2021 ASEE ANNUAL CONFERENCE
Virtual Meeting | July 26–29, 2021 | Pacific Daylight Time
Paper ID #34455

Work in Progress: On Teaching Requirements in Engineering Design

Mr. Alexander Pagano, University of Illinois at Urbana - Champaign

Alex Pagano is a PhD student studying engineering design. His work is focused on the early phases of design and the use of human-centered design or design thinking as a teaching tool. Alex holds a BS in Materials Science and Engineering from University of Arizona and a MS in Mechanical Science and Engineering from University of Illinois at Urbana - Champaign

Work-in-progress: On teaching requirements in engineering design

I. Introduction and Background

Novice designers must be given the opportunity to develop early-phase design knowledge, skills, and abilities (KSA) such as those used in requirements development. Well-developed requirements have the potential to frame unmet needs in a way that facilitates collaborative engineering problem solving, helping to reduce the gap between undergraduate engineering theoretical knowledge, and real-world problems. This paper presents findings from a review of literature that discusses requirements development and associated pedagogies and combines various perspectives from systems engineering, software engineering, human-centered design and design thinking, and others. The goal of this work is to identify opportunities to support undergraduate learning of early-phase design KSA through requirements development. In this first Section, a short overview of requirements development in engineering design is provided. In Section 2, the role and importance of requirements are discussed by considering how requirements are classified, created, and used throughout design. This includes a discussion of the influence of social dynamics on the design process and a brief overview of literature on the impact of requirements development. Section 3 gives insights into how requirements are taught in undergraduate engineering classes, and Section 4 provides a summary and overview of future work.

As stated plainly by Dym *et al.*, "Engineering design is conducted with imperfect models, incomplete information, and often with ambiguous objectives as well" [1]. Yet novice designers may approach a design problem as if it were straightforward and well-structured, acting prematurely to generate and commit to solutions [2]. Comparative studies between novice and expert designers suggest that novices spend less time on problem scoping and information gathering than experts do [3]. This could be influenced by a lack of familiarity with the early phases of engineering design which, compared to analytical methods, represents a very small portion of undergraduate engineering curricula. Instead, students may approach design problem-solving as they do textbook problem sets [4]–[6] where all necessary information is provided and the correct solution can always be found if the right methods are applied. In the early design phases however, it's the designers who are responsible for determining the scope of the problem, gathering the necessary information, and framing it such that analytical methods may be used to converge to an optimal design. An important part of this process is the development of engineering requirements or specifications which provide unambiguous targets for the system to be designed.

This process is ubiquitous in engineering design, though requirements statements may not be discussed explicitly. The variety in the presentation of this process can be seen in the early phases of engineering design process models, as reviewed by Wynn and Clarkson [7], where it is described as; *problem definition* [8], *clarifying the client's requirements* [9], s*tatement of the problem* [10], *clarifying the task* and *product planning* [11], [12], *preparation of problem*

assignment [13], *functional requirements* [14]. In software engineering design, the whole process is referred to as Requirements Engineering (RE), though RE is rooted in systems engineering and applies more broadly than just software-intensive projects [15]. In Human-Centered Design and Design Thinking, requirements development is intrinsically tied to the *Understand (Empathize)* and *Synthesize (Define)* phases where unmet needs are explored [16], [17] though requirements are not necessarily discussed explicitly. Similar information gathering through interviews, observations and related tasks is also described as *ethnographic* research, a term which is derived from anthropology but is increasingly common in engineering design education [18]. Crismond and Adams include *Understand the Challenge* and *Build Knowledge* as design strategies in their Design Teaching and Learning Matrix [2]. *Developing requirements* is also included in the ABET definition of what constitutes engineering design [19].

This variety may contribute to the breadth of information available, but it makes it hard to quickly introduce novice designers to relevant concepts and methods. The lack of universality means that without accessible descriptions and examples, it is easy to get lost, and searching available resources for any of the above terms will not give the same consistency and clarity as other engineering topics (e.g. compare the search results for *"Navier-Stokes"* with those of *"Problem Definition").* While there are ISO standards for Requirements Engineering [20], and codified best practices such as CMMI [21], these are not universally employed. Nonetheless, there are commonalities in requirements development. Typically, '*requirements*' refers to a collection of written statements which outline what is required of the solution. An example of a written requirement might be; *the chair shall withstand a static load of 300 lbs.* A requirements statement must clearly and unambiguously express an unmet need in order to enable the designer to address it [20]. Poorly constructed requirements can leave designers with an improper understanding of the problem and will greatly diminish the possibility of truly addressing the underlying need. For example, changing the requirement above by replacing the word '*shall'* with '*should*' and removing the clarification of loading condition (i.e. static load), the requirement becomes; *the chair should withstand a load of 300 lbs*, which is much more ambiguous. Is this capacity truly required, or simply a suggestion? Would a chair with a lower carrying capacity be sufficient if it can withstand a higher instantaneous load? Cleary, the structure, and content of requirements statements carry significant implications for the direction of the design, its outcomes and ultimately its success.

II. The role and importance of requirements

Any system will have a variety of requirements which specify different things, for example, operational conditions, external interfaces, regulatory compliance, or time-to-market. Different kinds of requirements will serve different roles in the design process, and while realistically requirements occupy a spectrum it may be useful to consider common categorizations to explore their roles. These categories are *functional requirements* vs. *non-functional requirements, stakeholder requirements* vs. *system requirements,* and *high-level requirements* vs. *low-level requirements*. Functional requirements are those which define what the system must do, whereas non-functional requirements (aka quality requirements) often define qualities of the system [15], [22]. Qualities such as comfort, longevity, maintainability, portability, and manufacturability

could be specified as non-functional requirements (note that many share the suffix *-ility*) [22]. Non-functional requirements are often poorly addressed in engineering design, which may be in part because they are often harder to measure objectively [23], [24]. Interestingly, in a case study analysis of an automotive OEM, Shankar *et al*. found that non-functional requirements act as goals for the design and constraints on the solution, driving design decisions [25]. Seemingly, non-functional requirements can define the niche that the design must occupy such that it can be feasibly produced by a particular engineering and manufacturing group to meet the needs of a particular stakeholder. This subjectivity can be illustrated by considering the requirements imposed by all relevant people or groups who are impacted by the design, such as users, clients, customers, manufacturers, distributers, etc., collectively referred to as *stakeholders*. For example, a driver may be concerned about how responsive the steering feels, and a mechanic may be concerned about how easy it is to reach bolts in the engine bay. In systems engineering non-functional requirement derived from the stakeholder's needs would be captured in the *stakeholder requirements* which would then inform the *system requirements* [15], [22]. Of course, not all stated requirements can be satisfied, and the chosen solution is often the one which 'satisfices' the highest priority requirements. Stakeholder requirements represent what stakeholders want and define unmet needs, while system requirements define the constraints and capabilities of a system able to satisfy those needs. System requirements are typically derived in response to stakeholder requirements. Stakeholder requirements may be functional or non-functional, but since they are a representation of what the stakeholders want, they should be written in non-technical language such that all stakeholders can understand them. Similarly, these objectives, goals, aims, aspirations, expectations, or needs often associated with stakeholders may be described as high-level requirements [15]. While high-level requirements are more likely to apply to the whole system, low-level requirements may apply to subsystems, assemblies, components, or specific interactions therein. Since the details of the design are often unknowable at the outset, low-level requirements are typically derived from or informed by high-level requirements [15], [26].

Throughout the design process, requirements may be developed and utilized by various groups. For example, a mechanical engineer working in a large company may be most immediately constrained by the low-level, functional system requirements of a specific component. However, this engineer's design will also be constrained by the stakeholders impacted by their decisions, e.g. machinists, other groups working on interfacing components, senior engineers, or project managers, and they in turn may be responsible for satisfying higher-level requirements. The interconnectivity of requirements is an inevitable result of the iterative nature of design and the complexity of systems. As concepts are developed, new constraints are imposed to ensure that the parts of the system work together as intended. If these dependencies are not well-understood, then changes in one part of the system can easily cause larger failures, for example upsizing a motor could cause a system-wide power failure unless the power supply and can meet the new demand. The ability to trace the connections between requirements throughout the system, to stakeholders, and among different hierarchical levels is called requirements *traceability* [15], [22], [23], [27]–[31]. In the previous example, the impact of changing the motor can be traced backwards to high-level systems or behaviors. This is called *backwards traceability*, and it is useful for understanding the implications of component-level changes on the larger system as

well as explaining *why* a low-level solution was implemented by tracing back to the related high-level requirement [31]. Inversely, *forward traceability* is the ability to draw connections from the high-level requirements to the low-level solutions that implement it, explaining *how* a need is met by the system [31]. Requirements management software may facilitate requirements tracing, though these tools are beyond the scope of this work and may not be publicly available (see DOORS in [15]).

The flow of requirements in a design process is well-represented by the V-model of systems engineering shown in Figure 1 [15], [22], [32]. Design progresses, moving down the left side of the vee, as stakeholder requirements are developed in response to unmet needs. These requirements are satisfied by the system, which is abstractly represented in the conceptual design (aka preliminary design) and is specified in the system requirements. The system is often broken up into subsystems whose relationships are described by the architectural design and specified in the subsystem requirements, which are in turn addressed at the component level. The right side of the vee relates to testing solutions against the stated requirements. Each level of the system should be evaluated to determine if it satisfies the associated requirements. Evaluation should be done for two purposes, *validation,* and *verification*. Validation addresses the question, "are you building the right system?", while verification addresses, "are you building the system right?" [26]. These questions are paralleled in descriptions of the two parts of a Human-Centered Design process, *getting the right idea* and *getting the idea right*, hinting at the potential for the applicability of HCD to requirements development. Design concepts and requirement statements should both be checked to ensure they are valid and verified. A valid design concept is appropriate for the application, for example a stepladder and an extension ladder both provide the functionality of reaching a higher elevation, but a stepladder is more appropriate for use in the kitchen. Similarly, the requirements statement specifying this functionality would not be valid if it fails to specify the context for use. Conversely, the stepladder concept should be

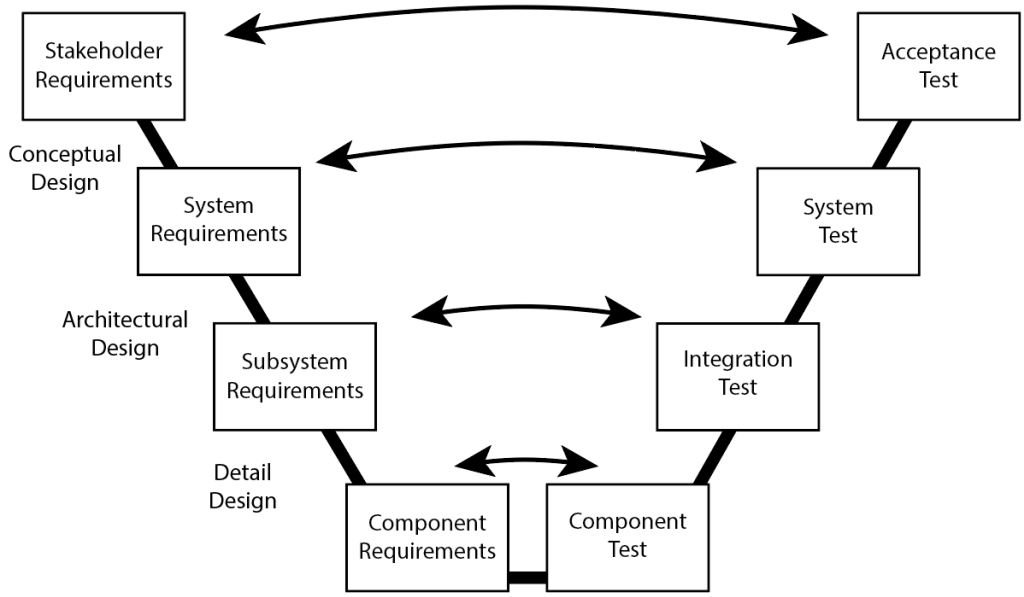

Figure 1: V-Model showing the development of requirements in an engineering design process.

verified to satisfy the requirement, i.e. that it allows the use to reach the correct height. The requirements statement specifying this height requirement should similarly be verified to confirm that the stated height is correct and includes units of measurement. Requirements Engineering standards specify that requirements should be *necessary, appropriate, unambiguous, complete, singular, feasible, verifiable, correct,* and *conforming* [20]. Likewise, sets of requirements, such as all stakeholder requirements, or all requirements for a subsystem should be, *complete, consistent, feasible, comprehensible,* and *able to be validated* (for definitions, see the cited standards) [20]. These metrics provide a solid basis for assessing the quality of requirements statements but do little to prepare a novice engineer to draft them.

Social and collaborative considerations

Requirements development is a creative and iterative process that is heavily dependent on communication and is therefore significantly impacted by social dynamics. The influence of social dynamics on engineering practice has been acknowledged in the literature as "heterogenous engineering", though despite engineering work being done by people, within social constructs, to serve human needs, there is often a distancing of these social influences from the technical ones, which may be perceived as the "real engineering" work [33]. Elena *et. al* report findings from an exploratory case study which builds upon anthropological definitions of culture through the expression of communication, artifacts and norms to identify evidence of a "Requirements Culture" in the engineering workplace [34]. The role of social dynamics in engineering design was also explored by Minneman who concluded that engineering design is fundamentally comprised of the series of social interactions among designers and prescriptive process models fail to describe the majority of design work [35]. Given the social underpinnings of requirements development, this is likely a realistic view to take. While prescriptive design methodologies may not properly map to every realistic instance of design, they are still useful for communication and instruction, providing structure and reference for how the tasks and deliverables may relate. This process of communicating with stakeholders to ensure understanding, resolve conflicts, and reach consensus is referred to as requirements elicitation in RE to infer the complexity and effort involved. Stakeholder requirements may be conflicting, infeasible, ambiguous, or otherwise improper, and it is likely not appropriate to treat all statements from stakeholders equally. One approach to structuring this interaction is to create an operational scenario (comparable to Concept of Operations [20], and Usage model [15]) which describes the necessary steps to achieve the desired goal [22]. By walking through these steps and asking stakeholders how the system should behave at each moment, and under various conditions, rough requirements statements can be devised. These should be developed collaboratively with the stakeholders to ensure that the designer's bias and assumptions do not influence the design requirements. Methods from HCD are extremely relevant here, for example, design teams conducting interviews could identify personas and create visual frameworks to explore the priorities of different stakeholder groups. Once stakeholder requirements are complete and valid, they should be formally approved by the client to ensure that they properly frame the problem to be solved and establish expectations. Then the design team must define system requirements informed by these stakeholder requirements. To do this, abstract models of the system should be developed so that the system can be explored without prematurely

specifying a solution and precluding other potential design directions. There are a variety of ways to model the system abstractly, such as functional decomposition, block diagrams, and flow charts. Unified Modeling Language (UML) is a collection of these types of models [15]. These abstract models facilitate the development of system requirements by providing a framework for their organization. For example, the behaviors outlined in the operational sequence developed with the stakeholders could be mapped to system functions outlined in a functional decomposition, and the functional requirements organized according to this hierarchy. These models and the requirements which extend them can then be used to provide structure by prompting and framing concept generation, providing assessment criteria for concept selection, and facilitating verification and validation of the system. While this example procedure may provide a useful framework for understanding how requirements may arise, additional work is needed to understand the mechanisms by which good requirements are developed and how they impact the design process.

Impact of requirements

Much of the literature on requirements development speaks to the importance of good requirements for successful engineering projects, often turning to retrospectives on failed projects to illustrate mishandled requirements [36], [37]. While the definition of a failed project is a point of discussion, we will follow Pinto *et al.* and relate the measure of success or failure of a project with three factors; the implementation process itself, the perceived value of the project and client satisfaction with the delivered project [38]. Well-developed requirements have the potential to support all three of these factors. A survey of project managers and requirements engineers found that "Good cooperation between development team and client" and "Good requirements" were the two qualities most beneficial to projects [37]. The Standish Group's CHAOS reports also emphasize the importance of user involvement and complete requirements, and pre-2000 reports are frequently used to illustrate the connection between requirements and project success [15], [29], [37]. Newer reports frame the factors differently, yet still point to a clear connection between user or stakeholder needs and clearly stated requirements, recommending that projects focus "on a narrow set of features and requirements, which users find easier to understand and absorb" [39]. Bahill and Henderson reflect on famous failures to consider if poor requirements development, verification and validation were contributing factors, finding plenty of examples where mishandled requirements are evident [26]. Despite these accounts, many of the potential benefits of requirements remain unsupported claims. We believe additional work is needed to describe the true impact of requirements development activities. Some of these potential benefits are outlined in the Table 1 below, without references. This list is not exhaustive, and it is the author's intent to include supporting references in future iterations of this work-in-progress paper.

Table 1: Potential Benefits of Requirements

Stakeholder Requirements	**System Requirements**
Make sure you're solving the right problem (validation)	Make sure you're solving the problem right (verification)
Inform priorities and resolve tradeoffs	Planning, scheduling, budgeting, risk management

Create a systematic overview of the problem to be solved so that solutions do not diverge	Establish acceptance criteria
Roots success in the satisfaction of unmet needs (prevent over-engineering, feature bloat/creep)	Represent the system abstractly, revealing dependencies and hierarchies
Engaging stakeholders in the process facilitates buy-in, and can improve collaboration	Inform concept evaluation/selection
Enable a wider view by getting more perspectives at the table (anticipate impact/risk management)	Enable the tracing of needs to solutions
Inspire confidence by supporting work with clear documentation	Frame problem-solving to facilitate ideation
Facilitate communication by providing a basis for discussion	Enable exploration of the design space by promoting solution-neutral framing

III. Requirements in Engineering Design Education

The importance of requirements development in undergraduate education is underscored by its explicit inclusion in the ABET description of what constitutes engineering design.

> "Engineering design is a process of devising a system, component, or process to meet desired needs and specifications within constraints. It is an iterative, creative, decision-making process in which the basic sciences, mathematics, and engineering sciences are applied to convert resources into solutions. Engineering design involves identifying opportunities, developing requirements, performing analysis and synthesis, generating multiple solutions, evaluating solutions against requirements, considering risks, and making trade- offs, for the purpose of obtaining a high-quality solution under the given circumstances." (formatted for emphasis) [19].

Engineering design courses, such as cornerstone and capstone courses, can provide students with experiential learning opportunities which approximate the real-world problems they will face after graduation by employing a project-based or problem-based learning (PBL) model [1] (note that project-based and problem-based learning started as distinct pedagogies, but are conflated commonly enough that they are practically interchangeable). Experiential learning may promote the transfer of learning, such that skills, knowledge, and abilities (KSA) developed through similar experiences may be more readily applied in new situations [1], [40]. Additionally, open-ended design problems involve iterative cycles of both convergent and divergent thinking which often utilize, and expand upon analytical methods taught in the engineering sciences [1]. In these PBL design projects, students work collaboratively in small teams and are subjected to inherent social influences and nuances which complicate real-world problem solving. Requirements development has long been included in capstone curricula. In a review of literature on early engineering capstone programs, "Needs Analysis" and "Requirements Definition" are included in a list of typical lecture topics [41], [42]. Elena *et al.* investigated the impact of a 50-minute lecture on the requirement statements generated by Mechanical engineering students in a capstone design course, finding that such an intervention can have positive impact on the novelty and variety of requirements generated [43]. Requirements are also included in sample capstone deliverables as "Functional Specification" [44] and "System Specs" [42]. However, limited exposure to this process in the senior year may not be sufficient. Joshi *et al.* reviewed 10 design

textbooks commonly used in undergraduate mechanical engineering courses and found that the majority of these do not mention the use of requirements in the "planning and clarifying stage of the design process." Interestingly though, the use of requirements is mentioned explicitly and implicitly in later design phases, though students are, "left to assess, under their own judgment, how requirements should be used within design tools" [45]. Developing non-technical or soft-skills, such as those necessary for requirements development, is a well-documented need for success in engineering industry [33], [41], [44], [46]. While engineering design experiences in the freshman and senior year may contribute meaningfully to the development of design KSA, there is a disconnect between these design experiences and the engineering sciences which are often the focus of the second- and third-year curricula. In their historical view of engineering design education, Froyd *et al.* describe this as creating, "a gulf between student experiences with engineering design in the first year and the capstone culminating experience" [47]. One possible reason for this is the difficulty in creating authentic design experiences which capture the technical learning objectives of engineering science courses. This is supported by Hung's chapter "Problem Design in PBL" in [40], which states, "However, as discussed, hierarchically well-organized content knowledge structure does not fit well in ill-structured real-life problems, which are the type of problems to be used in PBL (Barrows, 1994; Hung, Jonassen, & Liu, 2008). It is almost impossible to have a real-life problem that could perfectly afford the set of learning objectives in a given module" (original citation included). In response to this difficulty, design problems may be artificially constrained such that design deliverables align with learning objectives, though this means that students are no longer responsible for developing these constraints themselves. Conversely, design learning through open-ended design problems which involve real stakeholders, such as those in service-oriented projects and human-centered design projects, has been recognized to promote 21st century mindsets related to empathy, collaboration, creativity, experimentation, metacognition, and communication [48]–[50]. There is notable overlap between these and the skills suggested for RE courses such as, interview, groupwork, facilitation, negation, analytical, problem solving, presentation and modelling [28]. RE courses have since become more common in software engineering programs and Requirements Engineering Education has grown as a discipline. For a systematic review of this literature, see the work of Ouhbi *et al.* in [51]. Fundamentally, it is difficult to give all students the experience of interacting with stakeholders and attempts to simulate these real-world experiences in the classroom, through role-playing or gamification, while intriguing, may have limited impact [52], [53]. Though progress has been made, there is still a crucial need to better support requirements development in practice [51].

IV. Summary and Future Work

This work-in-progress paper presents a review of literature related to requirements development in engineering design. The lack of consistency in the way requirements are described makes it difficult to become familiar with the relevant tools and concepts. Thus, the intention of this paper is to combine various perspectives from systems engineering, software engineering, human-centered design and design thinking, and others and present these such that the role and importance of requirements in engineering design can be more easily understood. In summary, requirements statements describe what is required of the solution, and therefore provide an

unambiguous reference for the objective of a design process. They must be written carefully, validated, and verified to ensure the information they provide is accurate and complete, and while standards exist for the formatting of requirements statements, these are not always followed. Requirements are developed to translate unmet needs and often need to be iteratively refined to properly capture the problem at hand. While engineering systems are frequently described in terms of their functionality, non-functional requirements are also critical for delivering the expected result to the stakeholders. Stakeholder requirements serve a different purpose than system requirements and should be addressed first, documented separately, and used to inform the system requirements. Uncovering and documenting stakeholder requirements is an iterative and collaborative process that is significantly impact by social dynamics and quality of communication. While the role of social dynamics in engineering work has been documented, its influence is largely neglected. This presents a significant challenge to understanding the mechanisms by which good requirements are developed and the impact these have on design outcomes. Even without this theoretical understanding, requirements are commonly described as critical for the success of engineering projects, and developing requirements is a necessary skill for undergraduate engineering education. Unfortunately, open-ended design problems which necessitate the kind of problem-scoping work done during requirements development often don't fit within the confines of a PBL project in a standard semester-long engineering course. Thus, additional pedagogies are needed to support undergraduate's ability to engage in realistic open-ended design activities, such as requirements development. This work-in-progress paper will be extended to include a more thorough overview of pedagogies used to teach and assess requirements development skills in engineering design courses, including RE courses in software engineering. Additionally, the potential benefits of requirements will be substantiated with empirical studies, and where none are found, gaps will be identified for future work. It is expected that the role of social dynamics in requirements engineering will be underexplored, yet important for the practical use of this body of knowledge. This paper will support future work on the impact of requirements engineering education at the undergraduate level, as well as informing frameworks for understanding professional requirements engineering work.

References

[1] C. L. Dym, A. M. Agogino, O. Eris, D. D. Frey, and L. J. Leifer, "Engineering design thinking, teaching, and learning," *J. Eng. Educ.*, vol. 34, no. 1, pp. 65–65, 2006.

[2] D. P. Crismond and R. S. Adams, "The informed design teaching and learning matrix," *J. Eng. Educ.*, vol. 101, no. 4, pp. 738–797, Oct. 2012.

[3] C. J. Atman *et al.*, "Engineering Design Processes: A Comparison of Students and Expert Practitioners," *J. Eng. Educ.*, vol. 96, no. 4, pp. 359–379, 2007.

[4] N. J. McNeill, E. P. Douglas, M. Koro-Ljungberg, D. J. Therriault, and I. Krause, "Undergraduate Students' Beliefs about Engineering Problem Solving," *J. Eng. Educ.*, vol. 105, no. 4, pp. 560–584, 2016.

[5] C. J. Atman and K. M. Bursic, "Teaching engineering design: Can reading a textbook make a difference?," *Res. Eng. Des.*, vol. 8, no. 4, pp. 240–250, 1996.

[6] G. Rowland, “What do instructional designers actually do? An initial investigation of expert practice,” *Performace Improv. Q.*, vol. 5, no. 2, pp. 65–86, 1992.

[7] D. C. Wynn and P. J. Clarkson, “Process models in design and development,” *Res. Eng. Des.*, vol. 29, no. 2, pp. 161–202, 2018.

[8] C. L. Dym, P. Little, and E. J. Orwin, *Engineering design : a project-based introduction* , Fourth edi. New York: Wiley, 2014.

[9] C. L. Dym, *Engineering Design: A Synthesis of Views*. 1996.

[10] M. J. French, *Conceptual Design for Engineers*, vol. 80, no. 6. Berlin, Heidelberg: Springer Berlin Heidelberg, 1985.

[11] G. Pahl and W. Beitz, *Engineering Design: A systematic approach*. London: Springer London, 1996.

[12] J. Jänsch and H. Birkhofer, “The development of the guideline VDI 2221 - The change of direction,” *9th Int. Des. Conf. Des. 2006*, pp. 45–52, 2006.

[13] V. Hubka and W. Ernst Eder, “A scientific approach to engineering design,” *Des. Stud.*, vol. 8, no. 3, pp. 123–137, Jan. 1987.

[14] N. P. Suh, *The principles of design* . New York: Oxford University Press, 1990.

[15] E. Hull, K. Jackson, and J. Dick, *Requirements Engineering*, vol. 53, no. 9. London: Springer London, 2011.

[16] Stanford d.school, “Bootcamp Bootleg,” *Stanford d.school*, p. 47, 2010.

[17] L. Lawrence, S. Shehab, N. Tissenbaum, T. Rui, and T. Hixton, “Human-Centered Design Taxonomy: Case study application with novice, multidisciplinary designers,” in *Poster to be Presented at the American Education Research Association Virtual Conference*, 2021.

[18] I. Mohedas, S. Daly, and K. H. Sienko, “Design ethnography in capstone design: Investigating student use and perceptions,” *Int. J. Eng. Educ.*, vol. 30, no. 4, pp. 888–900, 2014.

[19] Board of Delegates Engineering Area Delegation, “Criteria for Accrediting Engineering Programs.” ABET, pp. 5–6, 2019.

[20] “ISO/IEC/IEEE International Standard - Systems and software engineering -- Life cycle processes -- Requirements engineering,” *ISO/IEC/IEEE 29148:2018(E)*. pp. 1–104, 2018.

[21] Software Engineering Institute, “CMMI for Development, Version 1.3,” *Softw. Eng. Process Manag. Progr.*, no. November, pp. 1–520, 2010.

[22] R. Stevens, *Systems Engineering: Coping with Complexity*. Prentice Hall, 1998.

[23] B. Nuseibeh and S. Easterbrook, “Requirements engineering: A Roadmap,” in *Proceedings of the conference on The future of Software engineering - ICSE ’00*, 2000, vol. 1, pp. 35–46.

[24] T. S. E. Maibaum, “Mathematical foundations of software engineering: A roadmap,” *Proc. Conf. Futur. Softw. Eng. ICSE 2000*, pp. 161–172, 2000.

[25] P. Shankar, B. Morkos, D. Yadav, and J. D. Summers, “Towards the formalization of non-functional requirements in conceptual design,” *Res. Eng. Des.*, vol. 31, no. 4, pp. 449–469, 2020.

[26] A. T. Bahill and S. J. Henderson, “Requirements Development, Verification, and Validation

exhibited in famous failures," *Syst. Eng.*, vol. 8, no. 1, pp. 1–14, 2005.

[27] M. Jarke, "Requirements Tracing," *Commun. ACM*, vol. 41, no. 12, pp. 32–36, 1998.

[28] L. Macaulay and J. Mylopoulos, "Requirements engineering: An educational dilemma," *Autom. Softw. Eng.*, vol. 2, no. 4, pp. 343–351, 1995.

[29] D. Damian, J. Chisan, L. Vaidyanathsamy, and Y. Pal, "An industrial case study of the impact of requirements engineering on downstream development," *Proc. - 2003 Int. Symp. Empir. Softw. Eng. ISESE 2003*, pp. 40–49, 2003.

[30] T. Beyhl, G. Berg, and H. Giese, "Connecting Designing and Engineering Activities," in *Design Thinking Research*, Cham: Springer International Publishing, 2014, pp. 153–182.

[31] R. J. Wieringa, *Requirements Engineering: Frameworks for Understanding*. New York, NY, USA: John Wiley & Sons, Inc., 1996.

[32] D. C. Wynn and P. J. Clarkson, "Process models in design and development," *Res. Eng. Des.*, vol. 29, no. 2, pp. 161–202, Apr. 2018.

[33] R. Stevens, A. Johri, and K. O'connor, "Professional engineering work," *Cambridge Handb. Eng. Educ. Res.*, pp. 119–138, 2015.

[34] M. V. Elena, C. Wentzky, and J. D. Summers, "Requirements culture: A case study on product development and requirement perspectives," *Proc. ASME Des. Eng. Tech. Conf.*, vol. 7, pp. 1–9, 2019.

[35] S. Minneman, "The Social Construction of a Technical Reality," Stanford University, 1991.

[36] F. P. Brooks, "No Silver Bullet," *IEEE Comput.*, vol. 20, no. 4, pp. 10–19, 1987.

[37] C. Atkins, "An Investigation of the Impact of Requirements Engineering Skills on Project Success ," no. May, 2012.

[38] J. K. Pinto and S. J. Mantel, "The Causes of Project Failure," *IEEE Trans. Eng. Manag.*, vol. 37, no. 4, pp. 269–276, 1990.

[39] S. G. International, "Chaos Report," 2016.

[40] M. Moallem, W. Hung, and N. Dabbagh, Eds., *The Wiley Handbook of Problem-Based Learning*. Wiley, 2019.

[41] A. J. Dutson, R. H. Todd, S. P. Magleby, and C. D. Sorensen, "A Review of Literature on Teaching Engineering Design Through Project-Oriented Capstone Courses," *J. Eng. Educ.*, vol. 86, no. 1, pp. 17–28, 1997.

[42] E. W. Banios, "Teaching engineering practices," in *Proceedings - Frontiers in Education Conference*, 1991, pp. 161–168.

[43] M. V. Elena and J. D. Summers, "Requirement generation: Lecture intervention impact on variety and novelty," *Proc. ASME Des. Eng. Tech. Conf.*, vol. 3, pp. 1–10, 2019.

[44] R. H. Todd, C. D. Sorensen, and S. P. Magleby, "Designing a Senior Capstone Course to Satisfy Industrial Customers," *J. Eng. Educ.*, vol. 82, no. 2, pp. 92–100, 1993.

[45] S. Joshi, B. Morkos, P. Shankar, J. D. Summers, and G. M. Mocko, "Requirements in Engineering Design: What Are We Teaching?," in *Ninth International Symposium on Tools and Methods of Competitive Engineering*, 2012, pp. 1319–1326.

[46] J. A. Donnell, B. M. Aller, M. Alley, and A. A. Kedrowicz, “Why industry says that engineering graduates have poor communication skills: What the literature says,” *ASEE Annu. Conf. Expo. Conf. Proc.*, pp. 22.1687.1-22.1687.13, 2011.

[47] J. E. Froyd, P. C. Wankat, and K. A. Smith, “Five major shifts in 100 years of engineering education,” *Proc. IEEE*, vol. 100, no. SPL CONTENT, pp. 1344–1360, 2012.

[48] S. Goldman *et al.*, “Assessing d.learning: Capturing the Journey of Becoming a Design Thinker,” in *Design Thinking Research*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 13–33.

[49] R. Razzouk and V. Shute, “What Is Design Thinking and Why Is It Important?,” *Review of Educational Research*, vol. 82, no. 3. pp. 330–348, 2012.

[50] F. Hesse, E. Care, J. Buder, K. Sassenberg, and P. Griffi, “Assessment and Teaching of 21st Century Skills,” *Assess. Teach. 21st Century Ski.*, pp. 37–56, 2015.

[51] S. Ouhbi, A. Idri, J. L. Fernández-Alemán, and A. Toval, “Requirements engineering education: a systematic mapping study,” *Requir. Eng.*, vol. 20, no. 2, pp. 119–138, 2015.

[52] R. L. Quintanilla Portugal, P. Engiel, J. Pivatelli, and J. C. S. Do Prado Leite, “Facing the challenges of teaching requirements engineering,” *Proc. - Int. Conf. Softw. Eng.*, pp. 461–470, 2016.

[53] R. B. Svensson and B. Regnell, “Is role playing in Requirements Engineering Education increasing learning outcome?,” *Requir. Eng.*, vol. 22, no. 4, pp. 475–489, 2017.