Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 28 (2014) 682 – 691

Procedia Computer Science

Conference on Systems Engineering Research (CSER 2014)

Eds.: Azad M. Madni, University of Southern California; Barry Boehm, University of Southern California; Michael Sievers, Jet Propulsion Laboratory; Marilee Wheaton, The Aerospace Corporation
Redondo Beach, CA, March 21-22, 2014

Design for System Lifecycle Properties – A Generic Approach for Modularizing Systems

Florian Schoettl*, Udo Lindemann

Technische Universität München, Institute of Product Development, Boltzmannstrasse 15, 85748 Garching, Germany

Abstract

Planning and development of technical systems under consideration of the whole system lifecycle is a state-of-the-art procedure, which is well-established in practical use. For system engineers, the systematic inclusion of system lifecycle properties becomes increasingly important, because agile system are required, which work optimally under current boundary conditions and can easily adapt to future requirements. Manifold system properties with specific dependencies within the observed system necessitate a systematic analysis to facilitate a targeted consideration in the early phase of development. Therefore we present a generic approach for modularizing systems depending on system lifecycle properties. Within a case study of automotive assembly a validation of this method was done and first implications were derived.

Keywords: System lifecycle properties; modularization; commonality; clustering, design structure matrix

* Corresponding author. Tel.: +49-89-289-15156; fax: +49-89-289-15144.
E-mail address: schoettl@pe.mw.tum.de

doi:10.1016/j.procs.2014.03.082

1. Introduction

In the development of large-scale technical systems, a deep knowledge about their system lifecycle properties (Ilities) becomes more and more important. The net of constraints around such systems gets closer and the dependencies crossing the system boundary are increasingly volatile. If one considers production systems, there is a need for agile systems[1]. That means a system which is flexible and responsive towards unpredictable internal or external changes[2]. But to meet that system requirement of agility, it has to be clear which subsystem or element requires which kind or amount of flexibility and responsiveness. Figure 1 illustrates the current gap between Ilities that are related to distinct elements (e.g. the geometry flexibility of a gripper) and more vague system lifecycle properties regarding the complete system (e.g. the agility of an assembly line). One key aspect to link these levels is the system structure, which builds up the complete system from single elements. To model these dependencies, we take a system theoretical perspective. In the sense of Systems Engineering, the interplay between Ilities and responsible system elements as well as the dependencies between different Ilities have to be analyzed thoroughly. Moreover, to make sustainable invests in facilities and optimize the effort of system planning and development, the system engineer has to be able to identify system components (clusters or modules) with the same Ility-demands. Bridging this gap offers potentials amongst others from an organizational point of view to optimize the efficiency of the development process as well as to rise the system transparency and consequently to be able to cope with complexity during system operation.

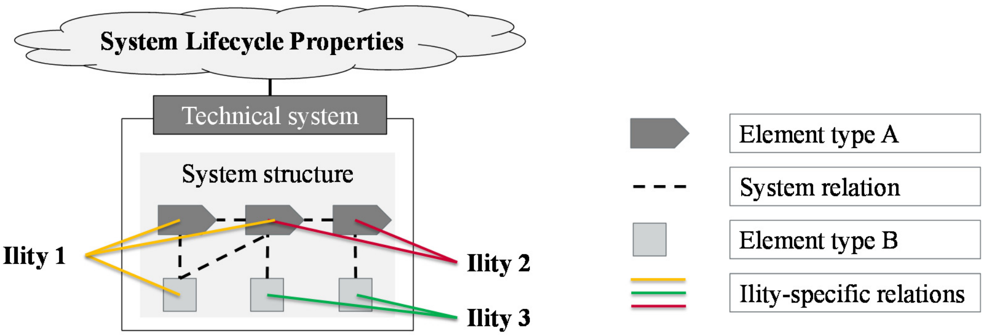

Fig. 1. Allocation of system lifecycle properties

In this paper, we focus on structuring system elements (depending on Ilities) and present an approach to modularize systems based on the commonality of Ilities in order to support the planning process in early phases. Therefore, we show a brief literature review, strongly focused on the related major research issues – system lifecycle properties, modularization and commonality approaches – and motivate this contribution (section 2). In section 3 we describe the concept of our modularization approach and its application in two phases. In a final step, a validation of the method using the example of a car assembly line is presented in section 4. Afterwards we critically discuss the benefit and weaknesses of our approach and its implications.

2. Literature Review

Based on the described problem three main research topics can be derived: The role of system lifecycle properties in planning and developing systems, approaches for clustering and modularizing systems and the corresponding objectives, the interplay between system properties and the system structure. Current publications and established state-of-the-art literature shows that the mentioned topics were mostly studied separately.

2.1. System lifecycle properties

In the last few years a lot of researcher groups gave attention to system lifecycle properties. A lot of studies investigated several Ilities of a certain technical system to meet specific goals, e.g. the publication from Beskesea et al.[3]: Quantification (specific goal) of flexibility (Ility) in advanced manufacturing systems (technical system) using fuzzy concept. But hardly anyone presents more general investigations, which are transferable to different use cases.

General considerations about Ilities from a system theoretical perspective can mainly be found in research projects of the MIT (Massachusetts Institute of Technology). These studies aimed at exploring and collecting the multitude of Ilities and to consequently implement them into a holistic classification system. Tracking the results of published, finalized studies chronologically, the following evolution becomes evident: In 2004, Crawley et al.[4] attempt to relate Ilities with the behavior and the architecture of complex systems. They postulate that the system architecture, as well as the system structure, influence or rather determine Ilities. Giving an outlook to further need for research they raise the question: „What are Ilities, and can they be classified?" Browning and Honour[5] present an approach to quantify the system value along the lifecycle. They generate a so-called lifecycle value (LCV) using weighted key parameters to describe stakeholder and their preferences. These key parameters are similar to Ilities. But Browning and Honour do not consider the system structure. McManus et al.[6] embed Ilities in a universal framework to use them as objectives in the conceptual design of systems. Thereby they focus on changes and financially quantify the survivability of a space tug in a comprehensive use case. Ross et al.[7] define and differentiate several Ilities (flexibility, adaptability, scalability, modifiability and robustness) aiming at the quantification of value robust systems by using an Ility taxonomy. More general conclusions are drawn by De Weck et al.[8] in 2012 coming again from the assumption that system lifecycle properties determine the system value. Hence, they present a basic review about Ilities and try to classify them. The quintessence of this work is that different Ilities interplay with each other more or less and a hierarchical structuring seems to be reasonable.

Publications related to specific Ilities are usually based on concrete technical problems and validated with detailed case studies. Nevertheless, general conclusions regarding higher-level systems as well as implications in the role of Ilities in the system structure are rare. Having a closer look at one specific Ility and related investigations, the contribution of such publications for our research becomes clear. Using the example of flexibility, there are numerous studies, which examine amongst others influencing factors and consequences of this Ility. For example Alexopoulos et al.[9] use the analogy of a dynamic mechanical system to quantify flexibility. This approach includes the system behavior as well as basic system parameters and indicates that flexibility depends on different system configurations. In the same manner, most approaches concentrate on diverse calculation and assessment methods to measure flexibility, e.g. fuzzy concepts[3], sensitivity of changes[10] or economic evaluation based on costs[11]. In summary, objectives like capacity, output, operational costs etc. and even the system behavior play the decisive role in these flexibility-related studies, but general implications from a system theoretical perspective are missing.

2.2. Modularization and commonality approaches

The upcoming trend in the development of technical systems is clearly towards modular instead of integral architectures, whereby adaptable parts (modules) and standard parts (platform) are linked by defined interfaces. This design principle possesses cost advantages and less updating effort in the context of volatile boundary conditions[12, 13], which we focus on. A well-known example is the architecture of a personal computer. Components like the processor as well as peripheral devices like a printer are connected via standardized interfaces and can be exchanged. The principle of commonality can be applied on different objects, using different levels of concretization. Nevertheless, basic types of commonality can be found in literature: physical, technological and functional commonality. The fundamental principle of commonality and modularity approaches consists of organizing systems in terms of common structures. For this purpose, Fixson[14] presents a broad literature review about modularity and commonality research. He concludes that most studies just aim on cost effects by investigating products, processes, organizations and even innovations. According to him, future work has to consider further performance measures than only costs, especially in the field of complex technical systems. Continuing Fixon's outlook, Simpson et al.[15] present applications of modularity and commonality in terms of product family design.

2.3. The role of system lifecycle properties in the development process

Previous remarks make clear that Ilities play an important, perhaps a decisive role in system design and development. In the case of concrete Ilities, as already shown by the example of flexibility, there are numerous methods for description, quantification and assessment. General considerations without specific instance are still limited and unsuitable to describe dependencies between Ilities or even certainly predict the system behavior. Necessary influencing parameters or rather structural criterions exist in part, for example presented by Scholz-Reiter et al.[16] in the case of complexity, by Fricke and Schulz[17] in the case of changeability or more general by Biedermann and Lindemann[18]. But these studies are not able to address Ilities and link them directly with the system structure. It becomes evident that Ilities cannot be considered separately without any knowledge about their impact on the complete system[8]. In accordance with Fixson[14], structuring by modularization methods has great potential in terms of agility, if suitable modularization criterions will be identified and used. Due to the need for an integrated consideration, we present a generic approach for clustering systems by system lifecycle properties.

3. Modularization approach

In this section we present a method to identify so-called Ility-modules using matrix-based methods and commonality analysis. This contribution is based on a modularization approach presented by Schoettl et al.[19], which supports the structuring and visualization of stakeholder networks in large-scale socio-technical systems. In order to clearly explain our generic approach, this section is split into three parts. First, we introduce the basic concept and point out the related scientific principles from literature. Then we present the application, which is divided into the part information acquisition and the part commonality analysis.

3.1. Part I – Concept

To bridge the gap between abstract Ilities on the level of the complete system, which are an important objective in the early phase of system design and concrete Ilities related to discrete system components, a system theoretical perspective is required. Furthermore the approach must be capable of handling, analyzing and visualizing large amounts of data. The approach of Schoettl et al.[19] meets these basic criteria, due to it is based on matrices. The central idea of their approach is to model dependencies between different types of system elements in matrices and use the principles of matrix multiplication to derive indirect dependencies. For example, if we just know that Person A is installing the suspension (direct dependency) and installing the suspension requires a torque spanner (direct dependency), we can reason that Person A needs a torque spanner (indirect dependency), because he has to install the suspension. Based on these principles, we model information about dependencies in three types of matrices: The Multi Domain Matrix (MDM), the Domain Mapping Matrix (DMM) and the Design Structure Matrix (DSM). A detailed explanation of the mechanics of these matrices in accordance to Lindemann et al.[20] can be found in the appendix (cp. figure A1). Figure 2 illustrates our basic approach and an extended version as MDM:

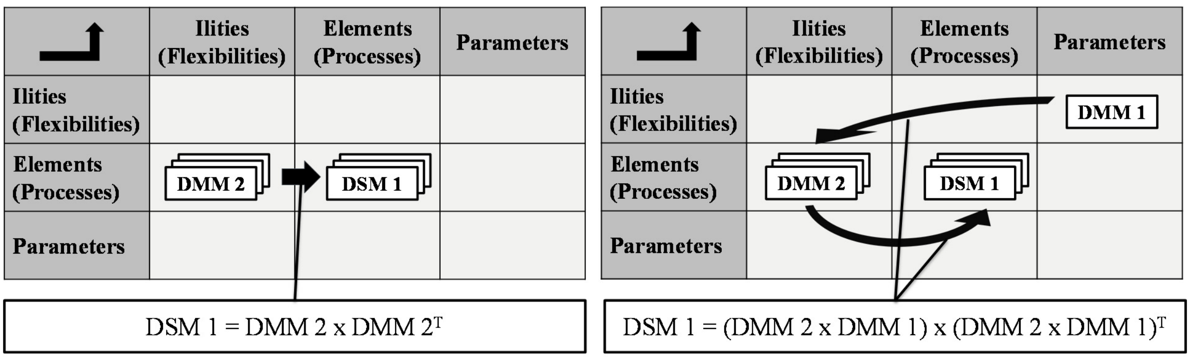

Fig. 2. (a) Basic modularization approach; (b) Extended modularization approach regarding parameters

Figure 2 (a) shows the MDM of our modularization approach. It consists of three domains: Ilities, system elements, and parameters. The last domain is necessary to reach a lower level of abstraction. Further explanations on that issue will be given in subsection 3.2. Starting with the DMM 2, which links system elements to Ilities, one can derive DSM

1 in one arithmetic operation (cp. Lindemann et al.[20]). DMM 2 has to be multiplied with the transposed DMM 2 in order to get the dependencies of system elements. Consequently, system elements are connected in DSM 1, if they are linked to one or more identic Ilities in DMM 2. Figure 2 (b) depicts three different matrices, that have to be processes in two arithmetical operations (cp. Lindemann et al.[20]). In this case, DMM 2 (the same as in the basic approach) has to be multiplied with DMM 1, which links Ilities to influencing parameters. This results in a further DMM, which includes dependencies between elements and parameters due to they are linked to same Ilities. We skip that intermediate step and multiply that DMM with the transposed to get DSM 1. Consequently, this matrix links elements, which are influenced by one or more identic parameters. Both MDM in figure 2 include more than one DMM 2 and DSM 1 (illustrated as stacks). This means that different types of dependencies can be used to model DMM 2 and consequently more than one DSM 1 results from that. If this is the case, the resulting DSM 1 are added to get one superimposed DSM 1. The analysis of these matrix-based system models will be explained in subsection 3.3.

3.2. Part II – Information acquisition

The depicted MDM in figure 2 include all relevant information for modeling the elements and dependencies. In addition, the arithmetical operations are known in order to transfer native dependencies (DMM 1 and DMM 2) to a derived system model (DSM 1) that can be analyzed in terms of commonality. Therefore we demonstrate how to apply the procedure in four or rather six basic steps, according to figure 3. But before we go into detail, some general remarks have to be made: Depending on the particular objective of using this approach, it is possible to model Ilities and elements on different levels of abstraction. An overview of some concrete examples follows:

- Ilities to characterize the complete system like robustness, agility, versatility, etc.
- Ilities to specify distinct system elements like geometry flexibility, size flexibility, product flexibility, etc.
- Abstract elements on a high system level like departments, factories, plants, etc.
- Distinct elements on a low system level like machines, assembly processes, tools, etc.

We recommend using the same level of abstraction for both domains, Ilities and elements. Consequently one could consider low level Ilities and distinct elements to derive practical recommendations for system design, as it will be shown in our case study. One additional boundary condition is the variety of elements that will be analyzed. In this contribution we concentrate on the dependencies between Ilities and one type of elements. To execute the basic procedure according to figure 3 (a), one has to run through the following four steps:

1. The Ilities domain has to be defined to pursue a constant level of abstraction of subordinated elements. Afterwards, all system elements of this domain have to be gathered. We use distinct Ilities: Flexibilities.
2. The elements domain has to be defined and subordinated elements have to be gathered. We use low level elements: Processes.
3. To link elements to Ilities or rather processes to flexibilities, the type of relation between these domains has to be defined. We use the term “process has flexibility”.
4. The last step is to calculate one or more DSM 1 according to the arithmetical operation in figure 2 (a). This matrix models dependencies like: Assembling the sunroof and installing the suspension are linked to each other, because both processes have ergonomic flexibility. Starting with a single DSM 1, element groups with a high commonality in the linked Ilities can already be identified after clustering the matrix. But the added value of this approach can be seen from superimposing the calculated Ility-specific DSM 1 to get element clusters with similar requirement profile, called Ility-set.

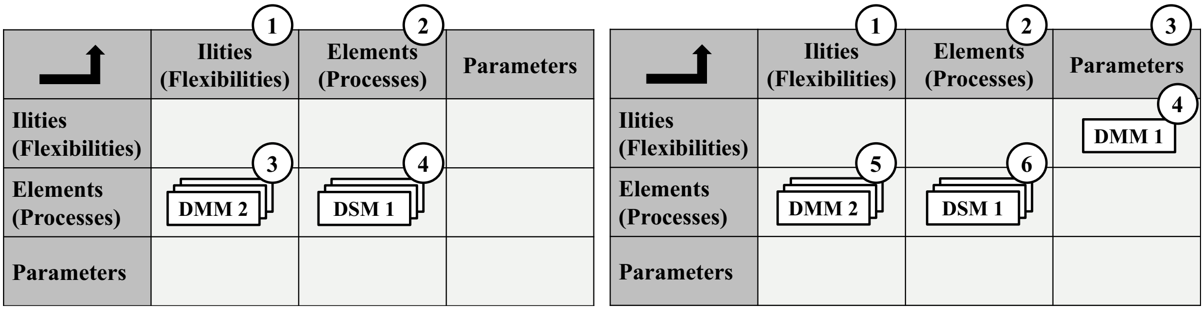

Fig. 3. (a) Basic procedure; (b) Extended procedure regarding parameters

An additional level of concretization is shown in figure 3 (b) using an additional domain. The parameters domain includes influencing factors which determine or rather control the considered Ilities. To execute the extended procedure according to figure 3 (b), one has to run through the following six steps:

1. See basic procedure step 1.
2. See basic procedure step 2.
3. The parameters domain has to be defined to pursue a constant level of abstraction of subordinated elements. Afterwards, all system elements of this domain have to be gathered.
4. To link Ilities to parameters or rather flexibilities to parameters, the type of relation between these domains has to be defined. We use the term "Ility is determined by parameter".
5. See basic procedure step 3.
6. The last step is to calculate one or more DSM 1 according to the arithmetical operation in figure 2 (b). Hence, the resulting DSM 1 connects processes, because the related Ilities are determined by the same influencing parameters. Based on this information, optimization potentials in the development organization can be found. If different elements (e.g. assembling the sunroof and installing the suspension) are planned by several developers which determine the same system parameters (e.g. working height in the assembly line) due to linked Ilities (e.g. ergonomic flexibility), they should work together. These connections are often unknown or just implicitly documented.

3.3. Part III – Commonality analysis

Due to the applied matrix multiplication of DMM 2 and the transposed DMM 2, a connection between two elements arises in the DSM 1, if both elements are linked to the same flexibility. When superimposing the resulting matrices DSM 1, the numerical value in each matrix cell represents the number of common dependencies. Thus, the degree of commonality of system elements can directly be extracted from this matrix. Further analysis and visualization requires arranging the elements in DSM 1 according to clusters or rather Ility-modules. The application of clustering algorithms such as, for example, described by Aggarwal and Reddy[21], is not possible at the moment, because additional boundary conditions of target clusters are required. In our use case, we have found out that clustering by hand tends to result in the best modularization with regard to the homogeneity of the Ility-modules.

4. Case study

We proofed the validity of our approach by a case study in automotive industry, exemplified by a vehicle assembly line. 31 sequenced assembly processes and four types of flexibility were analyzed to demonstrate the application of our method and the result: process clusters based on the commonality of flexibilities. Thus, these process modules have a high conformity in the flexibility-sets they pursue.

4.1. Application of the modularization approach

In a first step the domains were defined and assembly processes as well as flexibilities were gathered. All processes of the assembly line can be seen in figure 4. The following types of flexibility with subject were defined:

- positioning flexibility of a machine
- variant flexibility of a machine
- production time flexibility of a process
- automation flexibility of a process

As a next step the dependencies between assembly processes and flexibilities were modelled in DMM 2 using the term “assembly process has flexibility”. Due to the diversity of these flexibilities, four DMM 2 were created. Afterwards four DSM 1 could be derived and were added to the superimposed DSM 1, which is shown in figure 4.

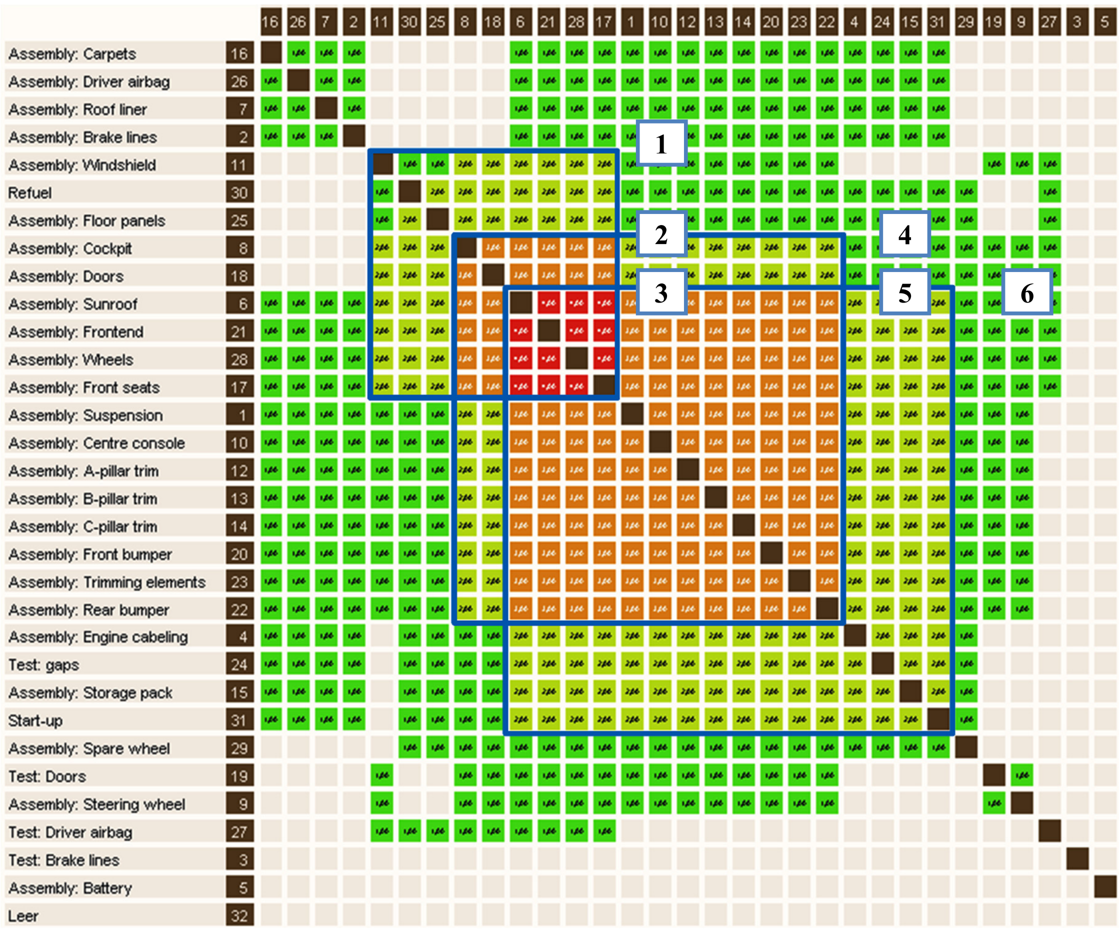

Fig. 4. Superimposed DSM 1 with potential process clusters

The coloring of the matrix cells represents the degree of commonality of the linked processes from green (low) to red (high). Possible modules are marked in the blue squares, which result from clustering the matrix by hand. The labelling can be found at the upper right edge of each cluster. Of course there are further alternatives besides the depicted result, depending on boundary conditions like the number of final clusters or the maximum number of elements per cluster. But this fuzziness of the result is the nature of the modeled issue. In this case, overlapping dependencies of assembly processes and different flexibilities result in a diffuse net of dependencies. Consequently, there can be no distinct solution for a multifaceted issue. Due to a quite general connection between elements (process has flexibility) was analyzed in the case study there are several possible interpretations of the clustered matrix. Thus, we choose the perspective of a system engineer, who is responsible for the whole assembly process from an organizational point of view.

4.2. Interpretation of results

To be able to derive explicit recommendations for action from the system connections, a clear objective is needed. If this study was made as a preliminary for a future reorganization, some basic boundary conditions of the future structure must be known. This may, for instance, be an upper or lower limit for the number of departments or the number of hierarchical levels. So, the number as well as the structure of target clusters can already be determined. With the demand for four departments on the same hierarchical level, one possible configuration results from the three big clusters (No. 1, 4, 6) and the remaining elements. In detail, the allocation of processes to departments looks like that:

- Department A: 11, 30, 25, 8, 18, 6, 21, 28, 17 (cluster 1)
- Department B: 8, 18, 6, 21, 28, 17, 1, 10, 12, 13, 14, 20, 23, 22 (cluster 4)
- Department C: 6, 21, 28, 17, 1, 10, 12, 13, 14, 20, 23, 22, 4, 24, 15, 31 (cluster 6)
- Department D: 16, 26, 7, 2, 29, 19, 9, 27 (remaining processes)

This solution shows a lot of overlap between the clusters. Such an organization would not be feasible. A classification of processes in different hierarchical levels seems to be more useful, because DSM 1 encloses a couple of clusters within bigger clusters. Two possible configurations considering that structural aspect are depicted in figure 5. The organization chart on the left side (a) shows all derived clusters from figure 4 in a hierarchical structure. The orange boxes have a dashed frame, due to they are completely included in the lower level, colored in light green. On the right (b), one can see a feasible configuration, consisting of four departments on two levels.

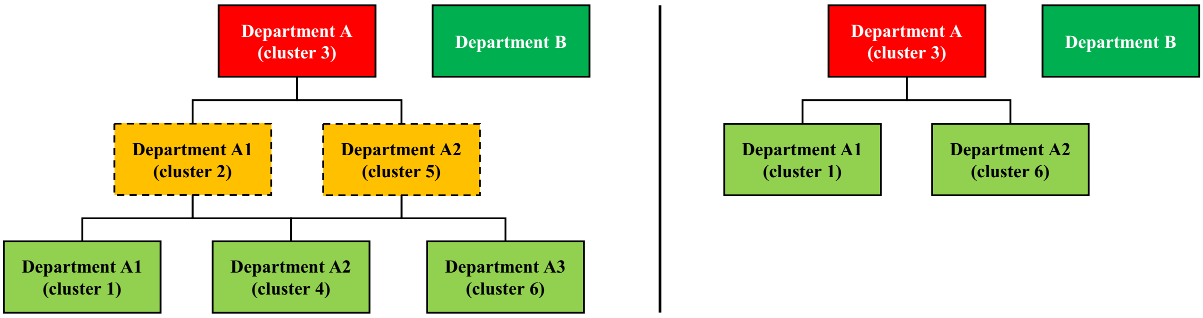

Fig. 5. (a) Organization chart with five departments on three levels; (b) organization chart with four departments on two levels

Assuming that the highest degree of commonality within the departments should be realized, in order to generate synergistic affects, the following flexibility-sets result from figure 5 (b):

- Department A: All flexibilities (each element depends on all flexibilities)
- Department A1: Positioning flexibility, variant flexibility, automation flexibility
- Department A2: Positioning flexibility, variant flexibility, production time flexibility
- Department B: All flexibilities (each element has just a single link)

4.3. Critical discussion

By applying the presented approach to support the design of systems, the dependencies between system lifecycle properties and system elements can be modelled and used to identify Ility-modules. But there are even some critical aspects affecting the significance of the results. The definitions of Ilities are manifold and not generally accepted. Consequently, it is hard to model the right links between Ilities and system elements. The more abstract Ilities are, the more difficult it is to define and handle them. As De Weck et al.[8] already tried to answer, is the question about the interplay between several Ilities. We disregarded that issue, because the net of flexibilities in our case would have been analyzed in detail, if certain flexibilities enable others. The second aspect is related to the matrix-based method. Using the basic approach, the modelling of dependencies is unproblematic, because native matrices are used. But the significance decreases, if one uses the extended approach with indirect dependencies via more edges. Nevertheless, the results of this pilot study indicate the operability of the presented approach. In order to use this approach by practitioners in industry, e.g. a plant developer, a software-based application is needed to hide the arithmetical operations and implement a suitable clustering algorithm.

5. Conclusion

Although system lifecycle properties play an increasing important role in planning and operation of large-scale technical systems, there are no consistent approaches to systematically structure system components regarding Ilities. We presented an approach, based on established matrix operations combined with commonality analysis to derive so called Ility-modules. These clusters represent parts of the system (a collection of different elements) with a high conformity in their Ility-sets. Exemplified by a car assembly line we demonstrated that process modules can be derived regarding the related flexibilities. Implications on feasible forms of organization emphasize the practical benefit. From an organizational perspective even bigger potentials can be unlocked, by extending our method to influencing parameters of Ilities, because such parameters are used in the daily business of system planers and developers. Consequently, this would be a reliable database with high acceptance in the practical application. Further research on this aspect and applications in additional case studies to validate the approach are ongoing.

Appendix A. Matrix-based methods

As depicted in figure A1, a MDM consists of different domains and the relation between these domains (cp. Lindemann et al.[20]).

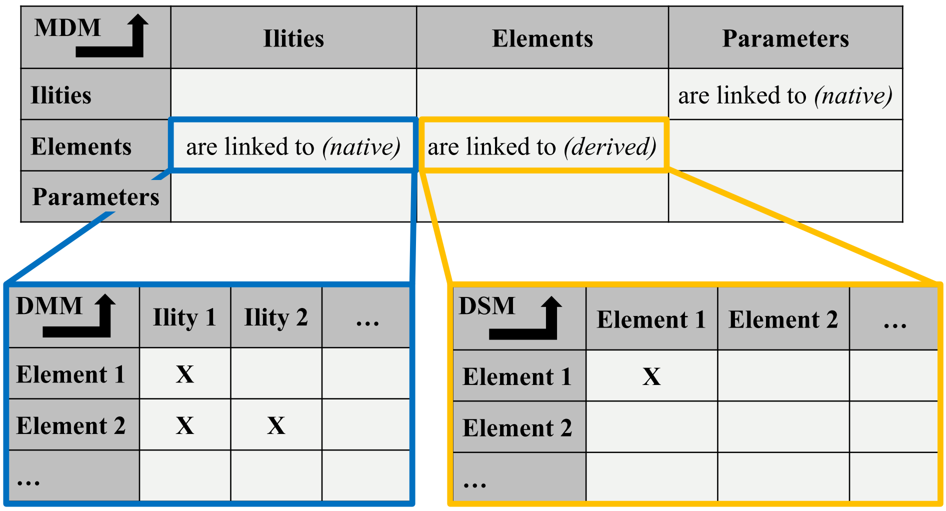

Fig. A1. Link between the used matrix types: MDM, DMM, DSM

In our case, the MDM consists of three domains: System elements, Ilities and parameters. Each cell of the MDM describes the verbalized dependency between two domains. The individual dependencies between elements of such domains are modeled in further matrices (DMM and DSM). If one matrix consists of elements of two domains, it is called DMM (marked in blue). If the matrix links elements within one domain, it is called DSM (marked in orange). Native matrices include direct dependencies. Derived matrices cannot be filled with existing information. These dependencies have to be calculated from native matrices.

References

1. Matt, DT. (Re-)Design to Agility using the Concept of Organisational Periodicity. In: Zeah MF. 3rd International Conference on Changeable, Agile, Reconfigurable and Virtual Production (CARV 2009); 2009 Oct 5-7; Munich, Germany. Munich: Herbert Utz; c2009. p. 683-92.
2. Gunasekaran A, McGaughey R, Wolstencroft V. Agile Manufacturing: Concepts and Framework. In: Gunasekaran A. Agile Manufacturing the 21st Century Competitive Strategy. Kidlington: Elsevier Science Ltd.; 2001. p. 25-49.
3. Beskesea A, Kahramana C, Irani Z. Quantification of flexibility in advanced manufacturing systems using fuzzy concept. Int. J. Prod Econ. 2004;89(1):45-56.
4. Crawley E, de Weck OL, Eppinger S, Magee C, Moses J, Seering W, Schindall J, Wallace D, Whitney D. Engineering Systems Monograph. The Influence of Architecture in Engineering Systems [Internet]. 2004 Mar [cited 2013 Jul]. Available from: http://esd.mit.edu/symposium/pdfs/monograph/architecture-b.pdf
5. Browning TR, Honour EC. Measuring the Lifecycle Value of a System. In: Proceedings of the 15th Annual International Symposium of INCOSE; 2005 Jul; Rochester (NY), USA. c2005.
6. McManus HL, Richards MG, Ross AM, Hastings DE. A Framework for Incorporating “ilities” in Tradespace Studies. In: AIAA SPACE 2007 Conference & Exposition; 2007 Sep 18-20; Long Beach (CA), USA. c2007. p. 1-14.
7. Ross AM, Rhodes DH, Hastings DE. Defining Changeability: Reconciling Flexibility, Adaptability. Scalability. Modifiability, and Robustness for Maintaining System Lifecycle Value. Systems Eng. 2008;11(3):246-62.
8. DeWeck OL, Ross AM, Rhodes DH. Investigating Relationships and Semantic Sets amongst System Lifecycle Properties (Ilities). In: 3rd International Engineering Systems Symposium CESUN 2012; 2012 Jun 18-20; Delft, Netherlands. c2012.
9. Alexopoulos K, Papakostas N, Mourtzis D, Gogos P, Chryssolouris G. Quantifying the flexibility of a manufacturing system by applying the transfer function. Int J Comput Integ M. 2007;20(6):538-47.
10. G. Michalos, S. Makris, N. Papakostas, G. Chryssolouris. A Framework for Enabling Flexibility Quantification in Modern Manufacturing System Design Approaches. In: Duffie NA, Westkämper E. 44th CIRP International Conference on Manufacturing Systems. 2011 Jun 1-3; Madison, USA. Madison, Wisconsin: Omnipress; c2011.
11. Wiendahl H-P, ElMaraghy HA, Nyhuis P, Zäh MF, Wiendahl H-H, Duffie N, Kolakowski M. Changeable Manufacturing - Classification, Design and Operation. CIRP Ann-Manuf Techn. 2007;56(2):783-809.
12. Gentile PD. Theory of Modularity, a Hypothesis. Procedia Computer Science. 2013;20:203-9.
13. Schapiro SB, Henry MH. Engineering Agile Systems through Architectural Modularity. In: 2012 IEEE International System Conference (SysCon 2012) Proceedings; 2002 Mar 19-22; Vancouver, Canada. c2012. p.28-33.
14. Fixson SK. Modularity and Commonality Research: Past developments and Future Opportunities. Concurrent Eng-Res. 2007;15(2):85-107.
15. Simpson TW, Jiao J, Siddique Z, Hölttä-Otto K. Advances in Product Family and Product Platform Design. New York: Springer; 2014.
16. Scholz-Reiter B, Philipp T, de Beer C, Windt K, Freitag M. [Influence of Structural Complexity on the Application of Self-Controlling Logistic Processes]. Einfluss der strukturellen Komplexität auf den Einsatz von selbststeuernden logistischen Prozessen. In: Pfohl H-C, Wimmer T, editors. [Controlling Logistic Systems – On the Way to Self-Control]. Steuerung von Logistiksystemen - auf dem Weg zur Selbststeuerung. Konferenzband zum 3. BVL-Wissenschaftssymposium Logistik. Hamburg: Deutscher Verkehrs-Verlag; 2006. p. 11-25. German.
17. Fricke E, Schulz AP. Design for changeability (DfC): Principles to enable changes in systems throughout their entire lifecycle. Syst Eng. 2005;8(4):342-59.
18. Biedermann W, Lindemann U. Designing Consistent Structural Analysis Scenarios. ICED 11 – 18th International Conference on Engineering Design - Impacting Society Through Engineering Design; 2011 Aug 18; Copenhagen, Denmark. c2012. p.133-44.
19. Schoettl F, Bauer W, Lindemann, U. Design for System Lifecycle Properties – Support of Planning Processes by Modularization of Stakeholder Networks. In: Scheurmann E, Maurer M, Schmidt D, Lindemann U. Reducing risk in innovation. Proceedings of the 15th International DSM Conference; 2013 Aug 29-30; Melbourne, Australia. c2013. p. 125-32.
20. Lindemann U, Maurer M, Braun T. Structural Complexity Management. An Approach for the Field of Product Design. Berlin: Springer; 2009.
21. Aggarwal CC, Reddy CK. Data Clustering. Boca Raton: CRC Press; 2014.