Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 153 (2019) 118–125

Procedia Computer Science

www.elsevier.com/locate/procedia

17th Annual Conference on Systems Engineering Research (CSER)

Review of Research into the Nature of Engineering and Development Rework: Need for a Systems Engineering Framework for Enabling Rapid Prototyping and Rapid Fielding

Shawn Dullen[a]*, Dr. Dinesh Verma[b], Dr. Mark Blackburn[b]

Combat Capabilities Development Command (CCDC) Armaments Center, Picatinny Arsenal, 07806, USA
Systems Engineering Research Center (SERC), Stevens Institute of Technology, Hoboken, NJ, 07030, USA

Abstract

Given the need to develop a systems engineering framework to enable rapid prototyping and rapid fielding capability for the U.S. Department of Defense (DOD) per Public Law 114-92 and the fact that historically rework has been a problem during product development, a literature survey of engineering and design rework was conducted to better understand its nature and causes. The intent of the survey is to present the current state of research in the understanding of this aspect of development and to articulate future research areas for developing a systems engineering framework during the Technology Maturation and Risk Reduction (TMRR) phase of the DOD life cycle that addresses rework concerns, accelerates iteration and enables rapid prototyping. Since much of the research on rework has been done on information exchange and organizational structure there is a need for future research in systems engineering to develop frameworks to: 1) mitigate the impact of information uncertainty and instability, 2) accelerate information evolution, and 3) reuse knowledge for engineering reasoning.

Keywords: Rework, Corrective Iteration; Firefighting, Negative Iteration; Product Development; System Engineering

Distribution Statement A: Approved for Public Release; Distribution is unlimited

* Corresponding author. Tel.: 973-724-5176; fax: 973-724-6757.
E-mail address: shawn.m.dullen.civ@mail.mil

1. Introduction

The need for a systems engineering framework to provide rapid prototyping and rapid fielding capability was written into Public Law (114-92) on November 25, 2015 [1]. This is a monumental task considering historically the Department of Defense (DOD) has had trouble implementing standard acquisition strategies. In 2014, the Government Accountability Office (GAO) released a report stating cost and schedule growth remain significant; 42 percent of programs have had unit cost growth of 25 percent or more [2]. From 1997-2009, there were 74 Nunn-McCurdy breaches (there are two types: 1) significant breach occurs when cost growth is at least 15 percent over the current baseline estimate or at least 30 percent over the original baseline estimate and 2) critical breach occurs when cost growth is at least 25 percent over the current baseline estimate or at least 50 percent over the original baseline estimate) involving 47 major defense acquisition programs where 18 programs breached more than one time [3]. The top three factors were: 1) engineering design issues, 2) schedule issues, and 3) quantity changes [3].

Engineering design issues lead to reworking the design for a variety of reasons. Rework is a significant component of product development cycle time that can represent as much as two thirds of project effort [4]. The DOD is not the only organization that experiences negative impacts due to reworking the design. Research conducted by Kennedy revealed larger companies had up to 70 to 80 percent of their development time spent reworking the design [5]. Most design efforts have significant rework, between 30 and 65 percent of the total design hours on a project [6].

Design decisions made in the concepting phase of the product life cycle account for 86 percent of the cost impact of all design decisions [7]. There have been other studies on the 80-20 rule that indicated similar results [8]. Based on this information there exists a need to address rework during concept development, which occurs during the Technology Maturation and Risk Reduction (TMRR) phase of the DOD acquisition life cycle [9]. The TMRR phase has the biggest influence on rapid prototyping and rapid fielding. Addressing rework concerns and accelerating iteration by developing a Systems Engineering Framework for the TMRR phase to enable rapid prototyping and rapid fielding will be the focus of future research.

Since design decisions made in TMRR effect the product development (PD) phase in the form of costly rework and schedule delays, the current state of the art in mitigating rework in the TMRR phase was studied. Therefore, the focus of this literature search was to understand the causes of rework and previous work to mitigate the rework. The remainder of the paper is as follows: section 2) Literature Review Methodology and Definitions, section 3) Reasons for Rework, section 4) Mitigation Strategies for Rework, section 5) Future Work, and section 6) Conclusion.

2. Literature Review Methodology

The literature search used the following key words: rework, iteration, firefighting, churn, design rework, design iteration. The literature was limited to journal papers and conference proceedings. The following journals were considered: Concurrent Engineering, Construction Management and Economics, Decision Sciences, European Journal of Operational Research, Guidelines for a Decision Support Method Adapted to NPD Processes, IEEE transactions on engineering management, Journal of Intelligent Manufacturing, Journal of Product Innovation Management, Management Science, Organization Science, Research in Engineering Design, and ASME Journal of Engineering Design, and Systems Engineering.

2.1 Definitions of Rework

For product development, there are many different definitions of rework. Costa and Sobek [10] concluded that the definition of rework is the repeating of an activity (design) at the same scope and abstraction level. Repenning [11] referred to fire fighting in the same context as rework, he defined this as the unplanned allocation of resources to fix problems discovered late in a product's development cycle. Wynn and Eckert [12] defined rework as redoing tasks in a similar way because inputs or assumptions changed. Mitchell and Nault [13] defined rework as the design change whose implementation alters work that was previously done upstream and downstream. Arundacahawat et al. [14] defined rework as unnecessary repetition of design effort. Kennedy et al. [5] defined rework as the work that occurs when a prior decision that was assumed to be final for that project is changed because it was later found to be defective. Taylor and Ford [15] referred to rework as a task that needs to be redone because of a change. Smith and Eppinger

[16] defined rework as the required repetition of a task because it was originally attempted with imperfect information (assumptions). Another term that has a similar context to rework is churn. Wynn and Eckert [12] defined churn as the ongoing corrective iterations where solving problems creates more problems without quick termination.

3. Reasons for Rework

There has been limited research on rework with respect to systems engineering and PD activities that can be used to reduce rework based on the scope of this survey. Much of the research on rework so far has explored information exchanges related to task dependency, process execution, project complexity, information evolution, and information completeness.

Task dependency can be of three types: 1) independent, 2) dependent, and 3) interdependent. Process execution can be of three types: 1) sequencing of tasks, 2) partial overlapping of tasks, and 3) full overlapping of tasks. With respect to rework due to information exchange independent tasks are not of a concern. Sequencing dependent tasks as shown in Figure 1 (a) can take a significant amount of time to complete. In order to speed up the process, there is a need to overlap tasks which are dependent depicted in Figure 1 (b and c). Task B needs information from Task A in order for the activity to occur. When this downstream activity starts with preliminary information that may be uncertain or ambiguous there is risk that the activity will need to be reworked.

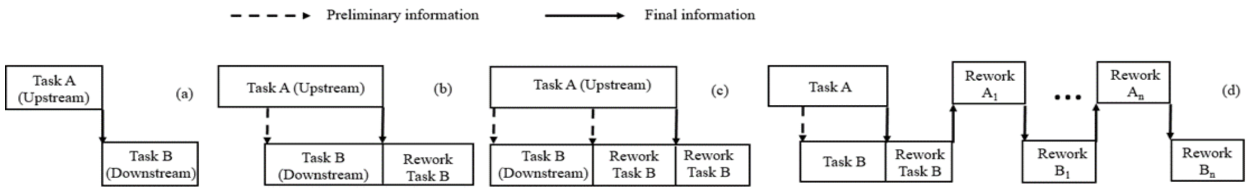

Figure 1. (a) Sequential dependent tasks; (b) Partial overlapped dependent tasks; (c) Fully overlapped dependent tasks; (d) Interdependent tasks

Schrader et al. [17] identified the difference between uncertainty and ambiguity for technical problems. The technical problem is uncertain when the problem is understood, but the value of these variables is unknown [17]. The technical problem is ambiguous when neither the variables themselves are known nor the mechanism to solve the problem to increase the knowledge [17]. Terwiesch et al. [18] also identified the precision and stability of the upstream preliminary information has an impact on rework. Information stability is the likelihood of the preliminary information no longer changing through the remainder of the process. Ambiguous problems are known to cause instability in information. The evolution of information influences the precision and stability of information. The extent of the rework will be a function of information evolution and downstream sensitivity. The faster the evolution of the upstream activity, the less likely it is that upstream information will substantially change [19]. The sensitivity of the downstream activity describes the extent to which changes in upstream information create rework in the downstream activity [19].

The next concern is with interdependent tasks which has both tasks reliant on each other. Downstream activity is dependent on upstream information and upstream activity is dependent on downstream feedback as shown in Figure 1 (d). The concern is that the downstream activity is dependent on upstream preliminary information and the upstream activity is dependent on downstream preliminary information. Uncertainty and instability concerns with the information creates high risk of reworking the tasks, ultimately causing a viscous set of iterations that leads to the churning effect.

Project complexity is another major factor that was considered for influencing rework. Complex systems require multiple integrated product teams (IPTs) to develop system, sub-system, and components. The more complex the system, the greater the chance that coupling exists within the design. Improved communication should exist between the IPTs, however, for more complex systems, there are more IPTs working on the system, increasing the risk that the product architecture and the organization structure are misaligned [20]. This misalignment leads to sub-optimal designs and interface issues usually identified later in the life cycle (testing).

4. Mitigation Strategies for Rework

4.1 Information sensitivity and knowledge evolution

Ha and Porteus [21] developed a dynamic model to optimize the number of design reviews for concurrent development of product and process design. The design reviews are used to prevent downstream problems through information exchange between product and process design. Instead of using design reviews to influence information exchange, many authors recommend using overlapping strategies. Krishnan et al. [22] developed an overlapping framework to determine the optimal timing of preliminary information release to minimize PD time. Other authors considered the communication of information and the overlapping of activities. Loch and Terweisch [23] developed an analytical model to account for uncertainty with preliminary information and dependency between teams performing overlapped activities. The analytical model can be used to understand the level of concurrency (overlapping) and the amount of communication required to reduce time to market. Chakravarty [24] developed an optimization model to: determine the optimal overlap strategy and understand the impact of build work quality on overlap decisions based on time and cost. The model considered three different overlapping modes: 1) interrupted build mode, 2) continuous build mode, and 3) preempt build mode. Uncertainty was considered as the probability of incompatible design segments. Roemer and Ahmadi [25] developed a deterministic model to determine the optimal overlapping and crashing strategy based on different upstream evolution and downstream sensitivity. Crashing is working with higher work intensities to reduce development times [25].

Some authors used surveys to develop frameworks for mitigating rework. For instance, Bogus et al. [19] developed a framework for identifying overlapping strategies for dependent design activities based on survey research that they conducted on industry. Their framework considered the upstream design evolution and the downstream sensitivity to upstream information. The authors propose two strategies for reducing downstream sensitivity: 1) overdesign and 2) set-based design. The authors propose four strategies for speeding up the upstream design evolution of information: 1) early freezing of design criteria, 2) early release of preliminary information and prototyping, 3) no iteration or optimization and 4) standardization. Mitchell and Nault [13] conducted survey research to understand the impact of: 1) cooperative planning on upstream and downstream rework, 2) uncertainty on upstream and downstream rework and 3) rework on the duration of project delay. They concluded that more cooperative planning reduces the magnitude of both upstream and downstream rework, decoupling the downstream delay from upstream uncertainty. Another survey approach was conducted by Arundacahawat et al. [26] to develop a framework for estimating design rework based on analogy, so that effective plans for design rework mitigation efforts can be developed in the early stages of PD. To develop the framework, two companies from the automotive industry were interviewed. The interviews were semi-structured, to identify the best practices to avoid design rework for the following rework drivers: 1) project complexity, 2) task dependency, 3) pre-communication, and 4) crashing.

Other authors considered improving the information that is being exchanged between dependent activities by incorporating testing earlier into the product development process. Through early testing, the evolution of information will be accelerated. Thomke and Bell [27] developed a mathematical model to determine the optimal timing, frequency, and fidelity of sequential testing activities in order to reduce uncertainty about technical solutions and customer needs (balancing the cost of repeated testing and the benefits of early information). This model did not consider the impact testing has on communication and coordination within development teams, nor did it consider conditions of rapid change. Tahera et al. [28] developed a framework to improve knowledge development by overlapping the testing and design activities. Since testing is costly and time consuming, the authors propose the use of a calibrated and validated "virtual model" to generate improved values quickly for downstream design tasks. The proposed method requires parallel virtual and physical testing.

The use of a design structure matrix (DSM) was proposed by many authors for mitigating rework. Cho and Eppinger [29] developed a generalized process model that used DSM method and advanced simulation techniques such as the Latin Hypercube Sampling and parallel discrete event simulation. The model can be used to evaluate the sensitivity of project lead time to variation in task duration, the impact to overlapping dependent tasks, and how to develop faster and few iterations. Smith and Eppinger [30] developed a modified design structure matrix using the Markov reward model to determine the ordering of coupled tasks and the estimation of their PD time. Yang et al. [31] developed a two-stage clustering framework using DSM for improving team coordination and reducing the coordination time. Their model uses the evolution DSM and sensitivity DSM to measure the interaction strength between teams performing overlapped activities. The first-stage clustering criterion is to maximize the interaction strength internally and the second-stage clustering criterion is to minimize total coordination time. Yassine et al. [32] developed a framework using DSM to study the conditions of the PD process that would cause churn due to information hiding related to interdependency between local tasks and systems tasks. The model incorporated two types of information flows: 1) information flows that reflect internal rework within local and systems groups and 2) information flows that reflect status updates from local to systems tasks and feedback from systems to local tasks [32].

Some authors considered task interdependency. Terwiesch et al. [18] developed a dynamic model for coordinating interdependent tasks based on data collected from a vehicle development project that can be used to help determine information coordination strategies. The coordination strategy can be either iterative or set-based, which depends on the stability and precision of the information. Stability is more beneficial when the cost of rework is higher than the cost of duplication and starvation. Yassine et al. [33] developed a probabilistic model that considered the execution structure, probability of design change, and the type of task dependency to understand the impact on development time and cost. Based on the model results the authors provided a few guidelines: use full overlap when development time has the highest priority, use partial overlapping when cost is the priority, and use partial overlapping when the probability of an engineering change is above 0.7. Jun et al. [34] developed a PD model to estimate cycle time that considered information dependency, relation cardinality, degree of overlapping, type of collaboration, type of routing, and type of synchronization. The authors classified seven design patterns: 1) feedback, 2) branching and merging, 3) no-overlap, 4) interaction, 5) overlap, 6) cycle and 7) communication. Nelson et al. [35] developed a Graphical Evaluation and Review Technique network model to understand the communication flows and information transfer in PD so that decision makers can determine what information needs to be transferred and when it needs to be transferred.

System dynamics and agent based models were developed by many authors to understand the different factors that impact rework. Repenning [11] developed a system dynamics model to study the causes of persistent rework. The model considered dynamics of resource allocation among competing projects in two different phases of the development process: 1) concept development phase and 2) the product design and testing phase. Based on the results of the model they concluded there are tipping points for multiproject development efforts. Once the tipping points are passed rework becomes a self-reinforcing and self-sustaining phenomenon. Taylor and Ford [15] came to the same conclusion using their developed system dynamics model to understand the features that control loop dominance. Ford and Sterman [36] developed a system dynamics model to understand the impact of concealing rework requirements on project overruns. The model considers the interactions between technical activities and human behavior. Yang et al. [37] developed a discrete-event simulation model to reveal how uncertainty related to iteration and ambiguity related to overlapping impact project duration. Levardy and Browning [38] developed an adaptive process model that provides insight on how to use the information to make informed decisions on making project changes during the course of the projects through simulation. The adaptive process model is an agent based model used to identify when to increase knowledge using iteration that considers the states of the project as a function of time, cost, technical performance, goals for each of them and the stakeholder utilities.

Braha and Bar-Yam [39] developed a dynamic network model for a complex PD process to study the PD dynamics. The model revealed focusing engineering and management efforts on central information-consuming and information generating PD tasks will likely improve the performance of the overall PD process [39]. The model can also be used to perform a trade-off between the elimination of task dependencies and the desire to improve the systems performance through the incorporation of additional task dependencies. Yassine et al. [40] developed a dynamic model to determine the optimal timing of information exchanges for distributed and collaborative environments based on the estimation of the information (uncertainty) and the stage the decision process is in (cost of incorporating the information). The model revealed there is a limit to when information should be incorporated into the PD process. Once, the limit is crossed further incorporation of new information can have significant impact on time and cost. Wang and Yan [41] developed an optimization model to determine optimal concurrency between one upstream product design activity and multiple downstream process design activities based on cost for design revision workloads. Their model considered the product design activity and all the related process design activities.

4.2 Complexity

Smith and Eppinger [16] developed a deterministic work transformation matrix (WTM) model to understand what group of design tasks are strongly coupled such that working on any of them creates significant work which are the controlling features of design iteration. The WTM is an extension to the DSM where the off-diagonal elements represent the strength of dependence between tasks and the diagonal elements represents the time to complete each tasks within one iteration. The eigenvalues and eigenvectors are used to determine the design modes that need the most attention where the largest eigenvalue is the slowest design mode. Once the controlling features are identified a strategy should be developed to accelerate the iterative process. The process can be accelerated by creating few iterations or by speeding up the iteration. Few iterations can be achieved by standardizing interfaces, decision analytics where performance metrics are integrated and evaluated at the same time, and/or effective coordination and communication channels between project members. Browning and Eppinger [42] developed a stochastic model that uses a discrete event simulation to understand the impacts of process architecture on process cost, duration, and risk.

Sosa et al. [20] developed a network model to evaluate how organizational and systems boundaries, design interface strength, indirect interactions, and system modularity impact the alignment of design interfaces and team interaction. The network analysis used information from an alignment matrix that was developed by overlapping a team interaction matrix (process DSM) and a design interface matrix (product architecture DSM). Hoedemaker [43] developed a mathematical model to determine the optimal number of modules (smaller problem sets tackled by distinct teams) to minimize the expected project completion time. The model included three factors that were assumed to influence development lead time: 1) communication, 2) rework, and 3) integration test time. The authors used a simple "base" model where they increased the complexity of the model by adding more modules to the model. They were able to prove that as the complexity of the system increases there is a limit to the amount of concurrency a project can have. Yassine and Braha [44] developed a framework using DSM to address four fundamental problems: 1) iteration problem, 2) overlapping problem, 3) decomposition and integration problem, and 4) convergence problem. The authors identified three sources of churn: 1) hidden, ignored, or forgotten interdependencies, 2) improper planning of feedback flows that drive the process to be unstable, and 3) feedback delays, which are the main reason why development problems, believed to be solved, tend to reopen at later stages of development [44].

5. **Future Research Areas**

The literature review cited set-based design as a methodology that could be used in situations when tasks are sensitive to information exchanges where new information causes the task to be reworked [5, 10, 18, 19] . In set-based design, engineers develop a set of design alternatives in parallel , as the design progresses, they gradually narrow their prospective set of alternatives based on additional information until they converge to a final solution [45]. This approach is believed to be robust to information changes such as design changes and customer needs changes. The literature did not provide any set-based design application for reducing rework or data to substantiate the reduction in rework. Al-Ashaab et al. [46] identified there is a lack of a clear and structured SBCE model. This indicates a need for future research to develop and investigate a set-based design framework for DOD systems that can reduce rework and accelerate information evolution.

Information evolution can be improved by accelerating learning [5]. There is a variety of ways design teams can learn based on engineering reasoning. Summers [47] defined three forms of reasoning: deductive, inductive, and abductive. Deductive reasoning is the main form of reasoning that engineers use. Tahera et al. [28] demonstrated improvement in information precision by overlapping test activities during early PD where they defined the need for trading off "virtual prototyping" and physical tests. In order for this to be a viable option for deductive reasoning, the virtual prototypes need to be validated and calibrated using physical tests which can be costly and time consuming. Future research using techniques such as optimal learning should identify efficient ways to verify and validate virtual prototypes using multi-fidelity physical prototypes [48]. There is also a research need to reduce the run time of executing these virtual prototypes which may require the need for multi-fidelity modeling [49], reduced order modeling [50], or enhanced parallel processing and high performance cluster computing. There is a need for future research in knowledge based engineering [51] to accelerate learning using digital engineering [52], multidisciplinary design optimization [53] to improve communication between IPTs and improve the knowledge curve for deductive reasoning, and data mining methods [54] for knowledge discovery.

Another future research need is accelerating learning by reusing data. This can be enabled through an ontology integration framework [55]. Ontology frameworks can be developed for systems, manufacturing and testing to include rework ontologies. Li et al. [56] developed an ontology based product design (OBPD) framework for manufacturability verification and knowledge reuse using Web Ontology Language (OWL). An ontology framework can also be used for inductive and abductive reasoning. This can be accomplished through an inductive learning semantic web framework using supervised machine learning techniques [57]. An ontology learning framework can be used to perform abductive reasoning [58]. Abductive reasoning can reduce the ambiguity of the design by providing potential design variables and relationships that influence system performance.

6. Concluding remarks

Rework has a significant impact to project cost and schedule. Most design efforts require significant rework, between 30 and 65 percent of the total design hours on a project [6]. In order to develop a systems engineering framework during the Technology Maturation and Risk Reduction (TMRR) phase of the DOD life cycle the reasons for rework must be addressed. Much of the research on rework has been on information exchange and organizational structure. Future research in systems engineering to develop frameworks is required in order to: 1) mitigate the impact of information uncertainty and instability, 2) accelerate information evolution, and 3) reuse knowledge for engineering reasoning.

7. References

[1] P. Law, "114-92,“," *The National Defense Authorization Act for Fiscal Year,* 2016.

[2] G. A. Office, "Defense Acquisitions: Addressing Incentives is Key to Further Reform Efforts," (GAO Publication No. GAO-14-563T). Washington, D.C.: U.S. Government Printing Office, 2014.

[3] G. A. Office, "Trends in Nunn-McCurdy Breaches and Tools to Manage Weapon Systems Acquisition Costs," (GAO Publication No. GAO-10-106). Washington, D.C.: U.S. Government Printing Office, 2011.

[4] S. M. Osborne, "Product development cycle time characterization through modeling of process iteration," Massachusetts Institute of Technology, 1993.

[5] B. M. Kennedy, D. K. Sobek, II, and M. N. Kennedy, "Reducing rework by applying set-based practices early in the systems engineering process," (in English), *Systems Engineering,* vol. 17, no. 3, pp. 278-96, 2014.

[6] K. Reichelt and J. Lyneis, "The dynamics of project performance: benchmarking the drivers of cost and schedule overrun," *European management journal,* vol. 17, no. 2, pp. 135-150, 1999.

[7] T. J. Y. James, K. Otto, and K. Wood, "A Comparison of Design Decisions made Early and Late in Development," in *ICED17: 21st International Conference on Engineering Design*, University of British Columbia, Vancouver, Canada, 2017.

[8] K. T. Ulrich and S. A. Pearson, "Does product design really determine 80% of manufacturing cost?," 1993.

[9] D. o. Defense, "DoD instruction 5000.02: Operation of the defense acquisition system," ed: Author Washington, DC, 2015.

[10] R. Costa and D. K. Sobek, "Iteration in engineering design: inherent and unavoidable or product of choices made?," in *ASME 2003 International design engineering technical conferences and Computers and information in engineering conference*, 2003, pp. 669-674: American Society of Mechanical Engineers.

[11] N. P. Repenning, "Understanding fire fighting in new product development⋆," *Journal of Product Innovation Management: AN INTERNATIONAL PUBLICATION OF THE PRODUCT DEVELOPMENT & MANAGEMENT ASSOCIATION,* vol. 18, no. 5, pp. 285-300, 2001.

[12] D. C. Wynn and C. M. Eckert, "Perspectives on iteration in design and development," *Research in Engineering Design,* vol. 28, no. 2, pp. 153-184, 2017.

[13] V. L. Mitchell and B. R. Nault, "Cooperative planning, uncertainty, and managerial control in concurrent design," *Management Science,* vol. 53, no. 3, pp. 375-389, 2007.

[14] P. Arundachawat, R. Roy, A. Al-Ashaab, and E. Shehab, "Design rework prediction in concurrent design environment: current trends and future research directions," in *Proceedings of the 19th CIRP Design Conference–Competitive Design*, 2009: Cranfield University Press.

[15] T. Taylor and D. N. Ford, "Tipping point failure and robustness in single development projects," *System Dynamics Review: The Journal of the System Dynamics Society,* vol. 22, no. 1, pp. 51-71, 2006.

[16] R. P. Smith and S. D. Eppinger, "Identifying controlling features of engineering design iteration," *Management science,* vol. 43, no. 3, pp. 276-293, 1997.

[17] S. Schrader, W. M. Riggs, and R. P. Smith, "Choice over uncertainty and ambiguity in technical problem solving," 1993.

[18] C. Terwiesch, C. H. Loch, and A. D. Meyer, "Exchanging preliminary information in concurrent engineering: Alternative coordination strategies," *Organization Science,* vol. 13, no. 4, pp. 402-419, 2002.

[19] S. M. Bogus, K. R. Molenaar, and J. E. Diekmann, "Strategies for overlapping dependent design activities," *Construction Management and Economics,* vol. 24, no. 8, pp. 829-837, 2006.

[20] M. E. Sosa, S. D. Eppinger, and C. M. Rowles, "The misalignment of product architecture and organizational structure in complex product development," *Management science,* vol. 50, no. 12, pp. 1674-1689, 2004.

[21] A. Y. Ha and E. L. Porteus, "Optimal timing of reviews in concurrent design for manufacturability," *Management Science,* vol. 41, no. 9, pp. 1431-1447, 1995.

[22] V. Krishnan, S. D. Eppinger, and D. E. Whitney, "A model-based framework to overlap product development activities," *Management science,* vol. 43, no. 4, pp. 437-451, 1997.

[23] C. H. Loch and C. Terwiesch, "Communication and uncertainty in concurrent engineering," *Management Science,* vol. 44, no. 8, pp. 1032-1048, 1998.

[24] A. K. Chakravarty, "Overlapping design and build cycles in product development," *European Journal of Operational Research,* vol. 134, no.

2, pp. 392-424, 2001.
[25] T. A. Roemer and R. Ahmadi, "Concurrent crashing and overlapping in product development," *Operations research,* vol. 52, no. 4, pp. 606-622, 2004.
[26] P. Arundacahawat, R. Roy, and A. Al-Ashaab, "An analogy based estimation framework for design rework efforts," *Journal of Intelligent Manufacturing,* vol. 24, no. 3, pp. 625-639, 2013.
[27] S. Thomke and D. E. Bell, "Sequential testing in product development," *Management Science,* vol. 47, no. 2, pp. 308-323, 2001.
[28] K. Tahera, C. Earl, and C. Eckert, "A method for improving overlapping of testing and design," *IEEE Transactions on Engineering Management,* vol. 64, no. 2, pp. 179-192, 2017.
[29] S.-H. Cho and S. D. Eppinger, "PRODUCT DEVELOPMENT PROCESS MODELING USING ADVANCED SIMULATION," in *ASME 2001 Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2001, pp. 1-10: American Society of Mechanical Engineers, September 9-12.
[30] R. P. Smith and S. D. Eppinger, "A predictive model of sequential iteration in engineering design," *Management Science,* vol. 43, no. 8, pp. 1104-1120, 1997.
[31] Q. Yang, T. Yao, T. Lu, and B. Zhang, "An Overlapping-Based Design Structure Matrix for Measuring Interaction Strength and Clustering Analysis in Product Development Project," *IEEE Trans. Engineering Management,* vol. 61, no. 1, pp. 159-170, 2014.
[32] A. Yassine, N. Joglekar, D. Braha, S. Eppinger, and D. Whitney, "Information hiding in product development: the design churn effect," *Research in Engineering Design,* vol. 14, pp. 145-161, 2003.
[33] A. A. Yassine, K. R. Chelst, and D. R. Falkenburg, "A decision analytic framework for evaluating concurrent engineering," *IEEE Transactions on Engineering Management,* vol. 46, no. 2, pp. 144-157, 1999.
[34] H.-B. Jun, H.-S. Ahn, and H.-W. Suh, "On identifying and estimating the cycle time of product development process," *IEEE Transactions on Engineering Management,* vol. 52, no. 3, pp. 336-349, 2005.
[35] R. G. Nelson, A. Azaron, and S. Aref, "The use of a GERT based method to model concurrent product development processes," *European Journal of Operational Research,* vol. 250, no. 2, pp. 566-578, 2016.
[36] D. N. Ford and J. D. Sterman, "The Liar's Club: concealing rework in concurrent development," *Concurrent Engineering,* vol. 11, no. 3, pp. 211-219, 2003.
[37] Q. Yang, T. Lu, T. Yao, and B. Zhang, "The impact of uncertainty and ambiguity related to iteration and overlapping on schedule of product development projects," *International Journal of Project Management,* vol. 32, no. 5, pp. 827-837, 2014.
[38] V. Lévárdy and T. R. Browning, "An adaptive process model to support product development project management," *IEEE Transactions on Engineering Management,* vol. 56, no. 4, pp. 600-620, 2009.
[39] D. Braha and Y. Bar-Yam, "The statistical mechanics of complex product development: Empirical and analytical results," *Management Science,* vol. 53, no. 7, pp. 1127-1145, 2007.
[40] A. A. Yassine, R. S. Sreenivas, and J. Zhu, "Managing the exchange of information in product development," *European Journal of Operational Research,* vol. 184, no. 1, pp. 311-326, 2008.
[41] Z. Wang and H.-S. Yan, "Optimizing the concurrency for a group of design activities," *IEEE Transactions on Engineering Management,* vol. 52, no. 1, pp. 102-118, 2005.
[42] T. R. Browning and S. D. Eppinger, "Modeling impacts of process architecture on cost and schedule risk in product development," *IEEE transactions on engineering management,* vol. 49, no. 4, pp. 428-442, 2002.
[43] G. M. Hoedemaker, J. D. Blackburn, and L. N. Van Wassenhove, "Limits to concurrency," *Decision Sciences,* vol. 30, no. 1, pp. 1-18, 1999.
[44] A. Yassine and D. Braha, "Complex concurrent engineering and the design structure matrix method," *Concurrent Engineering,* vol. 11, no. 3, pp. 165-176, 2003.
[45] D. K. Sobek II, A. C. Ward, and J. K. Liker, "Toyota's principles of set-based concurrent engineering," *MIT Sloan Management Review,* vol. 40, no. 2, p. 67, 1999.
[46] A. Al-Ashaab *et al.*, "The transformation of product development process into lean environment using set-based concurrent engineering: A case study from an aerospace industry," *Concurrent Engineering,* vol. 21, no. 4, pp. 268-285, 2013.
[47] J. D. Summers, "Reasoning in engineering design," in *ASME 2005 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, 2005, pp. 329-340: American Society of Mechanical Engineers.
[48] P. Frazier, W. B. Powell, and H. P. Simão, "Simulation model calibration with correlated knowledge-gradients," in *Simulation Conference (WSC), Proceedings of the 2009 Winter*, 2009, pp. 339-351: IEEE.
[49] B. Poethke, S. Völker, and K. Vogeler, "Aerodynamic Optimization of Turbine Airfoils Using Multi-fidelity Surrogate Models," in *International Conference on Engineering Optimization*, 2018, pp. 556-568: Springer.
[50] J. Oliver, M. Caicedo, A. E. Huespe, J. Hernández, and E. Roubin, "Reduced order modeling strategies for computational multiscale fracture," *Computer Methods in Applied Mechanics and Engineering,* vol. 313, pp. 560-595, 2017.
[51] W. J. Verhagen, P. Bermell-Garcia, R. E. van Dijk, and R. Curran, "A critical review of Knowledge-Based Engineering: An identification of research challenges," *Advanced Engineering Informatics,* vol. 26, no. 1, pp. 5-15, 2012.
[52] T. D. West and M. Blackburn, "Demonstrated Benefits of a Nascent Digital Twin," *INSIGHT,* vol. 21, no. 1, pp. 43-47, 2018.
[53] J. R. Martins and A. B. Lambe, "Multidisciplinary design optimization: a survey of architectures," *AIAA journal,* vol. 51, no. 9, pp. 2049-2075, 2013.
[54] S. Bandaru, A. H. Ng, and K. Deb, "Data mining methods for knowledge discovery in multi-objective optimization: Part A-Survey," *Expert Systems with Applications,* vol. 70, pp. 139-159, 2017.
[55] N. F. Noy, "Semantic integration: a survey of ontology-based approaches," *ACM Sigmod Record,* vol. 33, no. 4, pp. 65-70, 2004.
[56] Z. Li, X. Zhou, W. Wang, G. Huang, Z. Tian, and S. Huang, "An ontology-based product design framework for manufacturability verification and knowledge reuse," *The International Journal of Advanced Manufacturing Technology,* pp. 1-15, 2018.
[57] L. Bühmann, J. Lehmann, and P. Westphal, "DL-Learner—A framework for inductive learning on the Semantic Web," *Web Semantics: Science, Services and Agents on the World Wide Web,* vol. 39, pp. 15-24, 2016.
[58] M. Gavanelli, E. Lamma, F. Riguzzi, E. Bellodi, R. Zese, and G. Cota, "Reasoning on Datalog±Ontologies with Abductive Logic Programming," *Fundamenta Informaticae,* vol. 159, no. 1-2, pp. 65-93, 2018.