INTERNATIONAL DESIGN CONFERENCE - DESIGN 2016
Dubrovnik - Croatia, May 16 - 19, 2016.

INTEGRATED PRODUCT MODELLING THROUGH IFM-CPM/PDD

B. Eisenbart, Y. I. Khan and A. J. Qureshi

Keywords: functional modelling, integrated product development, product modelling, computational design synthesis (CDS)

1. Introduction

Determining a suitable solution to a given design problem is an interplay between reasoning about what is required and what particular combination of solution elements will provide for it [Chakrabrti and Bligh 2001], [Gero and Kannengiesser 2014]. This is typically referred to as 'co-evolution', i.e. a stepwise increase of knowledge about the problem in parallel to an emerging solution [Poon and Maher 1997]. It is an iterative process as in analysing, evaluating, and gradually synthesising the potential solution, some of its features or constraints may necessitate redefinition of the problem [Braha and Reich 2003]. This has to go hand-in-hand with continuously updating of models representing related information [Eckert et al. 2015]. These types of iterations, however, are not limited to conceptual design but carry through the entire development process. Frequently, embodiment and detail design work or eventual last-minute change requests by the customer demand (repeated) adaptation of the design concept. This again will almost inevitably require further adaptations to the solution to ensure consistent implementation of such conceptual changes [Dong, forthcoming]. This is hampered by the fact that coherent modelling of functions and solutions, which would link the conceptual and subsequent design stages (i.e. bridging between abstract/qualitative and increasingly quantitative information) is currently missing or has only limitedly been attained. A few modelling approaches have been proposed that already link function-oriented with initial system structural modelling, such as System Modeling Language (SysML, derived from Unified Modelling Language, UML [OMG 2012]) or Object-Process Methodology (OPM [Dori 1995]) and others. However, they are usually rather specific in their scope and/or reliant on a particular software environment and formalism [Stark et al. 2010]. This paper presents a first step towards a more generic approach using the relatively new Integrated Function Modelling (IFM) framework [Eisenbart et al. 2013] as vantage point which is to be coupled with the Characteristics-Properties Modelling (CPM)/Property-Driven Development (PDD) approach, respectively, by [Weber 2005, 2007]. Both approaches have been selected as they provide generic modelling and design frameworks capable of representing diverse information pertaining to multi-technology products as well as services, e.g. in combined offerings like product-service systems (PSS). The IFM framework uses simple, adaptable matrices for modelling functionality. PDD/CPM aims at consolidating a multitude of design methods and strategies into a common framework, eventually resulting in a comprehensive (and formalised) description of the system under consideration. IFM-CPM/PDD integration is eventually aspired to support the seamless and traceable transition from conceptual design into subsequent stages in practice.
The remainder of this paper is structured as follows: Sections 2 and 3 briefly introduce the IFM framework and the CPM/PDD approach. Section 4 then introduces the newly developed approach for

coupling both approaches conceptually, while Section 5 provides an example based on a simple PSS. Finally, Section 6 discusses the implications and directions for future research.

2. The IFM framework

The IFM framework is intended to provide designers with an integrated, cross-disciplinary approach for modelling system functionality, coupled with initial system structural modelling. Here, function is defined as an intended or already perceivable behaviour of a technical system intended to fulfil a task. In the framework, integration is facilitated through linking different types of contents prominently addressed in discipline-specific and cross-disciplinary function models (see [Eisenbart et al. 2013]). The respective information is presented in associated views, which are briefly described in Table 1. Their adjacent placement (see Figure 1) supports their parallel development and allows verification of their mutual consistency across the entire framework. Views are modular as they can be seamlessly added or omitted to ease flexible, demand-specific adaptation.

Table 1. Associated views in the IFM framework

View	Description
Process flow view	…qualitatively visualises the flow of sequential, parallel, alternative or even cyclic (interaction and/or transformation) processes related to a specific use case. For each use case an associated set of views is created. In the vertical direction, the process flow is visualised related to time which matches the gradual devolution of states in the associated state view. Horizontally, process blocks are spread from left to right to enable a direct link to the actor view.
State view	…represents the states from initial to final of operands and actors as well as their changes associated with related processes.
Actor view	…indicates the involvement of one or more actors in the realisation of individual processes related to a use case. Involvement may be active or passive. Actors can be differentiated based on whether they – from the designers' point of view – are part of the system or not.
Use case view	…indicates the involvement of individual processes in the different use cases.
Effect view	…represents the effects, which enable individual processes and are provided by actors. For each process block in the process flow view, a separate effect view may be created, preferably using a similar representation.
Interaction view	…maps the specific bilateral impacts between actors and operands as well as their complementary contributions (or any other kind of dependency between them) in the realisation of use cases, associated processes, etc. This view essentially results in an initial system structure or interface matrix of the system, respectively.

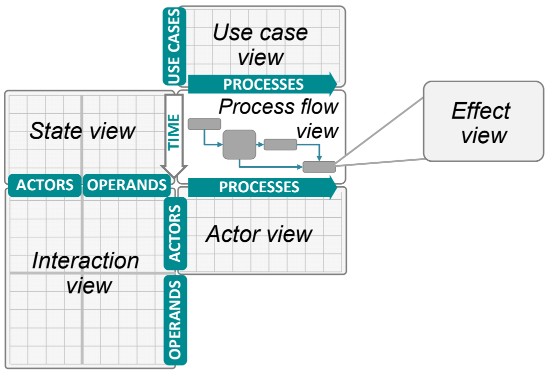


Figure 1. The IFM framework

The central process flow view represents the flow of transformation and interaction processes using standard symbols established for process/activity flow modelling similar to SysML, for instance. The remaining views link to it and comprise of matrices equivalent to Design-Structure-Matrices (DSM, see [Steward 1981]) encompassing different entities related to system functionality and their interdependencies. Entities comprise use cases, transformation and interaction processes, effects, states, operands and actors. Use cases represent different scenarios of applying the technical system for a specific purpose (e.g. fulfilling a goal, changing the state of the system or user, etc.). Transformation processes describe technical and/or human processes - realised by basic physiochemical effects - that result in a change of state of operands or actors within different use cases. Operands are specifications of energy, material and signals. Actors encompass stakeholders (referring to human or other animate beings) and technical systems (which may be hardware and/or software) that are actively or passively contributing to function fulfilment. Furthermore, actors include relevant parts of the environment that provide (unwanted) influences onto the system. Finally, interaction processes describe "cross-boundary" interactions between different actors jointly contributing to function fulfilment (similar to [Eder and Hosnedl 2008]). Figure 2 illustrates the particular entity-relations in the framework using a UML-based domain diagram.

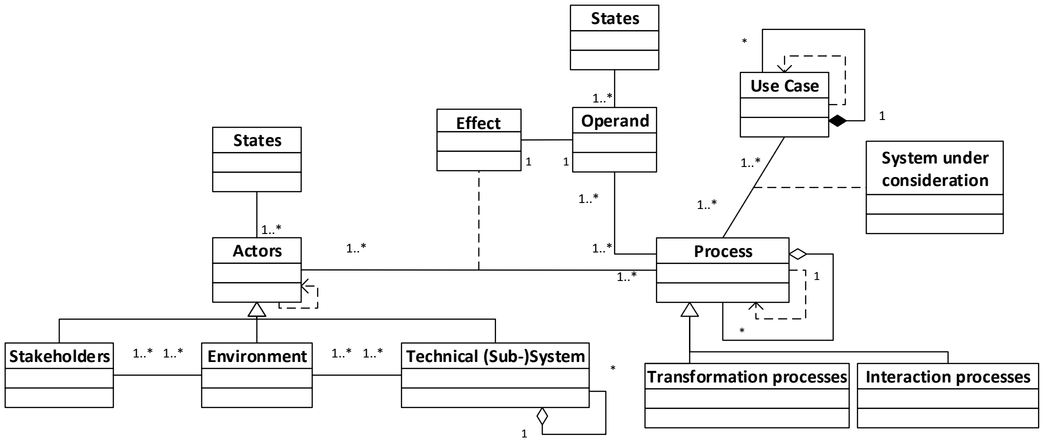


Figure 2. Domain diagram of the entity relations in the IFM framework

In the application of the framework, designers start by deriving the central use cases the system will have to fulfil from a requirements specification. Subsequently, main processes for the use cases are determined and gradually detailed also taking into account any related operands and already known actors. Then, the views are gradually filled with more and more information until, finally, the system is known well enough to determine the mutual interactions among all involved actors and/or operands to fill the interaction view, thus resulting in a comprehensive system description on functional level. Examples of the application can be found in [Eisenbart 2014], [Eisenbart et al. 2015].

3. Property driven development and the characteristics property model

Within Weber's CPM/PDD approach, CPM comprises the resulting product model, while PDD constitutes the systematic design process eventually leading to a comprehensive CPM. The approach's particular benefit lies in its ability to aggregate a plethora of product modelling approaches, design methods, and techniques from well-established theory into a common framework. It inherently endorses formalisation in the product description resulting from embodiment and detail design, eventually, easing transition to computer-based tools. The fundamental scheme underlying the PDD/CPM approach is the clear separation between the characteristics of a product and its properties. Therein,

- Characteristics (Ci) "describe the shape and the structure of a product (e.g. geometry, BOM, materials, etc.) and can be directly established, assigned and modified by the designer";
- Properties (Pj) "describe the current behaviour (and appearance) of a product (e.g. weight, manufacturability, function, cost, user friendliness, etc.)" which can only be indirectly influenced by designers through changing characteristics [Köhler et al. 2008, p.168].

These definitions are predominantly product-oriented, however, it is argued that they can be similarly transferred onto PSS. Therein, characteristics and properties are the structure and constituents of service-related aspect (e.g., the qualification of the service provider, division of work, etc.) as opposed to the parts of the PSS that are perceivable to the customer (e.g., duration of service, costs for consumer, etc.

see [Weber et al. 2004]). Furthermore, Required Properties (PRj) describe the properties that have to be fulfilled by the designed system and represent target specifications (e.g. from customers). Relations (Rj) relate the characteristics to the properties either through the laws of physical behaviour and/or fundamental tangible/intangible principles (e.g. related to individual perception). Relations may be deduced from physical objects (models, mock-ups, and prototypes) or by non-physical modelling (mathematical, numerical, computer-based, graphical, etc.). They may be differentiated in:

- Analysis: Based on a set of known or given characteristics the respective embodied properties are analytically determined through e.g. experiments, simulation, calculations or similar.
- Synthesis: Based on a concurrent set of (Pj) and required properties (RPj) the product's specific characteristics and corresponding values are established (or adapted from an existing configuration) that fulfil the RPs. Herein, designers may use calculation, tables, experience, etc.

In Figure 3 these two basic relations are illustrated. Dependencies (Dx) address the interdependencies between individual characteristics. External conditions (ECj) need to be respected during the analysis and synthesis, as they directly constrain the solution space and may introduce critical influences.

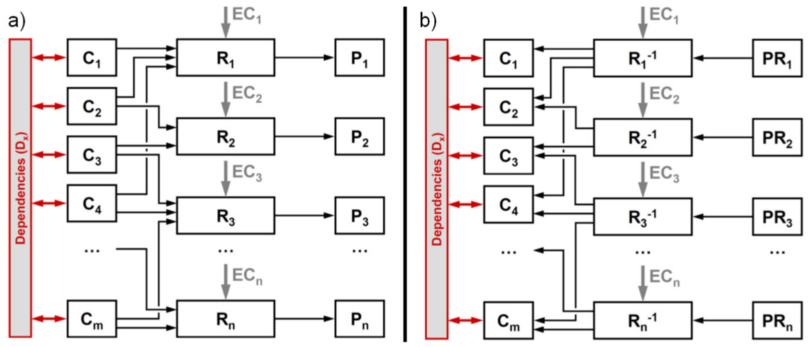


Figure 3. Illustration of the central analysis a) and synthesis b) steps in PDD, after [Weber 2007]

The starting point for the approach is a comprehensive requirements list - based on which the required properties (RPj) are derived - as well as an initial system structural model. Product development strives to define a set of product characteristics such that the resulting product properties (Pj) are sufficiently close to the required properties (PRj), i.e. the difference ΔPj = RPj - Pj ➔ 0. Minimisation of ΔPj is hence driving the development process. The process is expressively cyclic with recurring analysis and synthesis steps. For each cycle the designer selects the particular (set of) Pj(s) to address at a time.

The CPM/PDD approach has been developed further by Qureshi et al. [2013] and Dantan et al. [2013] for handling complex logical expressions including selective and complete validation, arc consistency and box consistency for so-called Quantified Constrained Satisfaction problems. This is predominantly aimed at algorithm-based, recursive optimisation of an initial conceptualisation of a system into a detailed design within set boundaries.

4. Data transition between qualitative and quantitative design via IFM-CPM

The IFM framework is a generic modelling approach that links comprehensive function modelling with initial structural modelling. Yet, it lacks a structured approach for integration and (computationally formal) processing of quantitative data. Such information, however, is predominant in the embodiment and detail design stages. The CPM/PDD approach, on the other hand, requires an initial structural model as a starting point. Therefore, both approaches lend themselves for consistent coupling at this particular point within modelling. That means, in extension, they lend themselves for realising the integration of more abstract, qualitative with increasingly quantitative modelling. So far, because of the rather unique separation into characteristics, properties, relations, etc. in CPM/PDD, there are no consistent links with models from preceding design stages. Designers are hence required to manually develop an initial CPM from scratch to be able to start the PDD process, which means losing efficiency, seeing that large parts of the required information is typically already available in other models at this point in the process. Manual data transition carries risks, e.g., related to ensuring consistency of information across

disconnected models (see [Stark et al. 2010]). Also, critical information may be compromised during manual intervention, e.g., due to unintentional falsification or false (re)interpretation in the process.
As a first step towards integration of IFM with CPM/PDD, this paper proposes an extended class structure for the IFM framework incorporating the data model of CPM for enabling multi-level modelling and designing of a system as by the iterative process inherent to PDD. Thereby, the designers will be able to complement a finalised or almost finalised IFM framework with initial descriptions that reflect the structure of CPM. This should then allow applying PDD based on the IFM framework and ease subsequent transition of data into computer-based tools. The fundamental element of the proposed adaptation is the definition of additional entity classes and attributes within IFM to afford coherent integration with CPM. These classes have been conceived based on the analysis of distinct types of information on meta-level comprised in both approaches. The resulting extended entity relation model of IFM-CPM is shown in Figure 4 (adapted or additional parts are highlighted by shading, c.f. Figure 2). It is imperative to mention here that one of the expressive capabilities of the IFM framework is to describe the impact of elements from the environment with other actors, operands, and processes associated with the system (as briefly discussed above). This, therefore, already inherently entails influences such as what is referred to as external conditions in CPM/PDD.

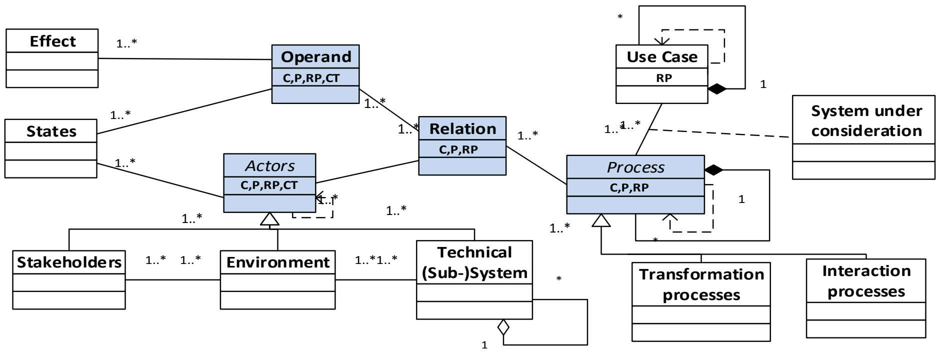


Figure 4. Domain diagram of the entity relations in the IFM-CPM integration model

Figure 4 contains two abstract classes i.e., Process and Actors, while the remaining classes are concrete. Use Case and Process classes have aggregation association with themselves, which means that they may comprise further sub-use cases or processes, respectively, within them. The Process class encompasses two further types of process classes, i.e. Transformation and Interaction Process. Stakeholders, Environment and Technical Sub-System classes are inherited from the Actor class. These classes have relationship with each other and may carry dependencies between them. Following the entity relations in the IFM framework which were briefly introduced in Section 2, it is particularly the transformation and/or interaction processes executed by actors and supported by operands that ensure function fulfilment and (desired) behaviour. They also carry the particular characteristics (Ci in CPM) that results in the system's properties (Pj in CPM). Therefore, these three entity classes have now been complemented by a set of specific attributes, namely 'Characteristics' (C in Figure 4), 'Properties' (P), 'Required Properties' (RP), and 'Constants' (CT). They are described in Table 2 and follow the fundamental definitions used in CPM/PDD.

Table 2. Attribute definition for the IFM-CPM integration

Attribute	Description
Characteristics (C)	…are quantitative attributes of an actor or operand that can be modified by the designers. Characteristics is a data structure element comprising strings of descriptions and units (if applicable) as well as a numeric portion allotting a tangible value to the string.
Properties (P)	...are the behaviour indicator attributes of operands, actors or the environment. These are the simulated or measured results of the interaction of characteristics of operands and actors through relations. They may be intermediate and serve as input characteristics for further relations, or may be final and used as an iterative benchmark to evaluate the soundness and maturity of a design solution to the required properties.

Required properties (RP)	...reflect the specifications of what is demanded of the system (from a client, the design team, etc.). They are the iteration determinants for the design analysis and synthesis cycles.
Constants (CT)	...are fixed values (e.g. from physiochemical laws) on which the designer does not have control over and which are not susceptible to change during the design process (if at all).

The main conceptual adaptation, however, is the introduction of a new class called 'Relation' into the model. The Relation class incorporates the mathematical and logical links between the attributes of a specific process and associated actors and operands. It describes the relations between the characteristics, properties, required properties and external conditions pertaining to these in analogy to CPM/PDD. The information contained in the Relation class can be recorded as analytical, numerical, statistical or experimental dependencies between the elements. It also has self-aggregation, i.e., a relation may be composed of sub relations. The general mathematical form of Relation can be described by the following equations (see [Qureshi et al. 2013]):

$$r_i = f(\bar{C}, p_i) \colon \bar{C} \subseteq C, p_i \subseteq P, f(\bar{C}, p_i) \in \mathbb{R}^\infty \vee \mathbb{C}^\infty \vee \mathbb{Z}^\infty \tag{1}$$

$$r_i = \begin{cases} IF \ldots & THEN \ldots \\ ELSEIF \ldots & THEN \ldots \end{cases} \tag{2}$$

$$r_i = \begin{cases} f(\bar{C}, p_i, rp_i) \leq 0 \\ f(\bar{C}, p_i, rp_i) \geq 0 \\ f(\bar{C}, p_i, rp_i) = 0 \end{cases} \tag{3}$$

Where r_i denotes a specific relation, C denotes the full set of characteristics for a model. The terms with bar accent, i.e. $\bar{C}$, denote specific characteristics that relation may contain, P denotes the full set of properties, and rp_i denotes a specific required property.
Once the full, extended IFM framework is established and populated with data, an iterative quantified constraint satisfaction problem (QCSP) algorithm can be used to iteratively simulate and model the system. This has previously been shown in the research by Dantan et al. [2013] which proposes a method for logically and iteratively reducing the design space for valid solutions in parametric and geometrical design. The general form of the system can be described as:

$$\exists s_i \mid s_i \subseteq D \tag{4}$$

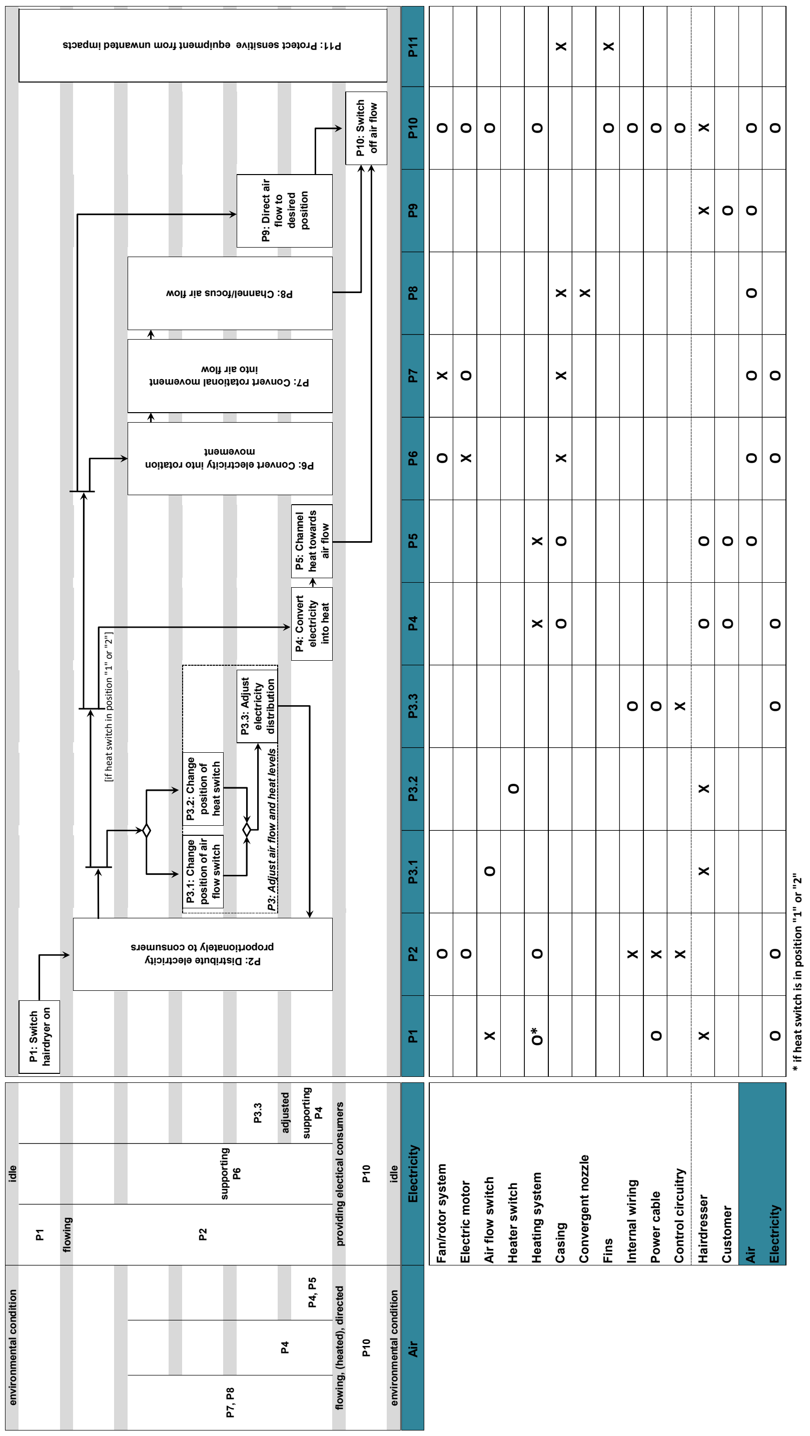
$$S = \{ \cup s_i \in S, s_i \vDash \forall C \, \forall EC \;\; R(C, P, RP) \} \tag{5}$$

Where s_i denotes a possible solution from the set of all possible solutions S, whilst C, P, R, RP, EC denote sets of characteristics, properties, required properties, and external conditions. The algorithm proposed in [Dantan et al. 2013] is based on definition of the design space in terms of characteristics, properties, and required properties in form of QCSP modelled by first order logic (FOL), which starts from an initial quantitative design space and - using interval arithmetic for all quantifiers, $\exists$ and $\forall$ - iteratively explores the design space for a possible solution through domain reduction technique. The next section provides an example of how information from the IFM framework can be transferred into an initial CPM.

5. Example of transition from IFM to CPM

The chosen example is an - arguably simple - PSS around an electric hairdryer (i.e. the product) used by a hairdresser (i.e. the service provider). The dryer has adjustable air speed levels ('0' - off, '1' - medium, '2' - strong) and heat levels ('0' - off, '1' - medium warmth, '2' - heated) that a user can shift between by manipulating embedded switches. In the main use case the dryer is used by the hairdresser simply to dry a customer's hair after washing them. Alternative use cases include, e.g., blowing cut hair off shop utilities or using attachments to the dryer for brushing through the customer's hair to achieve a particular hairstyle. Within the main use case, a few alternative function flows can be considered, which are related to the alternative heat levels and air flow speeds. Figure 5 illustrates the (for brevity reasons only partial)

IFM framework for this use case. The model comprises the process flow view, the operand state view, as well as the actor view ('X' indicating actors that are directly supporting respective processes and '0' indicating actors affected by them). For the ease of demonstration, the example shows an intermediate level of process decomposition and focuses on a small subset of processes of the system concerned with applying the hair-drying service, heating the air, and accelerating air through the hairdryer.



environmental condition
idle
P1
flowing
P2
P7, P8
supporting P6
P3.3
P4
adjusted
P4, P5
supporting P4
flowing, (heated), directed
providing electical consumers
P10
P10
environmental condition
idle
Air
Electricity
P1: Switch hairdryer on
P2: Distribute electricity proportionately to consumers
[if heat switch in position "1" or "2"]
P3.1: Change position of air flow switch
P3.2: Change position of heat switch
P3.3: Adjust electricity distribution
P3: Adjust air flow and heat levels
P4: Convert electricity into heat
P5: Channel heat towards air flow
P6: Convert electricity into rotation movement
P7: Convert rotational movement into air flow
P8: Channel/focus air flow
P9: Direct air flow to desired position
P10: Switch off air flow
P11: Protect sensitive equipment from unwanted impacts

	P1	P2	P3.1	P3.2	P3.3	P4	P5	P6	P7	P8	P9	P10	P11
Fan/rotor system		O						O	X			O	
Electric motor		O						X	O			O	
Air flow switch	X		O									O	
Heater switch				O									
Heating system	O*	O				X	X					O	
Casing						O	O	X	X	X			X
Convergent nozzle										X			
Fins												O	X
Internal wiring		X			O							O	
Power cable	O	X			O							O	
Control circuitry		X			X							O	
Hairdresser	X		X	X		O	O				X	X	
Customer						O	O				O		
Air							O	O	O	O	O	O	
Electricity	O	O			O	O		O	O			O	

* if heat switch is in position "1" or "2"

Figure 5. Partial IFM framework for the use case of drying a customer's hair after washing (including central process flow view, actor view and operand state view)

The example is simplified in two ways. Firstly, electrical and mechanical losses are neglected. Secondly, it is assumed that the designer is at an initial meta-model level and is interested in getting estimates of the design parameters for setting up initial design space. That means, using the IFM framework (Figure 5) as starting point, a number of processes (here: P3 to P8) can be assigned with basic relations to allow high level meta-model simulation. These are the following:

- P3 - Adjust heat and air flow: Boolean
- P4 - Convert electricity into heat: $P_e = I^2R$ (6)
- P5 - Channel heat towards air flow: $\dot{q} = h_cA(t_e - t_o)$ (7)
- P6 - Convert electrical energy to rotational mechanical energy: $P_m = 2\pi NT/60$ (8)
- P7 - Convert rotational movement into air flow: $v_e = \sqrt[3]{\frac{8P_m}{\pi\rho {d_e}^2}}$ (9)
- P8 - Channel/focus air flow: $Q = \frac{\pi}{4}{d_e}^2 v_e$ (10)

The mathematical Equations (6) to (10) assigned to these processes automatically convey an initial list of characteristics and properties which can now be defined within the processes, actors or operands. Table 3 lists these as well as the required properties and constants associated with the relations.

Table 3. Attributes and their associations

No	Description	Symbol	Type	Association	Type
1	Mains Voltage	V	Float	Operand	Constant
2	Current	I	Float	Operand	Property
3	Motor RPM	N	Float	Actor	Characteristic
4	Heating element Resistor	R	Float	Actor	Characteristic
5	Exit Velocity	v_e	Float	Process/Operand	(Required) Property
6	Air Discharge	Q	Float	Process/Operand	(Required) Property
7	Mechanical Power available at Motor	P_m	Float	Actor	Property
8	Exit Nozzle Diameter	d_e	Float	Actor	Characteristic
9	Ambient Air temperature	t_o	Float	Environment	Property
10	Electrical Power	P_e	Float	Environment	Property
11	Air Exit Maximum Temperature	t_e	Float	Process	Required Property
12	Density of Air	ρ	Float	Operand	Constant
13	Surface area of Resister	A	Float	Actor	Characteristic
14	Convective Heat Transfer Coefficient	h_c	Float	Actor	Constant
15	Heat per time unit	$\dot{q}$	Float	Actor/Operand	Property

The workflow based on the established IFM framework starts by designer specifying the type and quantitative attributes of the process, actor and operands. The designer then defines the relations between these entities. Using the simulation method as explained in [Dantan et al. 2013], [Qureshi et al. 2013] and briefly introduced above, subject to arc consistency and box consistency constraints being fulfilled, the designer is then able to perform the meta-model simulation. This, being an iterative method, provides the designer with the opportunity to gradually modify any characteristics pertaining to actors and operands to achieve the desired properties, which is consistent with the PDD approach. Due to the inherent flexibility of the IFM framework, granularity of the model can be increased to address more concrete information as it becomes available, resulting into a cascading increase in details that the designer can manage throughout the process.

The general CPM model resulting from Processes P4-P8 is illustrated in Figure 6. As described above related to CPM/PDD, the attributes on the left hand side of a relation R_i are considered characteristics, whereas the relations on the right hand side of the relation are considered a property. The objects R_i are the relations, stored as different instances of 'Relation' class, described and defined by the interaction of characteristics, properties and required properties of 'Actors', 'Operands', and 'Process' class instances. In case of Figure 6, R_{4-8} correspond to a 'Relation' (highlighted by shading) pertaining to a respective

'Process' P_{4-8} (Equations (6) to (10)). It is important to note that the interaction of the characteristics, properties, and relations is in accordance with dependencies and temporal flow of the processes in Figure 5 to maintain its direct links.

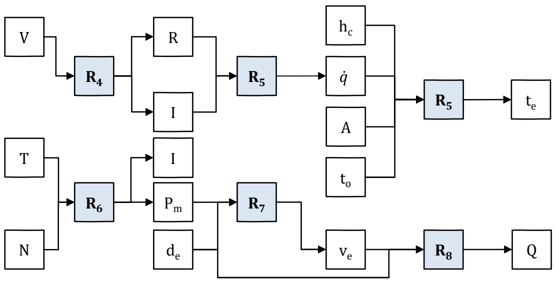


Figure 6. CPM for heating and moving air through the hair dryer

Using the created model, it is then possible to apply the algorithm proposed and successfully applied in [Dantan et al. 2013] to iteratively reduce the given design space. This will then help determining the entire system in detail during subsequent steps to, eventually, allow realisation.

6. Discussion and conclusion

Managing complexity is a challenging and important issue in the design process. One of the fundamental challenges faced in design formalisation and systematisation is managing the ambiguity and traceability of information and decision making [ElMaraghy et al. 2012]. This is compounded by the lack of information in early design stages and its exponential growth while moving to the detail design stage. This paper proposes a first step towards an integrated, yet flexible, method for early design space exploration both on an abstract functional analysis level as well as toward a meta-modelling level. The fact that the used approaches of IFM framework and CPM/PDD are both consensual, generic design and modelling approaches provides potential for the developed linkage between the two approaches to be transferred to other design automation endeavours. The proposed formal and conceptual integration of the IFM framework with CPM fills two important gaps. Firstly, it opens up the possibility to link more quantitative modelling (in CPM) with abstract design information such as use cases, processes, and states of related actors and operands. This may eventually enhance reasoning towards how changes to components in detail design may affect overall function fulfilment. Secondly, while the IFM framework currently predominantly promotes abstract modelling, the implemented adaptations are expected to allow for relatively seamless concretisation towards later design stages. The third aspect is the integrated description of external conditions through the environment class in IFM framework. Therein, the environment is explicated as an actor and interacts with the process and actors via the relation class and attribute sharing compliant with CPM/PDD. This should ease contextual product development, i.e. considering environmental and societal factor right from the onset of the design project. It is aspired that practitioners in industry will be able to use the proposed IMF-CPM/PDD integration to support decision making and gradual detailing of solutions during conceptual design and beyond.

One question that has not been tackled in this article is how process modelling related to services can be iterated on using formal relations. As service processes, especially when performed by humans rather than technical means, are more qualitative by nature, this question cannot be answered easily. However, research has already produced initial means for process simulation (see, e.g., the Cambridge Advanced Modeller [Wynn et al. 2009]). Therefore, the authors presume that this is possible in principle and future research will address this particular issue in detail. The authors expect that this will further require looking into business-related aspects of services and PSS as well, because these aspects may be of vital importance in terms of determining related required properties (RPj). Further potential for future research concerns a possible integration with STEP ISO10303 [ISO 1994] or similar established initiatives that promote standardised product modelling. Such initiatives, so far, are limited to quantitative or spatial modelling and have gaps towards early stage design modelling. The proposed IFM-CPM integration carries the potential to fill this gap in the future. Finally, provided that the proposed concept is taken further and implemented adequately, it may be transferred into established

PDM software packages, which would eventually unlock the full potential of the proposed approach, ideally on an interactive work surface used by a design team. Another direction for development may involve software implementation, e.g. using SysML, to ease implementation into industrial practice.

References

Braha, D., Reich, Y., "Topological Structures for Modeling Engineering Design Processes", Research in Engineering Design, Vol.14, No.4, 2003, pp. 185-199.

Chakrabarti, A., Bligh, T. P., "A Scheme for Functional Reasoning in Conceptual Design", Design Studies, Vol.22, No.6, 2001, pp. 493-517.

Dantan, J. Y., Qureshi, A. J., Antoine, J., Eisenbart, B., Blessing, L. T. M., "Management of Product Characteristics Uncertainty Based on Formal Logic and Characteristics Properties Model", CIRP Ann. - Manuf. Technol., Vol.62, No.1, 2013, pp. 147-150.

Dong, A., "Functional Lock-in and the Problem of Design Transformation", Research in Eng Design, forthcoming.

Dori, D., "Object-Process Analysis: Maintaining the Balance between System Structure and Behavior", Journal of Logic and Computation, Vol.5, No.2, 1995, pp. 227-249.

Eckert, C., Albers, A., Bursac, N., et al., "Integrated Product and Process Models: Towards an Integrated Framework and Review", Proceedings of 20th International Conference on Engineering Design - ICED, 2015.

Eisenbart, B., "Supporting Interdisciplinary System Development Through Integrated Function Modelling", Dissertation, University of Luxembourg, Luxembourg, 2014.

Eisenbart, B., Gericke, K., Blessing, L. T. M., "Integrating Different Function Modelling Perspectives", Proceedings of International Conference on Research in Engineering Design - ICoRD, 2013.

Eisenbart, B., Mandel, C., Gericke, K., Blessing, L. T. M., "Integrated Function Modelling: Comparing the IFM Framework With SysML", Proceedings of the 20th International Conference on Engineering Design - ICED, 2015.

ElMaraghy, W., ElMaraghy, H., Tomiyama, T., Monostori, T., "Complexity in Engineering Design and Manufacturing", CIRP Ann. - Manuf. Technol., Vol.61, 2012, pp. 793-814.

Gero, J. S., Kannengiesser, U., "The Function-Behaviour-Structure Ontology of Design", In: Chakrabarti, A., Blessing, L. T. M. (Eds.), An Anthology of Theories and Models of Design, Springer, Berlin, 2014.

ISO, "Industrial Automation Systems and Integration - Product Data Representation and Exchange - Part 1: Overview and Fundamental Principles", ISO/TC 184/SC 4, ISO 10303-1, 1994.

Köhler, C., Conrad, J., Weber, C., "A Matrix Representation of the CPM/PDD Approach as a Means for Change Impact Analysis", Proceedings of 10th International Design Conference - DESIGN, 2008.

OMG, "OMG Systems Modeling Language (OMG SysMLTM) Specification", Available at: <http://www.omg.org/spec/SysML/1.3/>, 2015, [Accessed 12.04.2015].

Pahl, G., Beitz, W., Feldhusen, J., Grote, K. H., "Engineering Design: A Systematic Approach", Springer, 2007.

Poon, J., Maher, M., "Co-evolution and Emergence in Design", Artificial Intelligence in Design, Vol.11, No.3, 1997, pp. 319-327.

Qureshi, A. J., Eisenbart, B., Dantan, J. Y., Blessing, L. T. M., "Design Automation with the Characteristics Properties Model and Property Driven Design for Redesign", Proceedings of CIRP Design Conference, 2009.

Stark, R., Beier, G., Wöhler, T., Figge, A., "Cross-Domain Dependency Modelling - How to achieve consistent System Models with Tool Support", Proceedings of the 7th European Systems Engineering Conference, 2010.

Steward, D. V., "The Design Structure System: A Method for Managing the Design of Complex Systems", IEEE Transactions on Engineering Management, Vol.28, No.3, 1981, pp. 71-74.

Weber, C., "An Extended Theoretical Approach to Modelling Products and Product Development Processes", In: Bley, H., Jansen, H., Krause, F.-L., Shpitalni, M. (Eds.), Proceedings of the 2nd German-Israeli Symposium on Advances in Methods and Systems for Development of Products and Processes, Berlin, 2005, pp. 159-179.

Weber, C., "Looking at "DFX" and "Product Maturity" from the Perspective of a New Approach to Modelling Product and Product Development Processes", Proceedings of the 17th CIRP Design Conference, 2007.

Weber, C., Steinbach, M., Botta, C., Deubel, T., "Modelling of Product-Service Systems (PSS) Based on the PDD Approach", Proceedings of 8th International Design Conference - DESIGN, 2004.

Wynn, D., Nair, S. M. T., Clarkson, P. J., "The P3 Platform: an Approach and Software System for Developing Diagrammatic Model-based Methods in Design Research", Proceedings of ICED 2009, 2009.

Dr. Boris Eisenbart, Assistant Professor
University of Sydney, Discipline of International Business
Delft University of Technology, 2628CE Delft, Netherlands
Email: boris.eisenbart@gmail.com