INTERNATIONAL DESIGN CONFERENCE – DESIGN 2024
https://doi.org/10.1017/pds.2024.65

Enhancing the IFM framework based on a meta-analysis of other design methods

Merlin Krüger ✉, Kilian Gericke and Stefan Zorn

University of Rostock, Germany

✉ merlin.krueger@uni-rostock.de

Abstract

The use of design methods across multiple design phases of the product development process often leads to inconsistency, the loss of transparency, and the rejection of design methods by practitioners. The authors of this work intend to develop a central modelling approach that supports consistency, based on the integrated function modelling (IFM) framework. Therefore, various design methods from the literature were examined for their techniques and content to identify indicators for supporting consistency. The results led to an enhancement of the IFM framework.

Keywords: functional modelling, requirements, design methods, model-based systems engineering (MBSE)

1. Introduction

The product development process (PDP) can be divided into the following design phases: Planning, conceptual design, system level design, and embodiment/detailed design (Blanchard and Fabrycky, 1998; Pahl *et al.*, 2007). The process is an interdisciplinary endeavour including e.g. mechanical engineering, mechatronic engineering, building design, or software design. Eisenbart (2011) has examined various design methodologies and the design states they address for the different disciplines across different phases of the PDP. Furthermore, the methodologies provide various design methods to reach these design states. An issue along the process is media inconsistencies that occur as different design states are modelled differently using different tools. Media inconsistencies may have been of less importance in the past, but in today's digital model-based systems engineering (MBSE) they lead to considerable complications and the rejection of design methods by practitioners. For this reason, the authors of this paper specifically examine specific design methods regarding content and techniques to avoid media inconsistencies in the development of a design method. The authors base their work on the Integrated Function Modelling (IFM) framework, a modelling approach developed by Eisenbart et al. (Eisenbart, 2013; Eisenbart et al., 2016) for function modelling of technical, moderately complex systems. The IFM framework has so far shown great potential in its application in terms of user-friendliness and the technical results achieved (Eisenbart *et al.*, 2015, 2016; Gericke and Eisenbart, 2017; Krüger *et al.*, 2020; Krüger and Gericke, 2022), but "making explicit the links between requirements and functions in the framework, however, will require additional research" (Eisenbart et al., 2016). The additional research carried out in this paper focuses on bridging the design states "requirements specification" and "product functionality" (Eisenbart and Gericke, 2011). Therefore, a literature review was conducted based on the research question of what content is described in commonly proposed design methods and what modelling approaches are included. Furthermore, these design methods were precisely examined to gather inspiration for an enhancement of the IFM framework.

2. Design methods proposed by methodologies of mechanical engineering and interdisciplinary engineering

This section begins with a brief clarification of the term "design method" as the terminology differs in literature. The definition serves as a basis for analysing methodologies across different disciplines in terms of which design methods they contain. The analysis presented in this section takes up the work of Eisenbart et al. (2011) - "design states addressed in methodologies across disciplines". But it will further focus on the design methods of the disciplines “mechanical engineering” and “mechatronics/ interdisciplinary approaches”. Furthermore, it compares the specific design methods only for the two specific design states "requirements specification" and "product functionality". The comparison also includes the proposed modelling approaches.

2.1. Comparing design methods recommended in design methodologies

Design methods are specifications *"of how a specified result is to be achieved. This may include specifications of how information is to be shown, what information is to be used as inputs to the method, what tools are to be used, what actions are to be performed and how, and how the task should be decomposed and how actions should be sequenced"* (Gericke *et al.*, 2017).
The following key questions, which can guide a comparison, and the key elements that help to answer them can be derived from this definition:

- How is information shown? -> Representation
- What information is used? -> Entities
- What tools are used? -> Techniques
- What actions are performed and how? -> Procedure
- How is the task decomposed? -> Decomposition
- How are the actions sequenced? -> Sequence

As the descriptions of the procedures are not reasonably comparable, they are not considered. In addition, the entities are not taken into account in the first step, but are added later on (see Chapter 3). Thus, the methodologies pre-selected by Eisenbart are further analysed using the following key elements:

- Representation (Design Model)
- Decomposition
- Techniques
- Sequence

The methodologies considered by Eisenbart et al. describe design activities and related design states. They provide recommendations of suitable methods but do usually not discuss related modelling approaches in detail. Focussing on the two considered design states, i.e. "requirements specification" and "product functionality", a distinction between approaches of mechanical engineering and interdisciplinary/mechatronic engineering is not necessary (e.g. Pahl and Beitz (2007), Ehrlenspiel, Hubka (1982), Tjalve (1978), VDI 2221 (1993), and VDI 2206 (2004)). They recommend very similar methods and related representations (see Fig. 1).
The design state “requirement specification” is usually modelled time-independent, in parallel and undecomposed by using a textual representation. Therefore, the use of requirements lists is recommended in many of the considered methodologies. The design state “product functionality” is typically represented using a graphical modelling technique to represent the given information and its dependencies. In most cases, these dependencies are time-independent but sequential. Some approaches propose a hierarchical decomposition instead of a sequential structure. Hence, from requirements specification to product functionality, there is a change from “textual, grouping, parallel” to “graphical, clustering, sequential”.
The considered methodologies provide similar methodical activities for the two considered design states (Fig. 1). Furthermore, these methodologies focus on the PDP overall and not on the specific use of individual methods or modelling approaches. As a result, almost all of the methodologies analysed lead to media inconsistencies across the individual design phases.

Design state	Key elements	Mechanical Engineering					Mechatronic/Interdisciplinary Approaches		
		Pahl and Beitz (2007)	Ulrich and Eppinger (2008)	Ehrlenspiel (2011)	Hubka (1982)	Tjalve (1978)	VDI 2221 (1993)	VDI 2206 (2004)	Salminen and Verho (1989)
Requirement Specification	Representation	requirements list	DSM	requirements list	requirements list	requirements list	requirements list	requirements list	requirements list
	Information Sequence	Time-independent, Parallel	Time-independent, Parallel	Time-independent, Parallel	Time-independent, Parallel	Time-independent, Parallel	Time-independent, Parallel	Time-independent, Parallel	Time-independent, Parallel
	Decomposition	Grouping	Level of Detail, Tree	Grouping	Grouping	Grouping	Grouping	Grouping	Grouping
	Technique	Text	Matrix	Text	Text	Text	Text	Text	Text
Product Functionality	Representation	Function Structure	DSM	Function Structure, Function List	Process structure	Function Structure	Function Structure	Function Structure	Function-Tree
	Information Sequence	Time-Independent, Sequential	Time-independent, Parallel	Time-independent, parallel or sequential	Time-dependent, sequential	Time-independent, sequential	Time-Independent, Sequential	Time-Independent, Sequential	Time-independent, sequential
	Decomposition	Clustering	Level of Detail	Clustering, Grouping, Tree	Clustering	clustering, tree	Clustering	Clustering	Tree
	Technique	Graph	Matrix	Graph, Text	Graph	Graph	Graph	Graph	Graph

Figure 1. Results for comparable design methods in methodologies

Ulrich and Eppinger's approach is different (Ulrich and Eppinger, 2008) by proposing the use of design structure matrices (DSM). These could be used via multiple design states when using multi-domain matrices (MDM).

Reflecting the results of the analysis with the aim of gathering inspiration for an enhancement of the IFM framework, the analysis was not satisfactory. According to an initial assessment, the approaches examined were too general. For this particular reason, a second literature search was conducted focussing on design methods that aim to bridge multiple design states without media inconsistency.

2.2. Comparing design methods without media inconsistency

Ten additional design methods that intend to support consistency across several design phases were found during the literature search. However, only four of these could be identified that enable consistency from the first to the last phase. A further five design methods support consistency across several phases, but not across all of them. A tenth design method supports the PDP in general without referring to a specific modelling approach at all.

The following five approaches avoid media inconsistencies:

- Design Structure Matrix (DSM), Multi-Domain matrices (MDM) (Ulrich and Eppinger, 2008),
- System Modelling Language (SysML) (Friedenthal *et al.*, 2015; Weilkiens, 2008),
- Requirements-Functions-Logical-Physical (RFLP) approach (Kleiner and Kramer, 2013), and
- System State Flow Diagram (SSFD) (Yildirim et al., 2017),
- Suh's Axiomatic Design (1998).

DSMs and MDMs (Ulrich and Eppinger, 2008) can be used to model complex data. They are particularly suitable for dense data (Ramos *et al.*, 2020; Schoenenberger *et al.*, 2021) and show the dependencies between different entities. However, there are limits to the number of entities that can be modelled simultaneously. In addition, the description of the entities is often abstract, which makes it much more difficult to read and handle the models in practice. As a result, DSMs/MDMs lack user-friendliness, which means that in practice they are usually only used for specific purposes, e.g. modelling component dependencies. Bridging all design phases through the concrete modelling of all necessary entities is therefore not the aim of the application of DSMs/MDMs.

SysML(Friedenthal et al., 2015; Weilkiens, 2008) is a much more readable approach, especially when the relationships between entities are important and the data is sparse (Ramos *et al.*, 2020; Schoenenberger *et al.*, 2021). Several publications discuss the limitations of SysML, including Eisenbart comparing SysML and the IFM framework (Eisenbart *et al.*, 2015). Among other things, the handling of the modelling language and its elements is often a hurdle for practical application. In addition, graph-based approaches are not well suited for modelling dense data (Ramos *et al.*, 2020; Schoenenberger *et*

https://doi.org/10.1017/pds.2024.65 Published online by Cambridge University Press

al., 2021), which has a negative impact on the later phases of the PDP. The biggest hurdle, however, is that although SysML can theoretically be used to model all the necessary entities in one approach; in practice, several parallel representations are required.

The Requirements-Functional-Logical-Physical (RFLP) approach (Kleiner and Kramer, 2013) is an interesting means to bridge different design phases of the PDP. On the one hand, RFLP connects the information between the design states in a centralized approach, on the other hand, the approach provides different models for the specific domains, which is a similar method to SysML. The connections of the domains are based on the supporting software and not the approach itself. However, in this case, the software is a central component of the approach. Consequently, RFLP also does not directly fulfil the requirement of an approach without media inconsistencies.

The system state flow diagrams (SSFD) (Yildirim et al., 2017) presumes the requirements specifications and relates functional requirements to attribute values that are further used as describing entities within the modelling approach. SSFD therefore can be used to support the design state “requirements specification”. Furthermore, SSFD does bridge multiple design states, especially for conceptual design and system design. However, SSFD simply lacks the entities needed to model the product development process and all its associated phases more holistically.

The same applies to axiomatic design (Suh, 1998), where the system architecture is modelled in separate domains through only functional requirements (FRs), design parameters (DPs) and process variables (PVs). While there is no direct distinction between design phases at all (also see (Weber, 2005)), the different domains are somehow connected, e.g. mapping the *"FRs of the functional domain into the physical domain"* (Suh, 1998) to conceive a design embodiment and to identify the DPs.

The analysed design methods provide interesting possibilities for modelling different entities in the course of the PDP, but there is a lack of either what needs to be modelled to guarantee the applicability of an approach across the different phases, or how it needs to be modelled to provide the necessary user-friendliness that an approach needs to be widely accepted in practice. The IFM framework has so far shone through its user-friendliness, i.e. how it has to be modelled. Hence, the main focus for an extension of the IFM framework is on what needs to be modelled. This requires an analysis of the entities to be modelled in the respective design phases. For such an analysis, it is not necessary that the modelling of the entities of the analysed methods is bridged across the design phases. Hence, all design methods considered in this chapter are analysed for their modelled entities, i.e. the design methods without media inconsistencies presented above and also the design methods with media inconsistencies that have not been presented so far. Therefore, the design methods with media inconsistencies are roughly explained in terms of modelling across the design phases in the following.

The different elements of the SAPPhIRE-Model (Chakrabarti and Srinivasan, 2009) can be used indirectly as a link to the requirements, but there is no direct requirements modelling. *„The model in its current form can support only conceptual and early embodiment phases and needs to be extended to support other detailed phases”* (Chakrabarti and Srinivasan, 2009). Another distinctive feature is the use of symbols and figures as a language within the model, which is a useful tool also used in SysML (Friedenthal et al., 2015).

Wong and Wynn’s (2022) Detailed Design Model (DDM) is based on a 3D CAD model and results in the Adaptive Redesign Method (ARM). It is primarily used for variant design and focuses on the conceptual design phase and system-design phase. It does not directly implement the embodiment design phase for the considered design process, but it is based on existing CAD models, hence it presumes the embodiment design phase of previous systems.

Other design methods deal specifically with change processes in the design process. These changes often occur at the system design level or the detailed design level but result in an iteration for a specific task. Hence, change management is the very beginning of an individual design process. However, the design methods themselves focus on the interactions and dependencies within the system concerning the design process overall. Nevertheless, it is important to analyse the entities that are relevant when using these design methods, as they can be the starting point for individual design processes. Thus, these design methods might provide a starting point for structuring modelling approaches that can bridge multiple design phases.

https://doi.org/10.1017/pds.2024.65 Published online by Cambridge University Press

Weber's (2005) characteristics-properties-modelling (CPM) and property-driven-development (PDD) are approaches to modelling products and PDPs but without provided design models or design methods. Weber mentions characteristics, properties and their derivation. This point of view is similar to Suh (1998) or Hubka (1988) and provides interesting perspectives and entities (characteristics, properties) that have already served as a basis for earlier work on the IFM framework (Krüger *et al.*, 2023).
The change prediction method (CPM) by Keller (2005) also focuses on engineering change and provides support by using DSMs, change risk plots, or propagation networks as basic representations. Hamraz and Clarkson's function–behaviour–structure (FBS) linkage method (Hamraz and Clarkson, 2015) enhances CPM with an FBS scheme and allows more detailed modelling and analysis of engineering changes. One difference in modelling lies in the decomposition, which in CPM corresponds more to a defined level of detail, whereas FBS enables a kind of tree structure. Furthermore, CPM provides various modelling techniques while FBS is primarily based on graphical modelling.
Attribute-dependency-graphs (ADGs) represent dependencies between attributes that further can be defined as design variables or design parameters. *"System engineers can use ADGs for systems design. They are built for a design problem in a specific context"* (Rötzer *et al.*, 2022). The dependencies are related to functional requirements (FRs). Hence, FRs are somehow considered, similar to the SSFD (Yildirim *et al.*, 2017). ADGs help the designer to identify how a change on the component level can influence the system level. *"This can provide transparency during the design of a complex system"* (Rötzer et al., 2022).
Several methods were presented and their advantages and disadvantages for bridging design states were implied. The core objective of this work is to enhance the IFM framework, so it can be used for continuous modelling across all phases of the PDP. Therefore, the next step is to examine all the design methods investigated in terms of the entities they contain, as well as the phases of the PDP in which these entities are preferentially considered.

3. Modelled entities

In a first step, all design methods of the previous chapter were examined to determine which entities they essentially represent based on the compared literature, as shown in figure 2.

Approaches	Representation	Design Phases				Entities												
		Planning	Conceptual Design	System Level Design	Embodiment/ Detailed Design	Use Cases	Transform. Processes	Interaction Processes	Functions	Actors	Operands	States	Design Parameters	Functional Requirements	Non-Functional Requirements	Time	Risk	Organs
Pahl and Beitz, Ehrlenspiel, Tjalve, VDI 2221, ...	Requirements list	x				x				x	x	x		x	x			
	Function Structure		x						x	x	x					(x)		
	Morphological Matrix		x			x			x	x								
Hubka (1984)	Process Structure		x			x	x			x	x	x				x		
Ulrich and Eppinger (2008)	DSM	(x)	x	x	(x)			x	x	x	x			x				
Yildirim (2017)	SSFD	x	x	x	x	x	x		x	x		x	x			x		
Kleiner and Kramer (2013)	RFLP	x	x	x	x			x	x	x	x	x		x				
Weilkiens (2008)	SysML	x	x	x	(x)	x	x	x	x	x	x	x	x	x		x		
Suh (1998)	*Axiomatic Design*	x	x	x	x					x			x	x				
Wong and Wynn (2022)	DDM/ARM		x	x				x	x	x	x	x						
Chakrabarti (2009)	SAPPhIRE-Model		x	x	(x)		x	x		x	x	x				x		x
Eisenbart (2016)	IFM Framework		x	x		x	x	x		x	x	x				x		
Keller (2005)	Change Risk Plot, Propagation Network			x	x			x		x							x	
Hamraz and Clarkson 2015	Product linkage model/FBS			x	x			x		x							x	
Rötzer (2022)	ADG			x				x					x	(x)				

Figure 2. Entities represented in different design phases

The entities of the IFM framework served as starting point for this, which are "use cases", "transformation processes", "interaction processes", "states", "actors" and "operands" (Eisenbart *et al.*, 2016). In addition to these entities, other entities were identified in the design methods analysed in chapter 2, which are "functions", "functional requirements", "non-functional requirements", "design

https://doi.org/10.1017/pds.2024.65 Published online by Cambridge University Press

parameters", "time", "risks" and "organs". At this point, it should be added that it is possible not to have recognized entities. Secondly, it was noted in which phases of the PDP the design methods are generally applied, again based on the respective literature and its recommendation for the application of the considered approach. Because some approaches provide similar representations as already discussed in section 2, they were summarized (see Pahl and Beitz, Ehrlenspiel, Tjalve, etc.).
In a third step, the previous results were superimposed as shown in figure 3. The number behind the dash shows how many of the analysed approaches consider modelling a design phase, e.g. "3/**6**". Hence, the number behind the dash is repeated for the whole row. The number before the dash shows, how many of these approaches consider a related entity, e.g. "**3**/6". For instance, six approaches consider the modelling of the planning phase, but only three of them consider the modelling of "interaction processes" during this design phase. This ultimately provides a percentage indication of how "important" the consideration of an entity can be for a design phase, e.g. the number in brackets "3/6" (**50%**). The higher the percentage, the more "important" an entity appears to be. Figure 3 is sorted by frequency, from left "often used" to right "rarely used".

Design Phases	Entities												
	Actors	Interaction Processes	Operands	States	Functional Requirements	Functions	Time	Transform. Processes	Design Parameters	Use Cases	Risk	Organs	Non-Functional Requirements
Planning Phase	6/6 (100)	3/6 (50)	4/6 (66)	4/6 (66)	5/6 (83)	4/6(66)	2/6 (33)	2/6 (33)	3/6 (50)	3/6 (50)	0	0	1/6 (16)
Conceptual Design	11/11 (100)	6/11 (55)	8/11 (72)	7/11 (63)	4/11 (36)	7/11 (63)	6/11 (55)	5/11 (45)	3/11 (27)	5/11 (45)	0	1/11 (9)	0
System Level Design	10/11 (90)	9/11 (81)	6/11 (55)	6/11 (55)	5/11 (45)	5/11 (45)	4/11 (36)	4/11 (36)	4/11 (36)	3/11 (27)	2/11 (18)	1/11 (9)	0
Embodiment Design	8/8 (100)	6/8 (75)	4/8 (50)	4/8 (50)	4/8 (50)	3/8 (38)	3/8 (38)	3/8 (38)	3/8 (38)	1/8 (13)	2/8 (25)	1/8 (13)	0

100 to 91	90 to 71	70 to 51	50 to 40	39 to 26	25 to 10	%

Figure 3. Frequency of the representation of entities in the PDP

Figure 3 focused on the "importance/frequency" of entities according to the compared design methods. The next step is to focus on the design phases in order to discuss in which design phase an entity should preferably be considered. To this end, the entities were reorganized according to the design phase in which they have the "highest" importance (Fig. 4). If an entity has approximately the same importance for multiple phases, then several markers have been set for this. However, it can be deduced from the percentage rating in which phase an entity is "primarily" represented - this is then the "main use". Further, entities with a percentage of zero for individual design phases have been declined from the list as they appear to not be expedient for providing consistency through all design phases. The sorting in figure 4 is again from left to right and additionally differentiates between the individual design phases. The green cells with markers represent the main use, the blank cells with markers represent "secondary uses".

Design Phases	Entities									
	Actors	Functional Requirements	States	Functions	Design Parameters	Use Cases	Operands	Time	Transform. Processes	Interaction processes
Planning Phase	x	x	x	x	x	x	x			
Conceptual Design	x		x	x		x	x	x	x	
System Level Design	x	x	x				x			x
Embodiment Design	x	x			x					x

Figure 4. Entities according to most frequently used in design phases (main use of entities)

Reflecting the results presented in figure 3 and figure 4, it can be indicated how to enhance the IFM framework. An indication for the IFM framework can be as follows:

1. The importance/frequency of entities indicates "what" could be added to the IFM framework - Entities that are not part of the IFM framework yet, e.g. "functions" or "design parameters"
2. The main use and secondary uses of entities on the one hand indicate "where" within the IFM framework the focus could be placed during modelling a specific design phase, and on the other hand, indicate which entities can be added to improve modelling a specific design phase while

https://doi.org/10.1017/pds.2024.65 Published online by Cambridge University Press

also improving consistency, e.g. actors, functional requirements, design parameters and interaction processes can support modelling embodiment design.

The results of the previous analyses served as a basis for an enhancement of the IFM framework.

4. Enhancing the IFM framework

The original IFM framework is explained in detail in Eisenbart's work (Eisenbart *et al.*, 2016). This paper focusses on changes to the original form of the IFM framework.

4.1. Implementing additional entities into the IFM framework

Krüger et. al. (2023) already added functional requirements (FRs) to the existing entities. In his work, he mentions two different types of requirements in relation to the work of Weber (2005), the indirect changeable properties as non-functional requirements (NFRs), and the directly changeable characteristics as functional requirements. The authors took up the work of Krüger et. al. (2023) and first enhanced the consideration of "directly changeable characteristics" - As characteristics are described to be directly changeable, there are those that relate to transformation processes and those that relate to interaction processes. Transformation process dependent characteristics are further still described as characteristics, but interaction process dependent characteristics are now described as Design Parameters (DP). Thus, the authors distinguish between FRs that relate to transformation processes (TFRs) and FRs that relate to interaction processes (IFRs). The TFRs are further on modelled in the state view (also see (Krüger *et al.*, 2023)), and the IFRs are now modelled in the interaction view (Fig. 5b). Figure 5a shows the arrangement of views in an overview of the enhanced IFM framework and figure 5b shows an exemplified filled IFM framework.

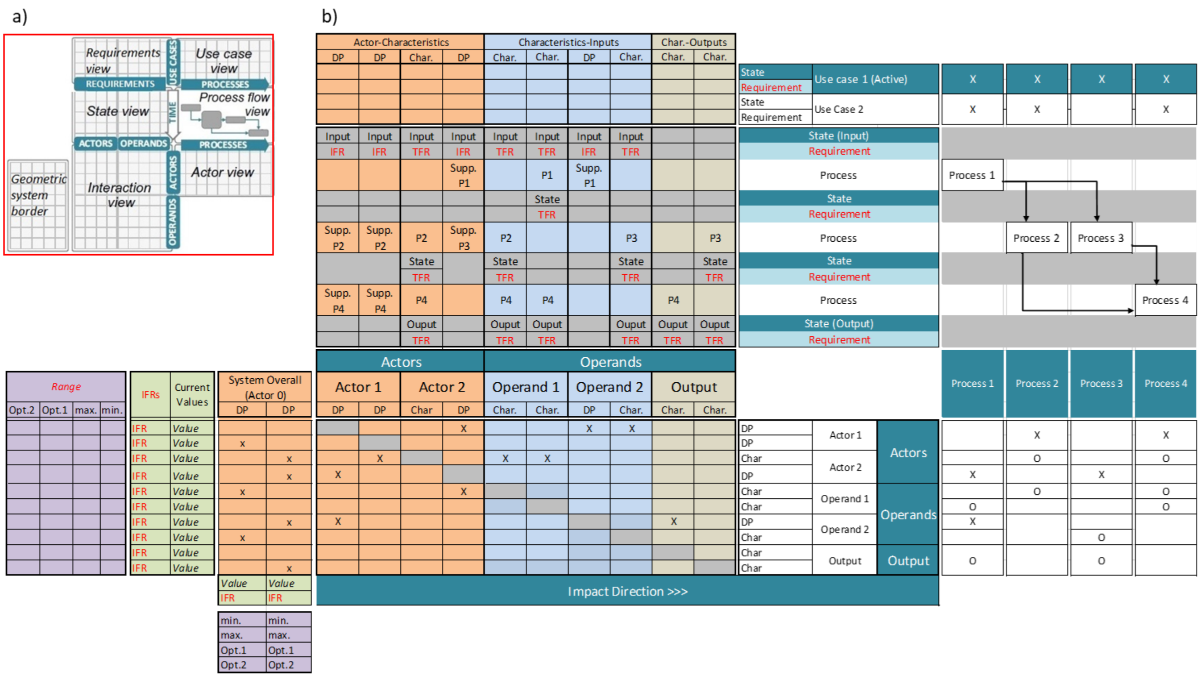

Figure 5. a) Overview of the enhanced IFM framework; b) Exemplarily filled enhanced IFM framework

Taking a coffee machine as an example (Eisenbart *et al.*, 2016; Krüger *et al.*, 2023), a characteristic "temperature" of an operand "water" could change due to a process "heat water", while the design parameters "length, width, depths" of the "heating system" remain the same. A TFR could then describe, what temperature is to be reached within the transformation process, while the IFRs could describe maximum, minimum or optional values for length, width and depth of the heating system.

Furthermore, the DPs are a powerful means to describe a "geometric system border". Therefore, and also to relate the dependent IFRs, the interaction view has been expanded by a separate area as shown

https://doi.org/10.1017/pds.2024.65 Published online by Cambridge University Press

at the bottom left of figure 5a. Within this area, it is also possible, to describe DPs not only of the single actors but of the system overall. This enables modelling dependencies throughout the considered system by modelling the geometric system border of the system. To describe the "functional system border" of the system, the authors introduced a new view to the top left of the framework (see Fig. 5a) - The requirements view. This view can support early phase design by basically describing the in- and outputs of use cases.

4.2. The application of the enhanced IFM framework

Even in its original form, the IFM framework tended to quickly become too complex. Therefore, it can be tailored to support the user focus on specific content. The tailoring allows rows, columns or even complete views to fade out. When adding the expansions, the authors paid particular attention to ensuring that tailoring is still possible. The following section highlights additional aspects of focussing, tailoring and using the enhanced IFM framework for specific design phases.
When applying the enhanced IFM framework, it can be useful to rather focus on specific content and views than to focus on the model overall - edit the content of a specific view, change the focus to the next view, edit the content, and iteratively repeat the process. The overall perspective can then be used to harmonize the content of the individual views. Figure 6 points out the shift in focus in relation to the design phase under consideration marked in red. The shift in focus as shown in figure 6 is primarily a recommendation illustrating only one possible scenario for using the enhanced IFM. The completion of the model is iterative, whereby each view is revised several times. Furthermore, the "geometric system border" is only shown in phase 4 to indicate the tailoring of parts of the framework due to the considered design phase.

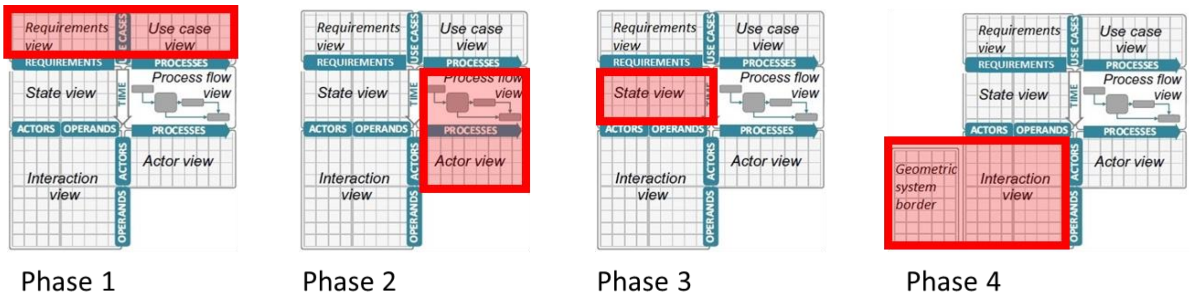

Figure 6. Focus on view for different design phases

In phase 1, the designer starts filling the requirements view and use case view by noting the use cases, in- and outputs of use cases and related requirements as TFRs. After that, transformation processes (TP) that fulfil the use cases can be noted. The TPs are allocated to the use cases, whereby it is possible that a TP is used for multiple use cases.
In phase 2, the designer focusses on the process flow view and actor view. The TPs, noted in phase 1, get specified to a use-case-dependent transformation process flow by detailing the sequence and duration of TPs. Next, the actors are defined and allocated to the TPs whether they have an impact on a TP or are affected by a TP. At this point, the user is given a means of self-checking: if a TP is used in several use cases, but the inputs and outputs of the TP differ for the respective use cases, it is advisable to differentiate the TP into multiple TPs.
In phase 3, the previous step continues while specifying the states of characteristics in the state view. Herein, the characteristics of all actors and operands are described as result to changes due to the TPs.
In phase 4, the interactions between actors and operands are marked in the interaction view. Result to the enhanced IFM framework, it is possible to describe them on a very high level of detail as interactions between characteristics and design parameters. Furthermore, the geometric system border is described in this phase.

5. Discussion and conclusion

In order to improve consistency of the IFM framework, a literature review was conducted. The design states addressed in methodologies across disciplines by Eisenbart et. al. (2011) served as starting point.

https://doi.org/10.1017/pds.2024.65 Published online by Cambridge University Press

An analysis of the methodologies has shown that the recommended design methods are usually associated with media inconsistencies throughout the design phases. As a result, the analysis was expanded to include ten additional design methods that focus on bridging multiple design phases without media inconsistencies. Afterwards, all design methods of the different analyses were examined for their modelled entities and a frequency analysis was carried out in relation to the individual design phases. Based on the entity-frequency and former work (Krüger *et al*., 2023), the IFM framework was enhanced by interaction process dependent function requirements (IFR), transformation process dependent functional requirements (TFR), characteristics and design parameters (DP). In addition to the implementation of these "entities", a new view was introduced (requirements view) to primarily bridging early phase design. The interaction view has been expanded for embodiment design. Furthermore, it is now possible to describe a functional system border and a geometric system border of a system. The enhanced IFM framework is now intended to improve consistency in the PDP.

In a design project, for instance, the starting point varies, as does the information of the project and the objects involved (systems, parts, assemblies, etc.). However, a designer does not start from scratch. Depending on the type of design (variant design, adaptation design, new design), individual objects of the project that serve as templates are to be replaced by the desired objects. This firstly leads to a change in the object-specific characteristics and DPs and secondly can lead to a necessary review of the functional system borders and geometric system borders. Therefore, the enhanced IFM framework should enable that the object-specific characteristics and DPs can be entered and that the dependent functional and geometric system borders can be checked. In this way, the enhanced IFM framework is intended to be used both to explore early phase design in the PDP and to determine geometry dependencies in late phase design. Furthermore, the geometric system border could be used to bridge variant management in computer aided design (CAD), i.e. variable tables are used in CAD to define values for variables and store equations. The values correspond to the range of the geometric system border (min., max., or optional values) and the equations are based on the interactions between characteristics and DPs that are modelled in the interaction view of the enhanced IFM framework.

It is essential to test the new features of the enhanced IFM framework in a real environment design project to validate its applicability. In particular, usability should be tested and the potential for simplifying the modelling approach should be discussed in a subsequent step. Another point to be discussed is the modelling of DPs. As DPs relate to actors and operands, they are used in the interaction view, but not all of them are needed to be modelled in the state view and the requirements view. The authors decided to fully model DPs through all views, but future work should focus on tailoring aspects, especially for needed content in specific views, because the enhanced IFM framework gets even more quickly too complex than the original IFM framework.

References

Blanchard, B. and Fabrycky, W. (1998), "Systems Engineering and Analysis".

Chakrabarti, A. and Srinivasan, V. (2009), "SAPPHIRE – AN APPROACH TO ANALYSIS AND SYNTHESIS", International Conference of Engineering Design, ICED'09, No. August, pp. 417–428.

Eisenbart, B. (2013), "Der Integrated Function Modelling (IFM) Framework", No. January.

Eisenbart, B. and Gericke, K. (2011), "A Framework for Comparing Design modelling approaches across disciplines", International Conference of Engineering Design, No. January, pp. 1–12.

Eisenbart, B., Gericke, K., Blessing, L.T.M. and McAloone, T.C. (2016), "A DSM-based framework for integrated function modelling: concept, application and evaluation", Research in Engineering Design, Springer London, Vol. 28 No. 1, pp. 25–51, https://dx.doi.org/10.1007/s00163-016-0228-1.

Eisenbart, B., Mandel, C., Gericke, K. and Blessing, L. (2015), "Integrated function modelling: Comparing the IFM framework with SYSML", Proceedings of the International Conference on Engineering Design, ICED, Vol. 5 No. DS 80-05.

Friedenthal, S., Moore, A. and Steiner, R. (2015), "A Practical Guide to SysML".

Gericke, K., Eckert, C. and Stacey, M. (2017), "What do we need to say about a design method?", Proceedings of the International Conference on Engineering Design, ICED, Vol. 7 No. DS87-7, pp. 101–110.

Gericke, K. and Eisenbart, B. (2017), "The integrated function modeling framework and its relation to function structures", Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM, Vol. 31 No. 4, pp. 436–457, https://dx.doi.org/10.1017/S089006041700049X.

https://doi.org/10.1017/pds.2024.65 Published online by Cambridge University Press

Hamraz, B. and Clarkson, P.J. (2015), “Industrial evaluation of FBS Linkage – a method to support engineering change management”, Journal of Engineering Design, Vol. 26 No. 1–3, pp. 24–47, https://dx.doi.org/10.1080/09544828.2015.1015783.

Hubka, V. and Eder, W. (1982), Principles of Engineering Design.

Hubka, V. and Eder, W.E. (1988), Theory of Technical Systems : A Total Concept Theory for Engineering Design, Springer, Berlin.

Keller, R., Eger, T., Eckert, C.M. and Clarkson, P.J. (2005), “Visualising change propagation”, Proceedings ICED 05, the 15th International Conference on Engineering Design, Vol. DS 35, pp. 1–12.

Kleiner, S. and Kramer, C. (2013), “Model Based Design with Systems Engineering Based on RFLP Using V6”, Proceedings of the 23rd CIRP Design Conference, pp. 93–102, https://dx.doi.org/10.1007/978-3-642-30817-8_10.

Krüger, M.F. and Gericke, K. (2022), “Adaptation of the Integrated Function Modelling Framework”, Proceedings of the 24th International DSM Conference, pp. 29–37, https://dx.doi.org/10.35199/dsm2022.04.

Krüger, M.F., Zorn, S. and Gericke, K. (2023), “COMBINING FUNCTION MODELLING AND REQUIREMENTS MODELLING WITH THE IFM FRAMEWORK”, International Conference on Engineering Design, No. JULY, pp. 24–28, https://dx.doi.org/10.1017/pds.2023.99.

Krüger, M.F., Zorn, S., Wichmann, R.L. and Gericke, K. (2020), “The Application of the IFM Framework and FIDD Method on an Industrial Cigarette Filter Maker”, No. September, pp. 10–10, https://dx.doi.org/10.35199/dsm2020.19.

Pahl, G., Beitz, W., Feldhusen, J. and Grote, K.H. (2007), Engineering Design : A Systematic Approach, edited by Wallace, K. and Blessing, L., 3. Edition., Springer Vieweg.

Ramos, G., Aguiar, A.P. and Pequito, S. (2020), “Structural Systems Theory: an overview of the last 15 years”, pp. 1–79.

Rötzer, S., Schweigert-Recksiek, S., Thoma, D. and Zimmermann, M. (2022), “Attribute dependency graphs : modelling cause and effect in systems design”, Design Science, Vol. 8, https://dx.doi.org/10.1017/dsj.2022.20.

Schoenenberger, L., Schmid, A., Tanase, R., Beck, M. and Schwaninger, M. (2021), “Structural Analysis of System Dynamics Models”, Simulation Modelling Practice and Theory, Elsevier B.V., Vol. 110 No. April, p. 102333, https://dx.doi.org/10.1016/j.simpat.2021.102333.

Suh, N.P. (1998), “Axiomatic Design Theory for Systems”, Research in Engineering Design - Theory, Applications, and Concurrent Engineering, Vol. 10 No. 4, pp. 189–209, https://dx.doi.org/10.1007/s001639870001.

Tjalve, E. (1978), “Systematische Formgebung für Industrieprodukte”.

Ulrich, K.T. and Eppinger, S.D. (2008), Product Design and Development, McGraw-Hill Higher Education.

“VDI 2206 - Design methodology for mechatronic systems”. (2004), .

“VDI 2221 - Systematic Approach to the Design of Technical Systems and Products”. (1993), .

Weber, C. (2005), “CPM/PDD – an Extended Theoretical Approach To Modelling Products and Product Development Processes”, Proceedings of the 2nd German-Israeli Symposium on Advances in Methods and Systems for Development of Products and Processes, pp. 159–179.

Weilkiens, T. (2008), Systems Engineering with SysML/UML, Morgan Kaufmann, Amsterdam.

Wong, F.S. and Wynn, D.C. (2022), “A systematic approach for product modelling and function integration to support adaptive redesign of product variants”, Research in Engineering Design, Springer London, No. 0123456789, https://dx.doi.org/10.1007/s00163-022-00401-3.

Yildirim, U., Campean, F. and Williams, H.U.W. (2017), “Function modeling using the system state flow diagram”, pp. 413–435, https://dx.doi.org/10.1017/S0890060417000294.

https://doi.org/10.1017/pds.2024.65 Published online by Cambridge University Press