineering. Int J Agile Syst Manage 7(3/4):365–382
81. McLay A (2014) Re-reengineering the dream: agility as competitive adaptability. Int J Agile Syst Manage 7(2):101–115
82. Moynihan P, Dai W (2011) Agile supply chain management: a services system approach. Int J Agile Syst Manage 4(4):280–300
83. Jacobs MA (2013) Complexity: toward an empirical measure. Technovation 33(2013):111–118
84. Carvalho H (2013) An innovative agile and resilient index for the automotive supply chain. Int J Agile Syst Manage 6(3):258–278
85. Modrak V, Semanco P (2012) Structural complexity assessment: a design and management tool for supply chain optimization. Procedia CIRP 3:227–232