CHOOSING THE RIGHT D FOR DESIGN

Joseph T. Foley, Marcel Kyas

School of Technology, Reykjavík University, Reykjavík, Iceland

ABSTRACT

In the current CDIO V2.1 standard (2016), there is no formal definition of what design is or what process should be employed. Instead, it is left to the educator to figure out what is the right way to proceed. Among philosophers of design, there is no agreement on the nature of design processes. Design is often taught as an iterative method taking a developed list of requirements and trying different combinations of elements until a satisfactory solution is found. Knowing which elements are worth investigating is often said to be only gained through reading background material and experience. There are alternatives in the form of formalized design methods, including Axiomatic Design and Google's Design Sprint. This paper presents an overview of these methods to provide opportunities in hybrid design frameworks for the CDIO educator. When properly informed, both students and teachers can choose or create the right D in CDIO to fit the project or discipline.

KEYWORDS

Design, Axiomatic Design, Sprint, Requirements, Standards: 5, 7, 8

INTRODUCTION

The second element in the CDIO name is "design", so one might assume a great deal of literature published on the subject and how it should be implemented. In the most recent CDIO proceedings, there are a few articles that give specific guidelines to using a particular method to design, such as Paul and Behjat (2016) explaining how to use SCRUM for an integrated design project. Tanarro et al. (2015) address the problem of teaching engineering design to a multidisciplinary audience. They leave the method of design open. What seems to be lacking is the meta-design phase of deciding what kind of design method to use. To make an informed decision, we must first consider what the CDIO standard states regarding design and suitable options to choose from. Perhaps the best question to start with is: "What does it mean to design?"

Background

While Crawley, Malmqvist, Östlund, Brodeur, and Edström (2014) mention *design* a lot, a clear definition of *design* is missing. The CDIO standards characterize design as (Standard 1, p. 293): "The Design stage focuses on creating the design, that is, the plans, drawings, and algorithms that describe what will be implemented."

Standard 4 (p. 296) suggests that: “[s]tudents engage in the practice of engineering through problem-solving and simple design exercises, individually and in teams.” Indeed, how to *design* is disputed.

Newell and Simon (1972) describes the design process as a sequence of discrete steps that are driven by a plan. The goal of designing is to optimize a candidate design for known constraints and objectives. Most design methods proposed in Section ”Traditional Design Methodologies” describe such sequences of steps.

Schon (1983), on the other hand, observes that designers do not follow such a model in practice. Indeed, such a model of discrete steps has been criticized in software engineering by Royce (1970) as infeasible for sufficiently complex projects. Indeed, the goals are often unknown when the design process begins and, and the constraints and objectives change (Brooks, 2010).

Today, software projects follow an *agile method* (Beck et al., 2001), which focus on customer interactions and short iterations to solve specific design problems. Design is instead viewed as a creative endeavor guided by intuition and emotion.

CDIO Standard

The CDIO standards recommend teaching about design (Standard 1) and provide design exercises (Standard 4). We report on different approaches to design, and how design exercises are used at Reykjavik University. Also, design-implement experiences should be included in the curriculum (Standard 5). Among others, the course on Internet of Things is such a design-implement exercise. This course is taught as a multi-disciplinary course (Standard 7) with mechanical engineering students, mechatronic students, computer science students, and software engineering students. This course exhibits that different approaches to design have to be addressed, and interpersonal skills have to be trained.

The course is also an active learning experience. After an initial guidance, the students are designing and implementing their solution, getting only the teacher’s guidance, and their support when requested.

TRADITIONAL DESIGN METHODOLOGIES

This section describes general design methods that are universally applicable in the sorts of projects commonly desired in courses. At an abstract level, the purpose of design is to move from an immaterial concept to something more concrete. C-K theory(Hatchuel & Weil, 2003) describes this as the process of mapping concepts to knowledge and examining the connections between these various nodes.[1] For use in a focused class trying to go from a concept to a prototype that can be operated, this method is too general for students to see how it applies to their specific design.

Instead, we consider the methods that share these commonalities: gathering stakeholder opin-

[1]It appears to be an application of Category Theory (Eilenberg & Mac Lane, 1945), where the term category for mathematical objects and their natural transformations got confused with Kant’s categories as concepts of knowledge (Kant, 1781).

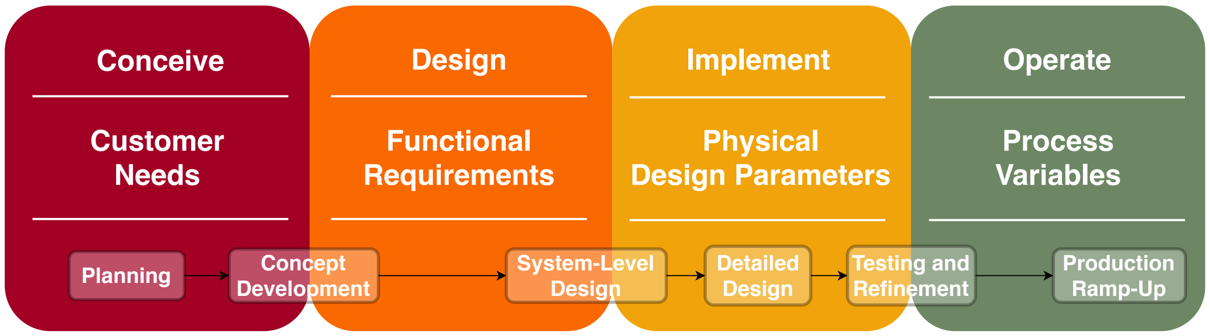

Figure 1. Mapping CDIO stages (top) to Axiomatic Design (middle) and Ulrich et al. (2019) product design process (bottom)

ion, developing requirements, refining concepts according to selection criteria, and developing a unified prototype. Each method within these constraints has a different focus which can make it more or less appropriate for a chosen CDIO project which we will attempt to describe by explaining how they are used in our curriculum.

Product Design

The Reykjavik University School of Technology Department of Engineering has been teaching the general product development process as described by Ulrich, Eppinger, and Yang (2019). This book is an industry-standard for product designers considering the end-to-end process of concept to implementation. The general process in Ulrich et al. (2019) is divided into these stages: Planning, Concept Development, System-Level Design, Detail Design, Testing and Refinement, and Production Ramp up. Of note, the Conceive-Design-Implement-Operate stages are inherently incorprated into this framework as shown in Figure 1

The design course curriculum[2] begins with students examining a problem or topic of interest and searching for stakeholders who will become “customers”. These customers become an integral part of refining the customer’s opinions into “Customer Needs”. Based upon these needs, the teams will benchmark existing products to develop competitive metrics and requirements. The requirements give insight into possible concepts, which are explored and refined for the rest of the semester. The deliverable artifacts are often a prototype, presentation, and paper suitable in quality and format for submission to a design conference.

Google Sprint

The Design Sprint was developed by Knapp, Zeratsky, and Kowitz (2016) at Google to design, implement, and evaluate a prototype of a problem solution in five days. Time-limited activities are scheduled each day. Starting from a challenge, a diverse team will work out a solution. On Monday, the team agrees on the goal and creates a map leading customers and other stakeholders towards the solution. The team consults with experts, collects problems, and selects a target, on that the sprint will focus. On Tuesday, the team collects old and new solution ideas to remix them and improve on them. The result of that day will be detailed solutions. On Wednesday, the team selects the best solution and makes a plan for the prototype. On

[2]which will become a requirement for all engineering lines in 2022

Thursday, the team will create a prototype, or if that takes too much time or effort, it creates a façade. On Friday, the prototype will be tested on five customers. The feedback will be used to plan the next steps.

The design sprint is not intended to be a complete design method. It is rather used to reduce the risk of bringing a new product to the market. It is inspired by design thinking (Asimow, 1962) and focuses on the user experience. The method may be used for other design tasks but will need to be adapted. Indeed, prototyping a product in one day limits its use.

Axiomatic Design

Axiomatic Design, developed by MIT Professor Nam Pyo Suh in the 1980's[3] also describes the design process as mapping but at a conceptual level, rather than a temporal one. As previously mentioned, this is a process focused on the mapping between domains. These domains (listed below) have a clear mapping to the CDIO framework as shown in Figure 1: Customer Needs, Functional Requirements, (Design) Parameters, Process Variables. Axiomatic Design has two Axioms which are believed to be inherent in all "good" design(Suh, 2001);

Independence Axiom: Maintain the independence of the Functional Requirements
Information Axiom: Minimize the information content of the design.

These can be translated into a less technical definition as:

Independence Axiom': Modular: minimize interference between requirements
Information Axiom': Robust: Choose implementations that maximize chances of meeting requirements

The general process is to develop a list of up to 7 elements in a domain, then move to map those elements to new elements in the next domain. Axiomatic Design desires solution-agnostic Functional Requirements which allow for further creativity. A customer need (CN) such as "keep my roof attached to my house" might become the functional requirement (FR) "bond roofing material to rafters" which would be mapped to "polyurethane-based glue applied to both surfaces" resulting in a process variable (PV) "Minimum curing of 24 hours at 25 degrees celsius". At each stage of the mapping, interactions *i.e.* couplings between the mappings must be examined: the optimal configuration is that each functional requirement is only affected by one design parameter as described by the Independence Axiom. Coupling is evaluated by making a Cartesian product of each domain element's transfer coefficient in a matrix called a Design Matrix. This matrix gives a simple mathematical expression that can be evaluated for the degree and type of coupling. See (Suh, 2001) for a more detailed discussion of analyzing design matrices.

HYBRID DESIGN METHODS

As mentioned previously, design methods are not necessarily a tool to be used in isolation. We provide in this section adaptations of the previous methods for new environments particularly cross-disciplinary.

[3] Conceived at Burger King in Cambridge Massachusetts

Collective System Design

While David Cochran was at MIT, he was exposed to Axiomatic Design and wished to spread the innovative way of thinking to fields such as Automotive manufacturing. Unfortunately, the technical nature of how it is described the taught is daunting to the average management or non-engineer. He decided to take the basic concepts of Axiomatic Design and make them more approachable to a non-technical manufacturing audience which resulted in the creation of the Manufacturing System Design Decomposition (Cochran, Foley, & Bi, 2016; Suh, Cochran, & Lima, 1998) leading to the more general Collective System Design process (Cochran, Smith, Sereno, Aldrich, & Highly, 2019). In the new system, Design Parameters became Problem Solutions. Both Functional Requirements and Problem Solutions have a metric that is used as an explicit test and/or target. He also realized that the design matrix was very hard for many people to intuitively understand, so instead, a tree structure is used to encode the same information. The final touch was to put the process in the context of a "Flame model" that sets the tone of why design is important as a collective agreement: 1. Standard work/actions 2. Structure 3. Thinking 4. Tone. In the collective system design process, diagnosis starts with standard work and drills down to the tone $(1 \rightarrow 4)$. At this point, the management of the organization makes a "conscious choice to change" involving workers at all levels. The design phase begins with this tone and works its way back to standard work $(4 \rightarrow 1)$.

This new "view" on Axiomatic Design has gained much traction in the automotive manufacturing industry as a method for becoming more "lean" Cochran et al. (2019)" and is an alternative to the INCOSE Systems Engineering process (INCOSE, 2015).

ADAPT: Axiomatic Design and Agile

In 2017, Jakob Weber and his team at Daimler AG (Mercedes-Benz Research and Development) began to adopt Axiomatic Design in a format that made more sense for how they designed automotive manufacturing in a "turbulent setting"[4]. They noticed that Axiomatic Design provided a top-level strategy for high-level goals in a research manufacturing project but did not indicate how to arrange the tasks and work. There was a realization that this missing component can easily be filled by Agile methods such as SCRUM. In the new method Weber, Förster, Stäbler, and Paetzold (2017) described, Axiomatic Design would first identify the design opportunities and implementation goals, then these Functional Requirement-Design Parameter pairings would become tasks for the product backlog. After a SCRUM had been completed, the information (and incomplete tasks) would be refactored into the FR-DP mappings to build a new design matrix (or design decomposition) and the cycle would continue. Joining these two provided a very useful guideline for Axiomatic Design: the right amount of decomposition is when an FR-DP pairing is a task that can be approached during a SCRUM period with an educated guess of its completion time. ADAPT became successful enough that additional enhancements were implemented including modularity indexing for sequencing and priorization (Kujawa, Weber, Puik, & Paetzold, 2018).

[4]Previously they had done so in small projects, but it had not gained wide-spread interest within the company (Weber, Förster, Kößler, & Paetzold, 2015)

Google Sprint and Axiomatic Design

In the academic year 2019, the two authors have taught the course "Introduction to Internet of Things and Embedded Systems" together. This course integrates the design of Things (embedded devices) with the design of software systems (connecting Things through the internet). This course also required integrating design methods as well: mechatronics students had previously learned about Axiomatic Design, while students of computer science and software engineering were only familiar with agile methods.

The authors discovered that While both methods have different origins, they work well together. The Sprint method provided stream-lined methods to explore the customer domain and the functional domain, while Axiomatic Design helped in structuring the requirements and building the prototype.

APPLICATION OF DESIGN SELECTION

Internet of Things Class

During the last four years, the second author taught a course on designing Internet of Thing (IoT) applications to computer science and software engineering students at Reykjavik University. The class is taught as a three-week block, in which the first week is devoted to examples of IoT, catching up on embedded programming practices, and the study of network protocols used in IoT systems. During the second week, students design an IoT system. During the third week, that design is prototyped and evaluated. Applications ranged from control systems for cooking sous vide to network-connected picture frames and smart food containers. During the term 2019, the authors taught the course as a cross-disciplinary course to students of mechatronics, computer science, and software engineering. The students that took the class learned different design methods in previous courses.

The authors have used the Google Design Sprint (see Section "Google Sprint" above). It works well to solicit requirements and test the viability of a product or solution. Students of mechanical engineering and mechatronics have had more issues adapting to the sprint because of their different experience and design approach.

Mechatronics

The inherent multi-disciplinary nature of Mechatronics[5] means that techniques from software, electronics, and mechanical engineering must be considered. For the last four years, the first author has been teaching a short introduction to Axiomatic Design as part of the course. Similar to the experience in the IoT class, this technique has resonated with students with a mechanical engineering competency, but often software-facing students do not understand the need for a formal process. These students would prefer a more Agile-style environment where there is an iterative rapid-prototyping mindset rather than trying to formalize a concept and requirements initially. The second author is considering a hybrid AD-SCRUM design method to satisfy both groups, similar to the ADAPT method developed by Weber et al. (2017).

[5]Often called embedded control systems or robotics in other curriculula

CONCLUSION

The CDIO standard gives a great deal of freedom in implementing a practical hands-on education style, particularly in the design area. With such a large realm of possibilities, it is comforting for both the educator and student to have standardized methods to employ. We have presented several generic design methods that we have found applicable in a large variety of student projects and labs. The methods share the same general idea of mapping an ethereal concept into a more concrete simulation or prototype that can then be evaluated and improved. Often the methods are optimized for a particular discipline such as mechanical engineering or software development. In multi-disciplinary teams, the one-size-fits-all approach of the ”generic” design method does not fit. A better approach was to mix the two design methods by realizing that they are different views of the same general process and educating the students about this universality. The need for this adaptation is present in large manufacturing industries demonstrated by the deployment of ADAPT at Mercedes-Benz’s Research and Development department(Kujawa et al., 2018). To conclude, there is no one clear D for *all* Design, but we believe the methods presented are a reasonable place to start.

REFERENCES

Asimow, M. (1962). *Introduction to design*. Englewood Cliffs, NJ: Prentice-Hall.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., … Thomas, D. (2001). *Manifesto for agile software development.* Web page. Retrieved from https://agilemanifesto.org/

Brooks, J., Frederick P. (2010). *The design of design: Essays from a computer scientist*. Addison-Wesley.

Cochran, D. S., Foley, J. T., & Bi, Z. (2016). Use of the Manufacturing System Design Decomposition for Comparative Analysis and Effective Design of Production Systems. *International Journal of Production Research*, *55*, 870–890.

Cochran, D. S., Smith, J., Sereno, R., Aldrich, W., & Highly, A. (2019). How to develop and sustain a lean organization through the use of collective system design. In H. B. Nembhard, E. A. Cudney, & K. M. Coperich (Eds.), *Emerging frontiers in industrial and systems engineering: Success through collaboration* (pp. 129–148). CRC Press.

Crawley, E. F., Malmqvist, J., Östlund, S., Brodeur, D. B., & Edström, K. (2014). *Rethinking engineering education: The CDIO approach* (2nd ed.). Springer. doi: 10.1007/978-3-319-05561-9

Eilenberg, S., & Mac Lane, S. (1945, September). General theory of natural equivalences. *Transactions of the American Mathematical Society*, *58*(2), 231–294. doi: 10.2307/1990284

Hatchuel, A., & Weil, B. (2003, August 19–21). A new approach of innovative design: An introduction to c-k theory. *Proceedings of the International Conference on Engineering Design ICED ’03*.

INCOSE. (2015). *Systems engineering handbook: A guide for system life cycle processes and activities, version 4.0*. Hoboken, NJ: John Wiley and Sons.

Kant, I. (1781). *Critik der reinen Vernunft*. Riga: Johann Friedrich Hartknoch.

Knapp, J., Zeratsky, J., & Kowitz, B. (2016). *Sprint: How to solve big problems and test new ideas in just five days*. New York: Simon & Schuster.

Kujawa, K., Weber, J., Puik, E., & Paetzold, K. (2018). Exploring and Adapt! – Extending the Adapt! Method to Develop Reconfigurable Manufacturing Systems. In E. Puik, J. T. Foley, D. Cochran, & M. Betasolo (Eds.), *12th International Conference on Axiomatic Design (ICAD)* (Vol. 223, p. 01006). Reykjavík, Iceland: MATEC Web of Conferences. Retrieved 2019-04-25, from https:/doi.org/10.1051/matecconf/201822301006 (October. 9–11) doi: 10.1051/matecconf/201822301006

Newell, A., & Simon, H. A. (1972). *Human problem solving*. Prentice Hall.

Paul, R., & Behjat, L. (2016). Using principles of SCRUM project management in an integrated design project. In (pp. 716–729).

Royce, W. W. (1970, August). Managing the development of large software systems. In *Proceedings IEEE WESCON* (pp. 328–338).

Schon, D. A. (1983). *The reflective practitioner: How professionals think in action*. Basic Books.

Suh, N. P. (2001). *Axiomatic Design - Advances and Applications*. Oxford University Press.

Suh, N. P., Cochran, D. S., & Lima, P. C. (1998). Manufacturing System Design. In *48th General Assembly of College International Pour La Recherche en Productique (CIRP), Annals of the CIRP* (Vol. 47, pp. 627–639).

Tanarro, E. C., Munoz-Guijosa, J. M., Lantada, A. D., Wiña, P. L., Otero, J. E., Sanz, J. L. M., … Mondéjar, S. G. (2015, June). Teaching engineering design to a multidisciplinary audience at master's level: Benefits and challenges of the cdio approach. In J. Björkqvist et al. (Eds.), *Proceedings of the 11th international cdio conference.* Chengdu, Sichuan, P.R. China: CDIO Initiative.

Ulrich, K., Eppinger, S., & Yang, M. C. (2019). *Product design and development* (7th ed.). New York, NY: McGraw-Hill Education.

Weber, J., Förster, D., Kößler, J., & Paetzold, K. (2015). Design of changeable production units within the automotive sector with axiomatic design. In M. K. Thompson, A. Giorgetti, P. Citti, D. Matt, & N. P. Suh (Eds.), *9th International Conference on Axiomatic Design (ICAD)* (Vol. 34, pp. 93–97). Florence, Italy: Elsevier ScienceDirect. (Sep. 16–18) doi: 10.1016/j.procir.2015.07.061

Weber, J., Förster, D., Stäbler, M., & Paetzold, K. (2017). Adapt! — agile project management supported by axiomatic design. In O. Dodoun (Ed.), *11th International Conference on Axiomatic Design (ICAD)* (p. 01018). Iasi, Romania: MATEC Web of Conferences. (Sep. 15–18)

BIOGRAPHICAL INFORMATION

Joseph T. Foley is an Assistant Professor in the School of Technology, Department of Engineering at Reykjavík University in Reykjavík, Iceland. He is on the Axiomatic Design scientific committee and the Axiomatic Design Research Foundation board. His current research focuses on mechatronic and mechanical design, emphasizing integrating Axiomatic Design and CDIO principles into the teaching curriculum. These principles are being integrated into a product design capstone course for all engineering students.

Marcel Kyas is an Assistant Professor in the Shcool of Technology, Department of Computer Science at Reykjavik University in Reykjavik, Iceland. He graduated from Christian Albrechts Universit ”at zu Kiel in 2001, and received his Ph.D. from Leiden University in 2006. Previously, he taught at the University of Kiel, University of Oslo, and Freie Universität Berlin. His current research focuses on ambient assisted living, indoor positioning, and the design of safe and secure embedded systems. Lately, he got interested in sustainable computing, looking at the resource costs of software. He teaches in the form of project-based courses.

Corresponding author

Joseph T. Foley
Reykjavik University
Menntavegur 1
Reykjavik 102
Iceland
foley@RU.IS