USING THE ACD³-LADDER TO MANAGE MULTI-PHASE REQUIREMENTS ON END-USER PRODUCTS

Berlin, Cecilia (1); Bligård, Lars-Ola (2); Simonsen, Eva (2)
1: Chalmers University of Technology, Division of Production Systems, Sweden; 2: Chalmers University of Technology, Division of Design and Human Factors, Sweden

Abstract

During the development of any end-user product, a multitude of design decisions need to be made. But if design activities and decisions happen at the wrong time, or not at all, unintentional and sometimes negative design outcomes can be the result. Determining all requirements early in the product development is traditionally recommended, but may force design decisions to be made prematurely on the basis of incomplete preconditions. Requirements at different degrees of resolution are useful and purposeful at different stages of the development process. To address these requirements management challenges, this paper proposes an approach for incrementally developing requirements in parallel with design, based upon a previously developed framework called ACD³, which draws on a combination of theoretically compatible ideas and concepts from Design Engineering, Human Factors/Ergonomics, Usability and Systems Theory. This approach helps designers identify and handle the possible interdependencies of design variables. The paper also theoretically motivates and demonstrates with an example how the different resolution levels of requirements relate within the framework.

Keywords: Organisation of product development, Design management, Multi- / Cross- / Trans-disciplinary processes, Requirements

Contact:
Dr. Cecilia Berlin
Chalmers University of Technology
Product and Production Development
Sweden
cecilia.berlin@chalmers.se

Please cite this paper as:
Surnames, Initials: *Title of paper*. In: Proceedings of the 21st International Conference on Engineering Design (ICED17), Vol. 4: Design Methods and Tools, Vancouver, Canada, 21.-25.08.2017.

1 INTRODUCTION

During the development of any end-user product, a multitude of design decisions need to be made. These design decisions are shaped by a combination of the individual product designer's unique knowledge and experience; input from multi-disciplinary perspectives; the organization's procedures, culture and routines regarding design work; constraints given by known end-user or customer demands; constraints given by previous production and sales of similar products, all of which shape expectations on the function and appearance of new products; and many more.

For products with a high degree of user-product interaction (e.g. involving manipulation of user interfaces), a multi-disciplinary design team is often tasked with the development. This may require complex work organization processes to optimize the use of each team member's expertise, leading to multi-disciplinary work at multiple levels of product definition. A successful series of design decision-making leads to the design team gradually narrowing down the maximal scope of design solutions to a well-defined set of requirements that indicate which desired criteria the end result should live up to. Given that end-user products can be extremely complex in functionality and performance demands, these design teams can be very large and diverse, with people whose scope of expertise ranges from the visionary and strategic realm to that of the highly specialized components-level perspective. Involving them all at the right time to submit their expertise into the decision-making process can be an organizational and temporal challenge, not to mention one of getting them to communicate their design intentions effectively and at the currently appropriate detail resolution. Orchestrating the input requires a holistic yet sequential approach to generating, selecting, refining and managing requirements. If activities and decisions are made at the wrong time, or not at all, it can have unintentional and sometimes negative impact on the final result of the development. For example: if the design of a product is not evaluated from a human factors perspective until the prototype stage, interaction problems may be discovered too late (due to other constraints) to effectively re-design the product's interfaces in such a way that human factors demands are met - and any attempted patch-ups may render the product a bad fit for the user population.

To cope with this, researchers like Pahl et al. (1996) have stressed the necessity for systematic design processes that support the definition of design activities to be carried out for a specific project, thus supporting design work without imposing the rigidity of a generic procedural template. To create such a systematic design process, many models have been suggested (Ullman, 2010; Ulrich and Eppinger, 2011; Pahl et al., 1996; ISO, 2010; Dubberly, 2005). Most of the models describe the design process in one dimension, i.e. one type of aspect or feature makes up the backbone of the process. Often, these processes are dived into phases or activities originating from one type of main flow; linear/sequential or iterative/cyclic.

In light of this, the conventional perception that requirements are to be finally determined early in the product development process (e.g. Ulrich and Eppinger, 2011; Cross, 2008) may "lock in" a sub-optimal design before all the necessary input has been given from appropriate stakeholders, which may force decisions to be made on uncertain preconditions. This may make the design vulnerable to late-stage inputs from overlooked stakeholder perspectives. Pew and Mavor (2007) emphasize the evolving nature of system requirements, since all the knowledge needed to write a complete requirement specification in beginning of a project usually does not exist. Rather, new requirements emerge during the whole design process, in a so-called co-evolution of requirements and design (Braha and Reich, 2003). This is particularly true of end-user products with multiple user categories (e.g. primary users as well as maintenance or disassembly personnel), since many requirement types do not emerge before user testing starts. Hubka and Eder (1988) describe the function-means tree as a structure for co-evolution, but without specific levels of abstraction.

Requirements are not uniform in scope or definition; they can be established at different degrees of resolution that are useful and purposeful at different stages of the development process. For example, System Engineering and the V-model (Stevens, 1998) in particular distinguishes between *User Requirements* and *System Requirements*, where the former concerns what the user wants to with the product, and the latter concerns what the product must do. Szejka et al. (2014) describe the difficulty of refining requirements throughout the development process, saying that "Requirements (…) must be controlled inside all these phases and domains to avoid misinterpretation and mistakes that would compromise the final results (...) there is still a semantic gap between all requirements definitions when

they are defined in different domains for the same engineering project and requirement consistency management in different systems life cycle phases".
Also, companies are becoming more interested in cradle-to-cradle perspectives, adding to the design complexity. Facing such challenges places large demands on the organization of design work, so as to draw the greatest benefit from involving various types of expertise in early phases without too quickly constricting the design space – this is notoriously difficult, since such a multitude of perspectives and demands may render the requirements list very long and difficult to address in a coherent order.
To summarize: finalizing requirements early in the development process, before enough knowledge of the product and its use is known, may lead to sub-optimal design solutions. To address the above challenges holistically, this paper proposes an approach for handling incremental development of requirements in parallel with design, in the domain of end-user product development. This is achieved by building upon a previously developed framework for product development and design work called ACD³. The specific contribution is a model with abstraction levels for the co-evolution of design and requirements. The levels are based on systems theory and termed *Effect, Usage, Architecture, Interaction* and *Elements*. The model is used as guide to requirement management and design.

2 THE ACD³ FRAMEWORK

This section describes the fundamentals of the ACD³ framework – previously described chiefly in Swedish (Bligård, 2015; Bligård et al., 2016; Berlin and Bligård, 2016) – to provide the reader with a good foundation to comprehend the paper's main concept, the ACD³ Ladder for managing multi-phase requirements on end-user products.

2.1 A brief overview of the framework

The explicit goal for ACD³ is to support multidisciplinary product development teams in making coherent design decisions that propagate logically from the product's intended effects down to the technical detail-level design. The purpose is to ensure a more successfully designed interaction between humans and a "*machine*" – this term is used instead of product to signal that the artifact being designed is complex and involves functionality, architectural structure and interaction with a user (Bligård, 2015). The "D³" in ACD³ refers to that the framework advocates development based on three fundamental dimensions: *Design levels*, *Design Perspectives* and *Design Activities*. When combined, the three dimensions provide a holistic guidance framework for which design decisions need to be made as logical consequences to the desired overall effect, all the way down to the technical detail-level solution. ACD³ is *Use*-centered rather than *User*-centered, i.e. it aims primarily to design a successful user-machine *interaction*; as a consequence, the hierarchy of design decisions consistently relates to intended functionality and activities. It also emphasizes that there is an end user, and thus the framework should not be evaluated as e.g. a tool for design optimization of individual machine components, as this is not what it is intended to facilitate.

2.2 Theoretical basis

The ACD³ Framework is based on a combination of theoretically compatible ideas and concepts from Design Engineering, Human Factors/Ergonomics, Usability and Systems Theory, as shown in Table 1:

Table 1. Theoretical basis for the ACD³ Framework

Theory	**Authors**	**Implications for the framework**
Systems theory	(Bertalanffy, 1973)	The Design levels in ACD³ are derived from the idea of "abstraction levels" and other terminology from General Systems Theory (GST), which has provided the overall philosophy of viewing the product development as a system. ACD³ follows the GST logic of defining a total systems boundary and the sub-systems it consists of, and gradually refining the description of elements, the hierarchical relationships between them, and how they affect each other (feedback).
Activity theory	(Karlsson, 1996)	The use- and activity-centered perspective is derived from the focus on agents, goals and means as described specifically in Karlsson's (1996) application of Activity theory.
Joint Cognitive Systems	(Woods and Hollnagel, 2006)	A Joint Cognitive Systems view means viewing the human (the user) as an integral part of the overall system, and not just designing the product as an independent technical entity.
Systems Engineering	(Stevens, 1998)	The Systems Engineering perspective provides concepts related to requirements management (e.g. the V-model) and the role of functions in technical systems. This theoretical framework also provides the name of the "Architecture" design level.
Cognitive Systems Engineering	(Rasmussen et al., 1994)	The contribution from Cognitive Systems Engineering is to view systems at several different abstraction levels based on the interplay between the human and the machine; in ACD³, this has been applied as Design levels.
Use-centered design	(Bennett and Flach, 2011)	Use-centered design focuses on the goals for the users and tasks that are carried out within the specific domain where the problems are. The match between human, activity and environment is central to Use-Centred Design, aiming towards a successful design of the machine (which in turn is defined as its achieving the intended purpose). The task is defined as the action taken by the user, with the machine as an assisting device, to solve the overall problem. This is a fundamental philosophy of the ACD³ framework.
Function-means law ("Hubka's law")	(Hansen and Andreasen, 2002)	The Function-Means law, a.k.a. Hubka's Law, states that goals and means can be organized in a hierarchical structure, and that in a hierarchy of effects, the behavioural and structural aspects of a machine – i.e., the functions and the means – are causally related. This is applied in how the design and requirements are developed interactively and in parallel throughout the design of the product.

As stated in Bligård et al. (2016), "ACD³ emerged as a result of combining different frameworks and process models in these areas into a coherent whole, with the aim of enabling a clearer understanding of a design process, as in making it easier to implement and teach." The framework advocates the viewing of the machine from multiple systems perspectives, to continually address different disciplines of machine design at all design levels. This helps the design team identify and handle the possible interdependencies of design variables. The levels are described in greater detail in Section 2.3.

The framework postulates that the *utility* of a machine only emerges when the product is successfully used; therefore designing the *use first*, and then a machine to support this use, is a philosophical cornerstone derived from the theoretical basis. Further, a *design decision* is defined as the action (taken by the designer) of constraining the possible value of a design variable; i.e. once the variable properties of the machine are specified (e.g. length, maximum weight, colour), determining their values (e.g. 4 metres, max 350 g, blue) constrains the space of possible solutions, and constitutes the design decisions. A design variable "(…) is always determined in a design process, regardless of whether this is due to an active decision or if it results unintentionally. (…) their interdependence results from the precedence relationship between the variables." (Bligård et al., 2016). The crucial element that regulates the intentional making of design decisions is therefore *requirements* – these narrow down the design space and emerge gradually (just like the design itself) as the result of the interplay between requirements and the design in the process. Accordingly, the requirements and the design give input to each other and ideally evolve in a top-down manner, from an initial overall principle to final technical details. The requirements from one phase frames the design in the next, and then that design generates new reasons to elicit requirements for the following stage. This structure, called the ACD³ Ladder, is presented in Section 3.

2.3 The three dimensions of ACD³

The first dimension of ACD³ provides a systemic perspective of the product development. Called *Design levels*, it refers to the abstraction level at which the product can be described, ranging across a continuum from the desired overall *Effect* (which defines the product's objective but leaves the design solution space wide open) down to the *Elements* (which mean that the design space has been maximally constrained and thereby the product specification has been defined in detail). Figure 1 provides an overview and a practical example of a vacuum cleaner, which will be revisited later.

Design level	System view	Description	Example: vacuum cleaner
Effect	Socio-technical system	The effect that the machine is intended to achieve in the context	A cleaner home
Usage	Human-machine system	The use of the machine by humans	Manually moving the device when cleaning
Architecture	Machine system	The technical architecture of the machine	An electrical motor that sucks air through a filter
Interaction	Machine interfaces	The interaction between human/ context and machine in details	Design of the physical form and user interface
Elements	Sub system	The technical elements of the machine	Structural design of the motor, the dust bag etc

Figure 1. The Design Levels, exemplified by a vacuum cleaner (from Bligård et al., 2016)

The second and third dimensions, *Design Perspectives* and *Design Activities*, are only briefly described here as they have a lesser role in requirements management, and are described in full elsewhere (Bligård, 2015; Bligård et al., 2016). *Design Perspectives* reflect the notion that many different types of foci can be useful to the development of a product, and that these need to be orchestrated. The Design Perspectives specify five multi-disciplinary perspectives of the product design (*Problem*, *Structure*, *Function*, *Activity* and *Realization*). In the ACD³ framework, design of the product from each of these perspectives happens at each Design Level, from Effects to Elements. Finally, the *Design Activities* involve the identification, determination and communication of the design variables that make up the solution, iteratively within each phase in the design work – the specified iterative activities are *Planning*, *Data Collection*, *Analysis*, *Ideation*, *Synthesis*, *Evaluation* and Documentation. These activities are understood to be part of a design process that is both linear, with distinct phases (defined in the ACD³ Process as *Needfinding*, *Design of use*, *Overall design*, *Detailed design* and *Structural design*), and also iterative, with the design activities repeating within each phase.

The pairwise combination of the three dimensions has led to two separate sub-frameworks called the *ACD³ Matrix* and the *ACD³ Process*, which are described in full in Bligård et al. (2016) and Bligård (2015).

3 THE ACD³ LADDER

3.1 Handling Requirements with The ACD³ Framework

One cornerstone of the ACD³ framework regards the interplay between requirements and the product design – they must be developed in parallel and successively, in order for detailed design decisions to become logical consequences of the overall desired effects. The requirements guide the design decisions in a particular direction, ideally to realize the overall effect goals. We argue that to achieve this, the resolution of requirements should be coupled to different design levels. In ACD³ each design level has been assigned a corresponding type of requirements (Table 2), which are developed in parallel with the design to an increasing degree of detail resolution.

Table 2. Summary of requirement purpose and resolution at each design level

Design Level	**Purpose**	**Requirement**
Effect	Frame the whole product development process	*Needs* - the needs that the human-machine system is expect to fulfil to achieve the effect
Use	Frame the machine in the context of the socio-technical system	*Use requirements* - Requirement on the human-machine system that needs to be fulfilled to enable the use
Architecture	Frame the design of machine as a whole	*Machine (system) requirements* - requirement on the machine as whole
Interaction	Frame the design sub-parts of the machine and how they interact	*Sub-system requirements* - requirements on the different sub-parts of the machine
Element	Frame the manufacturing of the machine	*Manufacturing requirements* - requirements on how the parts of machine should be produced

3.2 Setting requirements at each design level

The design levels with their respective requirement types are visualized in Figure 2, which shows the ACD³ Ladder of Requirements Management. The Ladder implies that the design work should advance as a result of an iterative interplay with the requirement specifications, evolving gradually and iteratively as the development progresses and the detail level increases. This means that there is a continuous exchange of influence, and the design and requirements become prerequisites for each others' incremental development.

According to this logic, every level in the ACD³ Ladder (except the first Effect) is determined and constrained by the design- and requirements levels preceding it, and in turn it determines and constrains all subsequent design- and requirements levels. Every level of requirements constrains the possible design solution space in the subsequent design work, i.e., the requirements operationalize design decisions made in preceding levels into demands for the subsequent specification and precision. This follows the structure of the function-means tree (Hubka and Eder, 1988), but the abstractions levels are defined beforehand.

Throughout the design process, the design and the requirements are made more precise and specified at more or less formalized intermittent "checkpoints" at which the design is evaluated, often called phases or gates. Accordingly, the requirements and the design give input to each other and iteratively evolve in a top-down manner. The requirement from one stage in the sequence frames the design in the next stage, and then that design is the basis for eliciting requirements for the following stage. This interaction between design and requirements continues until the machine is described at a level where it can be

produced in one decisive way; i.e., such that the subsequent design decisions made during the production phase do not affect the function and construction of the machine.

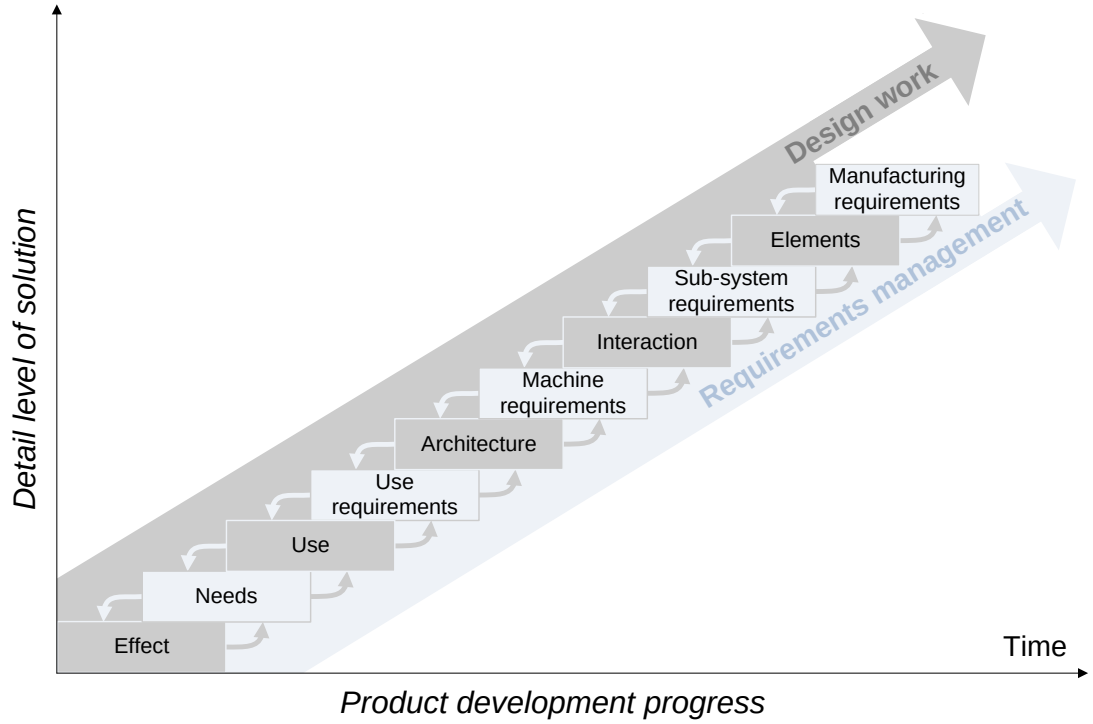

Figure 2. The ACD³ Ladder of Requirements Management

It is important to note that the Ladder of Requirements Management does not prescribe the order in which designs decisions are made, but rather the hierarchical relationship between design and requirements at different abstraction levels. The ladder should therefore not be interpreted as a strict working procedure. Within each of the design levels, the design and the requirements grow together as result of the iterative design activities. Furthermore, there is often a need to explore the succeeding design level to fully understand which aspects are relevant to put into requirements at the current level. Also here an interactive approach is advocated, proposing that the design and requirements may co-evolve at all design levels in parallel. This is especially important when requirements change and evolve due to increased learning and external restrictions and change. The role of the ladder is to provide the structure and logic for the co-evolution and iteration of all the levels of design and requirements.

4 EXAMPLE: VACUUM CLEANER

Table 3 shows an example product – a vacuum cleaner – to provide a guide for how the design and requirements at different levels interrelate when mapped onto the ACD³ framework. Here, we offer a walkthrough of the example to explain the logic of the interplay between design and requirements.

Table 3. Requirements at each level, with a vacuum cleaner as an example

Level	**Example design**	**Example of detail resolution of requirements**
Effect	A cleaner home	*(Use, User and Stakeholder) Needs, e.g.* - Cleaning should be performed in a healthy working posture - No damage to the dwelling and furniture - Cheaper than existing equivalent cleaning equipment - Possible to store in a normal closet - Be styled in the company profile colours - Conform with CE marking regulations
Use	Manually moving the device when cleaning	*Use requirements, e.g.* - Connect to the general power grid (if electric) - Reach at least 1.5m away from the user - Be used by 5-95 percentile users aged 10 and up
Architecture	An electrical motor that sucks air through a filter	*Machine requirements, e.g.* - Cord should be at least 8 meters - The suction pipe should be adjustable by 40 cm -The force when pressing the button shall be between 0,5 and 2 N - The hand grip / handle should be between 4 and 6 cm wide - The casing radii of the machine should be at least 2 cm - The length of the main unit must not exceed 35 cm
Interaction	Design of the physical form and user interface	*Sub-system requirements, e.g.* - Detailed requirements on the nozzle, hose, dust bag, filter, fan, motor, controller, software, cord, cord winder, etc.
Element	Structural design of the motor, the dust bag, etc.	*Manufacturing requirements, e.g.* - Tolerances for drilled holes - Smoothness of surfaces

The central design decision at the *Effect* level is that the result of the user-machine interaction should be “a cleaner home”. This is the basis for identifying the *Use- and User Needs* that must be fulfilled to achieve the effect in a desirable way for all stakeholders (not just users), e.g. by not forcing unhealthy working postures and by not being more expensive than existing solutions. Needs for other stakeholders are to "Conform with CE marking regulations" and "Be styled in the company profile colours". The needs should be fulfilled by the use, whereby the machine’s utility emerges. Central to the *Use* design level is what the machine and the user must do respectively. In this example, the human should be "manually moving the device when cleaning" while the machine should fulfil the rest of the functions. The design of the use then logically leads to demands on the machine related to the intended use context, like "Reach at least 1.5m away from the user" and "be used by 5-95 percentile users aged 10 and up", both of which must be fulfilled in order for the intended use to be made possible. The use requirements in turn constrain the technical *Architecture* towards the solution "An electrical motor that sucks air through a filter". From this architecture, *Machine requirements* can be set at a more precise level of detail, such as "Cord should be at least 8 meters" and that "The suction pipe should be adjustable by 40 cm". If these requirements are fulfilled, they should (due to their internal logical coherence) enable the intended use possible which in turn enables the achievement of the desired effect. The Machine requirements then constrain the *Interaction*, i.e. the "Design of the physical form and user interface" for the vacuum cleaner, and the aggregated design decisions up to that point amount to detailed requirements for all *sub-systems* (like the nozzle, hose, dust bag, filter, fan, motor, controller, software, cord, cord winder, etc). This is followed by the *Elements* level, at which e.g. the structural design of the motor, the dust bag etc. finally determine the *Manufacturing requirements*, e.g. tolerances for drilled holes and smoothness of surfaces.

5 DISCUSSION

Given the organizational, multi-disciplinary and complex nature of end-user product development, it may seem that there is a gap between the conventional paradigm to specify requirements as comprehensively and early as possible, and the growing understanding in contemporary research that requirements evolve during the on-going design process. We argue that handling requirements development incrementally and in parallel with the gradually emerging design, and benefiting from the interplay between them, allows the design decisions to not only become more internally coherent, but also more robust in the face of late-stage suggestions for change. New demands at odds with the overall desired effect can be addressed critically in order to avoid a solution that is a bad compromise, and the multi-perspective guidance of the ACD³ structure avoids this risk in the first place.

The idea of co-evolution is previously established and there is support in other literature for viewing design work at different levels and perspectives; for example, the *Requirements Abstraction Model*, RAM (Gorschek and Wohlin, 2006) describes software engineering at *Product Level, Feature Level, Function Level* and *Component Level*. ACD³ was developed independently of RAM, but the levels match quite well. Another model proposing abstraction levels is Leveson's model for safety-critical technical systems, called *Intent Specifications* (Leveson, 2017). It has seven levels, five of which map quite well onto the levels of ACD³. Table 4 shows a side-by-side comparison of the models' respective levels.

Table 4. Comparison of level structures of ACD³, RAM and Leveson's model

ACD³	Requirements Abstraction Model (RAM)	Leveson's model (2017)
Effect Level Use Level Architecture Level Interaction Level Element level	Product Level Feature Level Function Level Component Level	Level 1: System Purpose Level 2: System Design Principles Level 3: System Architecture Level 4: Design Representation Level 5: Physical Representation

The logical coherence of these three models demonstrates that current research supports a generalization of these combined theoretical perspectives to consider design from multiple abstraction levels, a notion backed by Savioja et al (2014). The ACD³ framework can also counteract problems teams may have with understanding requirements (Lehtinen et al., 2015; Szejka et al., 2015) since each level of requirements is direct related to a design level, i.e. it makes it easier to trace the raison d'être of the requirements and put them into context for appointing the correct stakeholder to interpret the need and choose a solution.

The validity of the ACD³ framework is currently being tested in a budding "Community of Practice" (Wenger, 2000; Wenger and Snyder, 2000) including Master thesis projects, an ongoing verification project involving Swedish product developers, and collaboration with a Swedish management consultancy firm specialized in organizational development.

6 CONCLUSIONS

This paper has presented an approach for incremental requirements management in parallel with the advancement of design work, based on a novel framework for product development. The specific contribution is a model with abstraction levels for the co-evolution of design and requirements thoroughly based on systems theory. It has also theoretically motivated and demonstrated with an example its utility for multi-disciplinary requirements handling.

REFERENCES

Bennett, K. B. and Flach, J. M. (2011), *Display and interface design : subtle science, exact art,* Boca Raton, Fla., CRC Press. doi.org/10.1201/b10774-2

Berlin, C. and Bligård, L. O. (2016), "An activity centered design framework for determining design decision levels in production systems", *Advances in Intelligent Systems and Computing.* https://doi.org/10.1007/978-3-319-41697-7_40

Bertalanffy, L. V. (1973), *General system theory : foundations, development, applications,* New York, Braziller.
Bligård, L.-O. (2015), *Utvecklingsprocessen ur ett människa-maskinperspektiv - ACD3-procesen,* Göteborg, Chalmers University of Technology. http://dx.doi.org/10.13140/RG.2.1.1954.4400
Bligård, L. O., Simonsen, E. and Berlin, C. (2016), ACD^3 - A new framework for activity-centered design. Proceedings of NordDesign, NordDesign 2016, 2016.
Braha, D. and Reich, Y. (2003), 'Topological structures for modeling engineering design processes', *Research in Engineering Design,* 14**,** 185-199.
Cross, N. (2008), *Engineering design methods : strategies for product design,* Chichester, John Wiley.
Dubberly, H. (2005), How do you design? San Francisco: Dubberly Design Office.
Gorschek, T. and Wohlin, C. (2006), "Requirements abstraction model", *Requirements Engineering,* 11**,** 79-101. doi.org/10.1007/s00766-005-0020-7
Hansen, C. T. and Andreasen, M. M. (2002), "Two approaches to synthesis based on the domain theory", *Engineering Design Synthesis.* London: Springer-Verlag. https://doi.org/10.1007/978-1-4471-3717-7_6
Hubka, V. and Eder, W. E. (1988), *Theory of technical systems : a total concept of technical systems,* Berlin ;, Springer-Verlag.
ISO (2010), ISO 9241- 210:2010 Ergonomics of human-system interaction – Part 210: Human-centred design for interactive systems. Brussels: ISO.
Karlsson, I. C. M. (1996), *User requirements elicitation - A framework for the study of the relation between user and artefact.* PhD Thesis, Chalmers University of Technology.
Lehtinen, T. O. A., Virtanen, R., Heikkilä, V. T. and Itkonen, J. (2015), "Why the development outcome does not meet the product owners' expectations?", *Lecture Notes in Business Information Processing.* doi.org/10.1007/978-3-319-18612-2_8
Leveson, N. G. (2017), "Rasmussen's legacy: A paradigm change in engineering for safety", *Applied Ergonomics,* 59**,** 581-591. doi.org/10.1016/j.apergo.2016.01.015
Pahl, G., Beitz, W. and Wallace, K. (1996), *Engineering design : a systematic approach,* Berlin, Springer. http://dx.doi.org/10.1007/978-1-4471-3581-4
Pew, R. W. and Mavor, A. S. (2007), *Human-System Integration in the System Development Process: A New Look,* Washington, D.C., National Academy of Sciences. doi.org/10.17226/11893
Rasmussen, J., Mark Pejtersen, A. and Goodstein, L. P. (1994), *Cognitive systems engineering,* New York, Wiley.
Savioja, P., Liinasuo, M. and Koskinen, H. (2014), «User experience: does it matter in complex systems?", *Cognition, Technology and Work,* 16**,** 429-449. doi.org/10.1007/s10111-013-0271-x
Stevens, R. (1998), *Systems engineering : coping with complexity,* London, Prentice-Hall Europe.
Szejka, A. L., Aubry, A., Panetto, H., Júnior, O. C. and Loures, E. R. (2014), "Towards a Conceptual Framework for Requirements Interoperability in Complex Systems Engineering", Proceedings - *OTM 2014 Workshops: Confederated International Workshops: OTM Academy, OTM Industry Case Studies Program, C&TC, EI2N, INBAST, ISDE, META4eS, MSC and OnToContent 2014, Amantea, Italy, October 27-31, 2014.* Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-662-45550-0_24
Szejka, A. L., Canciglieri, O., Jr., Loures, E. R., Panetto, H. and Aubry, A. (2015), "Requirements interoperability method to support integrated product development", *Proceedings - CIE 45: 2015 International Conference on Computers and Industrial Engineering,* 2015.
Ullman, D. G. (2010), *The Mechanical design process,* Boston, McGraw-Hill.
Ulrich, K. T. and Eppinger, S. D. (2011), *Product design and development,* New York, NY, McGraw-Hill/Irwin.
Wenger, E. (2000), 'Communities of practice and social learning systems', Organization, 7(2), pp. 225–246. doi: 10.1177/135050840072002.
Wenger, E. C., and Snyder, W. M. (2000), Communities of practice: The organizational frontier. Harvard Business Review, 78(1), pp.139-146.
Woods, D. D. and Hollnagel, E. (2006), *Joint cognitive systems: patterns in cognitive systems engineering,* Boca Raton:, CRC/Taylor & Francis.

ACKNOWLEDGMENTS

The authors wish to thank Chalmers Innovation and AFA Försäkringar for providing the funding to carry out this research. We also wish to thank the growing ACD³ Community of Practice for helping us test, challenge and grow this framework into something useful.