Modeling the Benefits of Frontloading and Knowledge Reuse in Lean Product Development

Endris Kerga[1], Roland Schmid[1], Eric Rebentisch[1], Sergio Terzi[2]
[1]Massachusetts Institute of Technology, Cambridge, MA - USA
[2]Politecnico di Milano, Milan, Italy

Abstract--**This paper uses system dynamics (SD) approach to model the lean product development strategy of set-based design (SBD) and the traditional product development strategy of point-based design (PBD). Using the Vensim System Dynamics (SD) tool, the paper investigates the performance outcomes of adopting the strategies within a new product development (NPD) context. The model simulates the development of five hypothetical projects. The model uses a classical SD project view as a rework cycle. It includes elements of both PBD and SBD which can be switched on/off to assess their impact on performance. Multiple projects are modeled so that the knowledge reuse issues are explored for differentiating PBD and SBD. In particular, we address the question of "what are the effects of frontloading in SBD on project durations, total project costs, and return on investment (ROI)?" From initial simulation results, it is found that in a typical PD, SBD can bring up to 25% reduction in average project durations, 40% reduction in total project costs and improves ROI significantly.**

I. INTRODUCTION

Companies are striving to compete globally by offering innovative, customer focused products as cheaply and timely as possible. However, it is challenging to design products that optimize customer interests, achieve time to market, and compete on cost. The design and management of the product development system and the choice of which strategies and processes to use will have a large impact on the company's ability to be competitive. Companies can choose, for instance, to design products one-at-a-time through a "clean sheet" design process that respond to perceived customer and market needs, base them on product lines or product families that reuse elements of the product and provide use continuity for the customer, or on platforms that leverage connections to related products in the platform network. Each of these approaches will have a different impact on the non-recurring and lifecycle costs of the product and profit for the company, as well as the company's responsiveness to market opportunities and product competitiveness. Understanding the relative merits of the different approaches and their associated benefits is therefore of strategic importance to the company.

A product development (PD) process can be viewed as "an organized group of interrelated activities that work together to create a result of value" [9]. While activities are performed by taking sequences of decisions, knowledge about the design evolves and uncertainties of information reduce [42].

An important feature of a PD process is uncertainty [42]. There are two classes of uncertainties in PD [24], [26]: (1) epistemic uncertainty (known unknown) which is the lack of knowledge; (2) aleatory uncertainty (unknown unknown) which is random. Epistemic uncertainty is reducible through reuse of previous knowledge, analyses, studies, measurement and experts' consultation [14, [34], [44]. Random uncertainty describes the inherent variation associated with a system or environment. Random uncertainty is often manifested as changes brought by events beyond the control of designers such as customer requirement changes or supplier changes, unexpected manufacturing changes, etc. [42]. Uncertainties cause changes, and the process should iterate to incorporate the changes [51]. To develop an optimal solution, strategies should be devised to reduce risks and tackle uncertainties [12], [13].

Approaches to handle uncertainties in PD include: point-based design (PBD) and set-based design (SBD). PBD is a prevalent and conventional approach in most companies who are developing new products [10], [11], [16], [32]. In PBD, product developers search for an optimal design or solution through a trial and error approach where a single alternative is selected at the onset of a project, and then rework on it until it becomes satisfactory. The rework is generated due to uncertainties during testing or validation. Rework is expensive, and doing design changes increases the development timeline [4]. Few resources are invested in PBD at the front-end of a PD process, but rework demands expensive iteration to handle defects. Moreover, in PBD, projects are not well-connected so that the knowledge generated in one project will not effectively be reused for subsequent projects [23], [49].

Set-based design (SBD) is recommended in the lean PD literature as an alternative to PBD to overcome the shortcomings of PBD [18], [39], [48]. Lean PD strives to eliminate waste of unnecessary rework in PD, and improve design value by offering optimal design and efficient utilization of knowledge [23], [49]. SBD considers a wider range of alternative sets in a design space, and an optimal design will be developed by rapidly converging on a preferred solution. In SBD, there is a relatively higher resource allocation at the font-end of projects for analyzing, prototyping, and testing multiple solutions to gather knowledge and reduce uncertainties, with the promise that the overall process will be more resource-efficient. The knowledge obtained in the process will be captured for future reuse.

Although discussions about the two approaches are studied in concurrent engineering (CE) literature, there is little quantitative research to compare the two approaches. In this paper, a system dynamics (SD) model is developed to analyze and compare the performance outcomes of the two approaches. SD modeling is a powerful tool to simulate a PD system which incorporates interdependencies between system variables, and consider several feedback loops in a PD system [40], [41].

Comparing the two approaches is paramount to show managers on the convenience of adopting lean strategy in PD. Companies although are interested to adopt SBD process, the proof of the SBC's productivity advantage is weakly articulated in previous researches [11], [32]. Thus, this paper aims to model and show the performance outcomes of adopting the strategies.

Comparing the two approaches would be challenging and expensive to tackle using other methodologies such as case studies or other analytical research methodologies. There are few opportunities to compare the two approaches in real projects while considering several differentiating parameters. Therefore, this paper uses a SD model to compare the outcomes of the two approaches to investigate the relative benefits of adopting the two approaches. It is the aim to study how adopting PBD or SBD potentially affect the PD outcomes in terms of project duration, project cost and return-on-investment (ROI).

Section 2 discusses in more details the differences of PBD and SBD through analyses of the literature. Previous studies on the performance outcomes of the two strategies, and the gaps this paper is aiming to fulfil are also the topics of section 2. Section 3 details the SD model developed and the assumptions considered while developing the model. The preliminary results obtained from the model are presented in section 4. Finally, conclusions of the paper and future researches are presented in section 5.

II. PBD VS. SBD

It is a primary task for engineering managers to prevent decisions being made too quickly using incomplete data [23]. Making decisions too early potentially causes changes in later phases of PD, which are expensive, suboptimal and degrade both product and process performances [23].

In PBD, the PD process rarely turns out to be linear in nature due to customer and downstream uncertainties. PBD results in iterative loops to either modify the selected solution until it satisfies the requirements or to start the process over again by selecting a completely different solution. Because of the iterative nature and the point to point search for feasible solutions, this process has been termed Point-based design (PBD) [39], [48]. Kennedy and Harmon [12] explained PBD approach as a 'design, and then test' cycle in which designers commit to a solution before gaining enough knowledge to do so [12]. However, once uncertainties (both epistemic and random) are revealed during testing, then rework loops become unavoidable. Since designers commit on several decisions, the cost of rework (changes) after testing is very expensive [4], [23].

With the objective of finding efficient strategies, researchers have compared western and Japanese approaches to PD [18], [39], [48], [50]. Among the companies that exhibit paradoxical PD approaches is Toyota. The so called "first Toyota paradox" deals with the efficient production system at Toyota, which had huge implications for companies in western countries [48]. Beside this unique production system, it is the way of doing PD that is atypical compared to companies in western countries. Toyota's approach of developing products is referred to as set-based design (SBD) and identified as the "second Toyota paradox" [48]. Researchers claim that this approach of developing products is as important as the Toyota production system. Toyota follows a seemingly inefficient approach to its development process upon initial inspection. In particular, it considers a broader range of possible designs, most of which will be abandoned along the way, and delays certain configuration and detailed design decisions longer than other car companies. However, Toyota has what may be the fastest and most efficient vehicle development cycles [23], [39], [48].

Sobek [37] defines SBD as when engineers and product designers 'reason, develop, and communicate about sets of solutions in parallel and relatively independent' [37]. The definition is further explained in three principles [39]: (1) 'Map the design space', which aims to achieve a thorough understanding of the sets of design possibilities; (2) 'Integrate by intersection', which ensures that subsystem solutions defined are workable/compatible with all functional groups involved; (3) 'Establish feasibility before commitment', that allows the aggressive elimination of inferior design solutions from sets.

Ghosh and Seering [8] argued that the above principles can be summarized into two principles, and it is possible to explicitly differentiate PBD and SBD approaches [8]. The principles are:

- Principle 1: Considering sets of alternatives concurrently
- Principle 2: Delaying convergent decision-making (elimination rather than selection)

The two principles provide a working definition for SBD. Examining the two principles, principle 1 can be exercised independently from principle 2. However, for exercising principle 2, principle 1 should be exercised a-priori. If sets are not considered, delaying decisions about the sets is not feasible.

Fig. 1 below shows the different scenarios based on these principles. In PBD, both principles are not exercised. This strategy is a traditional PBD where no sets of alternatives are explored and decision is made early. In SBD, both principles are exercised. Sets are explored in SBD and the convergence to the best alternative among the set is conducted by eliminating weak alternatives throughout the PD process. In trade-space exploration strategy, sets are explored but designers quickly converge on a solution. This strategy carries some set-based characteristics but sets are not pursued through, and progressive convergence is not exhibited. This strategy has been developed for the front-end of space system design [5]. Moreover, the prevalent Pugh method facilitates the trade-space exploration strategy [30]. In the Pugh approach, design alternatives are compared using criteria related to the performance of the alternatives, and the best is selected [30]. Although in the method principle 1 is exercised, principle 2 is not.

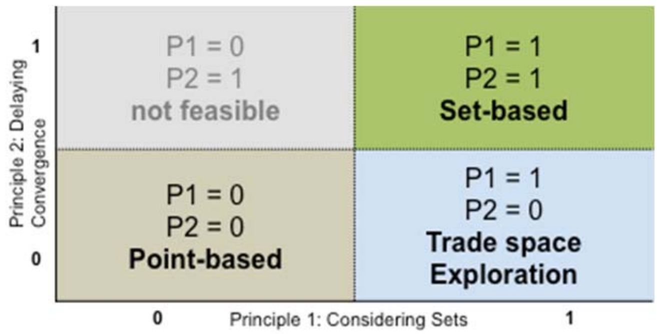

Fig. 1: Different scenarios for the principles (P1 = principle 1, P2= principle 2, 1 indicates the principle is exercised and 0 indicates the principle is not exercised).

The classification of the PBD and SBD using these principles facilitate modeling of the two approaches using a SD model. Furthermore, these additional aspects enable distinguishing between PBD and SBD:

i. Resources allocation and handling uncertainties
ii. Generating, capturing and reusing design knowledge

A. Resource allocation and handling uncertainties

To exercise the principles of SBD, upfront investment is required [39]. In SBD, a much larger number of alternative solutions are pursued early in the PD process. Then, designers test, analyze and build multiple solutions in parallel, and progressively converge to a feasible and optimal system solution [39], [48].

More resources are allocated upfront in SBD to explore a design space, and progressively converge into feasible solutions that balance customer, technical and business objectives. Fig. 2 (part II) shows the resource distribution in SBD. The frontloading effort in terms of resource allocation is to handle epistemic uncertainties within a design space. Frontloading is defined as "a strategy that seeks to improve development performances by shifting the identification and solving of problems to earlier phases of PD processes" [45]. Since there is a lack of knowledge about which alternatives meet customer and technical goals, higher resources are invested in the early stages of PD to explore and evaluate sets of alternatives. Moreover, feasible alternatives in the set are kept until they are proven infeasible. SBD is proactive strategy in handling uncertainties.

Kennedy and Harmon [12] explained SBD approach as a 'test, and then design' cycle in which front-end investments are made for pursuing/testing several options to gain knowledge to make decisions [12]. Thus, epistemic uncertainties are reduced at the front-end before testing, and a much leaner process can be achieved by reducing late changes and expensive rework [6], [12], [39], [48].

In most PBD practices, resources are allocated for a single or few alternatives at the beginning [32]. Fig. 2 (part I) shows the resource distribution in PBD. A solution is selected at the onset based on expert judgment, and assumes that the selected solution will meet requirements. Whenever there are failures due to uncertainties during testing or validation, more resources are allocated to adjust the selected solution or pick a different solution. Due to the uncertainties, expensive rework is likely to occur after validation, and resource allocation will be shifted near to the start of production in PBD. PBD is a reactive strategy to handling uncertainties.

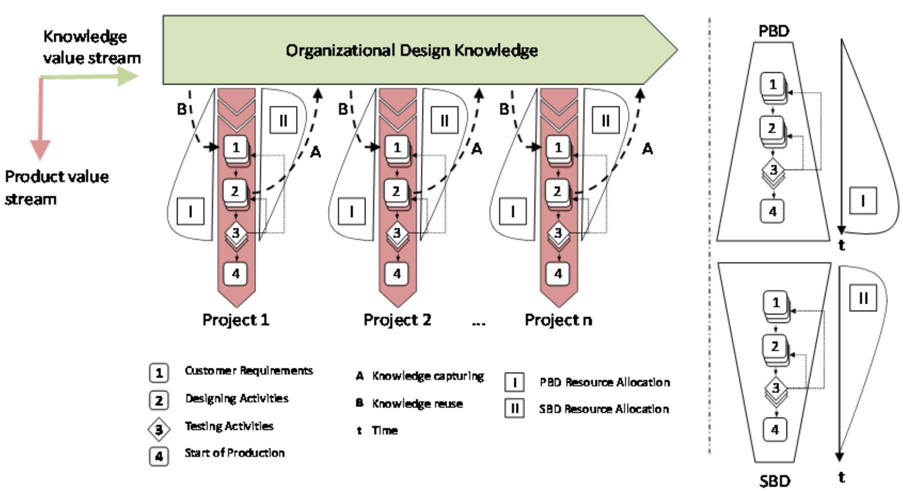

Fig. 2: Resource allocation, uncertainty handling and design knowledge in PBD and SBD.

B. Generating, capturing and reusing design knowledge

The quality of learning can be defined as the capability to effectively manage the design knowledge life-cycle such as the creation, capturing/ visualizing, transferring and reusing of knowledge to support decision making [25]. Knowledge is an organizational asset and value. Companies adopting lean thinking utilize and reuse design knowledge to respond to customer requests and solve problems faster for future projects [49].

The principles practiced in SBD significantly enhance the knowledge life-cycle [21], [49]. The higher front-end investments in SBD for considering sets, experimentation and learning provide the possibility to create more knowledge about design solutions and their performances. The more sets designers consider, the higher is the potential to create more usable and reusable knowledge [13], [27], [46]. The inflow of knowledge from projects (see Fig. 2, part A), is expected to be higher in SBD. In contrast, since there is no planned learning effort at the front-end in PBD, the possibility for generating reusable knowledge is minimal [20], [22], [36], [49]. The inflow of knowledge from projects (see Fig. 2, part A), is expected to be lower in PBD.

Also, SBD has important leverage for the effective capturing and visualization of knowledge [21], [23]. The knowledge representation tools used in SBD such as tradeoff curves, limit curves and performance checklists enable structured knowledge capturing and visualization [35], [38], [39]. The graphical representations of knowledge about design sets improve the readability and generalizability of data/information created concerning the technical and business related performances. Moreover, these tools improve knowledge transferability and reusability between projects. It is through these tools that knowledge from previous projects are used to 'front-load' the early phases of the subsequent projects in order to reduce uncertainties and avoid rework loops [13], [27]. Thus, the reusability of knowledge from previous projects (see Fig. 2, part B) is expected to be higher in SBD.

On the other hand, companies following a PBD approach have less opportunity to reuse knowledge from previous projects [13], [49]. The first reason is the lack of planned learning cycles at the front-end [13]. Pursuing a single or few solutions limits the capability to generate and capture generalized knowledge for future reuse. The second reason is due to the structuring of knowledge in PBD. Knowledge is generated on an ad-hoc basis after a failure occurs during testing, validation or integration [13]. Knowledge is generated to solve a specific problem when failure and rework are eminent due to uncertainties. The usage of knowledge capturing tools and methods are not prevalent in companies following PBD, and as a consequence knowledge reusability between projects is limited [32]. As a result, the reusability of knowledge from previous projects (see Fig. 2, part B) is expected to be lower in PBD. Table 1 summarizes the concepts presented in this section that distinguish the different strategies followed in PBD and SBD.

C. Previous studies on performance of PBD vs. SBD

Previous studies have investigated the performance differences based on case studies and modelling/simulation research methodologies.

The original case studies on Toyota's PD focus on collected evidence and interviews on the benefits of SBD over PBD [27], [39], [48]. In these studies, the main benefits of SBD are: (1) reliable and efficient communication among teams; (2) foster innovation in design and manufacturing; (3) better decision making early in PD process; (4) enhance institutional learning; (5) optimizing designs [39], [48]. Bernstein [2] conducted multiple case studies in the aerospace industry in the US. The study shows that SBD was able to reduce up to 50% of their rework costs [2]. Raudberget [31] conducted case studies on Swedish companies and reported the benefits of SBD. He observed that SBD's benefits go up to 75% reduction in project cost,

TABLE 1: COMPARATIVE SUMMARY OF PBD AND SBD.

Aspects	PBD	SBD
Search: how should solutions be found?	▪ Iterate on selected solution. ▪ Brainstorm new solutions if the selected solution fails.	▪ Define broader sets of alternatives. ▪ Delay decisions until feasibilities are proven.
Resources: how should resources be allocated?	▪ Budget resources for designing a single solution. ▪ Invest additional resources once uncertainties are revealed after testing and validation. ▪ Reactive approach to uncertainty reduction.	▪ Budget resources to consider sets and experimentation during design. ▪ Resources used to reduce epistemic uncertainties. ▪ Proactive approach to uncertainty reduction.
Epistemic uncertainties: how are knowledge gaps closed and handled?	▪ Use expert guess and judgment to make decisions during design. ▪ Knowledge is gathered through iteration.	▪ Consider broader sets and invest more resources to gather enough knowledge, and extra resources for uncover uncertainties.
Random uncertainties: how are random uncertainties handled?	▪ Unexpected changes are handled after test and validation.	▪ Unexpected changes are handled after test and validation.
Knowledge generation: how is design knowledge found to make decisions?	▪ Knowledge of a single or few solutions. ▪ Unplanned knowledge generation after validation (iteration).	▪ Knowledge of sets of solutions. ▪ Planned knowledge generation before validation.
Knowledge capturing/representation: how is design knowledge captured?	▪ Less structured. ▪ Knowledge of single or few solutions.	▪ Structure knowledge representation. ▪ Visual representation of knowledge. ▪ Knowledge of sets.
Knowledge reuse: how effective is design knowledge reusability?	▪ Ad hoc. ▪ Less effective.	▪ Deliberate. ▪ Highly effective.

50% reduction in lead time, 50-75% improvement in product technical performances (innovation), and 50-100 % reduction in warranty cost and number of engineering changes [31]. Kerga et al. [15] also studied the benefit of SBD on a case product and asserted that SBD enhances innovation and resulted in a 30% material and manufacturing cost reductions [15]. Ford and Sobek [6] and Schafer and Sorensen [33] developed analytical models showing that SBD can improve the expected value of a project in terms of project duration and costs [6], [33].

However, it should be noted that these studies, while focusing on evidence of SBD, found a number of related practices also to be in use. Consequently, this assessment of SBD is to some degree tainted by the understandable comingling of various PD practices in a real-world PD system. The studies are not sufficient to compare the two approaches. The studies are not presenting the comparisons of the two approaches taking similar projects running under similar PD inputs and constraints. This is due to the difficulty and expensiveness of running projects following the two approaches at the same time.

Comparing the two PD approaches require a systems view involving interrelationships among all factors to better predict outcomes. This is because the variations in specific practices employed between SBD and PBD result from interactions between strategies and parameters, rather than just on individual practices. Such a system thinking approach is the focus of recently emerging studies. Belay et al. [1] proposed a SD model to evaluate the impact of frontloading more resources in SBD [1]. The model investigates the shift of resource allocation in SBD towards the early phase (design) compared to the resource allocation in PBD after validation due to rework. The result shows that (under the assumptions considered), SBD reduces the total cost by more than half and improves lead time by 20% [1]. While the results show the benefits of adopting SBD over PBD, more modeling and analysis is required to advance the research. There are gaps in the concurrent literatures. In particular, the following considerations are given less attention in the modeling efforts to evaluate the outcomes of PBD and SBD:

i. *The effect of the reuse of the organisation design knowledge*. As previously discussed, one source of efficiency in SBD is the reusability of design knowledge. The previous SD model didn't consider this. The impacts of organisation knowledge to close knowledge gaps or reduce epistemic uncertainties in future projects should be included a modeling effort.
ii. *Single project vs. multiple projects*. Previous models analyze the outcomes of SBD considering a single project. However, the value analyses of SBD should include multiple projects [19]. Derivative projects significantly benefit from the investments of previous SBD projects by reusing knowledge generated from previous projects. Thus, simulation models should expand the span of project boundaries from single project to multiple projects.

Therefore, this paper considers a system view considering: (i) the effect of organisational design knowledge; (ii) evaluate project outcomes of the approaches for multiple projects in PD. Although previous researches claim that SBD improves design quality [6] and enhance creativity [15], the main focus of this paper is to compare and differentiate the efficiency or productivity of adopting SBD and PBD. The productivity is measured in this paper by using outcome measures such as project duration, project cost and ROI.

III. BUILDING THE SD MODEL

System Dynamics (SD) is a methodology for modeling, simulating, analyzing and designing of dynamic systems [6], [40]. It integrates engineering control theory with management theory and decision-making. What makes a SD model significant is the fact that managers can easily visualize the interaction between different factors and make it simple to control several non-linear parameters from causal loop relationships in PD [17]. In preparation for this research, variety of SD modeling tools has been closely examined, whereof the Vensim System Dynamics software was chosen.

A. Description of the SD model

The model is arranged in different views to ensure an intuitive understanding of the model. The model is structured as an IPO (input, process and output) model. The model has four views: parameter (input), project (process), knowledge management (process), and outcomes (output) views. The input part contains all exogenous parameters, the resource management and the context of uncertainty. The input part is the parameters view in the SD model. The process part contains the projects execution and the knowledge management parts. Projects activities and executions are included in the project level view of the model. Moreover, the knowledge management is included in the knowledge management view of the model. Finally, the output part is included in the outcomes view.

B. Parameters view

The parameter view includes the exogenous parameters or factors that affect different parts of the model and the outcomes of the two approaches. In the parameter view the resource management and uncertainty definitions are included. Uncertainty quantification is modeled using the Gompertz function to quantify both epistemic and random uncertainties [42]. Epistemic uncertainties are reduced at the rate called rate of uncertainty reduction (RUR) [42]. RUR is defined in the Appendix (Table 2). Resource allocation is different between PBD and SBD. The assumption in PBD assumes resources are allocated for the works in the design stock. In SBD additional resources are allocated considering the epistemic uncertainties in the design stock.

C. Project view

There are several SD models representing projects execution and management [4], [43]. The rework cycle model is the classical SD model used in several project management literatures [19]. This research is based on this classical rework cycle SD model. The illustration in Fig. 3 shows the central part of the SD model with a rework cycle. In this process view of the development process, the most important stocks, flows and variables are highlighted for a more detailed description.

The main part of the rework cycle in this research consists of three stocks, namely *Design Stock*, *Design Accomplished* and *Test Stock* (contains all undiscovered rework). These stocks are connected by four main flows as illustrated in Fig. 3. All tasks (requirements) that have to be worked on are in the Design Stock. Each task from the Design Stock that is done correctly is added to the stock of Design Accomplished. Tasks progress is constrained by resources, capacity and information availability, and management policies. A knowledge reuse flow contributes to the processing of tasks by accelerating the rate of progress due to the reuse of formerly captured knowledge. Five projects are simulated in the model. The projects can reuse knowledge from previous projects through the knowledge reuse flow. The rate of knowledge reuse depends on the innovation profile of the five projects.

Additionally, there is a certain amount of work that is not done correctly the first time. This error generation is accumulated in the Test Stock (Undiscovered Rework). The likelihood of a task being done correctly or incorrectly depends on the total uncertainty (epistemic and random) in the model. Epistemic uncertainty is handled during designing activity in SBD. Random uncertainties are handled as rework in both PBD and SBD. We can switch on/off PBD and SBD in the model to implement the strategies in the two approaches. The delay of the discovery of error is represented by rework discovery rate or time to detect errors. Additional work coming as rework causes additional work as new requirement through the rework penalty variable (RWP) [13], [31]. RWP is defined in the Appendix (Table 2). Total requirement stock thus accumulates the initial requirement and additional requirements. When Design Accomplished stock is the same as the total requirement stock, a project completes.

D. Knowledge management view

The knowledge manage view plays an important role in the model. Fig. 4 shows the model for the knowledge management view. The Knowledge Design Stock controls the knowledge capturing and reusing functions in the model. Engineering resources invested in project tasks translate tasks or activities into knowledge. Projects can also access the stock to reuse the knowledge stock to accelerate design tasks (see the knowledge reuse flow in Fig. 3 above). The inflow to the stock is affected by resources, project progress, innovation level of projects (ILP), knowledge saturation point (KSP), early and late knowledge capturing rates (EKC and LKC). A project with higher value of ILP is assumed to have higher knowledge generation productivity from a design task [3], [44], [47]. The outflow is affected by the knowledge decay rate (KDR). A project with higher value of ILP is assumed to have longer retention of knowledge (i.e., lower KDR). Expanded definitions of these variables are found in the Appendix (Table 2).

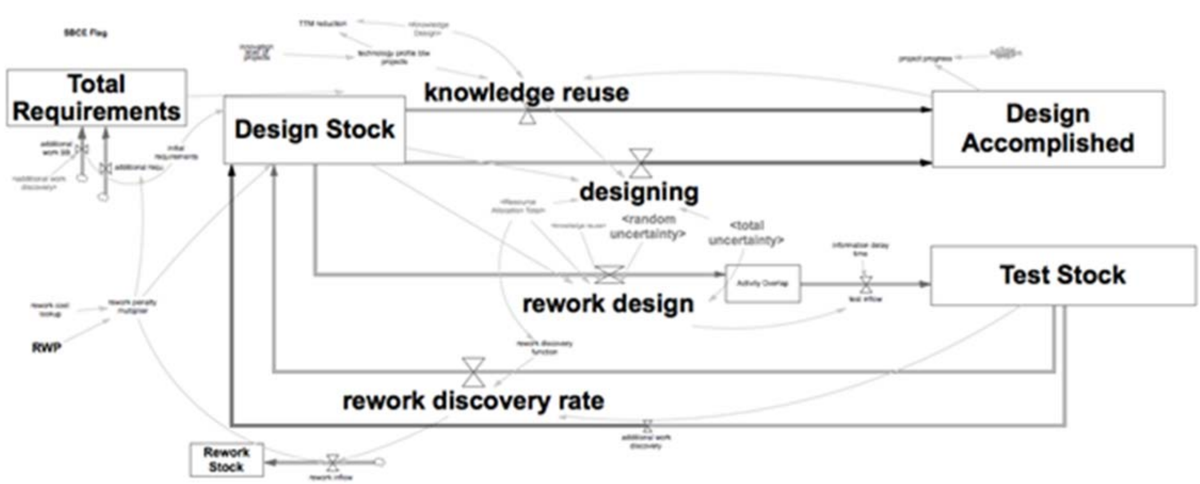

Fig. 3: Simplified project view as a rework cycle model.

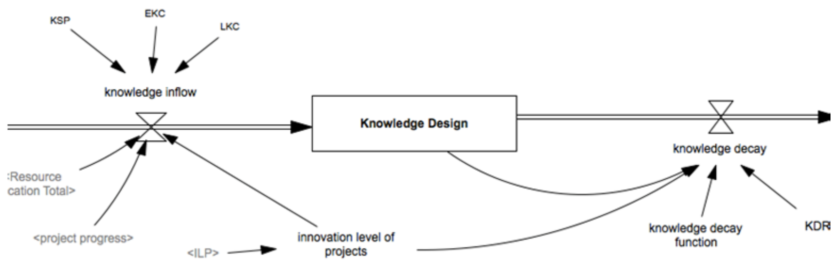

Fig. 4: Simplified knowledge management view.

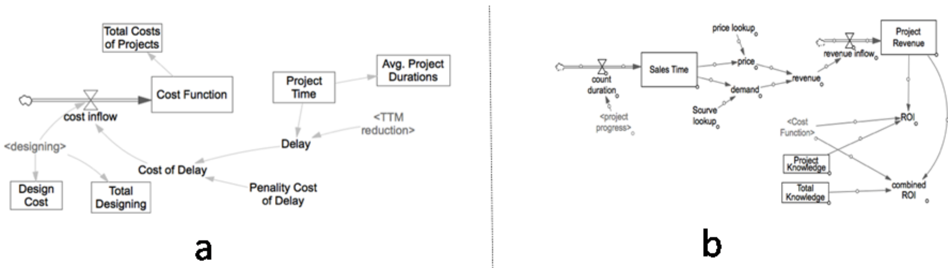

Fig. 5: Outcomes view (a) Project duration and cost (b) ROI.

E. Outcomes view

Figs. 5 (a) and (b) show the outcomes' views. The most common outcome measures considered for the comparison of the different simulations are project duration, project cost, and ROI. The average project durations, total project costs, and the combined ROI are also used to aggregate the results of the five projects simulated in the model.

IV. RESULTS AND DISCUSSIONS

As an initial investigation to the model, we run a scenario with specific sets of model parameters to determine the outcomes of PBD and SBD. The scenario is to simulate an assumed PD organization for 300 time units (~5 years). The following assumptions and settings are considered for the simulation of both PBD and SBD runs: (i) the initial requirement is considered to be 100 work units; (ii) the simulation assumes a rework penalty (RWP) of 10 (which means rework requires 10 times the level of effort of the initial design activity) [19], [44], [45]; (iii) five hypothetical projects are simulated to assess the impact of the different approaches over time. It is assumed that the five projects are derivative projects where the subsequent projects are derived from the first project. The first project is assumed to have a high level of innovation (ILP). The innovation contents of the subsequent projects are lower than the first; (ix) it is assumed that 10% of the knowledge generated in a project will decay or will not otherwise be reusable; (x) it is assumed that there is higher productivity of knowledge generation at the front-end or the beginning of projects compared to the later stage of projects. In particular, engineering work is translated to knowledge at an early knowledge capturing rate (EKC), which is higher than the late knowledge capturing rate (LKC).

We have conducted 50 simulation runs. Sensitivity and stability analysis consistent with SD modeling norms are conducted to ensure the model behaved appropriately. The following tests are considered for the model: parameter verification tests, extreme conditions tests, behavior prediction tests, behavior anomaly tests, and surprise behavior tests [7]. A full discussion of the sensitivity analysis and the tests results for the model are unfortunately beyond the scope and length limitations of this current paper. The following subsections discuss the outcome results of PBD and SBD considering the above assumptions and settings of the parameters.

A. Project durations

Project duration is a primary performance outcome to compare the two approaches. Figs. 6 (a) and (b) show the progresses and durations of the five projects in PBD and SBD. P1 completes in just over 90 and 60 time units in the PBD and the SBD cases respectively. The subsequent projects complete faster in both cases due to the knowledge reuse. Because the ILP profile (innovation level) is decreasing, derivative projects are able to reuse knowledge from previous projects. The five projects in PBD are completed in 240 time units, while in SBD case the projects are completed before 180 time units. The result shows that the resource invested for exploration and experimentation to handle epistemic uncertainties in SBD results a ~25% improvement in project completion time.

In the PBD case, each project takes longer compared with the SBD case. The rework penalty (RWP) plays an important role in longer project durations in PBD. In PBD, limited resources are invested during design. The resource invested for design activities in PBD is for the nominal value of the requirement without considering the epistemic uncertainties. However, once uncertainties are revealed after tests, resources are invested to reduce the uncertainties which results in expensive rework penalties. Although in SBD there are rework penalties due to random uncertainties, the strategy focuses on handling the partial uncertainties (epistemic uncertainties) before committing to a solution and conducting tests. In addition, the initial knowledge capturing activities in SBD are done early in a project in which there is high potential for generating knowledge efficiently. As a result, SBD benefits significantly the EKC rate whereas PBD utilizes the LKC to generate knowledge to tackle uncertainties.

B. Project costs

The project costs involve the engineering costs for design and testing activities, and the cost of delay. Figs. 6 (c) and (d) show the costs of the five projects in PBD and SBD. The total cost of the first project (P1) is about $350, 000 and $200, 000 in PBD and SBD cases respectively. The project costs of the subsequent projects decreases in both cases due to the knowledge reuse. The costs of the five projects in PBD are much higher (~40%) than the costs in SBD. Although in SBD there are higher engineering costs before testing (for exploration and experimentation to handle epistemic uncertainties in SBD), the costs of rework and delay are much smaller compared to PBD. The trends of the project costs are similar to the results found from previous researches [1], [28].

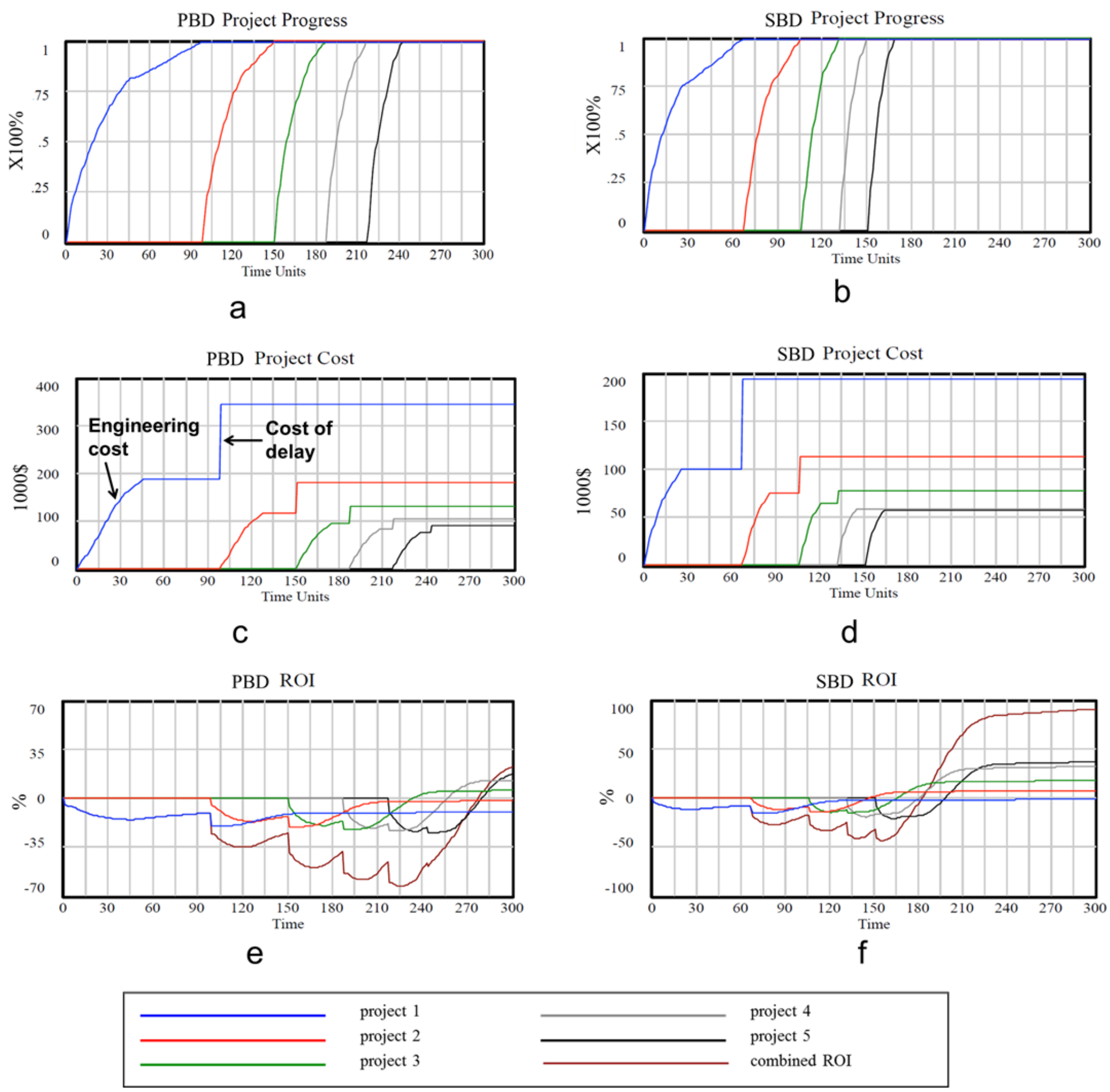

Fig. 6: Performance outcomes in PBD and SBD.

Looking at the trends of the cost of delay in Fig. 6 (c and d), the cost of delay for each project is higher in PBD than SBD. Project delays occur when a project finishes later than the planned time to market (TTM). The convergence process in the SBD approach enables fewer rework cycles, and the impacts of the RWP variable in the model generate less rework. As a result, the project delays are lower in the SBD case. Moreover, costs of delays reduce for subsequent projects due to the knowledge reuse in both PBD and SBD cases. However, much of the knowledge in PBD is generated through rework cycles. Using multiple rework cycles, epistemic uncertainties reduce in PBD at the expense of high engineering and delay costs.

C. ROI

ROI is a function of profit (revenue – project cost) and investments for each project. ROI is the ratio of the profit generated over the period of the simulation time and the investment in a project. Combined ROI is the total of ROIs of the five projects. Price follows a decreasing function over time, and demand follows an S-curve [29]. It is assumed that the revenue generation from a project starts as soon as the project is completed. Figs. 6 (e) and (f) show the project's ROI and combined ROI in PBD and SBD.

In PBD, projects 1 and 2 have negative ROIs. After project 3, the projects start generating positive ROI. The combined ROI is about 30% with a break-even point at 280 time units. On the other hand, the SBD case starts generating positive ROI from project 2. The combined ROI is about 90% with a break-even point at 180 time units. The PBD outcomes show that the delay in project delivery results in revenue losses. For example, projects 1 and 2 couldn't reach break-event points. Moreover, the project costs in PBD are high due to rework cycles and negatively affect project profits and ROI. Regarding investments, the total investment in knowledge in SBD is higher at the beginning but slows down near the start of production. The investment in PBD follows the opposite trend, where at the beginning of a project, few resources are invested, and near the start of production investments on knowledge increases. In general, the total investment in PBD is higher than SBD. Therefore, SBD has higher ROI compared to PBD considering the assumptions in the base case scenario.

V. CONCLUSIONS AND FUTURE RESEARCH

The value of adopting SBD over PBD has been argued in previous research. The current adoption level of SBD in companies is low. However, there is a growing interest to understand the benefits of SBD, and investigate the factors affecting the adoption of SBD over PBD. We have used a SD model to differentiate the two approaches, and investigate the performance outcomes. We differentiate between the two approaches by varying the allocation of resources, the method of uncertainty handling and the use of organisational design knowledge. By definition we have assumed that higher resources are allocated in SBD at the front-end (design stage) to consider baseline requirement and epistemic uncertainties. In PBD we assume that the epistemic uncertainties are resolved through rework cycles. Both approaches handle random uncertainties through rework cycles. Moreover, we assume that the PD system has the organizational design knowledge where knowledge is stored from projects, and knowledge is reused by projects.

Using SD as a modeling methodology has helped to uncover and understand the decision making complexity in PD system. The systematized modeling effort has focused on identifying the main sources of differentiating factors in PBD and SBD. It would have been expensive and very difficult to receive detailed comparative analyses between the approaches if other methodologies would have been considered. From the initial results obtained from the assumed scenario, SBD enables significant improvements compared to the PBD approach.

The model includes exogenous parameters that can further differentiate the performance outcomes of PBD and SBD. Some of these parameters are briefly mentioned in this paper such as RWP, RUR, ILP, KDR, KSP, EKC and LKC. A full discussion of the impact of these parameters on product development outcomes is unfortunately beyond the scope and length limitations of this current paper. Nevertheless, they have been formalized here in a single model of a product development system for perhaps the first time. Extensive sensitivity analysis of these factors promises to reveal much deeper insights into the dynamics of product development systems and strategies. Further research will present the results of the sensitivity analyses, and investigate the patterns of the outcomes over the parameters. Finally, this model should be more rigorously calibrated through collaboration with partners in product development organizations to further validate the results. While findings from prior empirical research were used to create and calibrate the model, it represents a composite view based on a number of different studies. Updating the model parameters with those obtained from a sample of specific product development systems would improve confidence in the findings and insights presented here.

REFERENCES

[1] Belay, A.M., Welo, T., & Helo, P.; "Approaching lean product development using system dynamics. Investigating front-loads effects," *Advances in Manufacturing*, 2(2), 130–140. DOI: 10.1007/s40436-014-0079-9, 2014.

[2] Bernstein, J.I.; "Design methods in the Aerospace industry: Looking for evidence of set-based practices," (Master's thesis, Massachusetts Institute of Technology), Retrieved from http://dspace.mit.edu/handle/1721.1/82675, 1998.

[3] Bogus, S.M., Molenaar, K.R., & Diekmann, J.E.; "A concurrent engineering approach to reducing design delivery time," *Journal of Construction Engineering and Ma*nagement, 131(11), 1179–85, 2005.

[4] Cooper, K.G.; "The Rework Cycle: Benchmarks for the Program Manager," *Project Management Journal*, 24(1), 17–21, 1993.

[5] Derleth J.E.; "Multi-Attribute Trade space Exploration and its Application to Evolutionary Acquisition," (Master thesis,

Massachusetts Institute of Technology), Retrieved from http://seari.mit.edu/documents/theses/SM_DERLETH.pdf, 2003.
[6] Ford, D.N., & Sobek, D.K.; "Adapting real options to new product development by modeling the second Toyota paradox," *IEEE Transactions on Engineering Management*, 52 (2), 175–185, 2005.
[7] Forrester, J. W. & Senge, P. M.; "Tests for Building Confidence in System Dynamics Models. System Dynamics," *TIMS Studies in the Management Sciences* 14: 209-228.A. A. Legasto, Jr. et al., Ed. New York, North-Holland, 1980.
[8] Ghosh, S., & Seering, W.; "Set-Based Thinking in the Engineering Design Community and Beyond," *Paper presented at the ASME 2014 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Buffalo, NY, USA. DOI: 10.1115/DETC2014-35597, 2014.
[9] Hammer, M.; "Seven insights about processes," *In: Proceedings of the Conference on Strategic Power Process Ensuring Survival Creating Competitive Advantage*, Boston, MA, US, 2001.
[10] Hoppmann, J., Rebentisch, E., Dombrowski, U., & Zahn, T.; "A framework for organizing lean Product Development," *Engineering Management Journal*, 23(1), 3–15, 2011.
[11] Hoppmann, J.; "The lean innovation roadmap: A systematic approach to introducing lean in product development process and establishing a learning organization," (Diploma thesis, Technical University of Braunschweig), Retrieved from http://dspace.mit.edu/handle/1721.1/81770, 2009.
[12] Kennedy, M.N., & Harmon, K.; *Ready, Set, Dominate: Implement Toyota's set-based learning for developing products and nobody can catch you*. Virginia, VA: Oaklea Press, 2008.
[13] Kennedy, B.M., Sobek, D., & Kennedy, M.; "Reducing rework by applying set-based practices early in the systems engineering process," *Journal of Systems Engineering*, 17(3), 278–296, 2014.
[14] Kerga, E., Rossi, M., Taisch, M., & Terzi, S.; "A serious game for introducing set-based concurrent engineering in industrial practices," *Concurrent Engineering Research and Application*, DOI: 10.1177/1063293X14550104, 2014a.
[15] Kerga, E., Taisch, M., Terzi, S., Bessega, W., & Rosso, A.; "Set-based concurrent engineering innovation roadmap (SBCE IR): A case on Adiabatic Humidification System," *Int. Journal of Design Creativity and Innovation*, 2(4), 224–255, 2014b.
[16] Khan, M.S., Al-Ashaab, A., Shehab, E. Haque, B., Ewers, P., Soril, M., & Sopelana, A.; "Towards lean product and process development," *Int. J. of Computer Integrated Manufacturing*, 26(2), 1105–16, 2013.
[17] Lai, C.; "Research on modeling of product development complex system based on system dynamics," *In: IEEE international symposium on knowledge acquisition and modeling (KAM08)*, pp. 361–365, 2008.
[18] Liker, J.K., Sobek, II D.K., Ward, A.C., & Cristiano, J.J.; "Involving suppliers in product development in the United States and Japan: Evidence for set-based concurrent engineering," *IEEE Transactions on Engineering Management*, 43(2), 165–178, 1996.
[19] Lyneis, J.M., & Ford, D. N.; "System dynamics applied to project management: a survey, assessment, and directions for future research," *System Dynamics Review*, 23(2/3), 157–189, 2007.
[20] Martínez León, H.C., & Farris, J.A.; "Lean product development research: current state and future direction," *Engineering Management Journal*, 23 (1), 29–51, 2011.
[21] Mascitelli, R.; *The lean product development guidebook: Everything your design team needs to improve efficiency and slash time-to-market*. California, CA: Technology Perspectives, 2007.
[22] Mebane, W.L., Carlson, C.M., Dowd, C., Singer, D.J., & Buckley M.E.; "Set-based design and the ship to shore connector," *Journal of Naval Engineers*, 123(3), 79–92, 2011.
[23] Morgan, J.M., & Liker, J.K.; T*he Toyota product development system: Integrating people, process, and technology*. New York, NY: Productivity Press, 2006.
[24] Mourelatos, Z., & Zhou, J.; "Reliability estimation and design with insufficient data based on possibility theory," *American Institute of Aeronautics and Astronautics Journal*, 43(3), 1696–1705, 2005.
[25] Nonaka, I., Von Krogh, G., & Voelpel, S.; "Organizational knowledge creation theory: Evolutionary paths and future advances," *Organization Studies*, 27(8), 1179–1208. doi: 10.1177/0170840606066312, 2006.
[26] Oberkampf, W.L., Holton, J.C., Joslyn, C.A., Wojtkiewicz, S.F., & Ferson, S.; "Challenge problems: Uncertainty in system response given uncertain parameters," *Reliability Engineering and System Safety*, 85(1-3), 11–19, 2004.
[27] Oosterwal, D.P.; *The lean machine: How Harley-Davidson drove top-line growth and profitability with revolutionary lean product development*. New York, NY: American Management Association, 2010.
[28] Pesonen, "Implementation of design to profit in a complex and dynamic business context", Retrieved 07/07/15 World Wide Web, http://herkules.oulu.fi/isbn9514264509/isbn9514264509.pdf
[29] Prasad, B.; "Analysis of pricing strategies for new product introduction," *J. of Prod. Brand Management*, 5(4), 132–141, 1997.
[30] Pugh, S.; *Creating innovative products using total design*. Reading, MA: Addison-Wesley Publishing Company, 1996.
[31] Raudberget, D.; "Practical applications of set-based concurrent engineering in industry," *J. of Mechanical Engineering*, 56(11), 685–695, 2010.
[32] Rossi, M., Kerga, E., Taisch, M., & Terzi, S., "|Lean product development: Fact finding research in Italy," P*resented at the International Conference on Industrial Engineering and systems Management (IESM)*, Metz, France. Abstract retrieved from http://www.i4e2.com/iesm11/free_access_documents/proceedings_outlines.pdf, 2011.
[33] Schafer, H. & Sorensen, D.J.; "Creating options while designing prototypes: Value management in the automobile industry," *Journal of Manufacturing Technology Management*, 21 (6), 721–742, 2010.
[34] Schrader, S., Riggs, W.M., & Smith, R.P.; "Choice over uncertainty and ambiguity in technical problem solving," *Journal of Engineering and Technology Management*, 10(1-2), 73–99, 1993.
[35] Shook, J.; *Managing to learn using the A3 management process*. Cambridge, MA: Lean Enterprise Institutes Inc., 2008.
[36] Singer, D.J., Doerry, N., & Buckley, M.E.; "What is set based design?" *Journal of Naval Engineering*, 121(4), 31–43, 2009.
[37] Sobek, D.K., "Principles that Shape Product Development Systems: A Toyota and Chrysler Comparison" (PhD. Dissertation, Industrial and Operations Engineering, University of Michigan), Retrieved from http://ir.library.oregonstate.edu/xmlui/handle/1957/19501, 1997.
[38] Sobek, D. K., & Smalley, A.; *Understanding A3 thinking. A critical component of Toyota's PDCA management system*. New York: CRC Press, 2008.
[39] Sobek, D.K., Ward, A.C., & Liker, J.K.; "Toyota's principles of set-based concurrent engineering," *Sloan Management Review*, 40(2), 67–83, 1999.
[40] Sterman, J.; *Business dynamics: Systems thinking and modeling for a complex world*. Boston: Irwin/McGraw-Hill, 2000.
[41] Sterman, J.; "All models are wrong: reflections on becoming a system scientist," *System Dynamics Review*, 18(4), 501–531, 2002.
[42] Suss, S., & Thomson, V.J.; "Improved Span Time Performance in NPD Using Better Coordination," *In L. Rivest, A. Bouras, and B. Louhichi (Eds.), IFIP WG 5.1 International Conference (PLM 2012): Towards Knowledge-Rich Enterprises (pp.310–321)*, Montreal, Canada. DOI: 10.1007/978-3-642-35758-9_27, 2012.
[43] Taylor, T., Ford, N.; "Tipping point failure and robustness in single development projects," *System Dynamics Review*, 22(1), 51–71, 2006.
[44] Terwiesch, C., Loch, C.H., & De Meyer, A.; "Exchanging preliminary information in concurrent engineering: alternative coordination strategies," *Organization Science*, 13(4), 402–19, 2002.
[45] Thomke, S.H., & Fujimoto, T.; "The Effect of Front-Loading problem solving on product development performance," *Journal of Product Innovation Management*, 17(2), 128–142, 2000.
[46] Vassilvitskii, S., & Yannakakis, M.; "Efficiently computing succinct trade-off curves," *Journal of Theoretical Computer Science*, 348(2), 334–356, 2005.
[47] Verganti, R.; "Planned Flexibility: Linking Anticipation and Reaction in Product Development Projects," *Int. J. of Product Innovation*, 16, 363–376, 1999.
[48] Ward, A.C, Liker, J.K., Cristiano, J.J., & Sobek, D.K.; "The second Toyota paradox: How delaying decisions can make better cars faster," *Sloan Management Review*, 36(3), 43–61, 1995.

[49] Ward, A.C., Shook, J.K., & Sobek, D.K.; *Lean product and process development*. Cambridge, Massachusetts: The Lean Enterprise Institute, 2007.
[50] Womack J.P., Jones D.T., & Roos D.; *The machine that changed the world*. New York, NY: Rawson Associates, HarperCollins, 1990.
[51] Wood, K.L., Antonsson, E.K., & Beck, J.L.; "Representing imprecision in engineering design: Comparing fuzzy and probability calculus," *Research in Engineering Design*, 1(3), 187–203, 1990.

APPENDIX

TABLE 2: EXOGENOUS PARAMETERS AND DEFINITIONS IN THE MODEL.

Parameter	**Definition**
Rework Penalty variable (RWP)	The magnitude of the penalty of the rework from testing to design. A low RWP value represents a process with low penalty cost of error due to rework. A high RWP value represents a high penalty of correcting error due to rework.
Rate of Uncertainty Reduction (RUR)	Speed of the feedback from testing to design activities. A low RUR value represents a slow (delayed) feedback process. A high RUR value represents a rapid (instantaneous) feedback process.
Innovation Level of Projects (ILP)	Degree in which a new product is novel. The ILP value indicates the innovation content of the first project. The subsequent projects are assumed to be derivative of the first with decreasing ILP profiles. A low ILP value implies the projects are improvement projects. A high ILP value implies at least the first project is radical project.
Knowledge Decay Rate (KDR)	The rate at which a certain amount of knowledge from the knowledge stock is useless. A low value of KDR represents a higher retention of knowledge in PD. A high KDR value represents a lower retention rate of knowledge.
Knowledge Saturation Point (KSP)	A point in the project timeline beyond which the model uses LKC, and before which the model uses EKC to translate design efforts into knowledge. A low KSP represents a PD that has high productivity of generating knowledge at the beginning of a project depending on EKC. A high KSP represents a PD that has high productivity of generating knowledge till the end of a project depending on EKC.
Early Knowledge Capturing (EKC)	Rate at which design efforts are translated into knowledge before the KSP. A low ECK represents a PD that has low productivity of generating knowledge before KSP. A high EKC represents a PD that has high productivity of generating knowledge before KSP.
Late Knowledge Capturing (LKC)	Rate at which design efforts are translated into knowledge after the KSP. A low LKC represents a PD that has low productivity of generating knowledge after KSP. A high LKC represents a PD that has high productivity of generating knowledge after KSP.