Ambiguity and Uncertainty of Requirements in Product Development.

E. Lutters, F. van Houten
Laboratory of Design, Production and Management,
Faculty of Engineering Technology, University of Twente, Enschede, The Netherlands

Abstract
In design engineering there are considerable discrepancies between academic theory and industrial practice. As concerns the design methods and working methods, a considerable part of the inconsistencies is caused by the fact that in everyday practice, decision making is inherently and inevitably based on fully expressed information as well as on incomplete, unreliable and even faulty information. The complexity engendered by the myriad decisions that constitute any development cycle can be addressed by developing human creativity of product developers and by computer power for routine tasks. To address the fluent amalgamation of the two, first and foremost the content en context of the development task must be established adequately at the appropriate levels of detail and aggregation. The publication illustrates how a requirement specification can do this, and how this specification can become a living and evolving constituent of a development cycle.

Keywords
Product Development, Requirement Specification, Uncertainty

1 INTRODUCTION
Many academic publications about design methods and tools have been written, but the common problem is that hardly any of these methods are being successfully used in the actual design practice. As Ullman states [1], design methods can be classified as prescriptive, descriptive or computational. Most of the prescriptive design methods come from a long German tradition in methodical design and construction principles, while descriptive design methods come from the Anglo-Saxon design research community and are generally based on protocol studies. The computational design methods are based on a combination of geometric modelling (CAD), analysis tools (CAE) and Artificial Intelligence methods (e.g. feature recognition). Science based geometric modelling started as early as 1798 with the publication of a general method of applying geometry to problems of mechanical construction by Gaspard Monge [2]. His solutions for descriptive geometry and 2D (orthographic) projections formed the basis for the joint development of accurate machines and by that for industrial revolution. After the invention of the computer, many generations of CAD systems have been developed, ranging from electronic replacements of drawing boards to sophisticated PLM systems with advanced capabilities for modelling, analysis and data management. In all cases, however, such tools assume a bottom-up way of working, whereas design is essentially a top-down process. Therefore, designers have to decompose the design problem to the level of defining part geometry before the composition of the product modelling can start.

1.1 Design as a composition effort
All classical design methods more or less presume that a fixed set of requirements has to be converted into an optimal or at least feasible product/solution. Against that background, all subsequent design process steps should be taken. Decomposition is the usual way to reduce complexity, because smaller problems tend to be easier to solve. However, the risk of introducing new problems that might arise in the composition phase is eminent, because of ‘broken links’ or ‘incompatible solutions’. In order to avoid re-composition problems, multiple solutions can be elaborated for the decomposed sub-problems and morphological schemes can be useful to avoid dead ends. However, the disadvantage of this approach is that designers usually consider the elaboration of multiple solutions as inefficient. This leads to the common practice that a lot of conflict resolution has to be done in the composition phase and that compromises and corrective solutions have to be found in the final stages of the design process. This increases throughput time, cost and loss of intrinsic quality.

To reduce the need for human effort in generating possible solutions, it has become feasible to implement computer support for design synthesis tasks. Instead of exploring only one or very few possible solutions and trying to achieve consistency with the set of requirements, like in common design practice, computing power can be deployed to generate large solution sets that dynamically fit (changing) sets of requirements. Subsequent reduction of feasible solution sets by specification of additional constraints leads to smaller sets of solutions that can be compared by humans.
1.2 Design as a collaborative effort

For the design of complex products but even for simple products that have to be produced with complex production processes, teams of experts have to work together. Actually, the product design process is more and more considered to be a group activity in which communication and collaboration between human actors plays a central role [3]. In addition to designers, these human actors may include people such as production engineers, maintenance engineers, marketers, representatives from legal institutions and potential users of the proposed product. They have to understand each other’s frame of reference, aggregation level and terminology. Additionally, they need means to communicate and to get an agreement on the problem statement and possible solution spaces.

New products are designed to fulfill a certain task or activity, i.e. solve the signalled problem. However, introduction of the new product will in return influence the identified problem. New problems may arise as well as new opportunities may be identified. As [4] explains: 'Design is always indeterminate, because design changes the world within which people act and experience'. Therefore, requirements always change.

2 REQUIREMENT EXAMINATION

The common assumption that design starts from a fixed set of requirements is generally not correct. In the early stages of problem definition, it is more likely that there only is a broad understanding of the requirements. Sometimes, it is even very difficult to characterise the possible users and use cases of a product. Moreover, in both the problem definition phase as well in the solution phase of the design process there usually are multiple stakeholders.

As a consequence, the set of requirements to start with is nearly always incomplete and sometimes even inconsistent. In the specification of product requirements, even those of low complexity, designers have to cope with the fact that the set of requirements remains variable during the process. New requirements may arise and existing ones might change or be dropped.

To reduce risk, it is very important to spend ample attention to the problem definition phase, before entering the problem solution space. To begin with, this involves thorough multi-stakeholder objectives synchronisation. Subsequently, possible product use scenarios should be evaluated and alternative solution principles should be tested. Because the production of physical prototypes is generally costly and time consuming, preferably testing is done in as much as possible in virtual worlds. Virtual environments for the definition and management of requirements as well as virtual test beds are becoming feasible and affordable. They will support the design methods and tools for the future.

3 REQUIREMENT SPECIFICATION

In product development cycles, stakeholders constantly deliberate about alternative solutions for design problems. During this decision making process, requirement specifications serve as a reference for judging the available alternatives and, as such, determine to a large extends the course of development cycles. Requirement specifications exist in multiple forms, at different levels of detail and with changing degrees of certainty and ambiguity. In its most essential state, the requirement specification has a so-called stated purpose as its basis; a stated purpose is a predefined, formalized and static reference of the development process [5]. Therefore, the stated purpose reflects the pre-imposed requirements of (external) stakeholders, like law, marketing and safety. Due to its static nature, this type of reference hardly influences the selection and use of the different requirement types. In addition, three types of requirements are distinguished here (see section 3.1): technical specifications, functional specifications and scenario-based specifications. The merge of these different requirement specifications is in itself a dynamic frame of reference that evolves as the product is being defined. This provides a benchmark for assessing the current development cycle and prefacing subsequent development processes.

The development cycle can be seen as a concatenation of decisions, and as such, it is essentially an exercise in problem solving. In order to exploit this adequately, the underlying information content of the development cycle has to be determined explicitly in a structured and accessible manner [5]. As a consequence of this (changed) perception, the requirement specification need no longer be the result of the initiation/revision in some of the design stages. Instead, it can become one of the contributing entities to facilitate the entire development cycle. Here, the notion requirement specification is defined as the registration of the constraints, demands and wishes that are established to state the design problem and its envisaged solution.

3.1 References for the design process

When decisions are considered to be instigators of the development process, the nature of the available context for taking a decision becomes increasingly important. As the requirement specification serves as a frame of reference through the development cycle, it is an indispensable part of that context. Throughout the development cycle, the subject of the decisions will change with respect to level of detail, level of aggregation, considered domains, etc. To support a well-substantiated decision, the expression of the requirement specification must ally with the characteristics of the decision at hand. As a structure for the alliance, three types of requirement specification can be discerned:
• **Technical specifications** are complete and unequivocal expressions of product requirements. In general, technical specifications express quantitative or easily quantifiable demands.

• Functional specifications provide a description of desired future product behaviour. In general, they express concrete demands to abstract product models.

• **Scenario based specifications** depict a 'possible future', thus placing emphasis on the product’s environment and the interaction between product and that environment. Product behaviour is indicated in terms of what the environment, e.g. the user, can do with a product and how it will interact.

### 3.2 Appropriateness of specification types

When requirement specifications are viewed as an evolving frame of reference for decisions throughout the entire development cycle, insight in the appropriateness of the distinct specifications for different decision types is required for an efficient application of the requirements. The degree of completeness, unambiguity and certainty of the involved information content and the influence of context information in this respect are important.

#### 3.2.1 Technical specifications

Technical specifications describe the minimum set of unambiguous and quantitative information necessary for an equivocal definition of a product model. Therefore, they leave little (or no) room for interpretation. These characteristics make them extremely useful as a reference for deciding between fully defined design options. Consider e.g. the maximum wing span of aircrafts; the value of this parameter is restricted by the layout of platforms and terminals of airports to serve. However, it is also a design parameter influencing the payload and fuel efficiency but also many other geometry and performance parameters.

The simple numerical formulation that is characteristic for technical specifications is experienced either helpful or restrictive by designers, depending on the context of use. From the perspective of e.g. consistency a single straightforward interpretation of product properties is desirable. If development engineers are not aware of the context that imposed the formulation of the specification they may consider it as a restriction to their design freedom.

#### 3.2.2 Functional specifications

Functional specifications describe product behaviour in a qualitative manner. The realisation, however, is still an open question. Humans are uniquely qualified to deal with such non-deterministic decisions. The description of a desired behaviour pretends to give an objective perspective on the future. However, the description of a certain desire is subjective by definition and the future behaviour of products therefore cannot be objectified. Moreover, functional specifications principally are defined on their own. As Suh formulates it, functional requirements should be independent in order to prevent coupled designs [6]. In actual design practice, it is nearly impossible to identify all mutual dependencies of functional requirements. The set of requirements is usually incomplete and often inconsistent. Therefore, possible conflicts cannot always be revealed on beforehand.

#### 3.2.3 Scenario based product specification

In this context, scenarios are defined as explicit descriptions of hypothetical events concerning a product during a certain phase of its life cycle. This approach is useful when exploring less quantifiable influences on the product life cycle, such as the usage of a product. A scenario can depict what happens in a particular situation without committing to details of precisely how things happen. Furthermore, a scenario can be deliberately incomplete to help developers cope with uncertainty.

Scenario specifications provide information about the conditions and modes of deployments of the objects being designed. Scenarios are useful in managing the trade-offs among competing design goals as they allow making decisions and keeping open options simultaneously. Scenarios allow for exploring ideas and obtaining feedback, whereas they can be revised quickly and easily, helping to avoid premature commitment. Scenarios also allow for incorporating the influence of the environment on the functioning of a product and vice versa in the development process by explicitly foreseeing the co-evolution of the environment with product interactions. Scenarios therefore allow relating the requirements placed on the product to its environment, even if the environment is dynamic.

#### 3.3 Compound & dynamic requirement specification

The solution space in which product development processes are allowed to take place is determined by the frame of reference. The specifications constituting the frame serve as an argumentation and negotiation basis for taking design decisions. In many cases, the specifications will be clear and unambiguous. However, especially in the early stages of a development trajectory, they can be uncertain, incomplete and even contradicting. In order not to introduce feint certainties in the process, it is important to adequately represent specifications applicable to a certain decision.

When the frame of reference is constituted by evolving requirement specifications (see chapter 2), the relations between the specifications must be dynamic as well. The product definition will concurrently evolve on different levels of aggregation, instigated by different viewpoints and with respect to different aspects. For the requirement specification to serve as a reference in the entire solution space, it must therefore be possible to interrelate the information constituting the different specification types.
3.4 Aggregation of requirement specifications
Given the depictions of the stated purpose, the technical, functional and scenario based requirement specifications, their mutual relations and proportions can be identified.

First of all, from stated purpose to scenario based specification, the rigour and expressiveness decrease. This takes shape in the structure and transparency of the stated purpose and technical specification versus the more narrative character of scenario based specifications. The latter gives essential, yet less prescribing directives. In other words, the more quantified a requirement specification is, the less room it leaves for interpretation and haggling.

For many product developers it is a significant challenge to translate scenarios via functional specifications into technical specifications. This is caused by the fact that the translation of any specification into a specification that is more univocal and quantifiable – by definition - entails subjectivity. This subjectivity stems both from the way in which requirements are met and from the prioritisation in which requirements to meet. As such, this translation is a kind of convergence process that is executed and directed by the same stakeholders, being the product development team. Consequently, the work of a product development team is not only aimed at problem solving, it emphatically also requires adequate ways of capturing, framing, converting and embedding the contingencies involved in selecting the individual (values of) requirement specifications to consider. In other words, development teams need to contextualise the requirement specifications against the uncertainties they presuppose and encounter.

4 INDEFINITE SPECIFICATIONS
In a sense, product developers unremittingly struggle to gain the most adequate insight and underpinning before tackling the multiplicity of decisions that separates them from the envisaged product definition. There are many reasons that a design team cannot make a certain decision. The most obvious is a sheer lack of argumentation (i.e. information). However, as important is the fact that design teams establish requirements while attempting to fulfil them at the same time (see also 3.4). This leads to indefiniteness in requirement specifications; causes for this are summarised in ‘VUCA’. Although the acronym stems from a completely different (military) background [7], and it’s use is biased towards strategic management, volatility, uncertainty, complexity and ambiguity together adequately cover the indefiniteness experienced. Volatility and complexity address the inherent dynamics of change and intricate arrangements in a project respectively. Because for product developers these two causes are extremely difficult to change, here the focus is on uncertainty and ambiguity.

4.1 Uncertainty
Uncertainty is a situation where the current state of knowledge is such that the consequences, extent or magnitude of circumstances, conditions or events are unpredictable - or credible probabilities to possible outcomes cannot be assigned. Although too much uncertainty is undesirable, manageable uncertainty provides the freedom to make creative decisions. This directly relates to how the working methods change in development cycles, ranging from the fuzzy-front-end to the nitty-gritties of the near-production stages. Interestingly enough, this coincides with the observed need for quantification when going from scenarios via functional specifications to technical specifications, and the convergence while doing that.

Precondition for this to be true is that the uncertainty is actually manageable. In other words, it must be credible that the uncertainty is ‘closed’ in the sense that it concerns a lack of knowledge of facts that are indeed obtainable. That is, a designer may be uncertain about whether a new design will work, but this uncertainty can be removed with further analysis and experimentation. However, the latter is often hampered by the limited time/resources available to product developers. One ‘escape' that is applied in engineering is the so-called ‘design by least commitment', which actually accepts the fact that decisive knowledge is not available – yet. At the same time, postponing decisions may be the cause of more vagueness in the overall development cycle.

Provided that capacity is accessible to address the issue at hand, decreasing or removing uncertainties is often a matter of routine work, by generating and evaluating alternatives. In terms of requirement specifications, this stresses the relation between uncertainties and the application area of especially technical specifications. Both address solution spaces that are confined, where more information can lead to better, more accurate or more underpinned solutions.

As mentioned (see section 1.1), computer power can be used to achieve more synthesis in design processes. In less abstract terms, computers can be used to perform routine tasks. In product design, the essence of routine tasks is that the constraints and variables/parameters are well-known. Consequently, computer power can be used adequately in situations where the possible solution space is limited, but can be large. This solution space inherently meets the requirement of ‘closed uncertainty’ above, depicting a sheer need for structured, procedural and meticulous effort to decrease uncertainties. As this way of working hardly allows for subjectivity, it is apparent that mainly the translation of functional specifications into technical specifications relates to reduction of uncertainty. In this, computer support can play an effective and efficient role in the collection and evaluation of ‘obtainable facts’.
4.2 Ambiguity

From a traditional solution-oriented perspective, engineers tend to courageously wage war against the uncertainties they perceive, whilst purposefully distributing part of the available resources/time over the uncertainties that call for elucidation. However, more often than not, uncertainties are encountered that can not be dispelled, irrespective of the amount of information available. This is the case for situations where it is not obvious or predictable which entities/uncertainties play a role. This type of uncertainty is referred to as ambiguity.

Mathematically speaking, ambiguity is the state of being unsure of the type of (statistical) distributions related to an uncertainty. In product development, however, ambiguity relates much more to the origin of uncertainty; it addresses the inherent insecurities of e.g., the subjectivity with which certain aspects in a design (are assumed to) prevail over others.

An important example in product design is ‘the voice of the customer’. As all market research builds on sets of hypotheses, the research in itself will influence the answers. Moreover, because answers are quantified against that set of hypotheses, this type of research usually is about finding indicative solution directions. Therefore, using it as a means to reduce uncertainty may render ostensible certainty.

Ambiguity is sometimes described as ‘second order uncertainty’, where there is uncertainty even about the definitions of uncertain states or outcomes. The difference here is that this ‘second order uncertainty’ is about the human definitions and concepts, not an objective fact of nature.

Given the engineering approach towards uncertainty (see 4.1), product developers inherently attempt to capture ambiguity and reduce it into uncertainty. The reason for this is that designers, but mainly engineers think they have the adequate tools to fight uncertainty. However, ambiguity is inherently different from uncertainty in the sense that uncertainty aims at answering a question where ambiguity is about determining what the question is.

Consequently, ambiguity hardly occurs in the translation of functional specifications into technical specifications. If at all, it generally is the result of an omission in earlier phases of the development process. Ambiguity relates much more to the hypothetical future of the product that is designed (see section 3.2.3). As such, ambiguity is involved in translating scenario based specifications into functional specifications. Here, it is about understanding what the real issues are in the development cycle instead of solving them.

Figure 1 depicts the relation between the different types of requirement specification and the relation with uncertainty and ambiguity.

5 NON-DETERMINISTIC CHARACTERISTICS OF DESIGN

From the previous sections it is clear that requirement specifications involve two dissimilar issues: a search for better/more answers and a search for better/appropriate questions.

Traditionally, engineering approaches are assumed to be deterministic in nature; identical questions will result in identical answers. As such, a deterministic system is a conceptual model that renders its outcome completely on causality. In a deterministic system, every action, or cause, produces a reaction, or effect, and every reaction, in turn, becomes the cause of subsequent reactions. The totality of these cascading events can theoretically show exactly how the system will exist at any moment in time.

However, scenario based specifications, due to the inherent underlying ambiguity, withdraw from the deterministic characteristics. As scenarios include the subjectivity and variety of viewpoints introduced by the various stakeholders, there is no reason to assume that design cycles adhere to concrete and definite causality.

Even uncertainty causes some indeterminism: if, for example, computer power is used to perform routine work, the identical question may get different answers over time, simply because more or better information to answer them is available.
6 VALUING UNCERTAINTY AND AMBIGUITY

In appreciating the role and impact of uncertainty and ambiguity in product development cycles, both sources of contingency can ‘escape’ from the traditional engineers’ urge to nullify them. Instead, they can become valuable means to understand e.g. the design process, the project at hand and the stakeholders involved. Moreover, they allow for the purposeful exploration and exploitation of the solution space that evolves with the design cycle.

Different stages in the design cycle call for different approaches. One way of addressing ambiguities focuses, e.g., on scenarios in so-called synthetic environments [8]. These bring together all influences on product development cycles, while achieving synthesis between information, resources and control mechanisms to reach adequate solutions.

For functional specifications, as the pivot point between ambiguity and uncertainty, addressing design rationale or decision support is very helpful. Moreover, working methods like ‘what-if’ design [9] can facilitate the transfer of functional specifications.

More focused on technical specifications, computational synthesis tools [10] aim at employing algorithmic procedures to automate the generation of designs. This is done by combining “low-level” building blocks in such a manner that “high level” functionalities can be achieved.

Next to these few examples, there are many tools and techniques that enable product designers to get a hold on indefinite requirements, provided these designers accept uncertainty and ambiguity as an inherent and practical part of development cycles.

7 CONCLUDING REMARKS

In the long history of product design, engineers traditionally mitigated the complexities of their work by aiming at reduction of the number of variables. Over time, this has become an apt approach to maintain overview of the development cycle. However, with the increasing complexity of products, use contexts and manufacturing environments, sheer annulment or enfeeblement of indefinitenesses is not sufficient. Product developers need to understand the nature, scope and purport of uncertainty and ambiguity in order to recognise and direct their influences on development cycles. With this ability, product developers can purposefully select tools and techniques to influence development cycles in the most favourable manner.

From the perspective of uncertainty and ambiguity, many existing tools and techniques can together produce a (hidden) coherence. This does not make product design a deterministic process; more important, it stresses indeterministic characteristics to allow product designers to fully exploit their craftsmanship in establishing innovative concepts in large solution spaces, while simultaneously employing computer power to address routine tasks.

8 REFERENCES


9 BIOGRAPHY

Eric Lutters obtained his PhD degree in Mechanical Engineering at the University of Twente, The Netherlands. Currently he holds a position as an associate professor in the faculty of Engineering Technology at the same university. He also is a professor extraordinary at the department Industrial Engineering at Stellenbosch University.

Fred. van Houten is the head of the research group "Design, Production and Management" at the University of Twente. He is active in a wide area of research in Design Integration and Process Modelling. Fred is member of many renowned international organisations and academies.