TASK COMPLEXITY AND DESIGN SCIENCE

T.Grandon Gill
IS/DS, University of South Florida
Tampa, FL , 33620, USA

And

William F. Murphy, Jr.
IS/DS, University of South Florida
Tampa, FL , 33620, USA

ABSTRACT

Design problems manifest themselves in many ways. Before anything resembling a true "science of design" can emerge, tools for classifying the design task must be developed. One such tool involves the task complexity construct, which comes in three distinct forms: objective complexity, problem space complexity and lack of structure. These forms, which can appear individually or in combination, each demand different approaches to the design task. Objective complexity entails a process dominated by search. Problem space complexity is often addressed by the accumulation of design rules. Lack of structure can benefit from the creation of different types of models. This paper describes the task complexity construct and explores—using a mix of theory and examples—its relationship to design.

Keywords: Task complexity, design science, goal setting, expertise, effectuation, entrenchment, artifacts, problem space.

1. INTRODUCTION

Design science is the emerging field that attempts to develop systematic rules and theories that can be applied to the design process. Design problems manifest themselves in many forms. These forms can range from highly artistic to highly technical, or both. They may be tightly constrained or unconstrained. It may be easy to recognize the strength of a given design, but for other designs, the fitness of the design may be highly uncertain. It therefore seems unlikely that any universal design principles will be discovered that apply equally to all design situations. Instead, to advance the field we must first focus of developing schemes for classifying and differentiating design tasks in a way that allows different approaches to be prescribed for different tasks.

One such classification scheme that offers plausible benefits is based upon the task complexity construct. In recent years, considerable progress has been made in distinguishing different types of complexity [8]. The present paper introduces three broad classes of task complexity definitions that have appeared in the research literature. Using examples, it then examines how each class leads to different design challenges and how those challenges may be resolved. Finally, it synthesizes these findings into a unified scheme for classifying design tasks. While no scheme of this type can fully determine the "best" design approach, we believe it represents a promising start.

2. TASK COMPLEXITY

Task complexity is a construct that, in the broadest terms, attempts to explain how the characteristics of a task impact the cognitive demands placed upon the task performer. The construct is most commonly applied in the fields of management and psychology and the most commonly referenced attempts to define the term are more than two decades old [1][15].

One aspect of task complexity that makes it particularly problematic is the myriad ways in which it has been defined. A recent study found 13 distinct definitions [8] that fell into 5 general classes:

1. Complexity as source of difficulty
2. Complexity as a source of information processing requirements
3. Complexity resulting from lack of task structure
4. Complexity as a measure of problem space characteristics (e.g., number of paths, amount of required knowledge)
5. Complexity as an objective function of task characteristics (e.g., number of task elements, degree of interrelationship between elements, dynamics of task objectives)

Because (1) and (2) are presumed consequences of complexity, while (3) and (4) are antecedents, these classes can be further simplified into three dimensions [6]:

I. Objective Complexity
Elements, Interrelationships, Dynamics →
Complexity → Ruggedness

II. Problem Space Complexity
Problem Space Characteristics →
Complexity → Information Processing

III. Unfamiliarity
Lack of Structure →
Complexity → Perceived Difficulty

The term ruggedness, presented as the consequence of objective complexity, draws upon the concept of a fitness landscape, introduced in evolutionary biology [10]. This form of complexity occurs when a task involves many attributes that interrelate, leading to a large possible solution space with many local peaks. A good example of this is the recipes in a between all the ingredients, as well as other actions (e.g., how they are mixed, cooking times, temperatures).

Problem space complexity views complexity in terms of the characteristics of the task-specific knowledge being applied by the task performer. Many of these characteristics would be familiar to computer scientists as metrics used to estimate the complexity of a computer program, such as number of paths (cyclomatic complexity [12]), amount of knowledge (e.g., lines of code, number of rules) or some theoretical minimum program size (e.g., Kolmogorov complexity [11]). In general, we would expect information processing rates to grow with accumulating knowledge, although such a relationship is better viewed as approximate than definitional.

Unfamiliarity is, in many respects, nearly the opposite of problem space complexity since it represents the absence of task-specific knowledge. For unfamiliar tasks, we are forced to rely on general knowledge and unreliable techniques such as analogy. Because these unfamiliar processes tend to place heavy loads on working memory [14], they tend to be perceived as difficult by the task performer. Once again, this relationship between unfamiliarity and difficulty is imperfect, but tends to hold true more often than not.

3. THE DESIGN TASK

Before considering the impact of task complexity on design, it is useful to define what we mean by "the design task". As illustrated in Figure 1, we treat the task as having three principal elements, two in the environment and one in the mind of the designer:

1. A design fitness landscape, which maps the characteristics of the design into a design fitness value.
2. Artifacts developed by the designer.
3. A problem space that the designer uses to perform the design task.

Naturally, where a design task is beyond the capabilities of a single designer, the problem space may need to span multiple individuals.

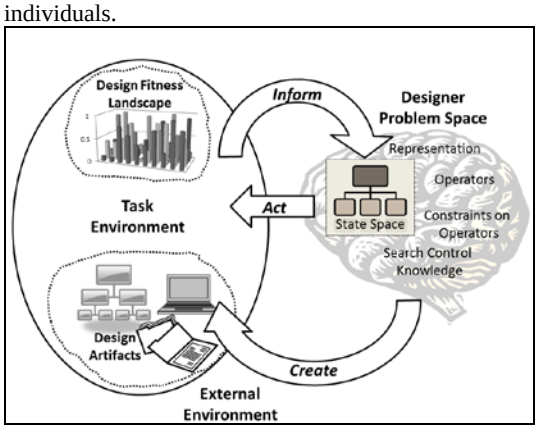

Figure 1: Overview of design task

cookbook, which can be viewed as mapping between ingredients and fitness outcomes such as taste and/or nutrition. In theory, at least, each recipe represents the author's local peak. Typical of this complexity, the contribution of an individual ingredient to that fitness is relatively meaningless, since a recipe's overall fitness is the result of the interaction

3.1 Design Fitness Landscape

The design fitness landscape maps design attributes, such as size, functionality, reliability, and so forth into a "fitness" value. As discussed elsewhere [7], what constitutes fitness can vary considerably. Its most direct translation might be to a metric such as immediate usefulness. On the other hand, it might also represent the likelihood that the design will diffuse, inspire other designs (reproduce) and evolve over time, a form more in line with the evolutionary concept of fitness.

The design fitness landscape can be characterized in terms of its ruggedness, linking it to objective complexity. On a rugged design landscape, many local maxima exist for different values of design attributes. By virtue of the string interaction between these attributes, we can also see very sharp jumps or drop-offs in fitness with small incremental changes. To return to the recipe analogy, think what happens to the resulting cake when you accidentally omit the ½ teaspoon of baking power called for by the recipe.

3.2 Artifacts

Within the design task, we treat artifacts in the most general way, consistent with design science research [9]. Thus, artifacts consist not only of the end products of a design process but also all the intermediate products. The latter group could include plans, design documents, communications between designers, prototypes, and so forth. These intermediate artifacts provide both a means of conserving working memory within the designer's problem space and a means of communicating design information to other stakeholders in the design process.

3.3 Problem Space

The problem space describes the internal representation within the mind of a designer (although, as mentioned earlier, collaboration between multiple individual designer problem spaces might have to be taken into account "real world" tasks). It is assumed to contain a representation of the problem within a state space, operators for transforming the state space, constraints that prevent inappropriate states and search control knowledge [2].

We refer to a series of transitions through the problem space as a *path*. To provide a term comparable to ruggedness and unfamiliarity, we describe the high complexity case as high *path entropy*. The choice of entropy here is meant to convey the qualitative sense that the most challenging design problem spaces are those where:

- *Uncertainty*: Considerable uncertainty exists in the choice of paths based on existing task performer knowledge.
- *Constraints*: Many paths exist that do not lead to the intended design end state.
- *Irreversibility*: Operators, once applied, are irreversible or costly to reverse

It should be emphasized that because problem space complexity exists within the mind of the task performer, new knowledge

may change the problem space in a manner that reduces the path entropy associated with a particular task. For example, imagine you are attempting to navigate between two locations in a city laid out in a grid (such as Manhattan) using only your position. In this case, any turn that brings you closer in distance to your destination will generally be a suitable path. Thus, while many path choices are available, the path entropy is low. Now consider how the task changes for a city laid out haphazardly (such as Boston). For such a city, many turns that appear to bring you closer in distance to the desired destination will, in fact, prove to be poor choices as a result of curves, dead ends and one-way streets. This is the high path entropy case. That high entropy is not intrinsic to the task, however. Rather, it is a function of your internal representation of the task. In the highly improbably event that you could find someone in Boston who was willing to give you accurate directions, the path entropy would drop since you would now know what choice to make at every turn.

It is also possible, of course, for a task to include uncertainty that is irreducible. No matter how good a solitaire player you are, some deals simply cannot lead to a successful outcome. This type of uncertainty could well be considered part of objective complexity. The problem is, outside of games of chance, it is often hard to tell what uncertainty is reducible and what is not. Thus, it generally makes sense to treat both forms as sources of path entropy within the problem space.

4. COMPLEXITY OF THE DESIGN TASK

Having described both the sources of task complexity and the nature of the design task, we can now turn to the key objective of this paper and consider the complexity of the design task. We do this considering at each of the three task complexity dimensions.

4.1 Design and Path Entropy

The presence of path entropy in the design tasks implies that no matter how clearly specified your design end state may be, and how great your familiarity with the task, achieving the intended goal will require considerable attention and care.

As an example, consider the contrast between how computers are configured today versus the situation that existed in the late 1970s. Today, the "designer" might be a customer who logs into a web site (such as *Dell*), chooses a model and then selects or deselects a series of options. For the most part, options do not interact in complex ways to determine fitness (e.g., a large disk drive is better than a small one, more RAM is better than less, faster processors are better than slower ones). Thus, the task of "designing" the system has become trivial and can be solved with a simple template. Many would not consider it design science at all [9].

In the late 1970s, however, the situation was very different. Some companies, such as *Digital Equipment Corporation (DEC),* chose to configure each computer to the customer's individual needs. That created a major design challenge because that lack of well-established standards at the time meant that each choice of component impacted what other components could be used. The problem was less one of figuring out what local maxima to choose (a ruggedness problem). Rather, it was finding any combination that met the requirements of the customer and the associated constraints of the components.

By 1980, the individuals designing each system were experiencing such a high error rate (> 60%) that DEC was considering abandoning its customization strategy. Eventually, however, they solved the problem by developing an expert system (XCON) that employed the same rules used by designers but did not make errors. Within three years the system incorporated over 4000 design rules and was 98% accurate in its configuration.

4.2 Design and Ruggedness

The presence of ruggedness in the design task occurs where features of possible design end states interact so strongly that the desirability (fitness) of a particular feature cannot be considered independent of the other features present.

Numerous examples of high ruggedness combined with limited path entropy and unfamiliarity can be found in the arts. Paintings and musical compositions, for example, have numerous interactions between elements of the piece (e.g., Would the Mona Lisa still be a masterpiece if Da Vinci has chosen to paint her eyes with different colors?) and yet the mechanics of creation are relatively straightforward (low path entropy).

For an IT-related example, we might choose a consumer technology such as designing cellular phones. Depending on the specific consumer's needs, design attributes such as small size, wealth of features, physical and/or virtual keypad, data capabilities, variety of protocols supported and carrier might all be considered either advantages or disadvantages. Presuming that most combinations are technically feasible, the design challenge then becomes to identify possible combinations that meet the needs of as large a market segment as possible. The designer that comes up with a unique high fitness combination hits a home run.

The challenge presented by ruggedness is that of ensuring sufficient and efficient search. In this context, expertise is a mixed blessing. It we consider the search process as being one of generating possible combinations and then assessing their fitness, psychological studies of games such as chess suggest that a major component of expertise involves unconsciously filtering the number of unfeasible combinations, such as poor chess moves, considered [3]. While this increases the time available for considering sensible options, it also means that outliers or radical new combinations may be filtered as well. This phenomenon is also referred to as expert entrenchment [4].

High ruggedness in the design space also means that designs that are "nearly right" combinations can exhibit relatively low fitness. As a consequence, we would expect that when a successful combination is created by one designer, it will tend to be copied.

A more realistic design scenario for most engineering and IT situations is one where both ruggedness and path entropy are high. Here, the challenge is that setting a high fitness end state as a goal may entail unmanageable path entropy. On the other hand, selecting a well-travelled path may lead to a low fitness end state. As a consequence, maintaining acceptable fitness while meeting the design constraints is likely to demand a continual balancing act.

The numerous attempts to create a successful tablet device can be viewed as an IT-based example of the rugged, high path

entropy design task. Finding an acceptable combination of form and function has proven to be an elusive goal. *Apple's Newton* was unable to make a dent in the market, largely failing to meet performance expectations. Moving towards smaller form factors, PDA devices were popular for a while, until rendered largely obsolete by cellular phones. Moving in the direction of larger form factor and greater functionality, *Microsoft's Tablet PC* was an attempt to graft the existing Windows architecture on to a tablet form factor. While the design proved to be highly fit for a small set of users [5], it never caught on as expected. eBook readers, such as *Amazon's Kindle* began to gain substantial attention by bundling cellular technology with an intermediate form factor and integrated marketplace. They were, however, single purpose devices. The first tablet that achieved breakout status with its features, capabilities, form factor and elegance was *Apple's iPad*.

The reasoning approach that seems particularly well suited to high ruggedness/high path entropy design tasks referred to as *effectuation*. Unlike the goal-drive decision rules of pure high path entropy task or the generate and test approach governing the pure high ruggedness task, effectual reasoning involves a constant shifting between goals and actions. Specifically, it is built around constructing the novel from the resources available. It is also the reasoning approach that is most commonly applied by entrepreneurs [13]. Effectuation-based design reasoning is also likely to produce user-modifiable artifacts, since such capability will facilitate the design process, ultimately increasing their flexibility.

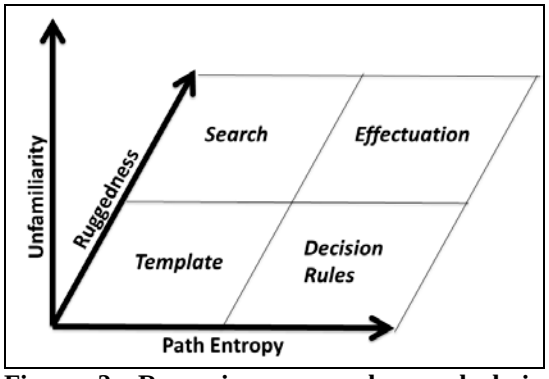

Figure 2: Reasoning approaches and design task complexity

In summary, the reasoning approaches combinations of ruggedness and path entropy are illustrated in Figure 2. To complete the picture, we need to consider the final complexity dimension: unfamiliarity.

4.3 Design and Unfamiliarity

Unfamiliarity typically corresponds to reducible uncertainty—or at least being able to assess what uncertainty is and is not reducible based upon current knowledge.

The key design tool for reducing uncertainty is the intermediate design artifact. These artifacts may be used for many, purposes. These include testing concepts (e.g., a test module), conserving working memory (e.g., diagrams), communicating between members of a design team or users (e.g., a model or prototype) and analysis (e.g., a project plan).

The types of artifacts that will dominate the design process are likely to depend upon the nature of the unfamiliarity. The mapping proposed in this paper is presented as the top layer of Figure 3.

For the trivial case, low path entropy, low ruggedness design projects, a prototype that evolves into a final project may be all that is needed; alternatively, a set of blueprints may be sufficient to reduce unfamiliarity without the need for any test artifacts.

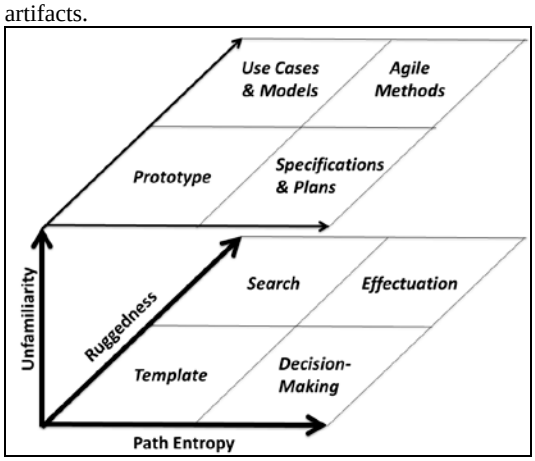

Figure 3: Three dimensions of design task complexity

Where ruggedness is high and path entropy is low, we can expect the design artifacts to rely heavily on use cases (so that the fitness of different combinations of attributes can be assessed in terms of user needs) and models of various sorts to provide a more concrete experience for potential users assessing the design. Such models are frequently seen in high ruggedness fields such as architecture and automotive design (e.g., concept cars).

Where path entropy is high but ruggedness is low, we can expect artifacts to depend heavily on specifications and planning. Since the desired end state is well understood for such design problems, we would expect many of the tools for project management as well as specification approaches such as UML to be effective in clarifying what needs to be done.

The case where all three dimensions of task complexity are high constitutes the truly "wicked" design problem. Any specification artifacts created are likely to need continuous modification. Use cases, while attractive in theory, may often lead to insoluble problems. A category of approaches that seem particularly well suited to high ruggedness/high path entropy design tasks fall under the heading *agile methods*. These methods involve continuous interaction between the users of a design and the builders of an artifact. Typical of these methods is the *scrum* approach to project management, where a low feature version of the design is constructed as quickly as possible and functionality is incrementally added while keeping a version of the design working nearly continuously.

5. CONCLUSIONS

Many individuals will argue that design is not and cannot be a science. While we are inclined to agree that design science will never resemble physics, we believe there is still room to attempt to construct conceptual schemes that allow the designer and manager of designers to think about what they are doing more

systematically. The framework presented in this paper offers a conceptual scheme that can be used to map a particular design task's complexity to the reasoning approaches and types of artifacts that seem are likely to be most productive. Before being applied to any mission-critical design project, it would definitely benefit from thorough empirical testing. We are optimistic that such testing will, in time, support a number of the proposition we have made, since what the framework we propose is grounded in many well researched studies of task complexity.

6. REFERENCES

[1] Campbell, D.J. (1988), Task complexity: A review and analysis. *Academy of Management Review, 13(1)*, 40-52.

[2] Card, S.K., Moran, T.P. & Newell, A. (1983). *The psychology of human-computer interaction*. Hillsdale NJ: Earlbaum.

[3] Charness, N. (1991). Knowledge and search in chess. In K. A. Ericsson & J. Smith (Eds.), *Towards a general theory of expertise: Prospects and limits* (pp. 39-63). Cambridge, U.K.: Cambridge University Press.

[4] Dane, E. (2010), Reconsidering the trade-off between expertise and flexibility: A cognitive entrenchment perspective, *Academy of Management Review*, 35(4), 579-603.

[5] Gill, T.G. (2007), Using the Tablet PC for instruction". *Decision Sciences Journal of Innovative Education. 5(1),* 183-190.

[6] Gill, T.G. (2010), *Informing business: research and education on a rugged landscape*, Santa Rosa, CA: Informing Science Press.

[7] Gill, T.G. & Hevner, A. (2011), A fitness-utility model for design science research, DESRIST 2011.

[8] Gill, T.G. & Hicks, R. (2006), Task complexity and the informing sciences: A synthesis, *Informing Science, 9.* 1-30.

[9] Hevner, A., March, S., Park, J. & Ram, S. (2004), Design science in information systems research. *MIS Quarterly. 28(1)*, 75-106.

[10] Kauffman, S.A. (1993). *The origins of order*, Oxford, UK: Oxford University Press.

[11] Li, M. and Vitányi, P., (1997), An introduction to Kolmogorov complexity and its applications, New York, NY: Springer.

[12] McCabe, T.J. (1976), A complexity measure, *IEEE Transactions on Software Engineering*, SE-2(4), 308-320.

[13] Sarasvathy, S.D. (2003), Entrepreneurship as a science of the artificial, *Journal of Economic Psychology 24*, 203–220.

[14] Willingham, D.T. (2009). *Why don't students like school?* San Francisco, CA: Jossey Bass.

[15] Wood, R. (1986). Task complexity: Definition of the construct. *Organizational Behavior and Human Decision Processes*. *37*. 60-82.