Industrial Feedback on Implementing Functional Decomposition Design Processes for Reliability and Functional Safety

Matt Pallaver[1*], *Ahed* Qaddoura[2], *Sung-hee* Do, Ph.D.[3]

[1]ZD-TRW, Livonia, MI 48150 USA
[2]Schlumberger, Sugar Land, TX 77498 USA
[3]Lexington, MA 02420 USA

Abstract. The value of Axiomatic Design functional decomposition as a design analysis point tool has been widely reported. This paper reports on the implementation of functional decomposition design processes as a system design tool on 23 industrial projects over a 5 year period with product development teams ranging from 6 to 35 engineers. The products developed were systems of systems with mechanical, electrical, firmware, software, and operational interface elements. Functional decompositions ranged from about 200 to 1600 Functional Requirements. A number of these projects are now in commercial production. This paper reviews the process definition and implementation process steps that evolved from these experiences. The paper then reports on the implementation lessons learned and the value propositions noted. Conclusions and recommendations are made. The experiences demonstrated that functional requirement decomposition processes aid in achieving on-time, on-cost and on-specification project development targets. The authors propose this paper summarizes the "Endgame" design process impact that axiomatic design can reasonably expect in industry design practices for system development.

1 Introduction and Overview

We applied Axiomatic Design (AD) processes to concept phase design of software, hardware, electronics and systems of systems. The complete process was referred to as Functional Decomposition analysis. This paper first summarizes the current working process and then discusses lessons learned.

Prior reported applications of AD have established a value of the process as a point tool for the analysis of specific design problems. We are not aware of discussions of large scale applications of AD to system design in industrial design settings.

The implemented process is much larger in scope than traditional AD methods. The matured process presented is based on an AD decomposition backbone. We refer to the process descriptively as a Functional Requirements and Decomposition process producing concept phase Function Models for subsequent analysis and design iteration. We use the FR-DP and zig-zag decomposition terms of AD methods.

Not intended as academic research, this paper assumes a general knowledge of the AD definition of Customer Needs (CNs), Functional Requirements (FRs), Design Parameters (DPs), Process Variables (PVs), Design Matrices, axiom 1 and axiom 2 as well as zigzag decomposition between the FR and DP domains. Terms that are defined in the Key Terms of Appendix 1 are capitalized in this text.

From 2011 to 2017 Functional Requirements analysis, Functional Decomposition processes and analysis of the resulting Function Maps were introduced to 23+ projects. Primarily, the projects involved products developed to perform a variety of functions within oil and gas down-hole exploration. Example functions include pressure sampling, liquid sampling, rock formation sampling, moving and manipulating down-hole process components and making measurements with various sensors of the down-hole rock formations. The products were typically processor controlled systems communicating with the surface and controlling hydraulics, motors, sensors and other active components. The environment is harsh being highly corrosive, having temperatures often in excess of 150 C (300 F) and up to 30,000 psi hydrostatic pressure.

Design teams were composed of trained engineers often at master and PhD levels from some of the finest schools in the world with typically 5 years prior design experience.

The primary need driving us to propose changes to design processes was improvement of functional reliability and safety of the products.

* Corresponding author: Matt@Pallaver.com

2 Implementation Issues

The methods we used evolved considerably over the period of use as we responded to implementation issues. Most process evolution was in the area of integrating Decomposition into other key functions of the company's concept phase processes. Areas of implementation evolution included:

- Definition of Terms
- Stakeholder requirements analysis
- Contractual requirement obligations
- Constraints management
- Product risk assessment
- Functional performance (reliability) risk analysis
- Prior product lessons learned and historic failure modes
- Design verification plans
- End to end traceability
- Fitting functional decomposition into the design V model and Process Assessment Models (PAMs)

2.1 Conflict with Aesthetic Design

We note that Functional Analysis applies to the performance aspect of product. It does not easily apply to design for consumer perception of aesthetics or goodness. Often aesthetic design will dominate the design equation of a successful product.

We further note that Functional Analysis and good functional design is often at odds with aesthetic design. For example, a functional requirement for a reasonable cell phone powered life (perhaps DP = Battery Capacity) conflicts with the aesthetic requirement for a visually slim package.

2.2 Conflict with Success

The authors' experience is there is not a strong correlation between good functional design and product success in many markets.

In part this is because Customers are generally incapable of judging good functional performance or fitness for use and their purchase decisions are more often driven by aesthetics, advertising or celebrity promotion of the product.

As a result the measure of good design and design processes in industry is defined largely by the subsequent product business success.

Thus, it is important to note in implementing a functional performance based design approach that the primary resistance to change in a successful company will be unsophisticated management who are afraid of changing underlying process 'recipes' and that will not see a connection between better functional design and business success.

3 Definition of important terms

Our design projects are often composed of internationally dispersed development teams. For all development projects, to minimize the risk of introducing communication based design and process errors, a common and unambiguous vocabulary is essential and the importance of having good working definitions cannot be overstated. See Appendix 1 for definitions of key terms of our Functional Decomposition and related analysis processes.

4 Summary of current working process

The current operational process is presented graphically in Appendix 2. The Functional Decomposition process passes through the following steps:

1. Process initiated by release of Contractual Requirements
2. Stakeholders Needs Assessment.
3. Requirements analysis.
4. Development of Operational Requirements and Input Constraints.
5. Functional Decomposition of Operational Requirements.
6. Systematic Product Risk assessment.
7. Validation of Input Requirements against the Function Map.

These process steps are discussed in turn.

4.1 Contractual Requirements document

Our overall design process starts with a Contractual Requirements document. A Contractual Requirements document is common industry practice. This document defines the contractual requirements between the development funding source and the development organization. This will typically contain a marketing needs analysis and business justification. The Contractual Requirements define, as detailed as possible considering that the design has not started, the development requirements to deliver on the business justification. In our Functional Decomposition process, this document is a significant source of many business needs.

4.2 Stakeholder Needs Assessment

Stakeholder Needs assessment is composed of determining stakeholders and collecting Stakeholder Needs. This phase has three tasks.

4.2.1 Determine Stakeholders

Stakeholder categories and potential contacts need to be developed. Stakeholder is a broad term which includes traditional people roles, regulatory agencies, interfaces with other systems and significant physical domain constraints. The assessment of whom or what is a stakeholder is driven by the question "With what or whom will our design need to interact during its complete product life cycle from creation to disposal?" The output of this task is a list of stakeholder categories,

with specific named stakeholders to be contacted when appropriate, with contact information and assignments of the staff responsible for each category.

4.2.2 Eliciting (human) Stakeholder Needs

The systematic contact and solicitation of stakeholder needs from persons is a qualitative research project, and all the rules of good research apply. Perform data gathering in waves of contacts. Between waves, during processing of prior wave research, adjust questionnaires and methods, and test for data saturation. Stop assessing within a stakeholder category when saturated (no new information is being gained by further research). The primary goal is to assess and reach data saturation within all stakeholder categories.

4.2.3 Collecting other Stakeholder Needs

There exist other 'stakeholder' needs. These include, for example, internal standards, regulatory authorities and interfaces. These need to be researched and documented. Generally these will convert to Input Constraints.

The outputs of the Stakeholder Needs collection tasks are stakeholder statements of need, organized by Stakeholder.

4.3 Requirements Analysis

To analyse the Stakeholder needs statements collected from people we first use K-J analysis [1] to categorize the stakeholder needs into Affinity Lists. Then a single step Quality Function Deployment (QFD) [2] House of Quality (HOQ) is used as a process to develop functional objectives to satisfy the customer needs affinity lists. We use stakeholder pairwise comparison ranking [3] of QFD input needs (typically in the form of Affinity Lists). The outputs of this HOQ task are lists of independent Critical to Quality functional statements with targets and prioritization. These statements are in the words of the analysts, traceable back through the QFD and KJ Analysis to the stakeholder statements of need.

Another key Requirements Analysis step is to analyze the Contractual Requirements. Specific product design performance needs from this document must be identified and extracted. The output of this process step is either a separate requirements document summarizing product needs of the Contractual Requirements, or a summary of which paragraphs are product needs, if the document is suitably structured, that are binding on the product design.

In virtually all companies, there are a series of controlling documents representing various organizational requirements applying to new designs. These will document standard needs for things like supply chain, manufacturing and good design practices. All of these documents need to be researched and included as requirement documents that apply to the current design. These requirements almost universally become Input Constraints and do not affect the Operational Requirements. If structured appropriately, these requirement documents can be appended directly to the Input Constraints document.

Separate from Stakeholders, but significant in establishing functional performance requirements, is an analysis of any prior product history. This is research on potential problems, issues and failure modes of current or past similar products. This is generally a research task of manufacturing and field records. Information discovered will often overlap with needs identified by Stakeholders. This research is reviewed and analyzed to define additional needs to resolve historic issues. The output of this process step is a list of addition needs traced back to the relevant research findings.

4.4 Development of Operational Requirements and Input Constraints

Operational Requirements are the system performance functions that must be met to deliver the required business value. These are not a complete set of requirements. Rather they are the complete set of top level Functional Requirements. These are controlled by the design team. These requirements should be viewed as a design response to the required functionality of the Contractual Requirements. This requirement list is developed specifically to drive Functional Decomposition.

Because Contractual Requirements documents in our experience are universally incomplete assessments of Stakeholder Needs, it is necessary to both conduct Stakeholder Needs analysis and examine this analysis along with the Contractual Requirements in crafting the Operational Requirements.

Operational Requirements come out of inspection and discussion of the Stakeholder's Needs analysis and the Contractual Requirements. This is not a deterministic process and will typically take weeks of work.

Operational Requirements determine important metrics like delivery times, project and product costs. Often, assumptions of cost versus technology performance trade-offs may be required in developing the Operational Requirements. There is a design process tendency to want to start engineering design to verify and de-risk these trade-off assumptions before finalizing Operational Requirements. We found it more useful to carry these trade-off risks through to completion of the Operational Requirements and a reasonable first pass Functional Decomposition. Often we found initial proposals on solutions don't survive the relatively quick Functional Decomposition phase. Thus we avoid spending expensive engineering time analysis of trade-offs that wouldn't survive Functional Decomposition. However, verifying these assumptions which carry significant risk should be the first engineering tasks of the design phase.

Operational Requirements are continually subject to revision, particularly during the early design phase as risks are either confirmed or mitigated resulting in conceptual design iterations.

Contractual Requirements or Stakeholder Needs that were determined to be Constraints and not functions are separated from the Operational Requirements and tracked as Input Constraints.

4.5 Functional Decomposition

Functional decomposition proceeds from the Operational Requirements. If there are N Operational Requirements, then there would be potentially N level one decomposition branches. In practice, not all branches are equal and often we would drop a few if insignificant or risk free.

We defined the Decomposition Node to contain:

- A unique ID to identify the node
- A title description of the FR
- A target FR Performance Measure with tolerances
- A title description of the DP
- A definition of the DP verification test
- An assessment of DP risk

We taught designers to read decomposition, when considering the child FRs of the parent FR-DP Decomposition Node, as "The minimum set of independent child FRs required to enable the parent DP to satisfy the parent FR." FRs are to be solution neutral.

We taught Axiom 1, one DP for each FR, and explained the independence requirement for the DPs.

DPs need to be checked to insure they do not violate any of the constraints of the Input or Derived Constraint lists. The DP verification test, to be used to demonstrate that the DP delivered on the FR, is defined.

In the early phases of decomposition, the DP test needs only to be defined sufficiently such that reasonable staff can agree on its definition. These tests will become more elaborate test specifications written during the design phase.

If the selection of a DP creates a binding constraint on the balance of the design process, the DP is added to the Derived Constraints list.

Completion of all node information is not required initially. We would start with FRs and DPs, perhaps defining the balance of the Decomposition Node information to create clarity if the FR-DP text was not clear. Once FR-DP Decomposition Trees are hashed out, and design starts to feel stable, then a more systematic completion should be done.

Trade studies, when needed, to determine DP selection would be documented, released and then noted on the decomposition node comments.

4.6 Systematic Product Risk Assessment

Risk assessment is presented as a question in the decomposition process at the Decomposition Node level, "How might the DP fail to satisfy the FR?" Risk tracking is implemented using an FMEA format. FMEA actions to mitigate risks resulting in one or more of the following options:

- Acceptance of the risk
- Scheduling of an engineering test during development to assess the design margins of the risk
- Mitigation of the risk by addition of child FRs to address the risk
- Changing of the DP to mitigate the risk.

Systematic risk assessment is a substantial axiom 2 value-add to engineering concept phase development processes. In a 1000 FR decomposition, we can easily add 100 FRs responding to DP risk mitigation opportunities. There will also be easily 200 engineering tests scheduled to evaluate DP performance risk during the engineering development phase. Most of these design phase engineering tests are assessing design margins.

4.7 Verification of Input requirements

When the Functional Decomposition is substantially completed, a quality process of Stakeholder Needs and Requirements verification begins.

Each of the Input Requirements documents, which are composed of the Contractual Requirements, crucial Stakeholder Needs, Prior Product History and Other Design Process Standard documents are systematically reviewed. For each identified need of these documents, the corresponding FR in the Decomposition Tree is identified, and the need traced to the FR.

This full traceability of selected Stakeholder Needs, affinity lists, and other input needs to FR-DP nodes to verification tests is an excellent quality tool to insure completeness of the proposed design. It also supports the management of thousands of system FRs that need to be implemented during the design phase.

If a corresponding FR is not found, then this 'Orphan' need is analyzed and a determination is made on its validity. If the Function Tree is at fault for missing a corresponding requirement, then the FR-DP Decomposition is refactored to include a corresponding FR and the trace link made. Often we find the need was not justified, and we would note this by linking the need to an explanatory summary document of unmet needs.

Politically, important stakeholders, customer and contractual needs were explicitly examined and if not met, more often than not, the Decomposition was revised even if the need was not justified by decomposition.

At this point, a design review is held on the Function Tree as a condition of a full release to begin the design work.

With release, design tasks are scheduled to reflect a priority of work on high risk portions of the design and logical dependencies.

Generally we maintained the Functional Decomposition during the early design phases to track and update the testing plans and track the overall design risk. As the design progresses onto engineering and manufacturing documentation control systems, and the work transitions from planned work to fire fighting, we found little value in maintaining the Functional Decomposition as it could not capture or contribute to

the myriad of details and issues of later stage development.

5.0 Process Findings and Lessons Learned

Findings and lessons learned deploying a working industrial functional decomposition process are presented in statements grouped in two sections, Deployment Process and Value proposition. These are presented individually in the general order of the process steps. The findings are contrasted to the general design and development experiences of the authors, not any specific processes of any one organization. The authors share more than 50 years of development experience in software, hardware and manufacturing across more than a dozen organizations.

5.1 Deployment Process Findings

We found that classroom teaching of Functional Decomposition is noteworthy for its ineffectiveness. Then how can we teach the process? We learned what is effective is to require that functional decomposition sessions take place facilitated by staff skilled in the process. People learn by doing.

We learned our toolkits needed to be updated to manage process updates. The Acclaro™ Toolkit [4] was revised to manage the following process issues:

- Management of Aggregate Constraints
- Management of Decomposition Node functional risk.
- Management of the engineering design testing plans (tests, status, traceability)
- Development of an FTA calculation for the Decomposition Tree
- Management of traditional Contractual Requirements and similar documents with full traceability into the Functional Decomposition
- Reporting of design risk mitigation status

Even the best design methods fail when applied to the wrong requirements. Garbage in, garbage out (GIGO) is an expression summarizing how flawed, or nonsense inputs produce nonsense outputs or "garbage" design. In both the spirit and practicality of implementing a Functional Decomposition design process, a design team needs to step up one level above the current design to analyse and understand the driving Functional Performance objectives. Otherwise Functional Decomposition is potentially producing better performing decoupled “Garbage” designs.

In the AD process, with the creation of the Operation Requirements document, our AD vocabulary changes from Needs to Requirements. However, we retained the use of Requirements in early document names, such as the Contractual Requirements, to simply avoid confusion with traditional practice.

We found it is not possible to separate the processes of requirements analysis from concept synthesis. This is primarily because higher level design decisions create lower level requirements that need further design analysis. Historically, organizations often chose to separate the requirements phase from engineering development, putting the requirements analysis function primarily in Marketing and assuming the Contractual Requirements represented 100% of the requirements analysis. In fact, Contractual Requirements are only a minor percentage of the requirements work remaining to be done for a good systems engineering effort. And the requirements work can only be completed in the presence of design analysis and trade decisions.

Additionally, we learned that marketing department driven Contractual Requirement statements are remarkably poor in defining functional requirements. In our experiences we noted that marketing driven requirements consist largely of features to catch up with the competition, marketing’s view of new features they perceive customers want to buy, marketing’s view of solutions to existing problems of current product and overly optimistic reliability and cost targets needed for management buy-in of the development budget. The majority of actual requirements, functional or otherwise, are assumed, and not explicitly defined.

We discovered, as discussed in section 2.2 Conflict with Success, working with business and marketing to change the requirement analysis paradigm is hard. The conversations would go like this:

> Marketing: “I don’t understand what you want; Just design the features we need.”
>
> Design: “We are looking at the problem from a system point of view and need to develop measures of functional performance for the entire system to best understand how to define the needed engineering performance requirements.”
>
> Marketing: “I have already suggested the solution you should use. If you find something better, fine, but I think you are making this too complicated.”
>
> Design: “Can we chat with some of our customers to understand better their objective measures of performance of our product.”
>
> Marketing: “I respect what you are trying to do, but there is not enough time. We are not going to bother our customers. We are not going to redesign the product. I have given you a budget and defined what we need to do, so now let’s get going…”

Even with historic first pass design yields being virtually zero, and not a single project delivering as initially scheduled, it was near impossible to change marketing behaviour, particularly if the company has been financially successful.

Furthermore, we have developed the opinion that it is not even useful to try and educate marketing and sales staff on the skills of requirements analysis. We feel that the most reasonable solution is to place functional requirements analysis processes in the engineering development team. The contribution of marketing should properly be considered the marketing needs input, certainly important, but simply one of many stakeholders.

We found early on that stakeholder needs and Contractual Requirements were insufficient to drive the Functional Decomposition. This drove the creation of the Operational Requirements document (discussed in detail later). Then how do we view the Contractual Requirements and other stakeholder needs? First, these are key inputs in the generation of Operational Requirements. Second, when completed, the Function Map is used to verify that stakeholder needs, Contractual Requirements as well as other input needs are met.

We found that there is a major difference between our Functional Decomposition and historic design processes. In the past, a development team would consider a marketing produced requirements document as the driving design document. With Functional Decomposition we introduce and design to the Operational Requirements and we merely verify against the more traditional Stakeholder Needs driven documents. With our Functional Decomposition approach we found many Stakeholder Needs were dismissed as irrelevant or even counterproductive to delivering on top level performance requirements. In prior design processes these needs would have been incorporated in the design. The magnitude of this difference cannot be overstated. It is a paradigm change. Bottom up "perceived" Stakeholder Needs design is driven by stakeholder experience. And when technology and business needs are changing rapidly, stakeholder experience is more often a handicap than an asset in defining and developing new solutions.

We learned that determining the Operational Requirements have proven to be a most difficult aspect. Stakeholders reliably fail to provide insight into the objective top level functions of the product under development. Instead, stakeholders discuss their needs from a framework of their experiences with current and prior solutions. From a Functional Decomposition process point of view, to determine the Operational Requirements, we had to invariably return to the business and marketing staff to figure out and define the development needs that had to be accomplished in the customer function space. Yet Business and Marketing staff, certain in their opinion of the required solution features, would battle their perceived waste of time to try and define what the solution actually needed to do.

We learned we needed Project Management, Design and Business Analysis skills and participation to complete the Operational Requirements and the subsequent upper level Functional Decomposition.

We found that all applications of Functional Decomposition need Operational Requirements. These are the AD top level functions. The text of Operational Requirements needs to be extensive to capture the performance requirements of a system. These are often statements with multiple paragraphs to define intent, and not simple single statement FRs that are common in axiomatic design papers.

We learned a technique to coach the development of Operational Requirements. Typically these requirements are key performance metrics of product cost, maintenance cost, reliability, and competitive performance functionality. To help staff envisage what top level functions are, we developed the term "Black Box" requirements. We tell staff to imagine they have multiple potential solutions for the marketing and business case requirements, all in black boxes. Staff is not allowed to see the solutions. To determine which of these solutions is best, staff can only ask questions or have the results of tests run on these solutions. What questions and tests would we ask to determine which of the black boxes has the best solution to achieve our business objectives? These questions and tests will correspond to key Operational Requirements.

We found that Functional Decomposition processes enable systematic risk assessment much earlier in the development process than any other process we had knowledge of. In our experience, previous systematic risk reduction processes were composed of table top reviews, design reviews, FMEAs and testing. Most of these traditional risk processes, by their very nature, were later stage exercises around design solution documentation. A Functional Decomposition tree enables a systematic risk review process during the concept phase around FR-DP nodes long before significant design effort has been invested.

We have found that experienced senior designers are often a handicap and a resistance to process change. The typical argument made by experienced staff against active participation is that this new process is clearly not required as proven by their personal past success. Functional Decomposition work is just introducing needless documentation and project delays. We found the only way to deal with this resistance is with +3 level management support. Without such support we would often observe that designers quickly generated their design, documentation and prototypes without systematic risk assessment or design margin analysis or verification to input stakeholder needs. Then, when the incompleteness of the design was established, usually during prototype evaluation, designs were patched, functionality dropped or problems ignored to stay on schedule. The +2 management, often bonus compensated by delivery schedule dates, readily permits delivery of substandard and incomplete product.

We learned, or rather concluded, there are two forms of relevant experience, environmental experience and solution experience. Environmental experience refers to understanding of the environment that the final solution under development needs to function in. This is very valuable, particularly in risk assessment processes. Solution experience refers to an understanding of how the design problem was solved in the past. Knowing prior solutions seemed to limit the ability of a team to identify or accept new superior solutions. Often their reasoning is that proven prior designs had less risk. Yet most staff would admit many new product problems came from past 'proven in use' designs failing under new conditions.

We found that defining good FRs was not difficult if a test driven approach is used. Historically, in AD literature, much is made of vocabulary and structure of forming FR statements. The better defined the FR statement was, the easier to understand the FR to derive the testing to demonstrate the FR is met by the DP. It is

much easier to skip the vocabulary exercise and just define the test.

We found that we needed six informational elements in an FR-DP node. These were 1) the FR, 2) the DP, 3) the FR measure and tolerance, 4) the definition of the test used to demonstrate the DP delivers on the FR measure within the tolerance range, 5) a DP risk assessment and 6) a unique FR-DP node identifier. This unique node identifier is not the node numbering system, which will be constantly changing for a give FR-DP pair during Functional Decomposition. As previously discussed, defining the DP verification test and the FR test measurement objective with tolerances effectively backward defines the FR as a function.

We learned a culture of understanding and practicing stakeholder needs analysis is important. Design teams in the concept phase need a business analysis focus with an understanding of the customer needs. In one project, a three month Stakeholders Needs assessment and Requirements Analysis uncovered that 7 of the 8 most important functional needs were absent from the Contractual Requirements. We theorize the reason for these kinds of gaps between true customer needs and the existing Contractual Requirements comes from a marketing culture of dependence on experience, re-use of prior solutions and a healthy dose of process laziness.

We discovered the formal source of customer needs, marketing, would know the current product problems we needed to fix, wish list features that customers had requested, and the features that competition had that we didn't. Marketing and development staff usually did not have a minimal understanding of how the system is used functionally by the customer, or other stakeholders, to meet their internal needs. Functional Decomposition processes reveal these information gaps and forces design engineering into the stakeholder needs analysis process to properly drive Functional Decomposition.

We found the DPs of the upper levels of decompositions are more "subsystem" placeholders than actual solutions (that can be linked to PVs). For example, we have an FR to slow down. The Target Measure might be a deceleration rate with tolerances. The DP acceptance test would be defined. The DP proposed could be "4 independent regenerative braking subsystems from 10mpg to max velocity, friction braking below effective regenerative braking, and friction braking supplement for emergency braking requests." This is a subsystem definition with performance decisions, but not having a detailed technical definition.

We found that Design Parameters (DPs) and Process Variables (PVs) do not feel like distinctly separate domains in our industrial applications. Rather, these are endpoints on an engineering development continuum. In a traditional AD representation of the design domains, initial conceptual solutions are proposed as a DP. Detailed product development work converts the DP to PVs. We found the proposed DP can often be defined directly as a final PV with no further development work required. Or other times the DP decision is just the beginning conceptualization of a multi-step process of development of PV solution during the implementation phase of the design V-model. Development and definition of DPs into PVs represents the bulk of the design effort. In summary we found that:

1. The PV domain didn't have much systematic value in the concept phase. In part, defining and completing the PV domain represents the bulk of the project design work, not the concept work.
2. PV concepts are often not relevant at the higher decomposition levels, where DPs represent more often architectural system and subsystem structures.
3. Practically, during the concept phase, the "PV domain" represented real world constraints on the implementation of selected DPs and contained the knowledge required to complete trade studies between alternative DPs. So a designer would "Visit the PV domain" to study the implementation risks of alternative DPs.
4. There was not a hierarchical zig-zag process between the DP and PV domains. We propose that DP-PV zig-zagging is a misconception in many axiomatic design writings. Instead the linking, if any exists, is one to one or many to one DP to PV (component or process) traceability. We still assume independence of PVs.
5. In the case of specific manufacturing process designs that might justify a zig-zag functional decomposition defining process decisions to implement a DP, we would re-purpose the FR and DP domains in a separate decomposition.

We have found we have virtually no new (from scratch) designs. Most industry design efforts are fixing or updating existing, highly constrained solutions. In part, we feel this is a self-imposed situation as we see many projects try to conserve or reuse "proven" designs from a perception that this practice minimizes development risk. We note that re-use creates coupling, so there is an inherent conflict between industry practice and axiomatic design independence goals.

We found that re-used solutions should be introduced as a DP in the Functional Decomposition. A characteristic of re-use is that typically, particularly with software, numerous independent FRs are all met by this DP 'block of code.' In such a case we would view the re-use risk as extremely high and systematically require design phase testing against the entire set of FRs serviced by the re-use DP.

We found over time that we should only apply Functional Decomposition when there was a structured Requirements Analysis effort. Early on in our efforts, design teams had a garbage-in garbage-out issue in that their requirements are poor, resulting in constant discovery of missed needs (requirements) during the project. This changing of requirements is typically referred to as "Requirements Creep" and explained by project staff as management's tendency to ask for more and more as the project goes on. We dispute this argument from our experiences. Our observations are that requirement creep is a direct consequence of poor requirements analysis. Requirements are not changing so much as they are being discovered late in the project.

As changing requirements blow the value proposition of any design process out of the water, we feel there is little advantage to introducing new design processes when not practicing rigorous requirements analysis. Functional Decomposition efforts fall flat as they reveal and try to deal with missed, poorly defined or incomplete requirements. Most of the toolkit functionality developed during our efforts was in the area of integrating needs analysis into the Functional Decomposition process.

We found that a traditional axiom 2 explanation of design versus process range analysis was not helpful. Axiom 2 conceptually represents an assessment of the systematic risk that the DP will (will not) deliver within the tolerances of the FR Measure. Axiom 2 does not seem to address random risks. The traditional academic textbook Axiom 2 representations feel simply like six-sigma based manufacturing process risk analysis. Instead we choose to use traditional risk assessment concepts to implement Axiom 2 and do not introduce the concept itself. We teach, using conventional reliability theory, that each FR-DP node needs an assessment of the systematic and random risk failure modes that might prevent the DP from delivering on the FR. How risk assessment is implemented varies between technical domains. Software risk assessment, for example, is different than hardware risk assessment. Traditional risk tools will apply. We implemented an FMEA format to capture these risks and their mitigations with our software toolkit.

We realized after a few projects that Function Maps were valuable largely in the early design and planning stages of a project. However, as the project progressed into the detailed engineering phase, and development staff started reacting to problems with testing and prototypes, Function Maps had decreasing value in design phase problem solving efforts. As such, once the major engineering effort began, we would effectively stop maintaining the Functional Decomposition unless there was a specific area of interest or perhaps a limited redesign effort.

5.2 Value Proposition Findings

We have found the following value propositions applying Function Decomposition process to system design and development as a concept phase process:

Thinking clearly in terms of solution neutral functional requirements rather than traditional features and solutions results in:

- Better requirements definition and clarity resulting in less project risk related to misunderstood, missing or creeping requirements

Function Maps enable excellent requirements and concept phase solution visualization resulting in:

- Increased collaboration and problem identification earlier in the development process which minimizes wasted time on substandard solutions and reduces design iterations.
- More complete hierarchical requirements definition that minimizes lower level design iterations due to unclear and un-reviewed subsystem requirements.

Driving the design top down from Operation Requirements and then verifying the solution against the input requirements results in:

- Functionally leaner designs simplifying the solution and lowering project and product costs

Functional Decomposition enables systematic Product Risk performance analysis structured by functions, resulting in:

- Better performance (less systematic and random risks) as significantly more performance risks are uncovered then testing could ever find.
- Better project delivery and cost metrics as more performance problems are discovered and dealt with systematically in the design phase, rather than appearing unplanned in test or deployment phases of the project.

We note that many AD proponents claim better design through design matrix analysis and decoupling, as an argument for process adoption. We did not observe this value proposition.

We found little overall value of systematic design matrix assessment in our industrial experience and dropped design matrix analysis from our process. Without prejudice as to this value in certain specific situations, why did we not see any value? In part, analyzing and maintaining a design matrix for systems of 500+ FRs that are constantly changing during the concept and early design phase is an intractable problem. Also, mentally practicing axiom 1 during design decomposition along with FR-DP test definition produces largely uncoupled design. Also, design matrix assessment is not reliable as systems evaluated as uncoupled during the concept phase end up being coupled anyway. Also, coupled systems pass the required tests and perform reliably. Finally, we find the level of competitive functional design so poor that designs are commercially competitive merely with sufficient functionality, not the optimal functionality of a fully decoupled design, making delivery, cost, feature set and other factors more important to achieving business goals.

The authors feel strongly that business strategy, process or software designs are more fertile ground than hardware design for applying Functional Decomposition processes. Yet typically staff in these areas is much less receptive to applying design process tools to their decision making. Management in particular, in our experience, resists these concepts. Convincing management that their business or operational strategies should be ‘designed’ and risk assessed is an incredibly difficult task.

6 Conclusions, Discussion and Recommendations

We found in our development projects that the value proposition of introducing Functional Decomposition processes was an increased first pass yield of designs better fit to deliver the Operational Requirements with increased functional reliability. This would also significantly reduce project costs and delays over the prior process(es).

We conclude, from our personal experiences and observations, Functional Decomposition processes applied during the concept phase and early design phase provide two primary underlying benefits to create this value proposition:

First, Functional Decomposition produces a more complete, internally consistent and mature set of system requirements verified against Contractual Requirements, Stakeholder Needs and other input requirements. This minimizes traditional requirements creep. This avoids errors of requirements omission.

Second, Functional Decomposition enables the earliest possible methodical approach to identification of systematic and random reliability failure modes. This permits a more complete mitigation of functional performance risks that would not traditionally be captured by verification or validation testing.

6.1 Discussion

In our experience virtually 100% of projects fail to deliver against the initial functionality, schedule and cost budget. This acceptance of such a high design process failure rate by industry is interesting. Common sense dictates that early processes will have the most impact on improving this project performance.

Yet we observe industry is only recently beginning to approach normalization of early stage design processes with Process Acceptance Models (PAMs). ISO/IEC 15504 is an example. Yet industry and PAMS focus on deliverables, which are called work products, and not the definition of the underlying processes themselves. This applies particularly to the design synthesis phase. Designers are left to their own devices and experiences, often ad-hoc, on how they develop the solutions to be documented by the work products.

We attribute this lack of interest in developing concept phase design processes to a couple of environmental variables. First, design teams in the concept phase are lulled into complacency as they have lots of time, lots of budget and lots of confidence in their abilities. Supporting these observations, we do not see design processes blamed for failures. In our experience, project failures are typically attributed to technical issues, insufficient time or inadequate design resources. Second, we see a pervasive belief that design, and in particular concept synthesis, is a “Creative” effort that defies structured processes.

Ironically, when projects result in commercial success, we observe that the development teams like to credit their creative and innovative design process skills for their success, even if the development processes technically failed to deliver as planned.

All of these conflicting observations lead us to conclude that it is common to have a lack of organizational process maturity in identifying, documenting and deploying concept phase design processes.

We also note that even with an AD Functional Decomposition framework the design processes to synthesize and propose individual DPs remain undefined and still equally difficult.

This recognition that AD operates a level above the bulk of the DP design synthesis work helps to highlight an interesting aspect of AD Functional Decomposition process implementations. We suggest it is more proper to state that Axiomatic Design is a technique to legitimately compartmentalize the hierarchical requirements architecture of a solution around functions using the independence axiom and functional decomposition. The design synthesis process, itself, as used to generate and select potential DPs, has not been systematized by AD. We invite comments on this observation.

The traditional definition of a quality process is repeatability. We suggest an interesting collaborative academic research project would be to have multiple student design classes independently take on the same design problem. This could be over time within a school, or over geography with different schools. Some efforts should apply AD. Others should be left to their traditional devices. The results should be compared. If AD has true elements of a process, then the AD design outcomes should be converging, and not be a scatter gram of solutions that we might predict with the less systematic non-AD process approaches.

Software design projects benefitted immensely from Function Decomposition processes. It is difficult for programmers to visualize performance requirements at the coding level from feature sets developed bottom up, by business analysts and architects, from stakeholder identified features. A top down functional decomposition with its inherent traceability of FRs makes the performance requirements of code modules very clear. The implementation of functional decomposition into an ISO/IEC 30003 SPICE PRM/PAM is underway.

This paper covers a lot of seminal ground. The topics within many of our brief paragraphs could be expanded into complete papers. Please contact the corresponding author for additional questions or discussion underlying the points made.

6.2 Discussion of Prior Process Reporting

As mentioned above, our paper is reporting on lessons learned from our industrial application. However we can contrast our findings against previous similar academic discussions around potential AD system design implementation processes.

Thompson discussed a requirements process to precede Functional Decomposition in great length [5]. She proposed a unique process for managing Stakeholder Needs analysis. We observe that adapting

existing conventional Stakeholder Needs analysis processes worked fine for us.

In addition, unlike the process Thompson suggested, we found no need for hierarchical zig-zag mapping within, between or around Stakeholder Needs and the FR domain. Primarily this is because we found Stakeholder Needs are not hierarchical. Nor did we have to create any new work products to supplement existing conventional Stakeholder Needs analysis techniques.

The significant difference in our process findings compared to prior practice was how we needed to use traditional Stakeholder's analysis to develop a discrete intermediate work product we called Operational Requirements which equates to AD top level FRs and driver of Functional Decomposition.

We also explicitly identified an early quality step of verifying the quality and completeness of the Function Map. We did this by traceability of QFD CTQs, Affinity List items and needs of critical stakeholders (such as management and customers) into the set of FRs of the Functional Decomposition Tree.

Brown and Henley discussed the structure of the FR-DP node and related process steps [6]. Our industrial practice findings were significantly different from the proposals of these authors.

First, interestingly, the authors seem to discuss that the design is the FRs. We suggest that in Functional Decomposition, the design, and the design freedom, is with the DPs. Solution risk, for example, is a characteristic linked to the DP. Constraints on the solution operate on DPs. FRs and their respected measures are the minimal but sufficient set of functions or sub-functions required to ensure the parent DP 'solutions' will deliver on its FR.

As expected and proposed by Brown and Henley, we also agree completeness and independence of the set of child FRs needed to implement a parent DP was a good working quality check of a Node decomposition. We did not find systematic Design Matrix independence checking critical to our value proposition. We attribute this tolerance of less than perfect independence to little need for optimal designs; sufficient designs are adequate. Coupled designs would easily pass required testing. Also, it is easy to see independence (or lack there-of) by examining any duplication of the DP verification tests without a design matrix.

Lastly, unlike the Brown and Henley proposal, we point out that we needed to supplement the FR-DP node with additional attributes of Unique IDs, FR measures, the DP verification test, and DP risks.

In particular, the DP verification test proved useful forcing designers to both understand and define the FR with one (or more) performance measures. This has the effect of eliminating onerous FR definition text. This test attribute is directly from practical test driven development concepts in industry.

Both systematic and random performance risk assessment at the Function Node level introduces Axiom 2 into the working Functional Decomposition process. This early systematic risk assessment is significant. Industrial projects do not reset when problems are found. Delivery target dates do not change easily. Rather than reset or refactor designs when problems are found, the design is usually patched and some functionality and manufacturability lost. Systematically identifying and mitigating risks earlier the design process made Axiom 2 concepts more valuable than Axiom 1 in most projects.

FR Measures were perhaps inherent in prior practice, but to drive good systematic DP selection thinking processes and reliability concepts at the concept phase we needed to make performance targets an explicit aspect of Functional Decomposition nodes.

6.3 Integration into PAMs

As mentioned earlier, Process Assessment Models (PAMs) are being used more often in industry to both define and assess good design processes. PAMs assess a process defined by a Process Reference Model (PRM). See ISO/IEC 33004 for a relevant discussion.

Functional Decomposition is just a small piece of the overall development process from needs elicitation to product retirement specified in these all-encompassing PRMs.

We propose that Functional Decomposition processes create one new process and two new work products that fit into a generic PRM under architectural engineering requirements analysis. They also similarly appear under engineering requirements analysis.

The new process is Functional Decomposition.

The first new work product is the Operational Requirements and Function Model. It will have elements of both traditional requirements analysis and design, in that design processes are required to establish lower level requirements.

The second new work product is the traceability between the FRs of the functional model Function Model back into the Customer Needs and Input Requirements and forward into implementation items. Whereas traditionally customer needs are considered requirements and traceable to design items and their verification steps, we propose that customer needs are traceable to functional requirements of the Function Model, which in turn can be traced to DPs, risks, and verification testing.

We also propose that risk assessment processes typically found in the Management Processes of the PRMs are extended forward in time into the requirements analysis process to take advantage of the Function Model framework's ability to support systematic risk assessment.

6.4 Recommendation

We recommend, particularly in the development of systems that have significant functional performance requirements, such as functional safety or high reliability systems, that design teams develop and implement a structured Functional Decomposition process during concept phase processes. The process focus should be on the benefits of the requirements analysis and process steps needed to create Function Maps and risk assessment and mitigation. The value is created by exercising the process, not by documenting the work

products. In practice, Functional Decomposition serves as an integrated requirements analysis, Function Model and risk management process.

References

1. R. Scupin, *The KJ Method: A Technique for Analyzing Data Derived from Japanese Ethnology*. Human Organization. 56. (1997)
2. L.K. Chan, M.L. Wu, *Quality function deployment: A literature review*, European Journal of Operational Research, Volume 143, Issue 3, Pages 463-497, (2002)
3. K.G. Jamieson, R.D. Nowak, *Active ranking using pairwise comparisons*. arXiv:1109.3701v1, (2011)
4. Acclaro Overview, Functional Specs Inc. Retrieved from Http://www.axiomaticdesign.com/products/default.asp (2018)
5. M.K. Thompson. *Improving the requirements process in Axiomatic Design Theory*. CIRP Annals - Manufacturing Technology, 62(62):115-118. (2013)
6. C. Brown, H. Henley, *Metrics for Developing Functional Requirements and Selecting Design Parameters in Axiomatic Design*. The 10th International Conference on Axiomatic Design, ICAD 2016. Procedia CIRP 53 113–118. (2016)

APPENDIX 1: Key Terms Defined

Affinity Lists: The categorized summary lists of Customer Needs produced by KJ Analysis

Aggregate Constraints: See Constraints.

Constraints: Constraints are derived from stakeholder needs that serve to limit the DP solutions that can be considered. (for example, "Product may not contain lead"). Constraints can be further defined into Input Constraints, Derived Constraints and Aggregate Constraints. Input Constraints are identified from stakeholder needs in the initial Requirements Analysis. Derived Constraints are Constraints created during the design process by DPs of the Functional Decomposition decision processes which become binding for the balance of the design effort. Aggregate Constraints represent either Input or Derived Constraints that can only be applied to total system aggregate measures such as weight or product cost. In application, Constraints are systematically checked to verify the acceptability of Design Parameters being considered.

Contractual Requirements: The contractual understanding between the funding source and the development team that exists prior to the start of development. Typically this is a statement of required business and customer needs. Although these are needs in an AD approach, we maintain the convention of calling this a Requirements document. Typically Contractual Requirements includes a list of project (cost, delivery, etc.) needs, proposed design solutions, lower level design features, and numerous Input Constraints of the system to be designed.

Customer: See Stakeholder. We prefer to reserve the term Customer for the specific category of stakeholders involved in the economic purchase of the design under development. But often in industry the term Customer is used interchangeable with Stakeholder.

Customer Needs: (AKA Needs) See Stakeholder Needs.

Customer Needs (CN) domain: This refers to the AD domain space of Stakeholder Needs that represents the gathering and analysis processes related to identifying and converting Stakeholder Needs and other inputs into Operational Requirements.

Decomposition: See Functional Decomposition

Decomposition Node: An FR-DP point on a numbered decomposition tree with 6 fields:
1. A unique node identifier
2. FR description
3. FR Measure
4. DP description
5. DP verification test
6. DP risk assessment

Decomposition Tree: (See Function Map)

Design Matrix (DM): A mapping of the dependencies between specific FRs and DPs when the FRs are mapped to the rows, and the DPs to the columns of an analysis grid. Used to expose and document FR to FR coupling by DPs.

Design Risk: See Performance Risk

Design Parameter (DP): The proposed conceptual design solution to deliver the Functional Requirement (FR) within target tolerances of the FR Measure. A FR-DP pair is defined completely with a DP verification test description and a risk assessment of systematic and random risks associated with the DP.

Design Parameter Attributes: Various measures of the physical characteristics of DPs that need to be tracked to manage an Aggregate Constraints of the total system. For example: weight, cost or power consumption.

Functional Decomposition process (AKA Decomposition, AKA Functional Analysis, AKA Functional Decomposition Performance Analysis): The process of synthesizing a top down hierarchical decomposition of a solution to the design under development in FR-DP pairs applying the rules of axiomatic design to develop a functionally independent, decoupled solution architecture that defines the FRs and DPs at all levels of the design.

Function Map: (AKA Function Tree, AKA Requirements Decomposition Tree, AKA Functional Model, AKA Functional Decomposition) A hierarchical top to bottom decomposition of requirements, starting from Operation Requirements, composed of Decomposition Nodes with parent and child relationships, consistent with Axiom 1 independence rules, is called a Function Map.

Functional Requirement (FR): A solution neutral description of the required functional performance. An FR is defined completely with a Description and a target Performance Measure with acceptable Tolerances.

Input Needs (AKA Input Requirements): The sum total of product needs from which Operational Requirements will be derived and against which the Function Map will be verified. These are composed of Contractual Requirements, Stakeholder Affinity Lists, prior product history and other design and process needs. In an AD perspective, these are Needs, but we will also use the traditional industry term of Input Requirements.

Needs: See Stakeholder Needs

Node: See Decomposition Node

Operational Requirements (AKA Top Level Functions, AKA System Performance Requirements): These are the objective requirements that summarize the expected and required business solution functional performance of the design under development in total absence of any definition of the solution. This defines functional performance requirements, which a subset of the total system requirements.

Performance Risk (AKA Functional Performance Risk, AKA Design Risk): An assessment of the risk that the selected DP will fail to deliver the FR Performance Measure? Systematic and random risks are analysed. Analysis is structured by failure modes in an FMEA format.

Performance Measure: The expected performance of the Functional Requirement. This will consist of a range of acceptable performance (Often a Target and a Tolerance). For example a requirement for weight control will have a target weight with acceptable tolerances.

Prioritization: The process of assigning an ordinal ranking of goodness to options under consideration. Prioritizing is used in trade analysis and QFD needs ranking.

Project Requirements: These are business requirements for the development process such as project costs, delivery dates and required review and approval cycles. These are usually contained within the Contractual Requirements.

Project Risk: Risks related to meeting project goals of delivering required business functionality on time and on budget. In contrast, Product Risk assesses the confidence at all levels of a proposed solution that the DPs will deliver on the FR Target Measure within required Tolerances.

Process Variable (PV) domain: This is referred to as the implementation or manufacturing design space associated with the implementation of a DP. A documented completed design has addressed all the "PV" issues of reliably replicating a solution to deliver on the Operational Requirements.

Requirements: See Functional Requirements

Requirements Analysis: A method of using process tools such as Kano, K-J Analysis, QFD and others to convert Customer Needs into a coherent de-conflicted set of Operational Requirements and Input Constraints.

Requirement Decomposition Trees: See Function Maps

Requirement Reference Numbers: (AKA Unique Node Identifiers) Unique identification numbers associated with FR-DP nodes. These are used in a practical tool to track nodes through edits and traceability.

Stakeholder (AKA Customer): The broad group of people, organizations, regulations, interfaces that place demands upon, and interact with the design under development from which needs for the design under development can be collected.

Stakeholder Needs (AKA Customer Needs, AKA Needs): A 'Voice of the Customer' collection of inputs on requirements (or features) of a potential solution that are essential or very important to stakeholders of the system being created. In Requirements Analysis, these needs are codified by KJ Analysis into summary need statements called Affinity Lists. After the development of Operations Requirements, Needs become Requirements in a vocabulary change.

Trade studies: the process of analysis to determine the preferred Design Parameter (DP) to select from a set of potential DPs

APPENDIX 2: Functional Decomposition process