
This is an electronic reprint of the original article.
This reprint may differ from the original in pagination and typographic detail.

Tan, James J.Y.; Otto, Kevin; Wood, Kristin L.

Relative impact of early versus late design decisions in systems development

Published in:
Design Science

DOI:
[10.1017/dsj.2017.13](https://doi.org/10.1017/dsj.2017.13)

Published: 01/08/2017

Document Version
Publisher's PDF, also known as Version of record

Please cite the original version:
Tan, J. J. Y., Otto, K. N., & Wood, K. L. (2017). Relative impact of early versus late design decisions in systems development. Design Science, 3, [e12]. DOI: 10.1017/dsj.2017.13

This material is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of the repository collections is not permitted, except that material may be duplicated by you for your research use or educational purposes in electronic or print form. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone who is not an authorised user.

Relative impact of early versus late design decisions in systems development

James J. Y. Tan¹, Kevin N. Otto² and Kristin L. Wood¹

¹ SUTD-MIT International Design Centre, Singapore University of Technology and Design, Singapore

² Department of Mechanical Engineering, Aalto University, Finland

Abstract

To better understand the impact of early versus late design decisions, a study was undertaken on the root causes of missed requirements in new product development and their impact on development cost through rework activities. The context is the industrial development of unmanned aerial vehicles. The aim is to understand the occurrence rate of missed requirements, their root causes, and their relative impact. A quantitative approach of counting requirements changes and using engineering documentation enabled traceability from observation back to root cause origin. The development process was partitioned into sequential program segments, to categorize activities to before and after concept and design freeze. We found that there was a significant difference in the rate of design defects arising before and after concept freeze; and found there was a significantly higher number of corrective activities required for design defects arising earlier before concept freeze. The revision rate of concept phase decisions was over 50%, and the rework multiplier if detected late was over 10X. In combination, design decisions made before design freeze accounted for 86% of the total expected program cost, and 34% was determined before concept freeze. These results quantify and support the anecdotal 80–20 impact rule for design decisions.

Key words: requirements, root cause analysis, conceptual design, empirical study

Received 10 April 2017
Revised 19 June 2017
Accepted 22 June 2017

Corresponding author
J. J. Y. Tan
esengcorp@gmail.com

Published by Cambridge
University Press
© The Author(s) 2017
Distributed as Open Access under
a CC-BY 4.0 license
(<http://creativecommons.org/licenses/by/4.0/>)

Des. Sci., vol. 3, e12
journals.cambridge.org/dsj
DOI: 10.1017/dsj.2017.13

the **Design Society**
a worldwide community

 **CAMBRIDGE**
UNIVERSITY PRESS

1. Introduction

1.1. Design decisions, design defects and missed requirements

The design of complex systems involves activities such as sizing subsystems and components, selecting configurations, and determining possible design solutions and implementations. Throughout, the process requires decisions (Hazelrigg 1998) that determine what needs to be analyzed, considered and determined. We call these *design decisions*. Design decisions define requirements, create concepts, form specifications from requirements, shape the final configuration and physical form, and define production and testing (Saad *et al.* 2013) and lifecycle/operating principles. A design research question of interest is the relative importance of design decisions made early versus late in the design process (Ulrich & Scott 1998; Ullman 2015).

One way to study relative decision importance of is to consider the relative impact of incorrect decisions made. An incorrect decision is observable in

terms of the corrective actions required to fix the problem. We define a *design defect* as a poor design decision made creating an attribute not desired in the design, which must either be fixed (decision undone) or accepted as a permanent non-compliance denigrating the produced design. Design defects come in form at different levels of abstraction, from high level missed requirement (e.g., insufficient flying distance range), to engineering sizing deficiencies (e.g., insufficient battery size) or unspecified design features (e.g., insufficient fillet radius) and inherent requirements that are not explicitly stated.

An engineering design process can be described in a requirements compliance framework; in this view, the process is a sequence of defining technical requirements and in turn developing and engineering concepts to meet the requirements, and then finally verifying and validating the design meets the requirements (VDI 1993; Clausing 1994; Otto & Kristin 2001). In this context, design decisions, when not met, can be tracked for impact based on traditional effort tracking against requirements within program management tools.

Therefore, we define *missed requirements* as design defects on managed requirements. Missed requirements are the important design defects with such impact that they require program management tracking during development. We propose to examine the importance of design decisions by studying managed requirements in industrial development programs and compare their relative timing and impact of missed requirements. Note that these requirements based approach includes the large early design decisions such as the decision on the concept selected. If the concept selected decision was not corrected and must be re-decided, it was due to an inadequacy in missed information needed early at the point when the concept was selected.

For more complex systems, this includes defining a network of requirements and managing their multiple interactions (Clarkson & Eckert 2010; Eckert, Isaksson & Earl 2012; Ladyman, Lambert & Wiesner 2013). *Requirements* are statements that describe targeted functionality, e.g., minimum range, minimum payload, etc. (Otto & Kristin 2001). *Constraints* on the other hand, are the physical limits based upon the chosen design implementation, e.g., maximum propeller tip speed when using propellers, or stress limits of chosen materials (Otto & Kristin 2001). When developing new systems, it can be unclear how these multiple requirements and constraints impact one another. These interactions may not be uncovered until late in development during testing (Clarkson & Eckert 2010), when it is realized the requirements and constraints are inconsistent. The consequence can be iterations to resolve these inconsistencies and longer development time (Keller *et al.* 2005; Clarkson & Eckert 2010).

The aerospace and defense industry in particular, has a history of late deliveries, cost overruns and missed requirements (Obaid *et al.* 2005; Mark *et al.* 2006; Bernard, Eric & Archag 2015). On average, the typical procurement process for a new defensive system takes nine (9) years with an average delay of 21 months (Charette 2008). As such, there is constant schedule pressure to deliver the product (system) on time and within cost. However, this time pressure is not necessarily the root cause of late deliveries; Chong *et al.* (2011), showed that time pressure can have a positive or a negative impact and is highly dependent on team level dynamics.

On the other hand, complex systems can have many nested interactions and hierarchical relations between requirements (Crawley *et al.* 2004). These

interactions and relations can introduce development time-lagged dependencies amongst requirements, and early concept phase decisions are typically made assuming other requirements can be met. The outcome of these assumptions will not be known until later in the program. This uncertainty can result in missed requirements, leading to in-service performance deficiencies (Keller *et al.* 2005). Late deliveries, peculiar behaviors in certain operating conditions, and outright failures in performance are some of the explicit symptoms of missed requirements (Bernard *et al.* 2015).

While it is known that complex programs exhibit cost and schedule overruns, the relationship amongst inconsistent requirements and when they are defined is less established. Requirements are often noted to have been inconsistent, but it is less reported whether these are inconsistencies between early-defined system-level requirements, inconsistencies between later-defined component requirements, or inconsistencies between early-defined system requirements and later-defined requirements, as an evolving system. These statements highlight the importance of this work as it provides clarification of how much more important early design decisions are by providing justifications for added investments in early design processes, methods and tools.

1.2. Related research

A well-established 80–20 rule has been noted where product design accounts for 80% of the manufacturing cost, and has been studied extensively (Ulrich & Scott 1998; Otto & Kristin 2001). The eventual cost committal accounts for manufacturing cost including material, labor, sourced components and other production overheads. This also includes the testing and certification costs. It generally does not include life cycle costs such as maintenance and repair. The 80–20 rule is also supported by Ullman (2015) who studied copier machines and argued that as high as 75% manufacturing cost savings is committed during the conceptual design phase, and further that design freedom decreases as the knowledge of the design problem increases. To a lesser extent, industrial standards such as S-3000L (Aerospace Industries Association 2009) use a similar premise as the basis to calculate opportunity for life cycle cost savings, although not explicitly stated.

Specific case studies demonstrating the 80–20 rule are less reported. In a product archaeology research study of coffee makers, Ulrich & Scott (1993, 1998) confirmed the 80–20 ratio as the design to manufacturing cost contributors, and found it could be made lower by adopting design for manufacturing and assembly methods (Geoffrey 1994). Standards on the cost impact of design for manufacture also provide useful guidelines to reduce developmental cycles and discuss the importance of early design decisions (Aerospace Industries Association 2009). In a study of space systems development costs, NASA reports defects not found until operations carry 50 times more cost than defects found in the early concept phase (NASA 2016). In a meta-study of multiple defense systems, the RAND corporation found the cost of changing a requirement escalate over time in a program (Obaid *et al.* 2005; Mark *et al.* 2006; Bernard *et al.* 2015) highlights that early decisions cost more. Software development process research also reports a similar phenomenon as those in electro-mechanical systems. The National Institute of Standards observed that the longer the time of the discovery of a bug from its creation, the greater the cost impact (Tassey 2002). Boehm reports bugs

found in operation carry 10 times more cost than defects found early in the coding phase (Boehm & Papaccio 1988; Boehm & Basili 2001). Hamada (1996) reported that with work done at Ricoh, the cost of fixing a design defect was \$35 in design compared to \$690,000 in field service. All these studies support that design defects discovered late cost more to fix, due to the costs of changing fixed assets such as tooling and repairing multiple units in operation.

Almefelt, Berglund & Nilsson (2006) found use of requirements management tools can significantly reduce requirements' errors. The likelihood of requirements compliance can be improved by using formalized processes like the VDI 2211 process or V-model for development to manage requirement changes (VDI 1993). The United States Department of Defense (US DoD) implemented use of *Engineering Change Requests* (ECRs) and configuration management protocols (Department of Defense 2001) to ensure the changes to design after Critical Design Review (CDR) do not affect the intended function of engineered defense systems. Capers & Lipton (1993) reported that the choice of integration and testing processes can greatly impact the performance of a system. This was illustrated in their reported Hubble Space Telescope case study, where failure to adequately test requirements caused expensive repairs.

While such requirements management systems help reduce design errors and help in their mitigation, understanding the root causes of missed requirements and their impact is less understood. De Weck, Olivier & Eckert (2007) studied the impact of various uncertainties on requirements management. Uncertainties can range from being stochastic in nature, such as fabrication or field-use variations, and those that are epistemic in nature, such as the unknown behavior response in the use of immature technologies. They note the lack of knowledge on the contextual background of the design, corporation, usage, market and politico-socio elements can lead to requirements being missed.

Overall, there are multitude interpretations and discussions of the 80–20 rule in the literature. This work seeks to clarify where in the early design stage that 80 percent cost committal is valid, from concept, preliminary, detailed or the verification design phases. Specifically, in terms of monetary resources, time or developmental work activities, we seek to understand the cost impact of decision from the various development process phases.

1.3. Examples of design decisions and consequences in the literature

The impact of design decisions has had a number of negative consequences, including performance failures, cost overruns and late delivery problems. As background to the industrial case study work presented here, we first also studied the recent literature for examples of particular failures due to design defects in complex systems. A sample of 10 case studies of design defects publicized in the literature are shown in Table 1. While these examples may have had multiple problems and incurred very high program costs, here we isolated the cost impact of an individual defect and studied that individual design defect's cost and delay impact alone. We examined the cost and schedule overrun and scrutinized for the published issue and self-stated primary root cause. For example, A380 the wiring design defect was discovered late in aircraft final assembly. This led to issues such as liquidity damage from delivery delays (2 years), recertification

Table 1. Examples of complex systems design defects

Example	Issue	Root Cause	Design Freeze Attribution	Reference
Millennium Bridge Vibration	Pedestrian vibrations leading to resonance	Lateral vibration mode not considered in the design.	Before	BBC, 2002
Chinook Mk 3 upgrade	Avionics uncertifiable	Inability to verify the software operating parameters with the full set of aircraft mission parameters, Requirements unable to be cascaded	Before	Burr, 2008
A400M Integration	Insufficient range and power	System requirement trade-offs not completed until after design work began	Before	Brothers, 2009
A380 Wiring	Wiring connection post assembly	Different versions of computer aided design tools used and assumed use of incompatible connector standards. Design standards not agreed upon.	Before	Wong, 2006
F-22 Integration	Delivery delays	Integration bugs and failures; concept and architecture was not adequately specified.	Before	Obaid <i>et al.</i> , 2005
SH-2G(A) Avionics	Unable to operate in low light conditions	Did not consider the compatibility between new generation avionics and older airframe.	Before	Allard, 2005
S-80 Submarine surfacing	Overweight design, unable to surface after submerging	Calculation error in weight allocation.	After	Govan, 2013
Citi Corp Center Wind	Building could not withstand quarterly winds	Initial calculations did not consider unique 'no corners' building design.	After	Vadaro, 2013
Boeing 787 batteries	Aircraft fire, thermal runaway in batteries	Failure to consider additional requirements of new lithium ion technology.	After	Williard <i>et al.</i> , 2013
Type 45 Destroyer electric power	Power failure, due to deficiency in cooling system	Intercooler undersized and failed in under-analyzed conditions.	After	Batchelor, 2016

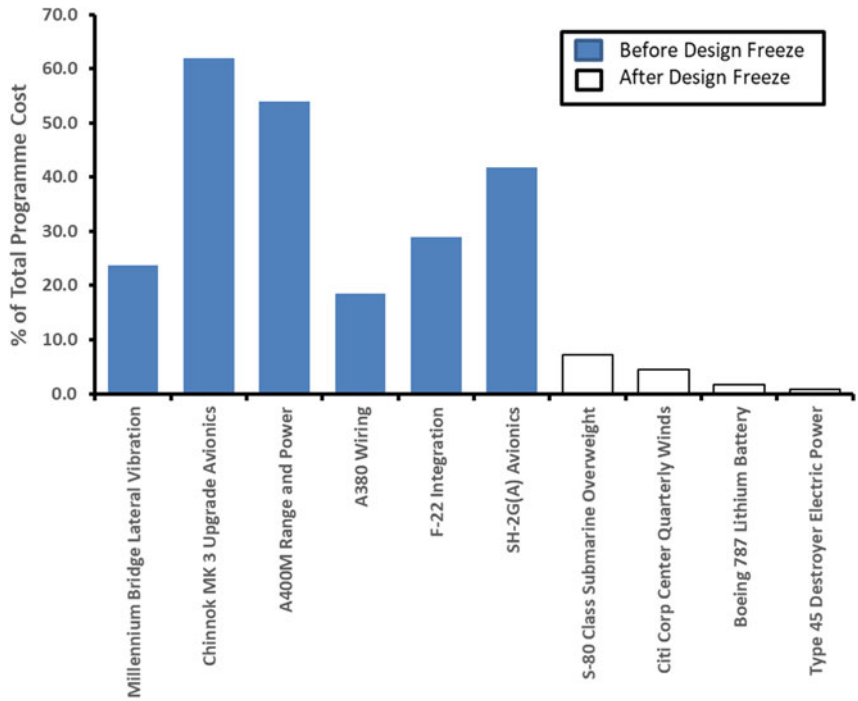


Figure 1. Cost impact of missed requirements.

after compatibility was established and other production rework cost. This single design defect resulted in 18.5% of the total program cost. Table 1 lists examples, issue and causes, and attributes whether the root cause is before (early design) or after (late design) design freeze. Figures 1 and 2 summarize the impact on cost and schedule overruns of the one design defect alone. These publicized design defects all have the same characteristic of not being detected until late. Further, from Figures 1 and 2, it is also observable that the earlier the design defect was attributed, the higher the cost of rework.

In this sample, all of the requirements changes made occurred late in their programs. Beyond the higher cost impact of changing a requirement late in a program, these results further show the cost of changing a requirement based on when the requirement was defined. From the figures, we can see that there is a clear indication that early/phase missed requirements have a distinctively higher cost than late/phase missed requirements.

The publicized design defect examples reviewed in Table 1 included analysis on their missed requirements. Analyzing these findings, in Table 2 we summarize the list of missed-requirement root causes, as self-reported within the works. The root causes include inadequate trade-off analysis of requirements, inadequate reviews, budget and time pressure, and immature technology or industrial base. Yet, the relative occurrence and impact of these root causes is less reported.

Given this, we next present a study of the root causes of missed requirements in the development process and their impact on development cost through rework activities. This is a longitudinal study over several years at the industrial development at a large aerospace firm responsible for developing and delivering UAVs.

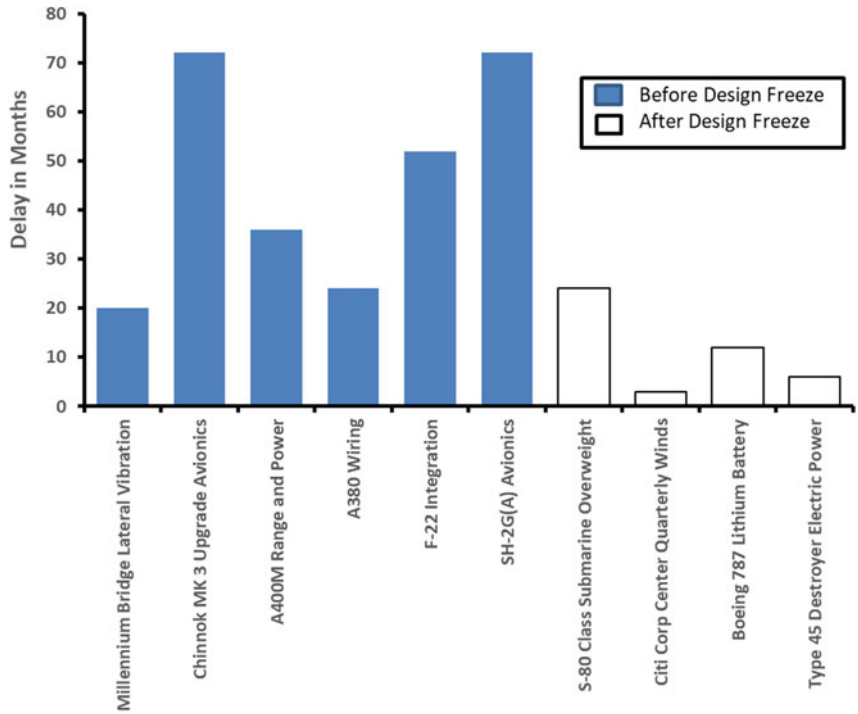


Figure 2. Schedule impact of missed requirements.

Table 2. Self-identified root causes of missed requirements

No.	Cause of Missed Requirement(s)	Reference
1	Inadequate trade-off targets	Simon, Charette, Bernard, Chong, Almefelt, Ulrich, Keller, Cooper, Aerospace Industries Association, Otto, Mark, Geoffrey, Camburn
2	Inadequate gate reviews	Obaid, De Weck, Ball, Agarwal, Pandey, Perez, Capers, Tasse
3	Excessive budget pressure and inadequate resources	Simon, Charette, Bernard, Chong, Almefelt, Ulrich, Hales, Aerospace Industries Association, Otto, Mark, Ulrich, Geoffrey, Capers
4	Excessive time pressure and inadequate design activity	Obaid, De Weck, Ball, Agarwal, Pandey, Perez, Chong, Cooper, Hales, Williard
5	Immature technology and inadequate industrial base	Obaid, De Weck, Ball, Agarwal, Pandey, Perez, Hales, Williard

1.4. Objective

An empirical perspective was applied to quantify and study why requirements are missed in the development of complex systems. While the results shown in Table 1 remain anecdotal evidence from publicized cases, it also creates a basis of hypotheses for root causes. We seek to understand if missed requirements occur more often or not in the early phase of design, as separated before and after concept freeze.

Hypothesis 1: Design decisions made earlier in development carry a higher probability of being incorrect than decisions made later in development.

Given that a missed requirement occurs, the cost impact can be different between early and late phase requirement misses. We seek to understand if early phase decisions cost more or less to fix than late phase decisions.

Hypothesis 2: There is a distinctively higher cost to rectify design requirements originating early in development than later.

From the literature, Table 2 lists several possible root causes of missed requirements. A third research question concerns the root cause of missed requirements.

Hypothesis 3: The most frequent root cause of missed requirements in complex systems development is the difficulty in trading off requirements.

The three research hypotheses will be tested by studying the available project documentation in the corporate setting, including the requirements documentation found in design reports and gate reviews documents, their time history, the *Engineering Change Request* (ECR) logs, the *Failure Reporting System* (FRACAS), and finally with Kaizen event workshops with the engineers discussing the problems and tracing their causes. The starting point of the analysis is an individual design defect as either an ECR, a FRACAS event, or a changed requirement in the gate reviews. Each event's relevant requirement was traced through the requirements documentation for the associated root cause inconsistency, and when those requirements were set. The cost impact can be determined through the chain of activities required to rework the design.

Beyond this, interviews were also conducted with the stakeholders to enhance understanding of the origin of the missed requirements and mitigation strategies. Finally, we will provide our recommendations and conclusions for theoretical contributions and design practice.

2. Study and analysis approach

2.1. Study background: design process at an industrial defense corporation

2.1.1. Corporate design process

With the background information and motivation, we now study a global aerospace company with turnkey expertise in aerospace systems development and modifications. The company's design process follows a stage gate design process (Cooper 1990) as shown in Figure 3, incorporating system engineering processes

Programme Segment		BCF	ACF	ADF	APGA
Development Phase		<i>Conceptual Design</i>	<i>Preliminary Design</i> <i>Detailed Design</i>	<i>Build / Test Prototype</i>	<i>LRIP Initial Deployment</i>
Design Review			SRR PDR	CDR	PGA
Program Task	<ul style="list-style-type: none"> - Data Gathering - Requirements Analysis - Functional Analysis 	<ul style="list-style-type: none"> - Modelling / Simulation - Trade-off requirements - Requirements Verification - Interface Management - Data Management - Sizing - Architecture Selection - Interface Selection - Process Selection - Work Breakdown Structure - Configuration Management - System Technical Review - Metrics Management - Risk Management 	<ul style="list-style-type: none"> - Embodiment - Test Plan - Reliability / Hazard Analysis - Modelling / Simulation - Trade – Off - Requirements Verification - Interface Management - Data Management - Work Breakdown Structure - Configuration Management - Technical Review - Metrics Management - Risk Management 	<ul style="list-style-type: none"> - Design Implementation - System Integration Test - Ground Test - Flight Test 	<ul style="list-style-type: none"> - Manufacture - System Inspection - Product Support

Figure 3. Overview of the corporate case study product development process, complete with segments, phases, gate reviews, and underlying program tasks (containing activities).

(VDI 1993; Department of Defense 2001). Design reviews and decision points are consistent with the DOD 5000-2P (Department of Defense 2001, 2015) systems development process, which is required of industrial defense companies. As an aerospace company, the design certification process conforms to the industrial standards as established by MIL-HDBK-516 (Department of Defense 2001) and STAGNAG-4671 (North Atlantic Treaty Organization 2009).

The corporate design process includes all necessary aircraft design program tasks as per industry practice (Anderson 2004), structured accordingly to DOD 5000-2P phases and complete with design technical reviews (Department of Defense 2001). Each developmental phase has its set of systems engineering and product development activities as summarized into program tasks in Figure 3. Activities are gated by design reviews in the following order; *systems requirements review* (SRR), *preliminary design review* (PDR), *critical design review* (CDR), *production go ahead* (PGA).

The description of the development process in Figure 3 further highlights the concept freeze decision made at the SRR gate review, so we can consider activities before and after this key milestone. To highlight this distinction, as shown in the top row of Figure 3 we introduce the concept of a *program segment*, where we divide our study scope into four ‘segments’: before the concept is frozen (*Before Concept Freeze*, BCF), after the concept is frozen (*After Concept Freeze*, ACF), after the design is frozen (*After Design Freeze*, ADF) and after the production go ahead (*After Production Go Ahead*, APGA). This segmentation divides the development activities at useful breakpoints from a research perspective. The activities before BCF are concept level activities; after BCF, the concept is frozen, and it is then embodied and designed. After ADF, all design work is deemed finished, the design is tested to ensure compliance, and the manufacturing process is developed. After APGA, the initial builds off the production line are expected to meet performance requirements.

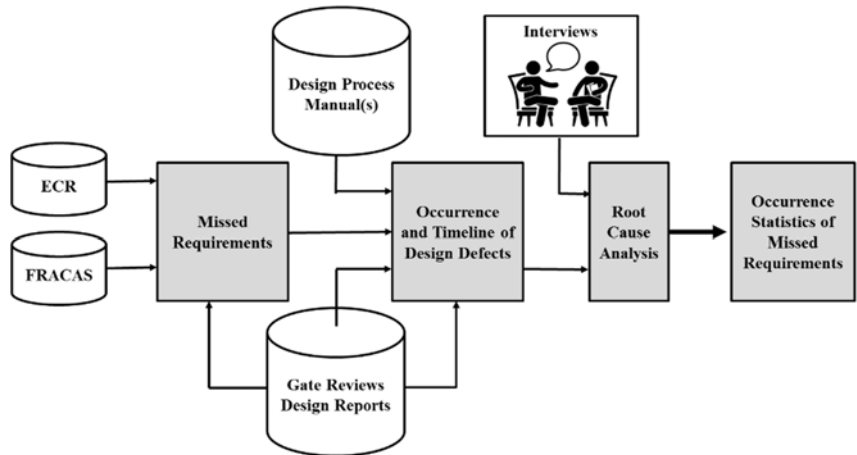


Figure 4. Data sources used in the study.

2.2. Data sources used in the study

For the basis of our analysis, two major programs were studied at the aerospace firm over a seven-year period. The effort represented an average of 20 person-years/annum of equivalent effort. These programs studied were the major UAV programs at the company.

2.2.1. Corporate product development documentation

Four independent sources of information within the company were used for this study, as shown in Figure 4. First, the corporate design process manual was used as a basis to structure the design processes (Figure 3). Then, information was sought on discussion of missed and changing requirements. This occurred in several document sets, depending on program phase.

For very late detection of missed requirements, the FRACAS contained the relevant information. For missed requirements detected after design freeze but before service, the ECR system contains the relevant information. For missed or evolving requirements before design freeze, gate review documentation provided information on the state of the requirements at each design review. Design reports were also used at the company for status of requirements up to design freeze.

For missed requirements detected after service, the FRACAS documented the prototype testing and in-service failures. An example of FRACAS is reporting of control departure of a payload gimbal in flight. Each design defect would have an outcome (or combination) of either drawing changes, recertification or becomes a permanent non-compliance.

For missed requirements detected after design freeze, the ECRs were the main documentation used and studied. The ECR logs the missed-requirement changes and associated necessary configuration changes to the design. An example of an ECR would be the design proposal of enlarging the tail cone area of an UAV to allow for greater airflow.

For missed requirements detected before design freeze, the considered missed requirements were those that failed a gate review. An example would be the design

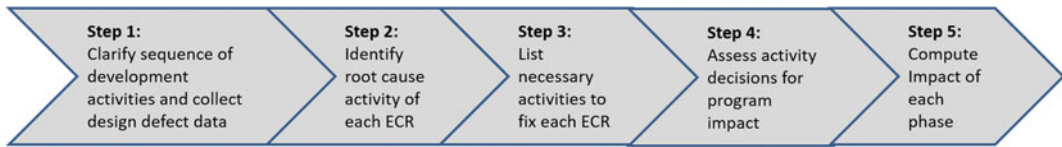


Figure 5. Analysis methodology.

proposal is likely to have excessive weight in the judgement of the gate review committee.

2.2.2. Interviews

After assimilating the timeline and occurrence rate of missed requirements, refined confirmation of the cause was developed through formalized discussions with engineers using Kaizen type workshops. The key objective of the interviews was to confirm and clarify the root cause aircraft systems and development process activities as identified from the documentation review. In this format, engineers were asked general why-type questions that invoked the perspectives from both managers and working engineers. This added depth to the identification of root cause systems and originating activities.

The interviews were carried out with the eight engineers and two program managers that executed the development programs. Each of the interviewees has more than three years of experience working in aerospace development programs and has deep expertise in their respective fields. The combination of both interviews and documentation provided the necessary empirical basis to understand missed requirements and design decision defects in the development of a new product.

2.3. Analysis approach

Four major steps were taken to determine the root causes of missed requirements, as shown in Figure 5.

2.3.1. Step 1: Clarify the sequence of development activities and collect design defect data

To define the development process as above for the two specific programs, the project Gantt charts of the corporate design process were studied, from segments, phases, tasks to activities. All task elements in the program schedule Gantt charts were abstracted to the level of the activities show in Figure 3.

Additionally, the missed-requirement design defects were listed from Gate review, ECR and FRACAS documentation. Each design defect was assigned an ID number and identified with the activity at which it was first detected.

2.3.2. Step 2: Identify root cause activity of each ECR

Next, a root cause analysis was conducted to trace systems engineering element from the onset of the design defect (through ECR and FRACAS) to the root systems engineering element. Each design defect was back-traced to the root cause through Kaizen event interviews. This was done starting with the design defect observation detection activity, and back-tracking through the chain of program

activities. At each such antecedent activity, the particular analysis, selection, test or other action was highlighted. Further back-tracing was done, until identifying the point at which the incorrect decision was made. This activity where the incorrect decision arose is the root cause activity. Such specific analysis of each activity that cannot be readily extracted explicitly from documentation and have to be accomplished manually.

2.3.3. Step 3: List necessary activities to fix each ECR

For each root cause requirement, the compliance-driven corporate development process standard work also defines the activities which must be reworked to ensure the requirement and its cascaded effects are completed. Correcting a design defect generally requires reworking of interlinked activities in the development process. The interlinked activities form a chain that begins from where the design defect was detected to the root cause activity.

2.3.4. Step 4: Assess activity decisions for program impact

Given steps 1 to 3, we next derive and assess the cost of rework to a program due to design defects. With this, we can assess the statistical significance of the root causes attributed to the four different program segments BCF, ACF, ADF, APGA described in Figure 3.

To derive the cost of rework, first consider the case of no rework. Assuming there are no design defects throughout the development process, the total cost of a program is simply the sum of individual costs,

$$E[\text{Total Cost}] = \sum_{\text{activity } j}^N W_j, \tag{1}$$

where W_j is the work cost required to complete activity j . If design defects occur, however, then rework effort is added to the cost. At any activity, though, it is uncertain if a design defect will happen and thereby incur rework costs. The total expected cost becomes an uncertain quantity,

$$E[\text{Total Cost}] = \sum_{\text{activity } j}^N (W_j + R_j Pr_j), \tag{2}$$

where R_j is the sum of rework activities needed if activity j must be reworked, and Pr_j is the probability of a design defect in activity j . Note the rework required for a design defect originating at activity j may involve more activities than activity j alone, depending on when the design defect was detected.

While Eq. (2) is the basis for the analysis, it breaks costs out by activity. In Eq. (2) the quantities W_j and R_j can be easily tabulated from the standard work and project documents. It is known how long each activity takes, and it is known what must be reworked if a design defect arises at each and every activity. However, the likelihood of a design defect Pr_j arising at any activity j is not so easily determined. Data was insufficiently significant at the activity level to calculate Pr_j . Instead, we compute the probability of a design defect over each program segment, discussed next.

2.3.5. Step 5: Compute impact of each program segment

We next determine the contribution of each program segment to total expected cost of a program. Data on work, rework and likelihood of design defects is available at this level of resolution.

To consider the rework costs by program segment, we compute average costs and probability of rework for the activities within a program segment. First, we determine the segment average values of Pr_i , \bar{W}_i and \bar{M}_i . The probability Pr_i of a design defect occurring within a segment i is the occurrence rate of defective activities within a segment:

$$Pr_i = \frac{n_{i, \text{defects}}}{n_i}, \tag{3}$$

where $n_{i, \text{defects}}$ is the number of activities with a defect arising within the segment i , and n_i is the number of activities within the segment i .

Note the data available from the 7 person-years of project data has sufficient occurrence rates at the segment level, but not at the activity level. At the activity level, many activities have zero defect counts and so were not statistically significant. At the segment level, all segments had over five occurrences typically deemed necessary for significance.

Similarly, the segment average rework rate \bar{M}_i is an interesting statistic for comparison of segments:

$$\bar{M}_i = \frac{1}{n_{i, \text{defects}}} \sum_{j \in \text{segment } i} R_j, \tag{4}$$

where $n_{i, \text{defects}}$ is the number of activities with a defect arising within the segment i , and R_j is the activity rework cost multiplier for activity j . Similarly, the segment average work cost \bar{W}_i is

$$\bar{W}_i = \frac{1}{n_i} \sum_{j \in \text{segment } i} W_j, \tag{5}$$

where n_i is the number of activities within the segment i , and W_j is the cost of activity j .

With these segment level equations, Eq. (2) for the expected program total cost $E[\text{Total Cost}]$ we can compute the expected cost $E[\text{Cost}_i]$ of a program segment i as;

$$E[\text{Cost}_i] = n_i \bar{W}_i + n_i \bar{M}_i \bar{W}_i Pr_i, \tag{6}$$

where n_i is the number of activities in segment i , \bar{W}_i is the average activity cost in segment i , \bar{M}_i is the average rework rate in segment i (Eq. (4)) and Pr_i is the average probability of reworking an activity in segment i . Note that all quantities in Eq. (6) can be tallied from the available project data. Equation (6) is the form used and computed in the paper.

Finally, we calculate the percentage cost contribution of rework of each segment, $E[\text{Cost}_{i, \%}]$ as a fraction of all the segments;

$$E[\text{Cost}_{i, \%}] = \frac{E[\text{Cost}_i]}{E[\text{Total Cost}]} (100), \tag{7}$$

where $E[\text{Cost}_i]$ is the total expected cost of segment i (Eq. (6)), and $E[\text{Total Cost}]$ is the total expected cost of the program (the sum of Eq. (6) over all segments i).

Table 3. Data source breakdown

Data Source	Count
ECR	22
FRACAS	16
Gate Reviews	11
Design Reports	9

3. Findings: Occurrence and impact of decisions

3.1. Activities and design defects

In step 1, we listed the design defects from the program data sources (Figure 4). Overall, out of 211 total system-level requirements, a total of 58 system-level design defect missed requirements occurred from the two major product development programs. Note there could be multiple misses over time for a particular requirement; for example, the endurance requirement changed at each gate review and after the final flight test.

Any *design defect* found is quantified with an engineering target specification for the rework activities to meet, so that the design would be made compliant to the desired specifications. For example, a particular avionics module may have had an over-heating problem when the operating temperature rose to 55 °C in test where the required operating limit was 50 degrees Celsius. The ECR proposed relocating the air scoop location to increase airflow to the cooling system such that the avionics module maintained a temperature range between 40 and 50 degrees Celsius. The root cause was traced to inadequate heat load analysis from the modeling and simulation design activity. By tracking these design rework measures, we could understand which subsystem was associated with the design defect.

In total, 264 ECRs, FRACAS, gate review and design reports were reviewed. The breakdown of the 58 design defects by subsystem is shown in Figure 6 and the breakdown by data source is shown in Table 3. Most of the design defects are attributed to the novel structure and avionics, which was expected as most of the design activities are concentrated on these systems. The electrical and mechanical systems were traditional designs available from vendors. The breakout in Figure 6 also indicates most of the design defects were on the in-house activities, consistent with that found elsewhere (Obaid *et al.* 2005). The engineering novelty was concentrated in these structure and avionics systems developed in-house. 66% of the design defects were from ECR and FRACAS data sources, which is an indication that most design defects were discovered late in the development process.

Next, the breakout by activity is shown in Figure 7. We found that eight of the design defects were detected before concept freeze, seven before design freeze, 19 after design freeze during testing, and 24 arose only once in field service. Of the 19 design defects detected during testing, 13 of them were not identified until final flight testing began. This result indicates that at least 13 of the 211 total requirements, required system-level testing activities to determine consistency.

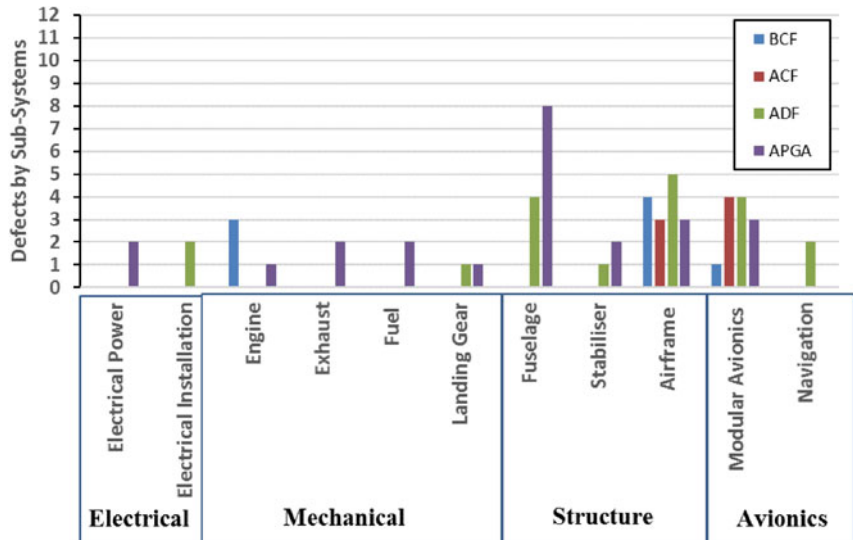


Figure 6. Breakdown of design defects detected by subsystem.

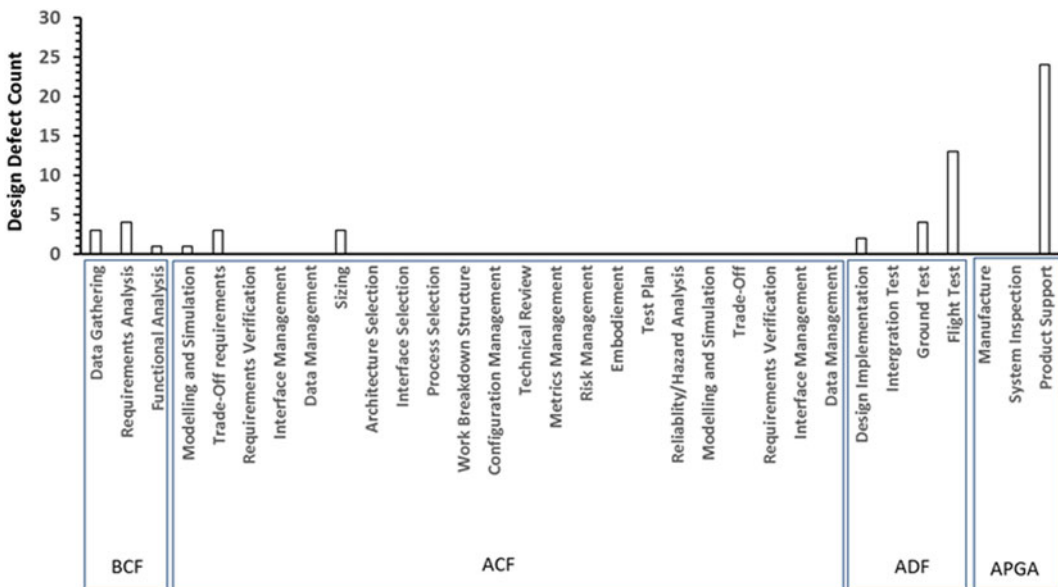


Figure 7. Breakdown of design defects by activity where detected.

Overall, 6% of the total system requirements could not be determined prior to final flight testing in the development process used at this company.

Next in step 2 (Figure 4), we trace each design defect to its root cause activity. Each design defect was followed regressively back through the design process Gantt chart, and included analysis of the design review documentation and design reports associated with each activity. From this information, we are able to isolate

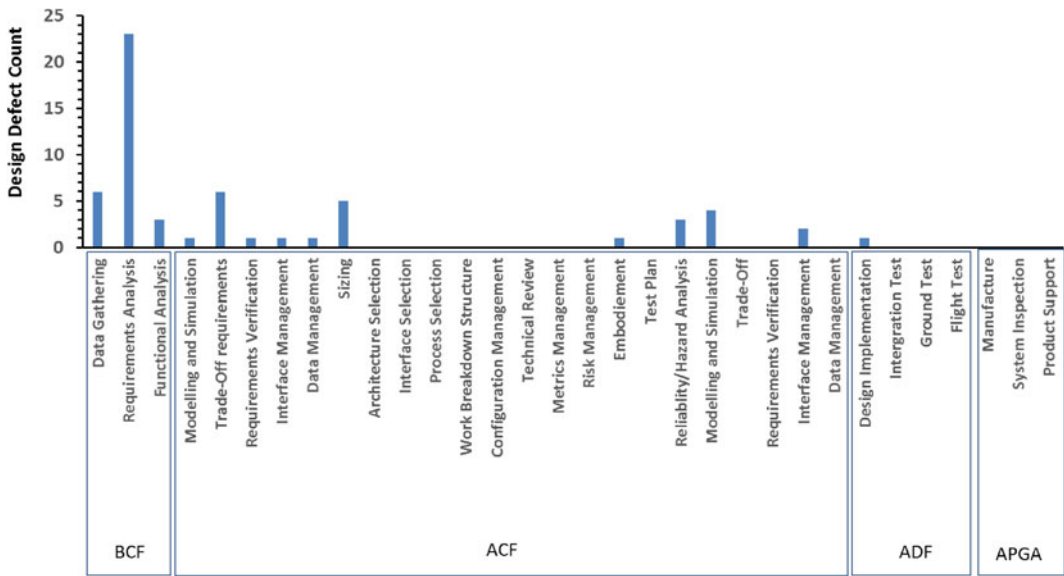


Figure 8. Breakdown of design defects root cause activity.

the onset activity for each design defect. This is plotted as the dark bars in Figure 8 for the 58 total design defects.

To ensure that the design defects are adequately categorized to its root program segment and subsystem, an inter-rater agreement measure was carried out. The design defect root cause analysis was repeated twice using two engineers on the projects, for all 58 design defects. The Cohen’s Kappa was 0.81 for the root cause activity identification (Figure 8) and 0.90 for the root cause subsystem identification (Figure 6). Both results show a good agreement for the categorization of root cause program.

Studying Figures 7 and 8, conclusions can be further drawn. At 23 design defects, the dominant root cause of design defects is attributed to the *requirements analysis*, which occurs in the early concept phase BCF program segment. This is the point in the program when requirement targets for the design are defined, and requires estimation of subsystem capabilities and their mutual impact. This result follows and confirms others who have noted the difficulty and impact of requirements definition (Almefelt *et al.* 2006). The BCF segment had more than half of the design defects as root cause (32 total), and the remainder were in the ACF and ADF design program segments (25 and 1 total). No root cause design defects arose in the testing or operation phases, as expected.

In addition to identifying the root cause activity, the root cause subsystem was also identified. This is shown in Figure 9. Notice the airframe had the most root cause design defects, which was due to its novel construction which was a major objective of the design concept. Conventional electrical and mechanical systems were used which the design team had extensive previous experience.

The count of activities and number of design defects is shown in Table 4, and the occurrence rate graphed in Figure 10. Notice the likelihood of a mistaken decision culminating in a missed requirement is much higher for early phase design decisions. The likelihood a decision made in the early BCF segment was

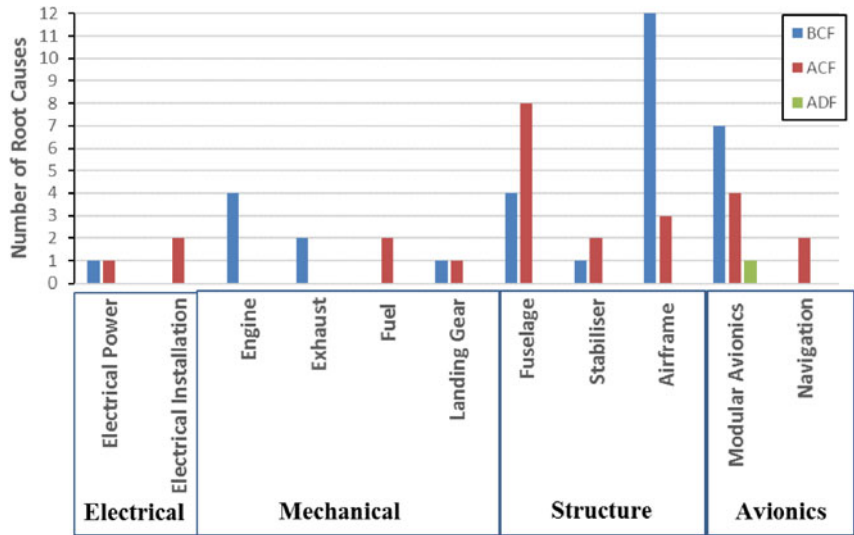


Figure 9. Breakdown of design defect root causes by subsystem.

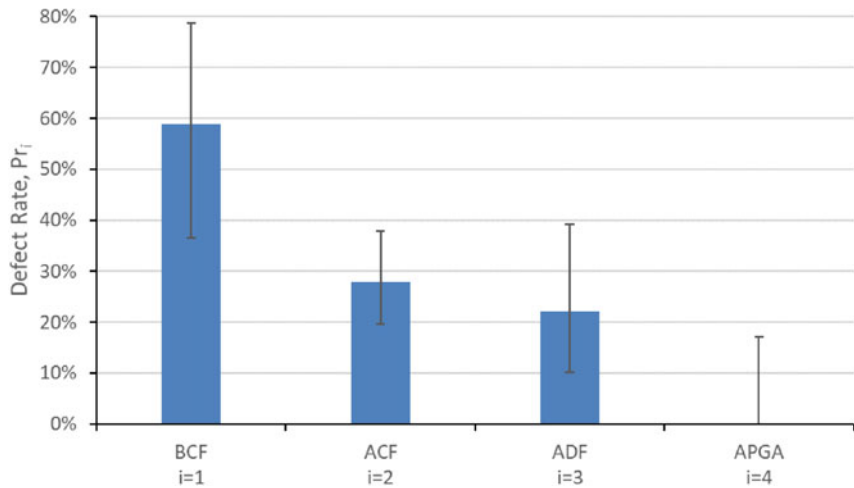


Figure 10. Occurrence rate of root cause of design defects by program segment.

computed as 59%, a very high rate, significantly different from the latter segments. This result holds for this company using their particular development process. Note the 95% confidence intervals were rather large given the small sample sizes (Table 4), but the concept phase versus later phases result remains significant.

3.2. Impact analysis

Following the next step 3 outlined in Figure 5, we next analyzed for the rework activities needed to fix each design defect. This allows for a comparison of impact. When doing this rework assessment of each design defect, a second point of comparison was computed. For each design defect, not only was the actual rework

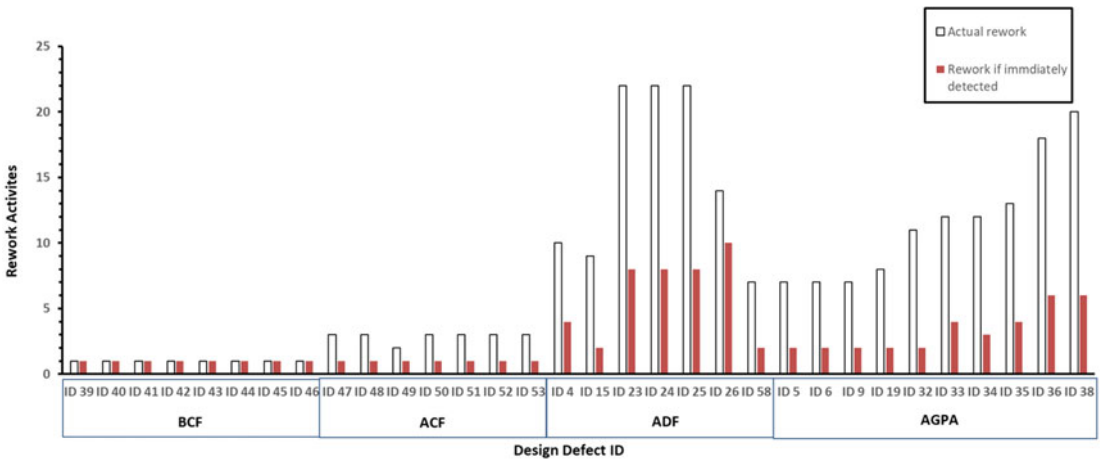


Figure 11. High rework activities for defects originating BCF.

Table 4. Design defect occurrence rates

Program Segment	Activities	Activities with Defects	Defect Rate, Pr_i
BCF	17	10	59%
ACF	75	21	28%
ADF	27	6	22%
APGA	16	0	0%

determined, but we also computed the difference in rework activities needed if the defect had been discovered immediately after it was detected.

These results are shown in Figure 11 for the concept segment BCF design defects (32 total) and in Figure 12 for the program segment ACF defects (25 total). One comparison to consider is the difference in count of rework activities needed between the concept phase and design phase defects. That is the comparison of rework activity counts (graph vertical axis height) between Figures 11 and 12. As can be seen, concept phase defects require more rework. A second comparison to consider is the difference in actual rework required and the theoretical rework that would have been required if the defect were immediately detected. In each graph, this is the difference in white and dark vertical bars for each activity.

An observation of Figures 11 and 12 shows that it is costlier in terms of rework activities to resolve design defects originating from the BCF program segment, as more rework activities have to be executed. This is shown in Figure 13, which considers the increased rework needed for concept phase design defects that are discovered later. Also, the rework activity would always be substantially less if it were to be discovered immediately at the point of occurrence. This is all consistent with observations of impact of defects determined late versus early reported elsewhere (Tassey 2002). We here build on those works in our analysis of root cause occurrence.

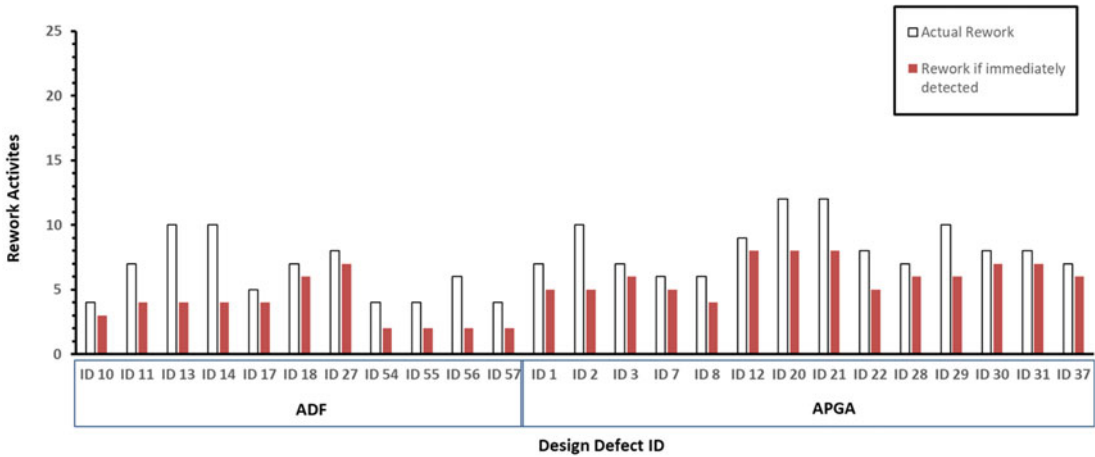


Figure 12. Lower rework activities for defects originating ACF.

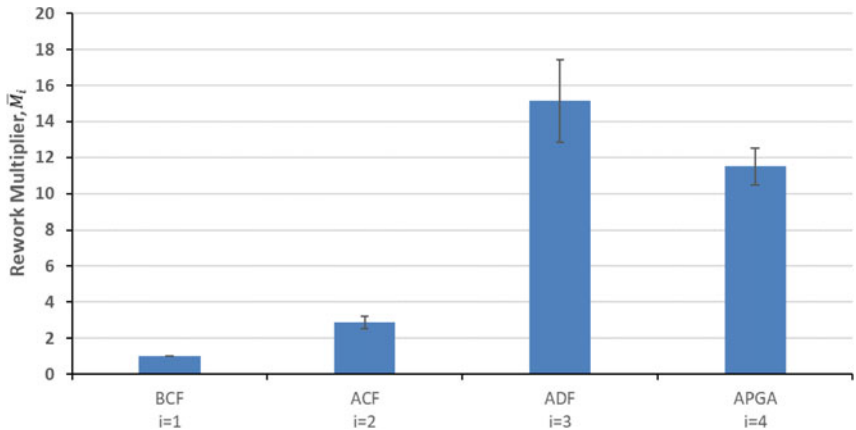


Figure 13. Average rework multiplier for when a BCF design defect is detected.

The average rate of increase in percentage of rework activities between the BCF and ACF program segments is as illustrated in Figure 13. It is observed that a statistically significant average of 13.0, or 13 times more activities were required for this company to fix a concept phase design defect not detected until after system-level testing and in-service. This statement is now quantified for this data set. A similar analysis shows a multiplier of 15.1 for the ADF program segment, and 2.9 for the ACF program segment.

This compares with earlier from Boehm & Papaccio (1988), Boehm & Basili (2001), who found a multiplier of 50 for the BCF segment, and with NASA who found a multiplier of 29-1000X for spacecraft systems (NASA 2016). Our results differ from previous in that the last column of BCF defects detected very late while in the APGA segment (in-field operation) had a lower multiplier than defects found during testing (Figure 13). This is because in our case, such defects were not actually fixed; instead usually the defect was interrogated and requirements revised and the issue accepted as a non-compliance and operations modified.

Table 5. Contribution of work and potential rework attributed to each program segment

	Segment Activities n_i	Rework Multiplier, \bar{M}_i	Probability of rework, Pr_i	Cost Contribution, $E[Cost_i]$	Percent Contribution, $E[Cost_i, \%]$
BCF	17	13.0	0.59	147.0	33.5%
ACF	75	7.4	0.28	231.2	52.7%
ADF	27	2.9	0.22	44.1	10.1%
APGA	16	1.0	0.00	16.0	3.7%

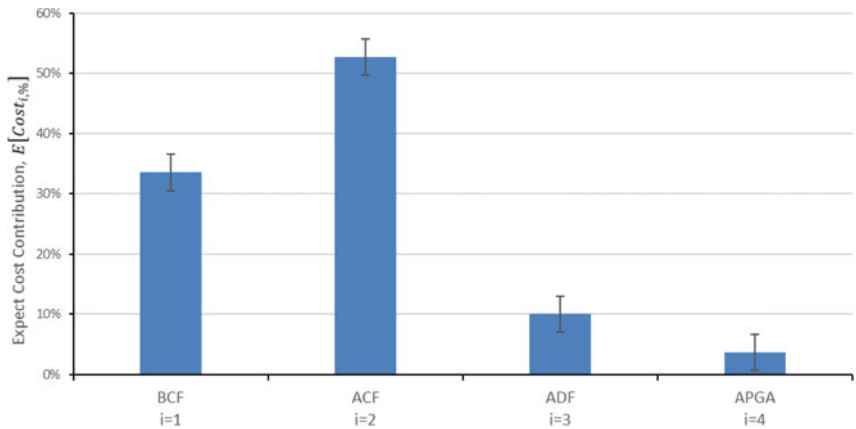


Figure 14. Cost contributions of each program segment.

Therefore, the actual rework activities simply involved issuing the non-compliance and so were less.

Next, we apply Eq. (2) and compute the contributions of each segment to expected cost. As a percent, this is in the last column of Table 5, as graphed in Figure 14. The results show that, for this company using their development process, 86% of cost was determined by design before design freeze. Within that, 34% had been determined before concept freeze. These results were statistically significant at 95% confidence (Figure 14).

3.3. Confirming results through interviews

After this analysis, interview sessions were conducted to confirm the results and ensure that the interviewee’s responses were complete, thorough, and meaningful. Six question categories were interactively asked in a recursive 5-why’s approach (Ohno 1988) as shown in Table 6.

The responses in the interviews were then holistically coded (Saldana 2015), and two (2) coders were used to conduct an inter-rater agreement measure. The resulting Cohen’s Kappa was 0.80, indicating good agreement. Typical responses

Table 6. Question categories to understand design defects

Question	
(1)	Why is it difficult to manage contractual information with design development information?
(2)	Why was it difficult to estimate, assess and manage requirements?
(3)	What are barriers preventing the adoption of new technologies and concepts?
(4)	What is missing from the current state development process?
(5)	What development resource constraints do you face?
(6)	Do you have any other comments to improve development?

Table 7. Typical interview responses

Response	
(1)	<i>'Lack of test facilities, limited access to testing facilities so such access is precious'</i>
(2)	<i>'Ability to translate past experience into simple guidelines to reduce learning curve time'</i>
(3)	<i>'A proper design flow with proper risk and impact assessment'</i>
(4)	<i>'With the separation of programme and technical management considerations'</i>

Table 8. Interview response agreement with design defect analysis

Interview Category	Defects	Statements	% Defect	% Statement
Customer Interactions and Data	6	15	10%	10%
Supplier Interactions and Data	1	11	2%	7%
Requirements Engineering	29	70	50%	47%
Modeling, Sizing	10	15	17%	10%
Concept and Technology	5	13	9%	9%
Technical Reviews, Process and Risk	6	26	10%	17%
Embodiment	1	0	2%	0%
Testing	0	0	0%	0%
TOTAL	58	150	100%	100%

are shown in Table 7; where the responses provided for better understanding of the root cause activity and subsystem.

These interview statements were then categorized into eight interview categories and the number of interview responses against these categories was summed across the interviews, as shown in the second column of Table 8.

To relate the interview responses to the previous activity quantification, the 58 design defects (Figure 7) were also grouped into eight categories. This is shown in the third column of Table 8. The relative percentage of design defects and interview statements is shown in Table 8, and a Chi squared test indicates the two agree with p -value <0.002 . This agreement indicates the earlier quantified analysis is supported by the engineers' perceptions.

4. Discussion: Impact of early phase decisions

4.1. Reviewing the evidence gathered

We have shown from ECR, FRACAS and gate reviews that there is indeed a distinctively higher number of corrective activities required to rectify missed design requirements originating from before a design concept is frozen. *H1* and *H2* are therefore supported.

For *H3*, our data has shown that 25 of the 58 (most frequent) missed requirements are attributed to concept phase requirements analysis. The interviews with engineers also reaffirmed that requirement changes could be better managed with better understanding of requirements trade-off and definition during conceptual design. Hence, *H3* is supported.

4.2. Limitations

This is a longitudinal study at one large company, limited to the tools, methods, experience and processes used to develop new products. The high impact root cause of requirements management may not hold true at other firms should they have a means to establish requirements targets early. However, other studies echo our results here (Almefelt *et al.* 2006). Therefore, if the company were to invest in improved conceptual design stage methods and tools, perhaps the error rate could be reduced. There is an incentive to do so, since 86% of the rework cost committed is determined before design freeze. Nonetheless, this study is typical of an aerospace firm today.

The second limitation is our findings with relation to the 80–20 rule. Based on the evidence, this validation can only hold for new product development in a mid-sized aerospace-level system complexity. This may or may not extend to very large-scale systems and may not extend to small-scale individual projects. Further investigations are needed.

Another limitation is the consideration of the means to mitigate design defects and their effectiveness. In the literature, we find at least two distinct approaches used to address concept phase design defects. One path is to build prototypes and a second is to analyze concepts using modeling and simulation. Both provide improved estimates of expected as-delivered system-level requirement outcomes at different configurations and sizes.

The first approach to mitigate design defects is through the use of physical prototypes and empirical testing of many concepts early in the conceptual design stage (Camburn *et al.* 2013; Camburn *et al.* 2014; Dunlap *et al.* 2014; Camburn *et al.* 2015a,b,c). They provide the designers with information on whether the conceptual design is sound. This applies the design philosophy 'Fail fast, fail often' (Ullman 2015) and the concept of a 'minimum viable product' (Ries 2011) to provide rapid assurance of system-level requirements. A difficulty with

prototyping can be the build time and cost of highly complex technology, e.g., jet engines, high performance aircraft, long range strategic submarines, etc.

A second approach to mitigating design defects also often used is model-based analysis of preliminary concepts (Agte, Shougarian & De 2012). Models can be used to provide rapid assessment and trade-off analysis of different requirements. This applies the philosophies 'Emphasize the key set of issues' (Almefelt *et al.* 2006) and 'get it right the first time' (Thomke 1998) to provide rapid assurance of system-level requirements. A difficulty with modeling can be the effort needed to build high fidelity models and also model error (Kenway & Martins 2014; Huang *et al.* 2015).

Finally, for the analysis approach used, another limiting factor is that the data originates from the design defect documentation and the case is built around correction activities on these design defects. This does not address whether some requirements are close to the threshold of failing. What is detectable is the rework activities required as a result of design defects.

5. Conclusions

Overall, this work showed the impact and occurrence rates of design defects. Our results would suggest that by assessing requirement change impact early, a design team could develop a forward perspective and plan strategies to mitigate or correct their impact. This indicates research is needed to establish formalized systems that can incorporate and manage not just design requirements but also their margins and mitigation strategies early and throughout development. This should be easily complemented with current product development processes to aid in the enhancement of design practitioners. These views are also supported by the work of Almefelt & Andersson (2004), Bergsjö, Almefelt & Malmqvist (2010), Bertoni, Bertoni & Isaksson (2016).

Our work showed that design defects occur equally before and after the conceptual design freeze. However, the design activities required to correct design defects arising before concept freeze are 13 times higher than after. This then translates into design decisions carrying 86% of total work plus potential rework costs. As such, the anecdotal 80–20 rule of design development cost is validated, that the 80 percent cost committal stems from before the design freezes for complex systems. We also found that the majority of the design defects originating from before concept freeze is attributed to requirement analysis. Consequently, we found that difficulty ensuring consistency and trading off requirements was the highest occurring root cause of design defects.

Acknowledgments

This work has been funded and supported by the Economic Development Board of Singapore under the Industrial Partnership Programme. The authors would also like to thank the SUTD-MIT International Design Centre (IDC, [idc.sutd.edu.sg](https://www.idc.sutd.edu.sg)) for financial and intellectual support. Any opinions, findings, or recommendations are those of the authors and do not necessarily reflect the views of the supporters.

References

- Aerospace Industries Association** 2009 S3000L International procedure specification for Logistics Support Analysis (LSA).
- Agarwal, H., Renaud, J. E., Preston, E. L. & Padmanabhan, D.** 2004 Uncertainty quantification using evidence theory in multidisciplinary design optimization. *Reliability Engineering and System Safety* **85** (1–3), 281–294.
- Agte, J., Shougarian, N. & De Weck, O.** 2012 Multistate analysis and optimization of a geared turbofan engine lubrication system. In *12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference and 14th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, p. 5637. American Institute of Aeronautics and Astronautics.
- Allard, T.** 2005 Navy's \$100m chopper can't fly in bad light. Retrieved September 30, 2016, from <http://www.smh.com.au/news/National/Navys-100m-chopper-cant-fly-in-badlight/2005/03/18/1111086017635.html>.
- Almfelt, L. & Andersson, F.** 2004 Requirements as a means to drive innovation: a reason based perspective. In *ASME 2004 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pp. 89–98. American Society of Mechanical Engineers.
- Almfelt, L., Berglund, F. & Nilsson, P.** 2006 Requirements management in practice: findings from an empirical study in the automotive industry. *Research in Engineering Design* **17** (3), 113–134.
- Anderson, J. D. J.** 2004 *Fundamentals of Aerodynamics*. McGraw–Hill.
- Ball, L. J., Evans, St. B. T. J., Dennis, I. & Ormerod, T. C.** 1997 Problem-solving strategies and expertise in engineering design. *Thinking and Reasoning* **3** (4), 247–270.
- Ball, L. J., Onarheim, B. & Christensen, B. T.** 2010 Design requirements, epistemic uncertainty and solution development strategies in software design. *Design Studies* **31** (6), 567–589.
- Batchelor, T.** 2016 Taxpayers foot 1billion after MOD 'failed to properly test' Type 45 destroyer engine. Retrieved September 30, 2016 from <http://www.express.co.uk/news/uk/692341/Type-45-destroyer-engine-problems-analyst-claims-MoD-failed-properly-test-engines>.
- BBC News** 2002 'Wobbly' millennium bridge fixed. Retrieved September 30, 2016 from http://news.bbc.co.uk/2/hi/uk_news/1026224.stm.
- Bergsjö, D., Almfelt, L. & Malmqvist, J.** 2010 Supporting requirements management in embedded systems development in a lean-influenced organization. In *11th International Design Conference, DESIGN 2010, Dubrovnik, Croatia 17-20*, pp. 1025–1034.
- Bernard, B., Eric, C. & Archag, T.** 2015 Stop multibillion dollar development delays. Oliver Wyman, <http://www.oliverwyman.com/insights/publications/2015/jun/stop-multibillion-dollar-development-delays.html#.V9akTCh96Uk>. Retrieved on 12th September 2016.
- Bertoni, M., Bertoni, A. & Isaksson, O.** 2016 Evoke: A value-driven concept selection method for early system design. *Journal of Systems Science and Systems Engineering*. doi:10.1007/s11518-016-5324-2.
- Boehm, B. & Basili, V. R.** 2001 Software defect reduction top 10 list. *Computer* **34** (1), 135–137.
- Boehm, B. W. & Papaccio, P. N.** 1988 Understanding and controlling software costs. *IEEE Transactions on Software Engineering* **14** (10), 1462–1477.
- Brothers, C.** 2009 Germany and france delay decision on airbus military transport. Retrieved September 30, 2016 from <http://www.nytimes.com/2009/06/12/business/gl>

obal/12airbus.html?_r=2&ref=europe&mttrref=en.wikipedia.org&gwh=7F9EBE538F F6A8649C4E390CE9169DBE&gwt=pay.

- Burr, T.** 2008 Chinook MK 3 Helicopters (Rep. No. HC-512). National Audit Office, London.
- Camburn, B. A., Jensen, D., Crawford, R., Otto, K. & Wood, K.** 2015a Evaluation of a strategic method to improve prototype performance with reduced cost and fabrication time. In *DS 80-4 Proceedings of the 20th International Conference on Engineering Design (ICED 15) vol. 4: Design for X, Design to X, Milan, Italy, 27–30.07.15*.
- Camburn, B., Dunlap, B., Gurjar, T., Hamon, C., Green, M., Jensen, D., Crawford, R., Otto, K. & Wood, K. L.** 2015b A systematic method for design prototyping. *ASME Journal of Mechanical Design (JMD)* **137** (8), Paper No: MD-14-1487, doi:[10.1115/1.4030331](https://doi.org/10.1115/1.4030331).
- Camburn, B., Dunlap, B., Kuhr, R., Viswanathan, V., Linsey, J., Jensen, D., Crawford, R., Otto, K. & Wood, K. L.** 2013 Methods for prototyping strategies in conceptual phases of design: framework and experimental assessment. In *Proceedings of the ASME 2013 International Design Engineering Conferences & Computers and Information in Engineering Conference, IDETC/CIE 2013, Portland, OR, August 4–7, 2013, DETC2013-13072*.
- Camburn, B., Sng, K., Perez, K. B., Otto, K., Wood, K. L., Jensen, D. & Crawford, R.** 2015c The way makers prototype: principles of DIY design. In *Proceedings of the ASME 2015 International Design Engineering Conferences & Computers and Information in Engineering Conference, IDETC/CIE 2015, Boston, MA, August 2–5, 2015, DETC2015-46295*.
- Capers, R. S. & Lipton, E.** 1993 Hubble error: Time, money and millionths of an inch. *The Academy of Management Executive* (1993–2005), pp. 41–57. <https://griffith.ri.talis.com/items/DA245EAD-ECCD-E776-5446-29AC358C3A35.html>.
- Charette, R. N.** 2008 *What's Wrong with Weapons Acquisitions*. IEEE Spectrum.
- Chong, D. S., Van Eerde, W., Chai, K. H. & Rutte, C. G.** 2011 A double-edged sword: The effects of challenge and hindrance time pressure on new product development teams. *IEEE Transactions on Engineering Management* **58** (1), 71–86.
- Clarkson, J. & Eckert, C.** (Eds) 2010 *Design Process Improvement: A Review of Current Practice*. Springer Science & Business Media.
- Clausing, D. P.** 1994 *Total Quality Development. (A Step by Step to World Class Concurrent Engineering)*. The American Society of Mechanical Engineers.
- Cooper, R. G.** 1990 *Stage-gate systems: a new tool for managing new products*. Business Horizons May–June 1990.
- Cooper, R. G.** 2011 *Winning at New Products, Creating Value Through Innovation*. Basic Books.
- Cooper, R. G. & Kleinschmidt, E. J.** 1988 Resource allocation in the new product process. *Industrial Marketing Management* **17** (3), 249–262.
- Crawley, E., De Weck, O., Magee, C., Moses, J., Seering, W., Schindall, J. & Whitney, D.** 2004 *The Influence of Architecture in Engineering Systems (Monograph)*. MIT Press.
- Department of Defense** 2005 MIL-HDBK-516B: Airworthiness Certification Criteria.
- Department of Defense** 2015 Department of Defense Instruction, Number 5000.02. Department of Defense.
- Department of Defense, System Management College** 2001 Systems Engineering Fundamentals. Department of Defense.
- De Weck, Olivier, L. & Eckert, C.** 2007 *A Classification of Uncertainty for Early Product and System Design*. MIT Press.

- Dunlap, B., Hamon, C., Camburn, B., Crawford, R., Jensen, D., Green, M., Otto, K. & Wood, K. L.** 2014 Heuristics-Based Prototyping Strategy Formation: Development and Testing of a New Prototyping Planning Tool. In *Proceedings of the ASME 2014 International Mechanical Engineering, Congress & Exposition, IMECE2014-39959*, November 14–20, 2014, Montreal, Quebec, Canada.
- Eckert, C., Isaksson, O. & Earl, C.** 2012 Product property margins: an underlying critical problem of engineering design. In *TMCE 2012*.
- Geoffrey, B.** 1994 *Product Design for Manufacturing and Assembly Volume 26*. Computer Aided Design.
- Govan, F.** 2013 £2 billion Spanish navy submarine will sink to bottom of sea. Retrieved September 30, 2016, from <http://www.telegraph.co.uk/news/worldnews/europe/spain/10073951/2-billion-spanish-navy-submarine-will-sink-to-bottom-of-sea.html>.
- Hales, C. & Gooch, S.** 2011 *Managing Engineering Design*. Springer Science & Business Media.
- Hamada, H.** 1996 The Importance of Getting it Right the First Time. *European Community Quarterly Review* 4 (3), 1996. As presented at the October 1991 EuroPACE Quality Forum.
- Hazelrigg, G. A.** 1998 A framework for decision-based engineering design. *Transactions American Society of Mechanical Engineers Journal of Mechanical Design* 120, 653–658.
- Huang, E., Xu, J., Zhang, S. & Chen, C. H.** 2015 Multi-fidelity model integration for engineering design. *Procedia Computer Science* 44, 336–344.
- Keller, R., Eger, T., Eckert, C. M. & Clarkson, P. J.** 2005 Visualising change propagation. In *DS 35: Proceedings ICED 05, the 15th International Conference on Engineering Design, Melbourne, Australia, 15–18.08.2005*.
- Kenway, G. K. & Martins, J. R.** 2014 Multipoint high-fidelity aerostructural optimization of a transport aircraft configuration. *Journal of Aircraft* 51 (1), 144–160.
- Ladyman, J., Lambert, J. & Wiesner, K.** 2013 What is a complex system? *European Journal for Philosophy of Science* 3 (1), 33–67.
- Mark, V. A., Robert, S. L., Shelia, E. M. & Obaid, Y.** 2006 Historical cost growth of completed weapon system programs. RAND Corporation, RAND.
- NASA Error cost escalation through the project life cycle. Retrieved September 2016 from <https://ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20100036670.pdf>.
- North Atlantic Treaty Organization** 2009 STAGNAG 4671: UAV system airworthiness requirements.
- Obaid, Y., David, E. S., Mark, A. L. & Frances, M. L.** 2005 Lessons learnt from the F/A 22 and F-18 E/F developmental programs. RAND Corporation.
- Ohno, T.** 1988 *Toyota Production System: Beyond Large-Scale Production*. CRC Press.
- Otto, K. N. & Kristin, L. W.** 2001 *Product Design, Techniques in Reverse Engineering and New Product Development*. Prentice Hall.
- Pandey, V., Mourelatos, Z. & Castanier, M.** 2014 Decision topology assessment in engineering design under uncertainty. In *ASME 2014 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, V02BT03A04*.
- Perez, R. E., Liu, H. H. & Behdinan, K.** 2004 Evaluation of multidisciplinary optimization approaches for aircraft conceptual design. In *AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Albany, NY*.
- Ries, E.** 2011 *The lean startup: How today's entrepreneurs use continuous innovation to create radically successful business*. Crown Business.

- Saad, N. M., Al-Ashaab, A., Maksimovic, M., Zhu, L., Shehab, E., Ewers, P. & Kassam, A.** 2013 A3 thinking approach to support knowledge-driven design. *The International Journal of Advanced Manufacturing Technology* **68** (5–8), 1371–1386.
- Saldana, J.** 2015 *The Coding Manual for Qualitative Researchers*. SAGE Publications Ltd.
- Simon, H. A.** 1973 The structure of ill structured problems. *Artificial Intelligence* **4** (3/4), 181–201.
- Tassey, G.** 2002 The economic impacts of inadequate infrastructure for software testing. National Institute of Standards and Technology, RTI Project, 7007(011).
- Thomke, S. H.** 1998 Managing experimentation in the design of new products. *Management Science* **44** (6), 743–762.
- Ullman, D.** 2015 *The Mechanical Design Process*. McGraw–Hill.
- Ulrich, K. T. & Scott, A. P.** 1993 *Does Product Design Really Determine 80% of Manufacturing Cost?* Cambridge Massachusetts, Sloan School of Management, MIT Press.
- Ulrich, K. T. & Scott, A. P.** 1998 Assessing the importance of design through product archaeology. *Management Science* **44** (3), 352–369.
- Vadaro, M. J.** 2013 *LeMessurier Stands Tall: A Case Study in Professional Ethics*. The AIA Trust.
- VDI-Standard** 1993 VDI 2221 Systematic approach to the development and design of technical systems and products. The Association of German Engineers.
- Williard, N., He, W., Hendricks, C. & Pecht, M.** 2013 Lessons learned from the 787 Dreamliner issue on lithium–ion battery reliability. *Energies* **6** (9), 4682–4695.
- Wong, K.** 2006 What grounded the airbus A380? Retrieved September 30, 2016, from <http://www.cadalyst.com/management/what-grounded-airbus-a380-5955>.