Set-Based Thinking in the Engineering Design Community and Beyond

Proceedings of the ASME 2014 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference
IDETC/CIE 2014
August 17-20, 2014, Buffalo, NY, USA

DETC2014-35597

SET-BASED THINKING IN THE ENGINEERING DESIGN COMMUNITY AND BEYOND

Sourobh Ghosh
Graduate Research Assistant
Department of Mechanical Engineering
Massachusetts Institute of Technology
Cambridge, MA 02139
sourobhg@mit.edu

Warren Seering
Professor
Mechanical Engineering and Engineering Systems Division
Massachusetts Institute of Technology
Cambridge, MA 02139
seering@mit.edu

ABSTRACT

Since a series of academic case studies had revealed Toyota's unique product development practices to the world, a flurry of research has been conducted into set-based design, also known as set-based concurrent engineering. In this paper, we review work related to set-based design across academic communities in efforts to find common themes and influences. After a review of this literature, we inductively arrive at two Principles of Set-Based Thinking: considering sets of distinct alternatives concurrently and delaying convergent decision making. These Principles allow us to articulate a working description of set-based design. We then examine these two Principles at work in a case example of a common theoretical construct in design.

PREFACE

Last year was the 25th anniversary of the Design Theory and Methodology (DTM) Conference. To mark the occasion, a series of essays commissioned by the DTM Conference and Program Chairs explored past winners of the "best paper award" in an effort to describe the development of past, present, and future research themes of DTM [1]. This led to a revisiting of Ward and Seering's 1989 DTM paper [2] and an informal review of the subsequent development of Set-Based Design (SBD) as a research topic. This paper seeks to continue the cursory review established in [2] and explore the issues it raised in a formal study.

1 INTRODUCTION

Set-Based Design (SBD), also referred to as Set-Based Concurrent Engineering (SBCE), has played an important role in the development of the Design Theory and Methodology (DTM) and the greater engineering design community, as established in last year's retrospective study of DTM best papers and underlying research themes [1]. The theoretical foundation for SBD was laid in Allen Ward's PhD thesis [3], the fundamental ideas of which he presented to the DTM community in [2].

Ward's work nominally describes a computer program that selects standard components from catalogs in order to implement a variety of mechanical designs [2]. However, the fundamental idea behind this work was that products should be designed with all viable options in mind, and that options should not be eliminated unless there is a logical reason to do so. This proposed approach of exploring concepts in parallel while gradually narrowing the solution space until a single solution is found was in contrast to a traditional, point-based design (PBD) approach which advocates generating alternatives at the onset and selecting only one for further development [4,5]. Thus, to keep all feasible options in play for consideration for as long as reasonable – this embodies the ethos of SBD. The way Ward achieved this was by reasoning with sets, lending to the name Set-Based Design.

While these ideas provided a fresh take on product development (PD) practice when they were introduced, the idea of considering design concepts in parallel was not new. Pahl and Beitz [6] and Hubka [7] described design processes for which alternate design concepts were considered simultaneously. The influence of processes associated with the consideration of sets of design alternatives was extended in the mid-1990's, when Ward and colleagues studied new PD at Toyota [8]. At the time, Toyota enjoyed a reputation for developing reliable, superior vehicles under short timelines and at low cost. During their study, Ward et al. found that Toyota's unique practices had embodied the principles of SBCE, or SBD, reasoning that these practices were fundamental to Toyota's success.

The influence of Ward et al.'s study was widespread. This study, along with related work [9], contributed to an environment which fostered the lean manufacturing movement that continues to impact new PD today. Furthermore, the influence of the Toyota case studies generated a flurry of research in the engineering design and management science communities, from where the original ideas of SBD were formulated. However, beyond these native academic communities, SBD began to influence areas such as civil and structural engineering, ship-building, and software development, to name a few examples. Among these diverse applications, SBD as a research area began to take a life of its own, potentially meaning something slightly different to each of its varied practitioners.

This paper reviews literature that addresses reasoning about sets of designs in an effort to find common ideas and themes. As this work is still preliminary, the literature reviewed in this paper is not exhaustive; rather, an emphasis was placed on demonstrating a breadth of work across domains. This includes reviewing 1) methods that are explicitly categorized by some to be SBD and 2) methods that have not been explicitly categorized as SBD but that enable decision-making or reasoning about sets of design alternatives, such as use of function structures and shape grammars. In this paper we will refer to the employment of both categories of methods as "set-based thinking."

With the diversity of work that has emerged over the last approximately two decades, we have struggled in our search of the literature to find a consensus over what SBD or set-based thinking represents today. Therefore, with this paper we hope to identify emergent principles which embody commonly espoused views regarding the core principles of set-based thinking from the literature across communities.

The paper is organized as follows. In Section 2, we discuss the research methodology for our study, outlining our inductive approach. In Section 3 we explore 7 Characteristics of Set-Based Product Development, and in Section 4 we use these 7 Characteristics to arrive at 2 Principles of Set-Based Thinking: considering sets of distinct alternatives concurrently, and delaying convergent decision making. In Section 5 we perform a case example on function structures, observing the ways in which the 2 Principles manifest themselves in this popular theoretical construct, and in Section 6 we discuss the implications of this study and conclusions.

2 RESEARCH METHODOLOGY

In this section, we discuss the theoretical framework for our study. As stated in the introduction, our goal in this work is to consolidate the diverse literature on SBD methods and techniques around common themes which illustrate set-based thinking. Due to the flurry of research over the last two decades that has led to the evolution of SBD as a topic across various domains, we have decided to take an inductive approach to our study.

Our approach shall be to review work which nominally exhibits characteristics or specific instances of SBD or, more generally, set-based PD processes. Then, from a holistic analysis of the work reviewed from these Characteristics, we shall arrive at Principles of Set-Based Thinking. This is in contrast to a deductive approach, which would involve formulating the Principles first, and then testing these Principles across each documented case of set-based affiliated research for validity. Not only would such a study be practically infeasible, but it would suggest that the premises (i.e. the derived Principles) be necessarily true in order for any proposed argument to be sound (and thus to have any value). To avoid this intellectual trap, we use an inductive approach to arrive at our probable conclusions: the Principles of Set-Based Thinking. We believe this analysis to be appropriate for this initial study.

The first step in this process is to identify common characteristics of set-based product development. Given the role of Ward et al. (1995)'s study of Toyota's PD processes in motivating the push for SBD research (a fact which can be substantiated by the frequent citations of [8] by the work reviewed in this paper), we believe that the case studies presented in the 1995 work are a good place to find and define these characteristics. After reviewing these case studies and selecting the most noteworthy, unique attributes of Toyota's PD practices as reported in [8], we arrive at 7 Characteristics of Set-Based Product Development (SBPD)[1], summarized below:

1. Emphasis on frequent, lo-fidelity prototyping
2. Tolerance for under defined system specifications
3. More efficient communication among subsystems
4. Emphasis on documenting lessons learned and new knowledge
5. Support for decentralized leadership structure and distributed, non-collocated teams
6. Supplier/subsystem exploration of optimality
7. Support for flow-up knowledge creation

The Characteristics, forming a notion of what research into set-based methods entails, act as a filter by which we choose what work to review in this study. Using this filter, we review two classes of literature since the Ward et al. (1995) study – work that is nominally classified as SBD, and work that is not nominally classified as SBD but that utilizes set-based reasoning. Both classes of literature are part of the narrative on the influence of set-based thinking in the design community and beyond.

3 7 CHARACTERISTICS OF SET-BASED PRODUCT DEVELOPMENT

This section is dedicated to exploring the manifestation of the 7 Characteristics of Set-Based Product Development across the literature. We detail the genesis of the Characteristics from [8], in addition to how the literature since this work supports each individual characteristic. We begin with the first characteristic, an emphasis on frequent, lo-fidelity prototyping.

3.1 Emphasis on Frequent, Lo-Fidelity Prototyping

At the time of Ward et al. (1995)'s case studies, Toyota was noted as a leader in lean manufacturing and was the industry standard-bearer for the most efficient PD processes, both in terms of time and cost. It was for this reason that one of the most counterintuitive revelations of Ward et al.'s studies was Toyota's practice of frequent prototyping. Ward et al. characterized Toyota's system as generating "excessive numbers of prototypes" when compared to other Japanese and US automakers. For instance, while another Japanese or US

[1] The difference between Set-Based Design (SBD) and Set-Based Product Development (SBPD) as we refer to them is nuanced. Where SBD refers specifically to methodology, SBPD refers to the practical situations and outcomes where SBD is applied

automaker may develop on average three 1/5 scale clay models of body designs, Toyota may develop anywhere from 5 to even 20 of such models.

3.1.1 Frequent Prototyping

These results generated interest in the value of prototyping in the design process, spurring further research. Yang recognizes prototypes as "early embodiment[s] of a design concept" [10], although it should be noted that prototypes may be generated at later phases of the design, and may embody several forms, from sketches [11,12], to 3D CAD models [13], to full-scale models described in [8].

Notably, the proliferation of prototyping throughout the design process is a clear manifestation of concurrent development, as multiple prototypes help designers explore multiple concepts – which Toyota clearly understood and practiced. As Yang notes, prototyping can be thought of as a "design language," or a "way to express design thinking" [10] – descriptions which support the understanding that prototypes enable the concurrent development of concepts. This reasoning will be important later on when we formulate Principles of Set-Based Thinking.

While Ward et al. (1995) document Toyota's correlation with frequent prototyping and ostensibly superior design outcomes (i.e. cheaply manufactured, high-quality automobiles), recent work from Häggman et al. seeks to explore this relationship in a series of controlled studies [14]. Their research demonstrates statistically significant correlations between prototyping early in the design process and design outcomes. Furthermore, they find no statistically significant correlations between the time that design teams spent developing their prototypes and outcomes. Therefore, their results advocate "prototyp[ing] cheaply and early in a project."

While the discussion above has generally supported the value of frequent prototyping, the other half of our argument is that prototypes developed should not intentionally exhibit very high fidelity or levels of resolution or detail – encompassed by Häggman et al.'s suggestion to develop them "cheaply." Here, it is instructive to consider examples from software design.

3.1.2 Prototyping with Sufficient Fidelity

Today, the agile software development movement continues to influence and inform much of the software industry's practices. The relationship between the agile software development movement and lean manufacturing has been well documented, suggesting that lean principles were important to influencing agile PD philosophies [15]. A popular method emerging from this movement was and continues to be Scrum. In general, Scrum and other Agile methodologies emphasize the rapid development of software concepts for trial and error without excessive commitment from developers. These "prototype" phase user interfaces (UIs) need to be tested for basic functionality, bugs, and other issues prior to implementation of the final details of the UI [16].

Similarly motivated practices are found at one of the world's leading design firms, IDEO. The prototyping philosophy at IDEO follows the three R's: rough, rapid, and right. This philosophy promotes the development of prototypes that have sufficient detail – not too much, but just enough to convey the concept. Aiming for "rough" and "right" ensures that time is not wasted developing concepts which eventually become "straw men" concepts and do not inform the final design [17].

In explaining their prototyping philosophy, IDEO founder David Kelley also notes that following the three R's prevents designers from becoming too attached to the concepts that their prototypes represent. This effect, known as design fixation, has been well-documented and studied [18,19]. Furthermore, recent studies by Viswanathan and Linsey demonstrate that design fixation can be mitigated by generating rapid prototypes [20]. Thus, by mitigating design fixation, designers enable themselves to consider a wider range of available options, encompassing much of Ward's original intent for SBD.

3.2 Tolerance for Under Defined System Specifications

Another aspect of Toyota's seemingly paradoxical efficiency from the Ward et al. (1995) study was its designers' tolerance for under defined system specifications. Toyota managers were intentionally reluctant to lock down system specifications prior to key decision making points throughout the design process. Examples included managers reporting only approximate subsystem specifications to target to their suppliers, and delaying the release of final specifications to these suppliers until late in the design process. A particularly poignant example was in body design, where Toyota postponed fixing body hardpoints in the vehicle architecture. As one of Toyota's general managers put it, "the manager's job is to prevent people from making decisions too quickly." This was in stark contrast to the more traditional design approaches adopted by other Japanese and US automakers, who would try to lock down specifications as soon as possible. In their view, this was the only way to move the design forward in the process.

3.2.1 Characteristic 2 in Civil Engineering, Management, and Software Applications

Other applications in the literature have also documented an ability to advance the project at hand without having defined all specifications up front, similar to Toyota. Let us consider a case study from civil engineering, specifically in airport design. Gil et al. undertake a study of upstream and downstream tasks in airport design, where upstream represents "base building" tasks that provide the service space for occupancy, whereas downstream entails fit-out subsystems that make the space functional (e.g. check-in baggage counters, baggage screening machines, etc) [21]. Traditionally, it would seem counterintuitive to delay decision-making on upstream tasks while developing downstream tasks. However, this approach is employed effectively to gain total development time between the concurrent development of upstream and downstream tasks. Managers on this project also pointed out the benefits of their "progressive fixity" to decision making, where intentionally delaying certain decisions built greater flexibility into their project. This mitigates their exposure to risk from events such as shifting requirements or availability of materials.

Further exploring these concepts on a theoretical basis from the management science literature are Thomke and Reinersten

[22]. They note the need for modern day corporations to incorporate agility in their PD processes by adopting "development flexibility." In a series of hardware and software applications in the computer industry, their results advocate "progressively locking down requirements," thus agreeing with the airport design and Toyota examples offered above. Furthermore, they support making "piecewise commitments vs. binary choices," where decisions are made on certain aspects of the project despite not having 100% of project specifications approved in advance. This increases flexibility, driving down the cost of a potential change down the line, and supporting the idea that Toyota's practices of decision delay may have actually helped promote efficiency.

Thomke and Reinersten's study of software applications was apt, as we find significant support for the second Characteristic of SBPD in agile software development. As alluded to earlier, the emphasis of rapid development in Scrum and other methodologies is done so that software can accommodate a highly dynamic environment of rapid technological change, steep competition, and quickly shifting customer requirements [16]. Hence, agile developers postpone decision-making on the final specifications of the UI, especially in an area where customers need to see a final product in order to get a sense of their preferences [22].

3.2.2 Characteristic 2 in Engineering Design and other Domains

In the discussion in Section 3.2.1, we observe that the tolerance of under defined specifications can be interpreted as a tool to manage uncertainty and to mitigate risk. Building upon this thinking, an excellent way to introduce tolerance for under defined specifications and to mitigate risk is to adopt a real options approach to PD [23]. Recognizing this connection, Ford and Sobek use a real options approach to analytically investigate the point beyond which delaying decision making no longer produces practical benefits [24].

Another method to manage uncertainty via this Characteristic, particularly for companies that manage large portfolios of individual products, is to develop a common product platform. Meyer and Lehnerd define a product platform as "a set of common components, modules, or parts from which a stream of derivative products can be efficiently developed and launched" [25]. Robertson and Ulrich put forth a similar definition, with emphasis on asset sharing across a "set of products" [26]. From these definitions, the link to set-based thinking has been established, with companies successfully employing product platforms to mitigate uncertainty, such as Black & Decker's universal electric motor [27]. Furthermore, Simpson notes that one can "stretch" or "shrink" platforms in order to create specific products capable of targeting market niches. Notably, to create a product from scratch to tailor niches implicitly assumes high risk, where a better strategy would be to utilize the under-specified system that is a product platform.

Finally, several in the engineering design community have and continue to mathematically characterize or model scenarios that deal with the uncertainty of specifications that is at the crux of Characteristic 2. Representations that utilize traditional set theory include [28–31], whereas those that explore employing fuzzy sets include [32,33]. Regardless of specific methodology, these works give credence to applying sets to analytically model the tolerance for under-defined system specifications.

3.3 More Efficient Communication among Subsystems

A common practical argument for using set-based methods or techniques is that sets enable fundamentally richer, more efficient communication. Ward et al. (1995) offer a simple thought experiment to illustrate their point. They argue that when a team of individuals is trying to schedule a meeting time, a set-based approach is most efficient. That is, if each participant simultaneously communicates their availability via sets of time, the meeting is scheduled much quicker than if one participant were to check with another if he/she is available at a given time, and so forth. Beyond this thought experiment, Ward et al. found support for their arguments in Toyota's PD processes, where Toyota and its suppliers were found to establish communication with each other less frequently for a shorter total duration of time than their US counterparts employing traditional design methods. For those practicing traditional design methods, constant communication among collocated engineering teams was a given.

The inherent richness of set-based communication has been noted by a few authors in the engineering design community. Madhavan, Seepersad, and colleagues have developed a set-based method for multidisciplinary, distributed design optimization problems. On test problems, they find that their set-based method yielded solutions within 10% of the optimum with only 1 global iteration, translating to 90% less computation expense [34–36]. These results are indicative of the shortcomings of traditional point-based approaches, which, void of the rich information of set-based methods, rely heavily on costly iteration to deliver similar results. Other engineering design authors motivated by set-based efficient communication include Malak and colleagues. Building upon previous work [30], Parker and Malak develop mathematical models which represent the capabilities of ill-characterized systems, which they call technology characterization models (TCMs) [37]. These TCMs intend to facilitate communication of subsystem level performance characteristics, allowing for the direct comparison of sets of heterogeneous concepts within the same performance space, similar to traditional Pareto frontiers.

Also inspired by Toyota's example has been the construction academic community, with Parrish et al. offering a SBD framework for the delivery of reinforcing bars in cast-in-place concrete [38]. They note that the ruling paradigm in the construction industry is a traditional, point-based design (PBD) approach featuring long delays in passing designs to different agents in the design process. These delays arise from the high amount of rework at each step when a new agent in the design process contributes to the overall design. Parrish et al.'s approach instructs all agents in the rebar design process to contribute their input concurrently, accommodating rich sets of information from each agent and producing a much more efficient communication scheme, bypassing the inefficiencies of iteration. The efficiency of this proposed approach has been recognized, leading other authors in the community to further develop Parrish et al.'s framework [39,40].

Of course, the efficient communication paradigm of SBD and set-based thinking in general can be found in other theoretical constructs in design, such as shape grammars [41]. Orsborn et al. describe shape grammars as "production systems created by taking a sample of the whole for which one is trying to write a language" [42]. Shape grammars, complete with a vocabulary (i.e. a set) of shapes and shape rules can represent "the language" of a given product's design. Orsborn et al. use shape grammars as a method to explore the development of crossover SUV body architectures by enabling the designer to produce a multitude of designs that fall with the constraints of the grammar. The intent is to enable the exploration of new and unique designs that may not have been imagined by the designer. Thus, using sets of shapes, designers can efficiently communicate their ideas on vehicle body architecture, while corresponding shape rules help constrain the design space.

3.4 Emphasis on Documenting Lessons Learned/Knowledge

While discussing Toyota's practices, Ward et al. (1995) note that in general "designers are notoriously resistant to documenting their work, partly because they sense documentation is generally useless." Toyota, however, expects rigorous documentation of design knowledge via "lesson-learned books." Ward et al. provide an example of a Toyota die-designer, who maintains a book of sixty to seventy different ranges of specifications that would ensure the fender design's manufacturability. They report that similar books are maintained for every body part, which, developed over fifteen years at that point, provide detailed knowledge of what potential designs can (and cannot) be implemented.

These lessons learned books are noted to promote institutional learning and to ensure that detailed technical knowledge of one's system is not lost over the years – rather, it is accrued. Similar case examples across domains are noted by Thomke [17]. For instance, he highlights the example of pharmaceutical companies stockpiling small quantities of discrete chemical compounds in "chemical libraries." These libraries are the result of years of prior drug development projects, maintaining "several hundred thousand compounds and information on their specific properties." In fact, an emphasis on documentation has gained traction in smaller scale firms such as IDEO. Thomke describes IDEO's "Tech Box," which is a "giant 'shoebox' for curiosities and interesting gadgets from prior IDEO projects." Of course, the intent of the chemical libraries and the Tech Box are the same, which is to help foster new PD.

While the industry cases above demonstrate how sets of information can be documented to help promote knowledge retention, a few academic case studies have established a direct connection between following SBD or SBCE processes and learning. In a series of case studies by Raudberget in which he introduces companies to SBCE methodologies, he notes that for one of the companies he studied, "the value of the SBCE project was to identify the knowledge gaps in their core technology" [43]. Furthermore, he notes that by "knowing the limits of their technology," the said company could "have an appropriate set of solutions ready for future offers." Similar results were found in Madhavan et al.'s study of applying SBD principles to Schlumberger, where "archiving design knowledge in the form of sets of solutions…can be used in future design activities" [44].

In fact, this idea can be found in other work in design theory, such as function structures. Pahl and Beitz describe the ability to define an overall function that expresses the relationship between inputs and outputs of system independently of the design solution [6]. They define a function structure as the combination of individual sub-functions that provides a representation of the overall function being analyzed. Pahl and Betiz point out potential efficiencies of employing function structures, noting that "if existing assemblies can be assigned directly as complex sub-functions, the subdivision of the function structure can be discontinued at a fairly high level of complexity." Thus, documenting system knowledge via sub-functions from prior experience in a set that is readily available for recall in the future may help save "a great deal of time and money," similar to how Toyota used their lessons-learned books.

Another tool that can enable set-based knowledge representation is affordances. Maier and Fadel view an affordance-based approach as an alternative to function analysis, where affordances describe a potential behavior between two or more subsystems within a larger designer–artifact–user complex system [45]. The goal is to articulate relationships that are unaccounted for in a function based approach to design, such as non-function based customer requirements [45,46]. Thus, an affordance-based approach, along with function structures, can enable set-based thinking.

3.5 Support for Decentralized Leadership Structure and Distributed, Non-Collocated Teams

This section is the first of three which explore insights from Toyota's unique organizational structure. Evidence for Characteristic 5 is apparent from examining Toyota's relationships with its suppliers, as reported in Ward et al. (1995). These relationships often provide considerable autonomy to the suppliers, with examples such as radiator and alternator manufacturer Nippondenso (now Denso) offering sets of potential alternators for Toyota to consider and choose from, rather than being tasked to develop an alternator that meets specifications initially determined by Toyota. Therefore, Nippondenso is accorded considerable weight in Toyota's decision making process, as the set of options offered to Toyota has been determined by Nippondenso's estimates of Toyota's needs – alluding to decentralized leadership (where decisions do not necessarily originate from the top). Furthermore, Nippodenso and many of Toyota's suppliers do not operate in-house, thus exhibiting support for distribution and non-collocation of subsystem designers.

3.5.1 Decentralized Leadership Structure

We begin by exploring support for a decentralized leadership structure in set-based methods. As noted earlier, the Scrum methodology of software development provides excellent examples of SBPD. The Scrum method is carried out by Scrum Teams. There are three core roles on a Scrum Team: Product Owner, Development Team, and Scrum Master [47]. While detailing the exact responsibilities of each role is beyond the discussion of this paper (details can be found in [47]), what is notable about the separation of labor in a Scrum Team is that there is no role which bears the responsibilities of a traditional project manager. For instance, while the Scrum Master "is responsible for ensuring Scrum is understood and enacted" [47], he/she does not bear personnel management responsibilities that are typical of a project manager in a company with a traditional, centralized leadership structure [48]. The reluctance to adopt a traditional leadership structure stems from the belief that it would produce sub-optimal results.

Exploring the role of varying levels of cooperation in concurrent engineering processes are Lewis and Mistree [49]. Using a decision-based design perspective, they develop a game theoretic model to represent non-cooperative situations in concurrent engineering processes. They note that the resulting model "is similar in concept to the idea of set-based concurrent engineering…explored in (Ward et al., 1995), except there is no communication among the various designers who are generating sets of solutions." Nonetheless, the reason why they observe similarities is due to their model's ability to represent decentralized leadership structures. This game theoretic model would later be further developed by Lewis and colleagues to investigate various distributed design problems [50–52]. It is noted that this framework works well to model nonhierarchical systems of designers [51] – which supports the decentralized leadership observation. Furthermore, the Nash equilibrium that represents the converged solution in the model is the result of intersecting each designer's rational reaction set, which is the set of solutions that optimizes that individual designer's objective relative to the decisions of other designers. Hence, we observe support for the use of set-based modeling and decentralized leadership, although we note that the models of Lewis and colleagues do not agree with each of the observed characteristics of SBPD as outlined in this paper (e.g. Characteristic 3). As we will find in Section 4, not all observed processes can be classified strictly as SBD or PBD; rather, they may fall somewhere on a spectrum between the two.

3.5.2 Arguments for Distribution, Non-Collocation

To test a set-based PD method's support for distributed, non-collocated teams, Sutherland et al. study whether the aggressive development schedule and culture of Scrum can be sustained across a multinational team of collaborators [16]. Given certain features of Scrum, such as frequent Scrum meetings, not many had ventured to explore whether Scrum would be practical beyond the familiarity of small, collocated settings that entailed its predominant usage. Thus, Sutherland et al. perform a case study tracking a distributed team of 56 developers across three countries and witness the most productive Java development project to have been documented at the time of their publication – a testament to set-based PD practices supporting distributed, non-collocated teams.

Another application area for set-based PD practices has been the Navy, specifically in the design of the Ship-to-Shore-Connector (SSC) [53]. Mebane et al. study the Navy's desire to develop the SSC on an "extremely aggressive" time schedule, leading the Navy to adopt an SBD approach to its design. While drawing conclusions from the study, they note "the Navy's ability to develop a design with a distributed team with much of the design team personnel remotely located in the field was certainly proven," demonstrating sources of support for this Characteristic of SBPD beyond just the private sector.

3.6 Supplier/Subsystem Exploration of Optimality

An interesting feature of a decentralized leadership structure that often provides subsystems with greater autonomy in the design process is that it encourages suppliers and subsystems to take initiative in exploring optimality. Continuing with the aforementioned example of Nippondenso, Ward et al. (1995) describe its approach to look for radical performance breakthroughs – rather than incremental improvement – in their designs, helping ensure their relevance in the market of automotive suppliers. Hence, Toyota benefits from having a supplier that prioritizes being at the forefront of its technology, helping propel Toyota's search for globally optimal designs.

Other examples in design practice which feature this Characteristic include Madhavan et al.'s previously discussed study [44]. The authors help substantiate this claim by pointing out how their set-based method encouraged Schlumberger to conduct tradeoff analyses at the subsystem level, such as the tradeoff between mass and stiffness in a given chassis design. Studying design spaces across interfaces was not prioritized prior to applying set-based methods, implying a less formal exploration of optimality when compared to applying SBD.

Examining the Navy's SSC project once again, Mebane et al. reported that Systems Engineering Managers would communicate "ranges of solutions with associated derived requirements for various systems…rather than develop a single point solution" [53]. The result was the consideration of a "far greater range of options…than would have been considered in traditional point-based design evolutions…in a much shorter time period." Of course, this extended exploration is a key feature towards the exploration of more optimal solutions.

Finally, beyond case study examples, the aforementioned example of technology characterization models (TCMs) illustrates this Characteristic well. As noted earlier, TCMs were developed to address the problem of how to present and explore the technical capabilities of certain types of ill-characterized complex systems [37]. Algorithms developed from this work feature a bottom-up flow of information [54]. The rationale for this approach is that when subsystems explore optimality, this should scale up to the greater system level in efforts to approach a global optimum. In essence, these efforts by Malak and colleagues are to mathematically model an approach that resembles the flow of information that exists between Toyota and its suppliers, such as Nippondenso.

3.7 Supports Flow-up Knowledge Creation

As alluded to in the discussion of the previous section, the supplier/subsystem exploration of optimality lends itself to supporting flow-up knowledge creation. What is truly indicative of this Characteristic is the timing of suppliers' presentations to Toyota. Delivered about 36 months prior to the start of new model production, these presentations allow suppliers to describe their latest developments to Toyota, which includes prototypes, large amounts of test data, and related information. The critical difference between Toyota and competitors as reported in Ward et al. (1995) is that while US companies would develop specifications in-house prior to these types of presentations, Toyota would develop its specifications almost two years *after* its supplier presentations. In other words, it would take information from these presentations to guide critical decision making behind setting final specifications, indicating a bottom-up flow of information, as opposed to the traditional top-down flow where specifications were handed down by US manufacturers.

An excellent example of how SBD was applied for its flow-up qualities is provided by the aforementioned study by Parrish et al. [38]. As noted earlier, the intent of Parrish's work is to involve all agents across the construction supply chain (e.g. owners, architects, structural engineers, concrete suppliers, etc.) in the design decision-making process. This is in contrast to a traditional model where a structural engineer will make all decisions up front, leading to a single, often suboptimal, design. This flow of decision making is decidedly top-down, as opposed to Parrish's approach which empowers subsystems (i.e. agents such as concrete suppliers) in the decision-making process, leading to a bottom-up flow of information.

Academic examples of this Characteristic include the work of Malak and colleagues as noted in Section 3.6 regarding TCMs. By optimizing subsystems first, the flow of information is certainly bottom-up. Other examples include work from Shan and Wang, who develop an algorithm using rough sets to map "the performance space to the design space" that "could be potentially used to explore and visualize the entire design space" [33]. The performance space that these authors refer to is attributes at the subsystem level, whereas the design space is the greater system level. Thus, similar to Malak and colleagues, these authors wish to use sets to map performance capabilities at the subsystem level to a higher level system level space to pursue globally optimal design – entailing a flow-up approach. Finally, more recent work is presented by Canbaz et al., who in studying distributed design processes, expand the "bottom-up design approach with exploring agent-based modeling techniques into the set-based design concept" [55].

4 THE 2 PRINCIPLES OF SET-BASED THINKING

Having reviewed evidence for the 7 Characteristics of SBPD, we now turn our attention to the development of 2 Principles of Set-Based Thinking (the 'Principles'). This section is organized as follows. In Section 4.1, we present the Principles, and in Section 4.2 we introduce the Set-Based/Point-Based Process Spectrum, which enables us to analyze the degree to which certain processes exhibit set-based or point-based characteristics. Finally, in Section 4.3, we detail how the 7 Characteristics of SBPD lead us to obtaining 2 Principles of Set-Based Thinking.

4.1 The Two Principles of Set-Based Thinking

At this point in our study, we make the following observations:

- SBD is not formally defined, yet numerous authors have studied its process inspired by the example of Toyota
- The 7 Characteristics of SBPD are a filter which allow us to review work that exhibits set-based practice

A major challenge in our task to define what SBD means is determining what work to include and what not to include in our review – which is the function of the 7 Characteristics. Having reviewed the literature of set-based practice in Section 3, our task now is to find the most common themes and ideas in this reviewed work that capture SBD, which we term the Principles. Our review supports the finding of two, independent Principles. These Principles are regularly woven into the ideas discussed in the work reviewed, both explicitly and implicitly.

At this phase of our preliminary study, we cannot guarantee that these are the only two Principles; however, our goal was to distill the network of complex ideas in Section 3 into the most simple, fundamental representation possible, which led to the articulation of these two Principles. As a preliminary validation of this approach, in Section 4.3 we explain how the 7 Characteristics of SBPD directly embody the Principles. But first, we present the Principles:

1. Considering sets of distinct alternatives concurrently
2. Delaying convergent decision making

Central to the ideas of SBD and related work is the concept of concurrent development, which is captured by Principle 1. Concurrent development describes the simultaneous development of distinct concepts. This is in contrast to traditional, point-based processes, where designers establish commitment early on to one concept which receives the singular focus of the designers involved. It is important, however, to establish the distinction between the development of sets of distinct concepts and the development of sets of variants of a single concept. The former captures the true ethos of SBD as noted in the introduction, which is to consider as many unique options as possible. The latter exhibits a limiting case of SBD and may regress to traditional, PBD processes.

To demonstrate the difference between sets of distinct concepts and sets of variants of a single concept, consider finding a solution to travel from the city of Rochester, NY, USA to Toronto, ON, Canada. One may consider renting a car, taking a bus, or hiring a limousine service. However, these options entail variants of a single concept: to travel by highway.

Other potential options include train, airplane, and even a ferry [56]. While this is a simple example, it illustrates that comparing sets of variants of a concept limits the solution space explored and does not provide the full intent of Principle 1. Nuances such as these will be important to keep in mind when analyzing the spectrum of set-based and point-based processes, as presented in Section 4.2.

Equally important to SBPD is the second Principle: delaying convergent decision making. Note that this Principle does not advocate delaying decisions for the sake of delaying decisions; rather, it supports delaying convergence to a single solution. Some authors highlight this distinction by pointing out that decisions should be delayed until the "last responsible moment" [21]. The emphasis on "responsible" highlights how decision delay as a strategy should not be exercised as an excuse to simply not make decisions. Instead, it should facilitate the consideration of more options. Specific to SBD, following this Principle results in a process where the solution space gradually narrows to a single solution, as opposed to traditional, point-based representations, which promote rapid convergence and reiteration on a single concept that may require rework calling for re-expanding the solution space down the road [4], which intuitively is a source of inefficiency.

4.2 The Set-Based/Point-Based Process Spectrum

In previous sections, we alluded to the idea that PD processes should be qualified on a spectrum that encompasses SBD and traditional, PBD processes. Such a spectrum can help highlight how some processes may exhibit certain characteristics of set-based PD but still not represent an ideal case of SBD. A question that has gone unaddressed in the body of SBD research is finding heuristics for determining when it is appropriate to apply set-based vs. point-based techniques. Thus, a spectrum that helps us classify to what degree a certain process represents a complete implementation of SBD is a first step towards answering that question.

In order to develop a set-based/point-based process spectrum, we first need a definition for PBD. This is a non-trivial task and is similar to attempting to define a primitive notion – in the sense that point-based, traditional design processes encompass that with which we are already familiar.
In his discussion of agile product development processes, Iansiti characterizes traditional design processes as those where detailed design and implementation occurs only after concept selection occurs [57]. Along these lines, Ulrich and Eppinger, commenting on a generic (i.e. traditional) product development process, posit the following about concept development, "alternative product concepts are generated and evaluated, and one or more concepts are selected for further development" [58].

While these definitions capture some of the distinctions of point-based design relative to SBD, they are generally incomplete and refer only to the interface between the concept development and detailed design phases. This is limiting since the Principles can manifest themselves at all phases across the design process. Therefore, we propose a working definition for point-based and traditional design processes that is defined relative to our definition of SBD. Similar to how temperature is defined on a relative basis to a given reference point (absolute zero), we wish to define point-based and traditional design relative to the 2 Principles of Set-Based Thinking. Let A = Principle 1 ("considering sets of distinct alternatives concurrently") and B = Principle 2 ("delaying convergent decision making). From these statements, we can define SBD succinctly:

$$SBD \equiv A \wedge B$$

In other words, SBD (or more generally, SBPD) is defined by Principle 1 and Principle 2. We define traditional design relative to SBD in the following way:

$$Traditional\ Design \equiv \neg SBD = \neg(A \wedge B) = \neg A \vee \neg B$$

In other words, traditional design is simply "not SBD." Thus, by DeMorgan's Law, it encompasses a lack of Principle 1 or a lack of Principle 2, or a lack of both Principles 1 and 2, since we use the inclusive definition of "or" in the notation above. In fact, where both a lack of Principle 1 and a lack of Principle 2 are exhibited, we have a special case of traditional design, which is pure PBD. The rationale for this working definition is that if a process exhibits one of Principle 1 or Principle 2, then it inherently bears some set-based characteristics, and thus should fall in a spectrum between the poles of SBD and PBD, where pure PBD represents cases that do not bear any set-based characteristics. This spectrum is demonstrated visually in Figure 1.

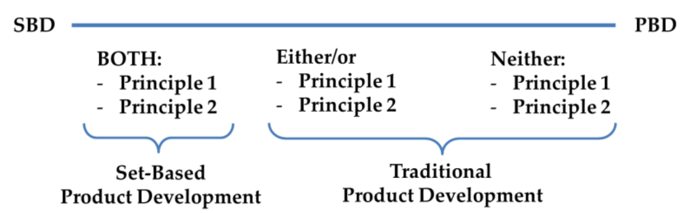

Figure 1: The Set-Based/Point-Based Process Spectrum

An important question to ask is whether the Principles can be considered independently in the proposed spectrum. That is, can Principle 1 act independently of Principle 2, and vis-versa? For instance, does a process exist where we consider sets of alternatives concurrently without delaying convergent decision making? The answer is yes – consider the example of a morphological matrix. This method hastens the consideration of a very large number of distinct alternatives [6]. Still, Pahl and Beitz note the "weakness of this approach" by noting "the very great, theoretically admissible but practically unattainable, number of solutions must be reduced at the earliest possible moment." Thus, using a morphological matrix is not conducive to promoting decision delay.

Now let us consider the opposite case: a process that delays convergent decision making but that does not necessarily promote considering sets of distinct alternatives concurrently. A good example here is the overuse of a product platform, such as Chrysler's K-car platform in the late 1980s [27]. While product platforms clearly enable the delay of convergent decision making (product can be tailored later to target niches, see Section 3.2), designers in a push for leveraging commonality may feel inclined to use it more often than they should, thus limiting the consideration of distinct alternatives.

Thus, we have demonstrated via example how the two Principles can act independently of one another, and critically how they must both be present in order to facilitate true SBD.

4.3 Deriving the 2 Principles from the 7 Characteristics of Set-Based Product Development

In this section, we detail the process by which the 7 Characteristics of SBPD reveal the 2 Principles of Set-Based Thinking. Note that some of the 7 Characteristics support one Principle more than the other. However, considering the 7 Characteristics holistically allows us to arrive at developing the 2 Principles. Thus, we go through each Characteristic demonstrating their collective support for the Principles.

Emphasis on frequent lo-fidelity prototyping. Prototypes enable designers to compare several distinct alternatives as embodied by the aforementioned design language argument in [10], thus exhibiting support for Principle 1. Furthermore, the emphasis on prototyping early and with lo-fidelity [14] enables further concurrent development, thus postponing convergence to a single concept and supporting Principle 2.

Tolerance for under defined system specifications. Designers using set-based techniques are comfortable with incomplete information, and thus can work with concepts without necessarily forcing convergence to continue development, as demonstrated in [8,21,22,53,57] – thus supporting Principle 2. In fact, incomplete information is often fostered by system managers [8,16,21,47], inducing designers to further explore the solution space and reinforcing concurrent development and support for Principle 1.

More efficient communication among subsystems. Communicating via sets is inherently more efficient than passing information sequentially and iterating on a point-based solution [2,8,37,38,44,53], supporting concurrent development and Principle 1.

Emphasis on documenting lessons learned/knowledge. Several examples described in Section 3.4, from Toyota's lessons learned books, pharmaceutical companies' chemical libraries, and IDEO's Tech Box illustrate how documenting knowledge can lead to the consideration of a wider set of options, supporting Principle 1 [15]. In addition, by allowing designers to reason about why options should remain in play such as with function structures [6], Principle 2 is upheld.

Support for decentralized leadership structure and distributed, non-collocated teams. The concurrent development of distinct alternatives is easily performed in distributed, non-collocated settings [8,16,53], unlike traditional processes that rely on iteration and require centralization. Thus Principle 1 is sustained.

Supplier/subsystem exploration of optimality. System managers of set-based processes will not provide predetermined specifications to suppliers [8,30,37], who in their search for optimality will need to adopt the concurrent development of distinct alternatives [38,44], fostering Principle 1. By tasking subsystems to explore optimality, the system manager also inherently delays convergence by fostering set-based exploration as opposed to point-based exploration [8,43,44], supporting Principle 2.

Supports flow-up knowledge creation. Following from the last Characteristic, supplier exploration of optimality, which is achieved via concurrent development, fundamentally enables a flow-up knowledge creation, thus promoting Principle 1. Similarly, from Nippondenso's example we saw how system managers (i.e. Toyota) delay convergent decision making until subsystems perform their tasks of exploring subsystem level optimality, enabling a bottom-up flow of knowledge that is conducive to supporting Principle 2.

5 CASE EXAMPLE: SET-BASED THINKING IN FUNCTION STRUCTURES

Having developed the Principles in the previous section, we now turn to a brief case example to illustrate how these Principles manifest themselves in a common theoretical construct in design: function structures. We intentionally choose to explore function structures as they are an example of work that is not nominally related to SBD but that we review in this paper as it features set-based reasoning. An example of a real-world function structure for the Black & Decker® VersaPak™ Cordless Screwdriver [59] is provided in the Appendix in Figure A1. Further reading on the background and theory of function structures is referenced in Section 3.4.

In a function structure, each sub-function can embody a wide array of physical configurations – the function structure does not discriminate in this regard. Refer to the example in Figure A1. While the sub-function "input signal" has been implemented via a sliding switch in the final product shown, the sub-function itself enables consideration of options such as a push-button switch or a rotary switch to name a few examples. Thus, function structures clearly enable the consideration of sets of distinct alternatives, or Principle 1.

However, the real value of function structures in the context of set-based reasoning is realized by plotting the flow of signals and other system interactions, nicely articulating constraints on the solution space in which designers explore. This allows designers to reason about which alternatives to explore and which to not explore – a line of reasoning that extends back to the genesis of SBD [2,3]. In this sense, it facilitates Principle 2, the delay of convergent decision making, because all physical instantiations that meet the requirements of the function structure are carried through the design process. This is a methodical way to practice decision delay, as opposed to morphological matrices (see Section 4.2), where coarse heuristics must be applied in order to rule out options that do not meet system physical constraints. Such heuristics may eliminate more options than required, thus preventing the delay of convergent decision making.

6 DISCUSSION AND CONCLUSION

In 1995, Ward and colleagues documented Toyota's unique PD practices as examples of SBCE, generating a flurry of research into SBD and set-based reasoning across academic communities, such as engineering design, management science, civil and structural engineering, software design, and ship building, to name a few examples. Over this time period, SBD has never been formally defined, despite many authors having studied its process inspired by the example of Toyota.

In this paper, our objective is to inductively identify common themes and ideas from the body of work that has emerged since Ward et al. (1995)'s original study. From their study, we identify 7 Characteristics of Set-Based Product Development, which essentially are observed attributes of set-based processes. We then employ these Characteristics as a filter by which to choose what work to review in this paper. We review work that is both nominally tied to SBD and also that which actively features set-based reasoning. After studying the sample of literature that we generate, we find two, independent principles that are both explicitly and implicitly integrated into the ideas of the literature reviewed in this paper. These are 2 Principles of Set-Based Thinking, which are 1) considering sets of distinct alternatives concurrently, and 2) delaying convergent decision making. We use these Principles as a basis by which to offer a working definition for SBD. We also introduce the Set-Based/Point-Based Process Spectrum, which provides a preliminary way to represent the idea that some processes may exhibit certain characteristics of Set-Based Thinking without being an ideal case of SBD, thus presenting a hybrid of set-based and point-based reasoning.

Given the discussion above, we summarize the contributions of this work as follows, in chronological order of how they are presented in this paper:

1. Provides a review of selected elements of the SBD literature over approximately the last two decades
2. Identifies the connection between work that is not nominally SBD via the lens of Set-Based Thinking
3. Provides a working description of SBD based off of the 2 Principles of Set-Based Thinking
4. Introduces the set-based/point-based spectrum by which to understand the degree to which a PD process exhibits set-based behavior

While most of these points have been addressed in the aforementioned discussion, one that has not yet been directly addressed in this section is item 2 above. An interesting insight from this work is that various methods that are a product of the engineering design research community and are not generally associated with SBD do in fact support set-based thinking.

Based on our preliminary understanding of SBD vs. PBD from this paper, areas for future work include determining what situations are most appropriate for SBD vs. PBD. For instance, are certain organizations more conducive to the application of SBD than others? Along these lines, what is the cost of employing SBD, and how does this relate to the value gained when compared to using traditional, PBD processes? Also, how has the practice of SBD generally changed real-world project outcomes in various fields? Finally, given that we are in the early stages of this work, are there other ideas embodied in SBD that have not been yet captured by the Principles of Set-Based Thinking that we articulate in this paper? Addressing these concerns and others will be vital towards our understanding of how designing products from the lens of set-based thinking may potentially ameliorate design practice.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. 1122374. The authors would also like to thank the King Fahd University of Petroleum and Minerals (KFUPM) in Dhahran, Saudi Arabia, for their support of the research reported in this paper through the Center for Clean Water and Clean Energy at MIT and KFUPM under project number R11-DMN-09.

REFERENCES

[1] Braha, D., Brown, D. C., Chakrabarti, A., Dong, A., Fadel, G., Maier, J. R. A., Seering, W., Ullman, D. G., and Wood, K., 2013, "DTM at 25: Essays on Themes and Future Directions," ASME International Design Engineering Technical Conferences, Portland, OR, pp. 1–17.

[2] Ward, A. C., and Seering, W., 1989, "Quantitative Inference in a Mechanical Design 'Compiler'," ASME International Design Engineering Technical Conferences.

[3] Ward, A. C., 1989, "A Theory of Quantitative Inference for Artifact Sets, Applied to a Mechanical Design Compiler," Massachusetts Institute of Technology.

[4] Sobek, D. K., Ward, A. C., and Liker, J. K., 1999, "Toyota's Principles of Set-Based Concurrent Engineering," Sloan Management Review, **40**(2), pp. 67–83.

[5] Singer, D. J., Doerry, N., and Buckley, M. E., 2009, "What Is Set-Based Design?," Naval Engineers Journal, **121**(4), pp. 31–43.

[6] Pahl, G., and Beitz, W., 1988, Engineering Design: A Systematic Approach, The Design Council, London.

[7] Hubka, V., Andreasen, M. M., and Eder, W. E., 1988, Practical Studies in Systematic Design, Butterworths, London.

[8] Ward, A., Liker, J. K., Cristiano, J. J., and Sobek, D. K., 1995, "The Second Toyota Paradox: How Delaying Decisions Can Make Better Cars Faster," Sloan Management Review, **36**(3), pp. 43–61.

[9] Womack, J. P., Jones, D. T., and Roos, D., 2007, The Machine That Changed the World, New York.

[10] Yang, M. C., 2005, "A study of prototypes, design activity, and design outcome," Design Studies, **26**(6), pp. 649–669.
[11] Yang, M. C., 2008, "Observations on concept generation and sketching in engineering design," Research in Engineering Design, **20**(1), pp. 1–11.
[12] Shah, J. J., Vargas-Hernandez, N., Summers, J. D., and Kulkarni, S., 2001, "Collaborative Sketching (C-Sketch) - An Idea Generation Technique for Engineering Design," Journal of Creative Behavior, **35**(3), pp. 168–198.
[13] Nahm, Y.-E., and Ishikawa, H., 2006, "A new 3D-CAD system for set-based parametric design," The International Journal of Advanced Manufacturing Technology, **29**(1-2), pp. 137–150.
[14] Häggman, A., Honda, T., and Yang, M. C., 2013, "The Influence of Timing in Exploratory Prototyping and Other Activities in Design Projects," ASME International Design Engineering Technical Conferences, Portland, OR, pp. 1–12.
[15] Poppendieck, M., and Poppendieck, T., 2003, Lean Software Development: An Agile Toolkit, Addison-Wesley Professional, Boston, MA.
[16] Sutherland, J., Viktorov, A., Blount, J., and Puntikov, N., 2007, "Distributed Scrum : Agile Project Management with Outsourced Development Teams," Proc. of the 40th Hawaii International Conference on System Sciences, pp. 1–10.
[17] Thomke, S., 2003, Experimentation Matters, Harvard Business School Publishing Corporation, Boston, MA.
[18] Jansson, D., and Smith, S., 1991, "Design fixation," Design Studies, **12**(1), pp. 3–11.
[19] Linsey, J. S., Tseng, I., Fu, K., Cagan, J., Wood, K. L., and Schunn, C., 2010, "A Study of Design Fixation, Its Mitigation and Perception in Engineering Design Faculty," Journal of Mechanical Design, **132**(4), pp. 1–12.
[20] Viswanathan, V. K., and Linsey, J. S., 2013, "Role of Sunk Cost in Engineering Idea Generation: An Experimental Investigation," Journal of Mechanical Design, **135**(12), pp. 1–12.
[21] Gil, N., Beckman, S., and Tommelein, I. D., 2008, "and Ambiguity : Evidence From Airport Expansion Projects," IEEE Transactions on Engineering Management, **55**(3), pp. 508–522.
[22] Thomke, S., and Reinertsen, D., 1998, "Agile Product Development: Managing Development Flexibility in Uncertain Environments," California Management Review, **41**(I), pp. 8–30.
[23] De Neufville, R., 2003, "Real Options: Dealing with Uncertainty in Systems Planning and Design," Integrated Assessment, **4**(1), pp. 26–34.
[24] Ford, D. N., and Sobek, D. K., 2005, "Adapting Real Options to New Product Development by Modeling the Second Toyota Paradox," IEEE Transactions on Engineering Management, **52**(2), pp. 175–185.
[25] Meyer, M. H., and Lehnerd, A. P., 1997, The Power of Product Platforms, Free Press, New York.
[26] Robertson, D., and Ulrich, K., 1998, "Planning for Product Platforms," Sloan Management Review, **39**(4), pp. 19–31.
[27] Simpson, T. W., 2004, "Product platform design and customization: Status and promise," AI EDAM: Artificial Intelligence for Engineering Design, Analysis and Manufacturing, **18**(01), pp. 3–20.
[28] Finch, W. W., and Ward, A. C., 1997, "A Set-Based System for Eliminating Infeasible Designs in Engineering Problems Dominated by Uncertainty," ASME International Design Engineering Technical Conferences, Sacremento, CA, pp. 1–12.
[29] Malak, R. J., Aughenbaugh, J. M., and Paredis, C. J. J., 2009, "Multi-attribute utility analysis in set-based conceptual design," Computer-Aided Design, **41**(3), pp. 214–227.
[30] Malak, R. J., and Paredis, C. J. J., 2010, "Using Parameterized Pareto Sets to Model Design Concepts," Journal of Mechanical Design, **132**(4), pp. 1–11.
[31] Panchal, J. H., Fernandez, M. G., Paredis, C. J. J., Allen, J. K., and Mistree, F., 2007, "An Interval-based Constraint Satisfaction (IBCS) Method for Decentralized, Collaborative Multifunctional Design," Concurrent Engineering, **15**(3), pp. 309–323.
[32] Antonsson, E. K., and Otto, K. N., 1995, "Imprecision in Engineering Design," Journal of Mechanical Design, **117**(B), pp. 25–32.
[33] Shan, S., and Wang, G. G., 2004, "Space exploration and global optimization for computationally intensive design problems: a rough set based approach," Structural and Multidisciplinary Optimization, **28**(6), pp. 427–441.
[34] Carlos, S., Madhavan, K., Gupta, G., Keese, A., Maheshwaraa, U., and Seepersad, C. C., 2006, "A Flexibility-Based Approach to Collaboration in Multiscale Design," 11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Portsmouth, VA.
[35] Madhavan, K., 2007, "A Framework for a Flexibility-Based Approach to Multiscale and Multidisciplinary Design," University of Texas at Austin.
[36] Seepersad, C. C., Shahan, D., and Madhavan, K., 2007, "Multiscale Design for Solid Freeform Fabrication," Solid Freeform Fabrication Symposium, Austin, TX.
[37] Parker, R., and Malak, R. J., 2011, "Technology Characterization Models and Their Use in Designing Complex Systems," ASME International Design Engineering Technical Conferences, Washington, DC.
[38] Parrish, K., Wong, J., Tommelein, I. D., and Stojadinovic, B., 2007, "Exploration of Set-Based Design for Reinforced Concrete Structures," Proc. of 15th Conf. of the Int. Group for Lean Construction,, Michigan, USA, pp. 213–222.
[39] Lee, S.-I., Bae, J.-S., and Cho, Y. S., 2012, "Efficiency analysis of Set-based Design with structural building information modeling (S-BIM) on high-rise building structures," Automation in Construction, **23**, pp. 20–32.

[40] Sacks, R., Koskela, L., Dave, B. A., and Owen, R., 2010, “Interaction of Lean and Building Information Modeling in Construction,” Journal of Construction Engineering and Management, **136**(9), pp. 968–980.
[41] Stiny, G., 1980, “Introduction to shape and shape grammars,” Environment and Planning, **7**(November), pp. 343–351.
[42] Orsborn, S., Cagan, J., Pawlicki, R., and Smith, R. C., 2006, “Creating cross-over vehicles: Defining and combining vehicle classes using shape grammars,” Artificial Intelligence for Engineering Design, Analysis and Manufacturing, **20**(03), pp. 217–246.
[43] Raudberget, D., 2010, “Practical Applications of Set-Based Concurrent Engineering in Industry,” Journal of Mechanical Engineering, **56**(11), pp. 685–695.
[44] Madhavan, K., Shahan, D., Seepersad, C. C., Hlavinka, D. A., and Benson, W., 2008, “An Industrial Trial of a Set-Based Approach to Collaborative Design,” ASME International Design Engineering Technical Conferences, New York, NY, pp. 1–11.
[45] Maier, J. R. a., and Fadel, G. M., 2008, “Affordance based design: a relational theory for design,” Research in Engineering Design, **20**(1), pp. 13–27.
[46] Cormier, P., Olewnik, A., and Lewis, K., 2013, “Towards a Formalization of Affordance Modeling in the Early Stages of Design,” ASME International Design Engineering Technical Conferences, Portland, OR, pp. 1–16, DETC2013–13170.
[47] Schwaber, K., and Sutherland, J., 2013, “The Scrum Guide ^{TM},” (July), pp. 1–16.
[48] “Scrum Primer” [Online]. Available: http://www.infoq.com/minibooks/Scrum_Primer.
[49] Lewis, K., and Mistree, F., 1998, “Collaborative, sequential, and isolated decisions in design,” Journal of Mechanical Design, **120**(12), pp. 643–652.
[50] Chanron, V., and Lewis, K., 2005, “A Study of Convergence in Decentralized Design Processes,” Research in Engineering Design, **16**(3), pp. 133–145.
[51] Devendorf, E., and Lewis, K., 2011, “The Impact of Process Architecture on Equilibrium Stability in Distributed Design,” Journal of Mechanical Design, **133**(10), pp. 1–12.
[52] Ghosh, S., Devendorf, E., and Lewis, K., 2014, “Testing the Parallel Character Hypothesis for Distributed Design Procceses Subjected to Stochastic Disruptions,” Artificial Intelligence for Engineering Design, Analysis & Manufacturing, **28**(4), accepted.
[53] Mebane, W. L., Carlson, C. M., Dowd, C., Singer, D. J., and Buckley, M. E., 2011, “Set-Based Design and the Ship to Shore Connector,” Naval Engineers Journal, **123**(3), pp. 79–92.
[54] Galvan, E., and Malak, R., 2012, “A Genetic Algorithm Approach for Technology Characterization,” ASME International Design Engineering Technical Conferences, Chicago, IL, pp. 1–12.
[55] Canbaz, B., Yannou, B., and Yvars, P.-A., 2013, “Expanding the Bottom-Up Approach Through Intergrating Design Attitudes into Set-Based Design,” ASME International Design Engineering Technical Conferences, Portland, OR, pp. 1–10.
[56] 2013, “HSC Dolphin Jet” [Online]. Available: http://en.wikipedia.org/wiki/HSC_Dolphin_Jet.
[57] Iansiti, M., 1995, “Shooting the Rapids: Managing Product Development in Turbulent Environments,” California Management Review, **38**(I), pp. 37–58.
[58] Ulrich, K. T., and Eppinger, S. D., 2012, Product Design and Development, McGraw-Hill, Inc, New York.
[59] Dahmus, J. B., Gonzalez-Zugasti, J. P., and Otto, K. N., 2000, “Modular Product Architecture,” ASME International Design Engineering Technical Conferences, Baltimore, MD, pp. 1–11.

APPENDIX

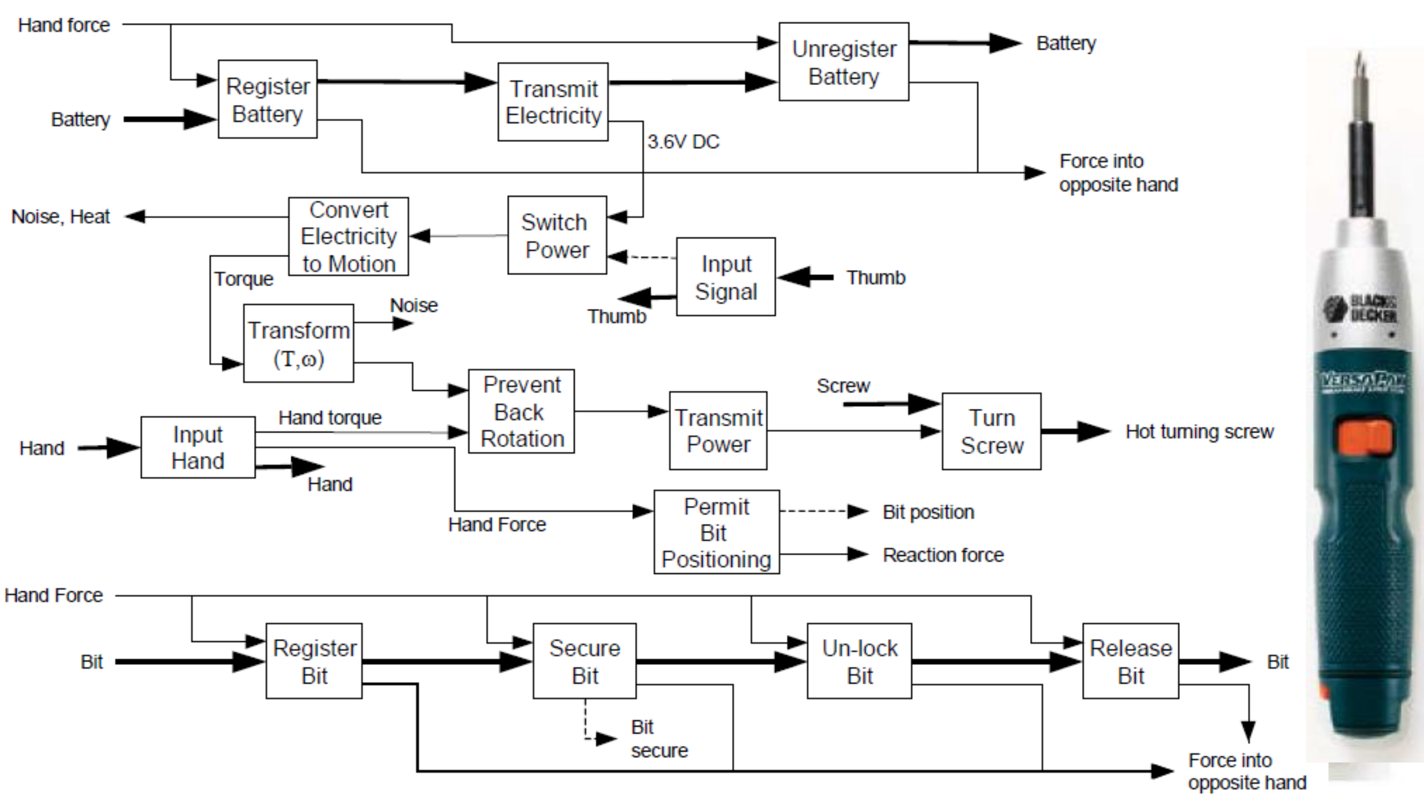

Figure A1: Function Structure for a VersaPak™ Cordless Screwdriver (from [59])