INTERNATIONAL DESIGN CONFERENCE - DESIGN 2004
Dubrovnik, May 18 - 21, 2004.

DESIGN 2004

PRODUCT - DEVELOPMENT COMPLEXITY METRICS: A FRAMEWORK FOR PROACTIVE-DFA IMPLEMENTATION

C. Rodríguez-Toro, G. Jared and K. Swift

Keywords: complexity metrics, design complexity, complexity management, proactive DFA, decision-making theory

1. Introduction

The term "sandpit" has recently gained currency as a term for any environment oriented to experimentation and learning. However, despite an inherent ethos of innovation through experimentation and learning, designers currently have little to rely on other than their own "mental sandpit". Rather they are provided with CAD (a misnomer in itself), which only serves to record the geometry of a final design, together with analysis tools which allow for fine adjustments and CAM tools which can generate manufacturing instructions. What is desperately needed is an environment for decision making at an earlier stage of design, when it is possible to have a much greater effect on the outcome, together with tools which support "strategic" decisions rather than those present ones that address "tactical" issues. Such an environment is currently under development in the Designers' Sandpit project [Designers' Sandpit Project 2003]

It is widely accepted that it is much more cost-effective to produce an initial design that is simple, feasible and assembly-efficient, than it is to rectify design problems after the product has reached the shop floor. Similarly, it is essential to get a good concept before putting in the effort to make it a fully developed design. The Designers' Sandpit aims to enable this by allowing comparison of radically different designs rather than fine tuning of a single option. It will allow the designer to focus on function, product structure and higher level design evaluation metrics, but also provides for exploration of detailed issues – possibly even through to CAD / analysis – within the same context. In other words it enables iteration and parallel development of competing ideas – hence the Sandpit title. Clearly, one key tool within such a sandpit will be the evaluation of product complexity at various stages of development of a design. The route to providing this type of tool is the main focus of this paper.

2. Background

Concepts and measures of complexity have been increasingly reported in journals, conferences, books and on the Internet in recent years. It has been considered in disciplines including psychology, physics, management, biological and information sciences, to name a few. This is but a small sample of the enormous diversity of considerations given to the concept of complexity. Authors such as Edmonds [Edmonds 2000] have carefully compiled some of the definitions and considerations tried out in several areas. He particularly emphasises that generalisations are necessarily abstract and, more specifically, that estimations of complexity heavily depend on the context in which they are used. In the context of engineering complexity plays a major role in activities such as assembly and manufacturing and these both desperately require techniques to manage it.

There is a risk of becoming too philosophical about the concept of complexity and losing oneself in definitions, without actually settling on a proper meaning or circumstance of application. This is why some initial descriptions of various types of complexity, in this limited domain of engineering, were highlighted and reported in previous work [Rodriguez-Toro, Tate et al. 2002]. The authors are well aware of the risk of creating a model that is mathematically viable but may not reflect reality because of the amount of assumptions made during its creation. The idea, nevertheless, is that of limiting the scope of what is considered complex, without creating rigid boundaries. In other words, defining the scope of specific considerations of complexity creates a distinct situation that makes the concept more approachable and, perhaps, even practical.

This paper focuses on the evaluation of product architecture in a proactive manner in the context of an assembly-oriented design environment. The aim of such an environment is to present the designer with valuable information throughout the design process so as to encourage the production of an efficient design in the first instance. There is, however, no direct attempt to direct the designer as to, for example, which product structure is the most convenient, it is he/she who is ultimately responsible for making such choices. The intention is, however, the automation of certain repetitive tasks, as well as keeping track of changes that occur during the design process. This tracking allows the handling of a large number of emerging and possibly contradictory options, thus sparing the designer from having to deal with them. This should be an automated process, preferably carried out in the background, continuously empowering the designer, instead of holding him back by adding extra burdens to his / her duties. The implementation of this necessitates the creation of a framework of complexity metrics that will ultimately assist the designer during the product development process.

Managing factors or sources of complexity can help to reduce the 'design effort' [Bashir and Thomson 2001], thus shortening development time and cutting project costs. Such benefits have been previously offered by existing evaluation tools like 'Design for Assembly'. Furthermore, there are a whole range of similar 'design-for-X' methodologies, but these recommend scoring systems that, so far, have only been applied to finished product structures. This late use of tools has the obvious negative aspect of design time being misdirected, since at that point the product will have been outlined and be waiting to be validated. There are no reasons to believe that the designer would happily apply such tools since a significant amount of resources would have already been committed.

The following sections will present the theory necessary to achieve the objectives of product complexity evaluation through a proactive-DFA implementation.

3. Theoretical considerations

Firstly, it is worth mentioning how complexity is to be considered here. The aim is to measure complexity, thus it should be described or defined first. Such definitions tend to fall into one of two major schools of thought, namely:

(a) *Structural-based definition*: Complexity considered as a property of the object (system, product, etc.), which depends on the number of parts, part interaction, possibilities in their interactions and other characteristics.

(b) *Information-based definition*: Complexity considered as a property of the subject (observer), that is, it depends on the description of the system given by the observer, specifically the language used and the number of different descriptions given [Casti 1992].

Although it is difficult to ascertain which of these is more appropriate to this application, the first option – complexity as an intrinsic property of the system – has been followed. It is probably the most suitable and is, as might be expected, less subjective to measure, thus making the control of the complexity of a product also less subjective. Having chosen this path, there are further questions that must be taken into addressed. *Is the complexity of the product something required or is to be avoided? And if it is to be avoided, is it entirely avoidable*? The rationale behind these questions is the fact that complexity has been related to 'design effort', which has in itself been linked to design duration and number of resources consumed.

From an exploration of these questions, it appears that complexity is not only unavoidable, but actually required! Hence the evident need for a management technique. The difference between a beneficial type of complexity and another, which is not, has been stated by Tang and Salminen [Tang and Salminen 2001] and Suh [Suh 1999]. The latter devised a theory that proposes that complexity is the vector sum of complexities, *imaginary* and *real complexity*. Imaginary complexity is nothing but a lack of understanding of the product itself. This type of complexity diverts the designer's endeavour, blurring his view of the main goal. Real complexity, by contrast, is regarded as the uncertainty in fulfilling the functional requirements of a design – a concept that will be revised later on in this paper. Conversely, the first two authors write about complexity in terms of "an inherent and beneficial property, as long as it reduces the complicatedness of the system." This has several interesting implications, because it identifies complexity as a property of the product, which can be taken as a characteristic of its composition. Thus, any attempt to increase the level of complexity will ultimately increase the complicatedness of the product instead. The remaining question is, *how can such distinctive complexity be identified?* If what is considered complex could be detected, then it could be measured, and consequently it could be used to make comparisons between two structures of the same product. Comparing the complexity of different structures for their complexity will reveal the structure that requires less design effort, hence lower costs.

To end this section, some of the vocabulary introduced by Michael Behe in his confrontational book "Darwin's Black Box: The Biochemical Challenge to Evolution" will be considered. He introduces the concept of "irreducible complexity" to describe systems (biological or man-made) whereby the removal of any one of the parts causes the system to effectively cease functioning. In the context of this paper, it can only be said that, the book defies the findings of evolutionists as it introduces the idea of 'intelligent design' as the only way to achieve systems of 'irreducible complexity' in biological systems, leaving no room for the creation of systems formed by numerous, successive and slight modifications. The book has certainly caused controversy and generated passionate discussions that can be followed all over the Internet. However, it has also defined terms that can be useful in engineering (man-made systems), and these are ones that fit in well with the theory presented here, partly helping to define the bottom of the scale in a complexity measure scheme.

3.1 Surrounding constraints and design-for-all tradeoffs

As mentioned above, the intricacy of a product can be measured, controlled and reported once its environment and surroundings have been defined. This environment comes from the construction of an idealised world, which is gradually modified to account for the more complex world of experience. It is composed of several levels of abstraction of the product, thought of as being surrounded and in continuous interaction with tooling, handling and manufacturing processes, as well as other products. All of these demand something from the product and, consequently, add extra variables that need be considered during the design process. When the boundaries of the product have been established, a set of factors are given emphasis as being the sources of complexity. Such list has been compiled primarily by reviewing DFA and DFM methodologies. These two methods can produce contradictory information, as the first considers assembly time, performance and integration, whereas the second one is concerned with ease of manufacturing, part simplicity and variety. Striking a balance between sources of complexity after following DFA/DFM methods needs to be placed in context. Such context could be explained within a hierarchical structure that would help in evaluation and monitoring of changes in the product architecture.

4. Complexity taxonomy for assemblability and manufacturability analysis

It has proved hard to put in context the notion of complexity, for there is a generally held view which may be expressed as: "no-one has ever succeeded in giving a definition of complexity which is meaningful enough to enable one to measure exactly how complex a system is". If the goal were that of producing a single value that would convey the idea of complexity, then there would be, indeed, no way to know exactly how complex that system is. Products cannot and should not be reduced to one single complexity metric. Products are not only the end result or solution to a given opportunity; but also the source of an entire system of manufacturing, transport and economic evolution.

Therefore, if instead of a suitable definition, that accounts for all that is known to produce complex systems, a description of it were achieved, then complexity could be measured, monitored and mapped into what complex products will generate. A classification of the concept, such as the one depicted in Figure 1, provides the framework necessary to monitor sources of complexity. Factors that can be traced and recorded, thus keeping historical data of the design process, will help in understanding the process itself. Recording historical 'know-how' is not at all new, it has been thoroughly reported [Hatch and Tsoukas 1997]. It has been thought of as a means of considering the different bifurcations encountered in (design) processes, whereby the interpretation of point C implies knowledge of the history of the product in the system, which had to go through points A and B. Furthermore, when this kind of data has been recorded and plotted as the development process evolves, it is easier to spot the change that originated an increase in the overall complexity rating.

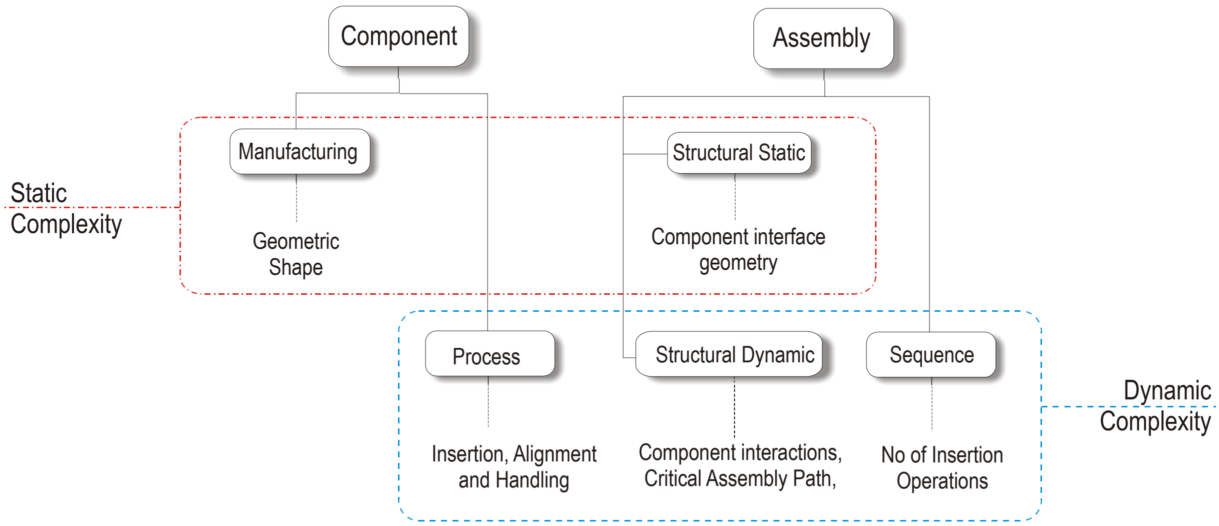


Figure 1. Complexity taxonomy for proactive DFA

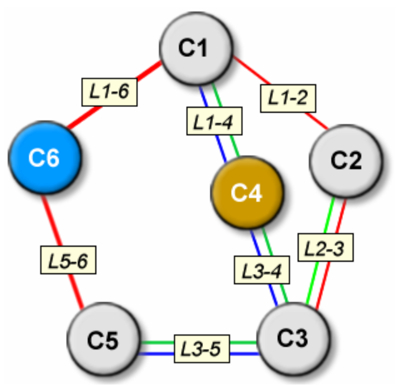
The hierarchical decomposition of complexity portrayed above has been considered in terms of DFA and DFM methodologies. This framework has been further divided into two levels: component and assembly complexity. The '*static vs. dynamic*' view, on the other hand, appeals as a much more abstract notion altogether, but it is perhaps this intangible condition that makes it more useful for a mathematical representation of the complexity evaluation methodology. Static complexity can be seen as a snapshot of the product, whereas dynamic is time dependent and related to the actual process of building up the product. There is not space here to go into detail about the complexity of components, which has been studied extensively and scoring systems proposed.

4.1 Assembly complexity

Assembly has been properly considered as the activity that actually creates the product. When products are assembled, activities such as design, manufacturing, manipulation and logistics converge [Whitney, Mantripragada et al. 1999]. Assembly is also the activity that dictates most of the complexity of the product itself. Complexity of the product is, accordingly, more than the sum of the complexity of the components, this being a holistic approach to product complexity identification. Moreover, most of the aspects to be considered during assembly can be classified into two groups, namely:

1. *Structural/architectural* complexity, where the configuration of a product in terms of its structure is not addressed by current DFA analyses, other than to eliminate non-functional parts wherever possible. This category includes the number of components, number and type of interactions, type of components, subdivisions (subassemblies) and parallel processing, amongst other things.
2. *Sequence complexity* is one of the most studied areas when dealing with assembly. It has produced a large amount of information, which can be used to extract valid metrics or means of detecting sources of complexity. This type of complexity mainly addresses the prediction of the most convenient assembly paths. Such predictions imply a certain number of possibilities that are directly influenced by decisions made at the design stage. As expected, they heavily depend on the architecture of the product, whereby adding new components, new possibilities of assembly are presented; this creates continuous bifurcations in the sequence analysis. Bifurcations are not only present in assembly sequence, for they actually populate the whole product – as already mentioned.

The first category has been the least studied of the two – or at least, it has not been considered under the title of structural complexity, as it is here. The product architecture is developed as the designer formalises concepts into components and systems, whereupon integrating and linking those components, product structures are generated and later on re-considered as the designer explores various configurations. Every arrangement or architecture retains specific 'signatures' that will pass on valuable information such as: component connectivity, number and type of connections, type of interfaces, as well as the types of components, to name just a few.



(a)

Name / Component ID		1	2	3	4	5	6	7	8	9	10	a	b	c	d	e	
Component 01	1	1	b										1				1
Component 02	2	b	2	b	a	c						1	2	1			4
Component 03	3		b	3	b	e							2			1	3
Component 04	4		a	b	4	a						2	1				3
Component 05	5		c	e	a	5	a	a	d			3		1	1	1	6
Component 06	6					a	6		a	a	a	4					4
Component 07	7					a		7	a	a		3					3
Component 08	8					d	a	a	8		d	2			2		4
Component 09	9						a	a		9	d	2			1		3
Component 10	10						a		d	d	10	1			2		3
Geometric Docking	a		1		2	3	4	3	2	2	1	18					
Geometric Coupling	b	1	2	2	1								6				
Contact: Tribocontact	c		1			1								2			
Contact: Constraint	d					1			2	1	2				6		
Contact: Sporadic	e			1		1										2	
Total No of Interactions		1	4	3	3	6	4	3	4	3	3						34

(b)

Figure 2. Interaction graph and matrix

The interaction of components has already been studied - it even has a methodology solely dedicated to its analysis, as in the case of the 'Design Structure Matrix (DSM)' [Eppinger, Whitney et al. 1994]. However, the interaction matrix used in this paper differs from that of the DSM methodology in several important aspects. It is based on the view that accounting for the amount of interactions alone is not enough, as there are many different ways of interacting. The various types of interactions are referred to as interfaces – also called "assembly features" by some authors, although similar, they are not intended to represent the same notion. Interfaces can be further sub-classified and, therefore, assigned weighting using data such as: tolerances, geometry of the interface, kinematic constraints and even tribological information.

Graph and matrix-based representations are the two most commonly used ways of symbolizing component interaction (Figure 2 – *shown for illustration purposes only*). Whatever the method used, every binary interaction can be classified within one or more of the groups presented in Table 1.

Table 1. General classification of interfaces

Classification	Description	Involves
Geometric docking	Perfect male-female match. No extra components needed.	Geometry
Geometric coupling	Male-female match, a third component is required to hold them together.	Geometry
Tribocontact	Matching components with relative movement	Tribology
Constraint	Not necessarily matching. Permanently restraining each others movement.	
Sporadic	Not necessarily matching. Sporadic or coincidental constraint.	

Although the groups presented above must be further refined, they are used to add weighting to the interactions encountered in every product. This characterisation of interfaces requires ranking kinematic constraints (lower and higher pairs) and tolerance values (the higher the surface finish, the higher the 'penalties') amid other measurable features.

Adding a weighting to the interface will help to distinguish between a product with large number of interactions and clear-cut interfaces, and another product with a lower number of more intricate ones (hence, greater interface weighting).

*Interface weight = Geom. shape * adjacency type * kinematic constraint * (tol. value)*$^{-1}$ ** other values* (1)

Equation 1 is a rough and generalised, mathematical representation of the intended plan. Due to space restrictions in this paper, it will not be further developed, but it will be presented as part of ongoing work. The reader should by now have an idea of how an abstract concept, such as the complexity of a product, can be quantified. Detecting sources of complexity, which are generally linked to measurable quantities, can be used to produce more objective and quantifiable metrics of complexity. This is, nevertheless, one of the several variables that can be recorded as part of a quantification of complexity.

4.1.1 Periodicity – subassemblies and arrays

Products do not normally have all-unique components. Basic arrays of fittings (e.g. the universally discouraged bolt-washer-nut array), more often than not, populate product structures in such a way that they have a propensity to create patterns. The discussion of pattern formation, in this paper, falls into the domain of 'patterns of interaction in a product.' [Eppinger and Salminen 2001]

Any measure of complexity which is proportional to the number of interactions will be sent "off-scale" because of the formation of patterns of interaction. Their appearance in the product structure will ultimately be reflected an overstated level of complexity. This can be corrected by detecting the template that created such pattern, which in itself has a certain level of complexity due, primarily, to its own interactions. Once revealed, any other similar occurring instances will have the advantage of a known procedural style, thus reducing the complexity level.

As expressed above, structural and information-related complexity are two different ways of looking at the same thing. Periodicity, seen as a matter of information theory, can be regarded as a way to economise information in the assembly process. The assembly process is composed of a set of instructions for putting components together. Instructions such as component position and orientation, next component, etc. are commonly found in sequence analysis and procedures. Information regarding neighbourhood, interfaces and interactions are commonly found in architectural decomposition of the assembly (product). Considering periodicity in assembly complexity makes it possible to distinguish between the amount of instructions needed for one component, also regarded as *specific component information*, and the amount of instructions needed for recurrent components. For a component used once in an assembly, instructions are created and used once, whereas for recurrent components, assembly instructions are produced and recycled, with the obvious benefits.

4.1.2 Standardisation

A logical consequence of highlighting the periodicity of components in assemblies is that, it represents a lower product complexity. A characteristic of the product, considered constant for a given product structure, can be actually lowered by modifying the type of components, yet keeping the same number of them. Standardisation, as a means of reducing complexity and component variants, actually boosts the manufacturability of the product itself. It also increases the chances of automated assembly, for it presents a repeated mode of assembly.

5. Concluding remarks

One of the aims of drawing up complexity metrics is the elimination of subjective estimations and hence detection of which product architecture is the most cost effective. Such metrics will not only help to control the design process, as they would constantly draw attention to critical changes, but they will also help to compare two product architectures. Such comparisons will ultimately point out which architecture requires less design effort and lower production costs.

Notice that the term *product* and *system* have been used indiscriminately throughout this paper. Acknowledging that, ‘a product’ can be regarded as ‘a system’ accounts for the variations in considerations given to the term ‘part.’ ‘A part’ could be ‘a component’ or ‘a sub-assembly’. Take for instance ‘a car door’ (with all the components within), which for some designers/manufacturers can be taken as ‘one part’, it can be also thought of as a ‘sub-assembly’, whereas ‘a handle’, from the same door, could be considered as ‘one part’ belonging to ‘a door’, hence taken as ‘a component.’
Nevertheless, a product, unlike a part, can be considered as the nucleus of the system. So it does not matter much if a product *is* a system or it *belongs to* a system, in this case, what really matters is that the evaluation of a product will raise an interesting question: Is the product leading the complexity of the production system? Or is there a mutual constraint? The reason for this question, amongst others, is the need to estimate the time and effort required to manufacture the product, as well as the costs involved in building and planning a whole process around it.

The paper has presented a framework for the estimation of complexity metrics within a proactive implementation of “design for …” methodologies. It has mainly focused on metrics for the detailed design stage of the product, where at least a first approach to the product architecture has already been considered. Such metrics are intended for use in conjunction with those drawn up for the different product stages (concept design, detailed design, construction and system use). Although the concept of complexity implies non-linear estimations, it is known that recording historical data will serve as the ‘know-how’ for predicting the repercussions for product construction and use, as the product architecture changes throughout the design process. Unfortunately, due to space restrictions, many issues cannot be elaborated on here - these include a more detailed description of component interfaces and interactions, evaluation of periodicity in assembly and the relation between product and system complexity.

The authors are aware that there are still shortfalls in the analysis proposed, for example, it has not given information about the functional decomposition of the product. Yet it has presented one aspect of the metrics required for complexity analysis. The assessment of complexity as such cannot be reduced to a single number. The very nature of the concept requires a multi-dimensional estimation that needs to be complemented with other metrics such as formalisation (from concept to detailing), functionality, modularity and product viability, to name a few. This is part of an ongoing research process that has promised to be as complex and the subject it studies!

References

Bashir, H. A. and V. Thomson, "Models for estimating design effort and time." Design Studies, 22(2), 2001, pp. 141-155.

Casti, J. L., "The simply complex: trendy buzzword or emerging new science", Santa Fe Institute, Santa Fe, New Mexico, USA: pp. ALL, 1992.

Designers' Sandpit Project, "The Designers' Sandpit Project: An Assembly-Oriented Design Environment (website)." at: http://www.eng.hull.ac.uk/research/sandpit/, Last accessed on:17.Mar.2004

Edmonds, B., "Complexity and Scientific Modelling." Foundations of Science, 5(3), 2000, pp. 379-390.

Eppinger, S. D. and V. K. Salminen, "Patterns of product development interactions", International conference on engineering design, S. Culley, Bury St Edmunds, Glasgow, 2001: pp.283-290.

Eppinger, S. D., D. E. Whitney, R. P. Smith and D. A. Gebala, "A Model-Based Method for Organizing Tasks in Product Development." Research in Engineering Design, 6(1), 1994, pp. 1-13.

Hatch, M. J. and H. Tsoukas, "Complex Thinking About Organizational Complexity: The Appeal of A Narrative Approach to Complexity Theory." Warwick Business School Research Papers, 1997, pp. ALL.

Rodriguez-Toro, C. A., S. J. Tate, G. E. M. Jared and K. G. Swift, "IMECE2002-39413 Shaping the Complexity of a Design", ASME; Design Engineering Division, Asme, New Orleans, LA, 2002: pp.641-650.

Suh, N. P., "A Theory of complexity, periodicity and the design axioms." Research in Engineering Design, 11(2), 1999, pp. 116-131.

Tang, V. and V. K. Salminen, "Towards a theory of complicatedness - framework for complex systems analysis and design", International conference on engineering design, S. Culley, Bury St Edmunds, Glasgow, 2001: pp.125-132.

Whitney, D. E., R. Mantripragada, J. D. Adams and S. J. Rhee, "Designing Assemblies." Research in Engineering Design, 11(4), 1999, pp. 229-253.

Mr. Carlos Rodríguez-Toro
School of Industrial and Manufacturing Science
Cranfield University, Cranfield, Bedfordshire
MK43 OAL – United Kingdom
E-mail: c.a.rodriguez-toro@hull.ac.uk