The discipline of Natural Design

Jon G. Hall, Open University, UK

Lucia Rapanotti, Open University, UK

Abstract

If we define *design work* as those cognitive and practical things to which designers give their valuable effort, then our Natural Design framework allows the recording and replaying of design work. Natural Design provides a meta-structural framework that has developed through our observations of engineering design in safety and mission critical industries, such as aircraft design. Our previous work has produced parametrisable models of design work for software intensive systems, and we now look to make an initial assessment of our natural design framework for its fit to the more creative design practices. In this paper we briefly sketch the framework and subsequently attempt to locate 'creativity' in it. We find that, although there are good strong hooks for what the designer does, we are forced to find a role for the community of the designer in the creative process in our framework, something that was only implicit in our previous work.

Keywords

Natural design; Engineering design; Creativity

It is widely accepted that design is a creative, risky and iterative endeavour. The designer is a source of creativity, a mitigant of risk, with the patience and persistence to produce designs that satisfy the client. However, there is more: the designer must have insight enough to see how a solution to a client's problem fits with the other problems they face. For, in reality, the designer will never be faced with a single problem: problems come together, intertwined in pairs or triples or more; they fit together like the clues and answers in a crossword–they are *crossproblems*. Our term crossproblem is chosen to be inclusive: although designers will face more or less wicked problems (Rittel & Webber, 1973), we do not need to exclude tamer problems (q.v.) from our framework.

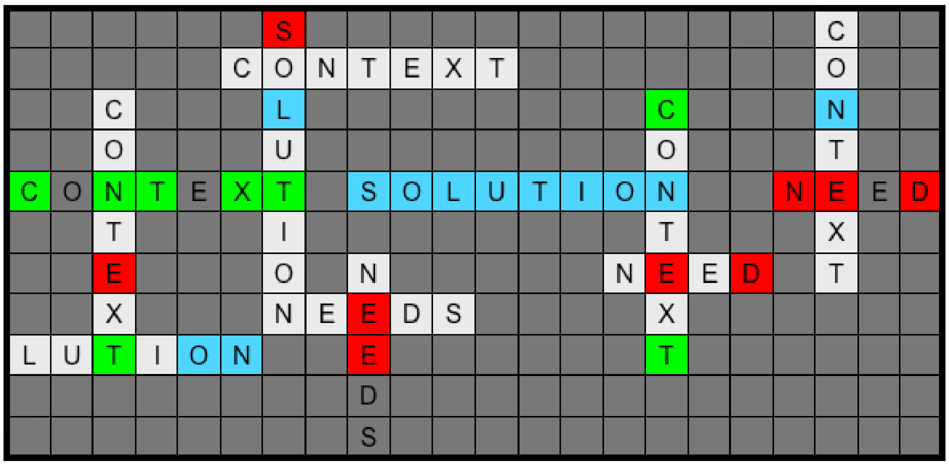

Figure 1: A 'crossproblem' is like a crossword, an intertwined collection of problems facing a designer.

Figure 1 tries to convey the complexity of a crossproblem, to illustrate a designer's difficulties: In the figure, think of the most colourful horizontal problem as a client-provided *focus problem*–here, that of finding a blue solution in the green context to satisfy the red need. The problem and its elements are only partially known or knowable. They intertwine in complex ways with the many other problems facing the designer, each of which constrains or makes more complex the focus problem. Each context, need and solution is concomitant and/or contingent on the focus problem, and each is expressed in a different way in different languages. Moreover, the focus problem may not be solvable with current knowledge and technology.

Over a period of two years, we have worked closely with an engineering designer in a commercial practice to observe the 'crossproblems' he faced (Hall, Mannering & Rapanotti, 2007). His crossproblem had many facets, including:

1) understanding the client's problem;
2) solving the client's problem to the satisfaction of the client;
3) solving it to the satisfaction of the chief architect, his project managers and senior practice management, whose organisation was partially sustained by that solution;
4) producing a design rationale that would satisfy a safety regulator (it was a safety critical system);
5) ensuring that appropriate post-commission and pre-decommission activities, such as maintenance, could be performed on his deployed solution;
6) exploring novel techniques for early solution validation to contribute to process improvement research.

In this paper we describe the framework-in-progress that is developing to support such observations. The framework-in-progress is called *Natural Design* and is derived from Problem Oriented Engineering (POE) (Hall, Rapanotti & Jackson, 2007, 2008). Natural design uses mathematics to provide a lightweight high-level structure that identifies inconsistency in crossproblems and that keeps problems and their solutions consistent whenever possible. It constrains neither the ways in which the problems relate to each other nor the steps that can or should be taken to solve them. In addition, we observe in this paper that natural design may provide a location for creativity. Although it does not, by itself, create anything natural design provides a lasting record of a designer's generously reported design steps, rationale and validation, at a granularity and level determined by the designer. Once captured the design steps can be replayed; sometimes the steps can, with work, be made generic to apply in other design contexts.

Design, creativity and software processes

Our work has thus far been located in the context of engineering design, specifically that of the safety-critical software and systems design industry. To sum up briefly, the organising notion of design in this area is still nascent: much work has been focussed on process definition to constrain perceived harmful invention, the goal being software development as an engineering discipline with focussed specialists working in professional communities akin to other engineering disciplines (Jackson, 2001). In this computing has been much influenced by the mathematical and scientific approach. In other work, the exploration of creativity in software design focusses on the need for creative approaches to the whole and to its separate phases: Glass (2006) explores the dichotomy between stubborn creative practitioners and ignorant managers, the one inventive and undisciplined, the other stifling and plodding. (Maiden & Robertson, 2005) provide examples of process improvement through a creative approach: extensions of software processes back to the customer, embodied in software requirements engineering, are argued to add value the more creative they are. At the other extreme, model driven development (Schmidt, 2006) removes much of the need to write code, focussing on intermediate models that can, at the push of a button, produce correct systems. Simply put, it is not yet known whether software design is a necessarily creative discipline.

This against the backdrop of over 40 years of design research (Cross, 2007) which has cast light on what distinguishes design from other disciplines. Rittel's (1973) characterisation of planning and design problems as *wicked* and Schön's (1983, page 49) reflective needs for `artistic, intuitive processes which some practitioners do bring to situations of uncertainty, instability, uniqueness and value conflict' show design to share little with the disciplines of mathematics and science, and suggest great value for a study of design separated from that of those disciplines. The subsequent consolidation of design research has, for the great part, excluded most of mathematical and scientific enquiry from creative design, and much from engineering design.

Is it too simplistic to say that design is waiting for mathematics and science to catch up? If so, the current distance of the core mathematical and scientific academic (and practitioner) communities from design means design will wait for a long time for an *organic* mathematics and/or science of Schön's `uncertainty, instability, uniqueness and value conflict' to arise (q.v.). The need of a designer to satisfy what we have above termed crossproblems may go someway to explaining why current scientific and mathematical approaches will remain unsuccessful, and why a divide will exist for some time.

A scientist, specialised to the extreme, may see the key to problem solving as the description of the variance of one dependent parameter on another; the creativity in a perfect experiment or model is in perfected isolation. However, a designer's problems admit no single description, nor do they exist without their crossproblem 'baggage'. On the other hand, mathematicians can deal with many problems–solving simultaneous partial differential equations, for instance–but, necessarily, such problems are expressed in a single precise language and with a single absolute notion of correctness. For the designer there is no single language that can bring all problem stake-holders together, nor is there a single notion in which one can place one's trust for expressing adequacy separately and together.

But bridging the divide from the design bank is also fraught with difficulty: there is no sophisticated aspect of design that can be represented by a single mathematical idea; there is no design process that is reducible to a linear, *bang-bang* process without gross approximation; and there is no sub-discipline of design that can be fixed into a logical framework. Academic evangelists of mathematical and scientific approaches have often, unintentionally, misrepresented the power of their work, resulting in failure, time after time, to deliver to the extent that avenues of expression of and discussion for creative design processes between the respective communities are exhausted (for instance, Glass (2004) and Gutmann (2004)).

If the unwarranted use of mathematics and science in design has removed any welcome felt there even for useful parts of mathematics and science it is important to understand, as (Buchanan, 1992, page 6) observes, that:

> *The significance of seeking a scientific basis for design (...) lies in a concern to connect and integrate useful knowledge from the arts and sciences alike, but in ways that are suited to the problems and purposes of the present.*

Only the careful and sympathetic reassessment of potential contributions from mathematics and science and their positioning within creative design process has any chance of leading to an acceptable reconnect for Buchanan's *useful knowledge*; but there is value on both sides: a reflective scientist and mathematician could benefit much from the right connection, as mathematics and science share a need for creativity in theory and practice.

If the goal of this work were to synthesise a single model for creative design consistent with what has gone before, we might ideally look, like Howard *et al.* (2008), for an 'umbrella' for (the valuable parts of) what has gone before–(Howard et al., 2008) lists twenty three engineering design process models plus nineteen more creative process models that they used as the subjects for integration. This goal is worthy. However, it is not the path we chose: for to synthesise a whole faces difficulties, not least that "design processes observed in practice are more erratic than most representations suggest" (Howard et al., 2008, page 176) introducing doubt into any effort not based on observation. Even the number of candidates alone indicates incalculable difficulties of a suitable consistency/breadth trade-off.

The goal of this paper is to look for creativity in our existing theory of engineering design, which we know to have practical explicative and predictive capability in that it provides a framework into which observations of the engineering designers at work fit, and that can predict their need for resources and for communication.

Natural Design

The mathematician Gerhard Gentzen observed mathematicians at work and up-shifted to define the paradigm of natural deduction (Gentzen, 1934) (English translation in: (Gentzen, 1969)). Previously, the search of absolute truth embodied by Hilbert's axiomatic proof systems (see, for instance, (Kleene, 1964)) looked for the proof of the true proposition in its structure. Gentzen's system supports the mathematician that creatively wishes to short-track a proof by inventing balancing 'positive' and 'negative' conjectures with which to split it: For a proposition, *P*, its proof $\bullet P$ can be staged through *Cut* (Kleene, 1964) with an invented, *C*, which serves to break $\bullet P$ into a proof for *C* together with a proof that *P* follows from *C*. That a *Cut* succeeds in a difficult proof rests on the ability of the mathematician to design the correct form for *C*: too strong, and *C* will be harder to prove that *P*; too weak and *P* will not follow from *C*; navigating the search space for *C* is the design flair of the skilled logician.

Because of the mathematical and scientific disconnect of Turski's problem (Turski, 1986)–that the real world is not expressible in any single linguistic system nor does the logical notion of proof apply–our framework cannot be just *mathematical*. Rather it needs to be *sympathetic* to the creative expressions of the designer. We have therefore substantially extended Gentzen's system in providing the mathematical foundations for our work: through the extensions we do not require the designer to commit to any single language for expression, any strategy for discovery, nor any detail in description, each of which can be left without a definitive formulation. Moreover, as Howard (2008) observes, creativity is not solely located to finding the right 'conjecture,' but in various amounts in establishing a need, problem analysis, conceptual design, embodiment, detailed design, and implementation. Each of these aspects of creativity can be located within our framework.

In the discursive introduction to our framework-in-progress that follows, we present both mathematical definition and explanation. For more details of the whole framework-in-progress, and a complete development, other papers should be consulted; for instance (Hall, Rapanotti & Jackson, 2007, 2008).

Problems

After Gentzen, our intention is to observe designers, fitting to them our framework-in-progress. Our novelty is to replace the *proposition,* upon which natural deduction is based, with the *problem*:

Definition: A *Problem* is a triple *Context, Solution* □ *Need.*

Explanation: a problem has three parts that must fit together in a particular way; specifically, the *Solution* must fit into its real world *Context* to adequately satisfy (□) the preferential stake-holders' *Need*.

Example: if the problem is to "make recommendations as to what to do with the Louvre," as was given to Emile Biasini by the President of the French Republic in the early 1980s.

French Republic, Solution □ What to do with the Louvre?

Biasini's 'solution' was to talk to I.M. Pei (Lecocq, 1989).

Problem solving

We have written extensively on problem solving in POE (for instance, (Hall et al., 2007), from which the mathematical definitions are taken), but that was in an engineering design context. The intention here is to progress our framework-in-progress a little more in the direction of design.

Definition: A problem transformation is a mathematical relation between a conclusion problem (written below the line) and its (many) premise problems (written above the line):

$$\frac{\text{Context 1, Solution 1} \bullet \text{Need 1, ..., Context n, Solution n} \bullet \text{Need n}}{\textit{Context, Solution} \bullet \textit{Need}}\; J$$

together with a justification, *J*, that includes but is not limited to the design rationale. The relationship between the various parts is left to the designer.

Explanation: to solve the conclusion problem transform it to other problems that are more amenable to solution or that lead to something more amenable to solution. Possible transformations include: focusing on one of many stake-holders' problems; exploring a choice made for the solution structure or architecture; simplify the context and/or requirement by ignoring domains; work to creatively detail part of the problem more fully.

Example: Returning to Biasini's problem above. Pei interpreted, i.e., transformed, Biasini's problem into many smaller problems:

> *"The center (sic) of gravity of the museum had to be in the Cour Napoleon, that's where the public had to come. But what do you do when you arrive? Do you enter into an underground space, a kind of subway concourse? No. You need to be welcomed by some kind of great space. So you've got to have something of our period. That space must have volume, it must have light and it must have a surface identification. You have to be able to look at it and say, `Ah, this is the entrance.'"*
>
> *(Cannell, 1995)*

A very special transformation is when the designer records only a single step to solution:

$$\frac{}{\textit{Context, Solution} \bullet \textit{Need}}$$

as there are no premise problems, the designer has *solved the problem*. Figure 2, for instance, gives an early sketch for the Louvre pyramid; with this sketch the problem of what to do with the Louvre is solved, as least to the satisfaction of Pei.

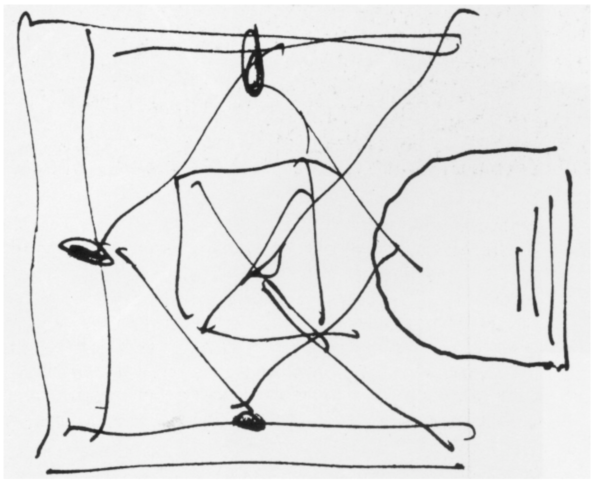
Figure 2: An October 1983 sketch of a solution of the *What to do with the Louvre?* problem by I.M. Pei.

A design process

For creativity, *satisfaction-belief*, by which we mean a stake-holder's belief that the solution *will* be adequate, arises during the design process. It is discharged only later, however, through direct experience of the solution in its context. Design is, therefore, inherently risky and belief in the expertise of the designer may be the only mitigation initially available for that risk. Risk exists for all stake-holders, and the sharing of risk between parties and its transfer from one party to another is the lifeblood that drives the design process: we have observed that, when the quality of the argument that will convince stake-holders of the adequacy is critical, i.e., when the argument is an additional mitigator of risk, an engineering designer will alternately explore the problem space looking for a validatable problem and the solution space looking for a validatable solution. Based on our observations of engineering design we have proposed the process, illustrated in Figure 3, that shows the iterations between *problem* and *solution exploration*.

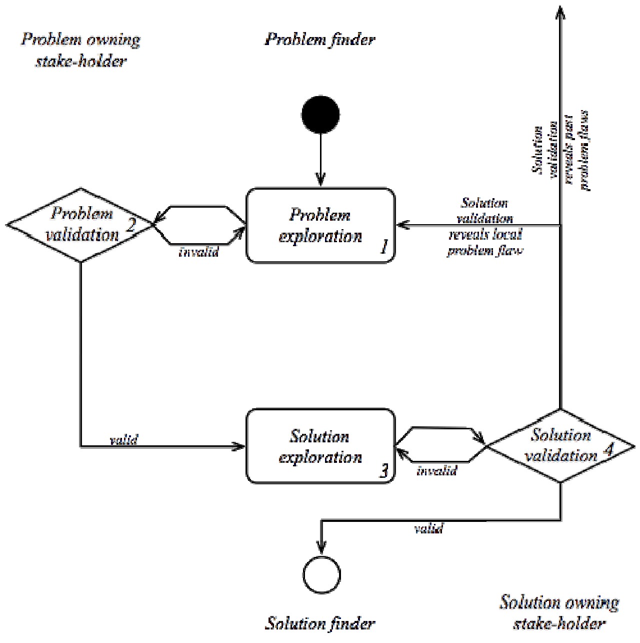
Problem owning stake-holder
Problem finder
Solution validation reveals past problem flaws
Problem validation 2
Problem exploration 1
invalid
Solution validation reveals local problem flaw
valid
Solution exploration 3
invalid
Solution validation 4
valid
Solution finder
Solution owning stake-holder

Figure 3: the natural design problem solving process. There is a transfer of risk between designer and other stake-holders at diamonds 2 (problem validation) and 4 (solution validation) which drives the process around the loop. Each of Problem Exploration and Solution Exploration are, themselves, problem solving activities.

When a candidate solution, such as that sketched by Pei in October 1983, is accepted it is typically a source of other problems. In our process, we admit this possibility by allowing instances of Figure 3 to be cascaded in sequence–one solution is source of other problems–and in parallel–one problem is worked on together with others. Cascading problems from Pei's initial study could be those of detailing the design, of constructing the edifice, and of maintaining it.

There are two points at which risk is transferred or shared in the figure corresponding to validation of the problem and solution with relevant stake-holders. We do not intend to exclude the various stake-holder from involvement in problem exploration and solution exploration; indeed, this is a necessary part of these activities, reflecting the perceived wisdom on stake-holder involvement. Rather we suggest only that there is a point at which risk is transferred and shared between the parties to the design:

Managing the risk of solving the wrong problem

Problem space validation is when the downside risk of solving the wrong problem is transferred from and/or shared with problem finder to an external stake-holder (that we term a *problem owning stake-holder*) willing to say that, 'Yes, the problem finder has understood the problem correctly'. Note that, problem validation will not prevent the designer from solving the wrong problem; it is just the means of mitigating the downside risk involved.

There are many familiar examples of problem owning stake-holders, including, but are not limited to, customer (those that pay for a product), clients (those that pay for a service), regulator (those requiring safety, for instance), and end-user (those who will use the product or service when commissioned).

Managing the risk of providing an unsatisfactory solution

Solution space validation is when risk is transferred from and/or shared with the solution space to an external stake-holder (that we term *solution owning stake-holder*) willing to say that, 'Yes', the solution finder has understood the solution correctly.' Again, solution validation does not prevent the designer from providing an inadequate solution; it is just the means of mitigating the downside risk involved.

The roles of solution owning stake-holders may be less familiar to the reader. They include, but are not limited to, a development house's chief software architect–who knows which architectures their organisation uses in solutions, an oracle–who determines which of a number of features should be included in the next release, or a project manager–who needs to time-box particular activities; there are many other roles that fit solution owning stake-holder.

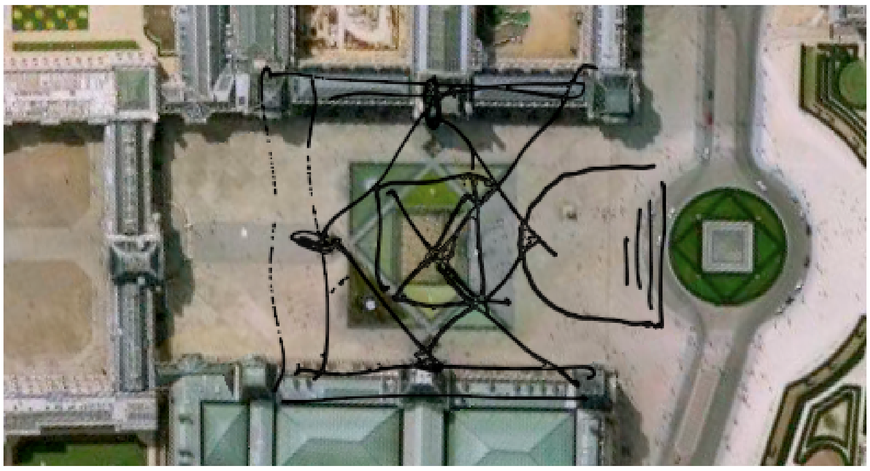
Figure 4: Pei's creative way of seeing the Louvre context, as a square with inscribed diagonals rather than a distorted rectangle, and his hint of a solution. (Figure 2 overlaid on an aerial view of the Louvre.)

Locating creativity

Definition: Natural design is *the process whose intent and practice is understanding and arrangement of a (collection of) problems sufficient to allow a process whose intent and practice is the understanding and arrangement of problem or solution parts which satisfies the concomitant needs for the relevant stake-holders. There is creativity in the amount by which the means of achieving this is* known adequate *in the designer's reflective context.*

Unpacking this definition a little:

First of all, but notwithstanding the complexity added to the process below, design deals with understanding and navigating (which together we denote exploration) both problem and solution spaces, and the processes by which each is structured or by which structure is arrived at therein.

Secondly, the arrangement of parts is applicable to problem and solution: creativity provides new ways of seeing the designer's problems' context(s) or need(s) as well as the solution. In this sense, the quote above and Figure 4 (simply the overlay of Figure 2 onto an aerial view of the Louvre) demonstrate Pei's 'novel understanding' of the Louvre entrance problem's context and need: Pei squares what, in reality, is a distorted rectangle showing novelty in the arrangement of the problem's context and needs. This interpretation was not, we suppose, an externally known adequate representation for this problem. Pei's reasoning behind the squaring of the rectangle and why, in October 1983, he thought it an adequate solution is unknown. Creativity may see the whole problem–its context, needs and solution–differently.

Thirdly, *known adequate* is a notion that is difficult to quantify as it depends upon community participation (the *reflective context* mentioned above) in systems of record and for the representation of known adequate arrangements of parts. In traditional engineering disciplines, specialism dictates that effort is placed in the focus on recording adequacy of particular designs and design parts: Michael Jackson, a well-respected engineer, has recently written that:

> *To some extent this focus on the particular in the established branches is a natural consequence of their already highly evolved specialisation. But it is also a precondition and a cause of specialisation. By recording very specific studies, or carefully documenting specific designs, researchers and teachers offer practitioners a continually updated corpus of detailed knowledge that they must not ignore. If only because each practitioner can master and exploit only a small part of this corpus, specialisation is an inevitable outcome.*
>
> (Jackson, 2008)

This requirement of reliance on known adequacy constrains much of engineering design creativity and it is possible to glimpse a difference in the two forms of design–in engineering, the emphasis is of fitting problem and parts into known representations, limiting creativity, engineers may even say "Necessarily so."

The strength of the reflective community is, then, one determinant of the creativity of its practitioners. Outside of engineering design–in scientific enquiry, for instance–*independent* discovery, i.e., in the absence of record of some result, is recognised as creativity, even though nothing new is produced. Of course, in these case, much hinges on the practitioner's honesty.

Lastly, Vincenti (1990) calls the process by which known adequate solutions are applied at *normal design* and opposes it to the creative *radical design* with much of engineering design being, necessarily, of the normal quality. We suggest that there is a continuum between normal and radical, with creativity and application of the pre-known being reciprocal notions on this continuum.

Process creativity; higher-order creativity

The reader that doubts the ability of the simple bang-bang process that we have so far described to describe a creative process is correct. To see why, we need to go fractal...

According to Wikipedia (2008), a *fractal* has a recursive structure at arbitrarily small scales, and is self-similar (at least approximately or stochastically). Problem solving in natural design is structurally simple but admits recursive application in that both problem exploration and solution exploration are themselves problem solving activities (as would be their recursive instances). The reflective context and the known adequate are therefore incident on the process itself. Creativity is present not only in the solving of the problem, but in the structuring of problem and solution exploration, and in *their* problem and solution explorations, and so on...

The fractal nature of problem solving also has the ramification that, no matter how deep the knowledge of particular designer's practice goes, there will always be deeper levels, certainly unknown and perhaps even unknowable, preventing even those who study from emulating that designer. They must, at some point, stop emulating perhaps to develop their own deep levels.

And this is another reason computers will never be able to replace designers: we know that bare searching the problem and solution space is no replacement for creativity, but now it is clear that there is a limit to what can be encoded of any designer's recursive problem solving process, even if a master designer's intention is to encode as software all of their design skills.

Discussion and conclusions

We have presented a framework for natural design. The framework is inspired by Gentzen systems for mathematics, but relaxes the need for absolute correctness to one that encompasses the needs of clients, developers and other stake-holders in the design process, and allows a lasting record of a development to be made. In this paper, we have attempted to locate creativity in the framework and have postulated its appearance in the fractal nature of the problem solving process; specifically, we locate it in the problem and solution exploration phases, each of which is another creative design activity. We are aware of the need to be able to represent collections of intertwined problems, that we term crossproblems, each member of which faces the designer or design team, each of which has multiple, perhaps conflicting, stake-holders and each of which poses risk for the designer that must be mitigated by a solution. The sharing and transfer of risk drives the process of design.

The framework was originally derived from observations of engineering design, for whom the community involvement in design–the recording of adequacy of particular solutions, of fractal problem solving activities and the like–and we had not realised until widening our attempts to capture more creative design, that the involvement of the community was a constraining characteristic on the expression of creativity in engineering design processes.

Our research programme at the Open University, the aim of which is to formulate engineering design processes from software engineering, safety critical systems and other systems engineering, is gathering momentum. Peaked by what we have learned in this theoretical strand of research, our interest turns to study more creative design processes, reformulating and adapting our framework-in-progress as we go, to be able to form lasting records of design. The ability of our framework-in-progress to work to capture the structure of problem exploration and solution exploration (and their fractal instances) holds some hope of capturing not only the many traditional linear problem solving processes that populate creative design courses (Howard et al., 2008), but also higher order processes that drive the fractal instances of problem and solution exploration.

Natural design is an extension of Problem Oriented Engineering, our validated framework for engineering design. We hope that its basis in mathematics is sufficiently sympathetic to provide some acceptable basis for a discussion between creative and engineering designers, as well as with ourselves, on the nature of the design disciplines.

Bibliography

Buchanan, R. (1992). Wicked Problems in Design Thinking. *Design Issues*, *8*(2), 5-21.

Cannell, M. T. (1995). *I.M. Pei: Mandarin of Modernism*, . Retrieved June 5, 2008, from http://www.washingtonpost.com/wp-srv/style/longterm/books/chap1/im_pei.htm

Cross, N. (2007). Editorial: Forty years of design research. *Design Studies, 28,* 1-4.

Gentzen, G. (1934). Untersuchungen über das Logische Schliessen. *Mathematische Zeitschrift, 39,* 176-210.

Gentzen, G. (1969). *The Collected Papers of Gerhard Gentzen.* (M. E. Szabo, Ed.). Amsterdam, Netherlands: North-Holland.

Glass, R. L. (2004). Practical programmer: The mystery of formal methods disuse. *Communications of the Acm, 47*(8), 15-17.

Glass, R. L. (2006). *Software Creativity 2.0.* developer.* Books.

Gutmann, P. (2004). *Cryptographic Security Architecture: Design and Verification.* Springer.

Hall, J. G., Mannering, D., & Rapanotti, L. (2007). Arguing safety with Problem Oriented Software Engineering. *10th IEEE International Symposium on High Assurance System Engineering (HASE),*

Hall, J. G., Rapanotti, L., & Jackson, M. (2007). Problem Oriented Software Engineering: A design-theoretic framework for software engineering. *Proceedings of 5th IEEE International Conference on Software Engineering and Formal Methods,* 15-24.

Hall, J. G., Rapanotti, L., & Jackson, M. (2008). Problem-oriented Software Engineering: solving the Package Router Control problem. *Ieee Trans. Software Eng., 34*(2), 226-241.

Howard, T. J., Culley, S. J., & Dekoninck, E. (2008). Describing the creative design process by the integration of engineering design and cognitive psychology literature. *Design Studies, 29,* 160-180.

Jackson, M. A. (2008). *The Name and Nature of Software Engineering.* Unpublished manuscript.

Jackson, M. A. (2001). *Problem Frames: Analyzing and Structuring Software Development Problems* (1st ed.). Addison-Wesley Publishing Company.

Kleene, S. C. (1964). *Introduction to Metamathematics.* Van Nostrand, Princeton, NJ.

Lecocq, C. (1989). *The New Louvre.* Connaissance des Arts.

Maiden, & Robertson (2005). Integrating creativity into requirements processes: experiences with an air traffic management system. *Proceedings of the 13th IEEE International Conference on Requirements Engineering,* 105-114.

Rittel, H., & Webber, M. (1973). Dilemmas in a general theory of planning. *Policy Sciences, 4,* 155-169.

Schmidt, D. C. (2006). Model-Driven Engineering. *Computer, 39*(2), 25-31.

Schön, D. (1983). *The Reflective Practitioner: How Professionals Think in Action.* Temple Smith.

Turski, W. M. (1986). And no philosopher's stone, either. *Information Processing, 86,* 1077-1080.

Vincenti, W. G. (1990). *What Engineers Know and how they know it: Analytical studies from Aeronautical History.* The Johns Hopkins University Press.

Wikipedia (2008). Fractal. Retrieved June 5, 2008, from http://en.wikipedia.org/wiki/Fractal

Jon G. Hall

Jon G. Hall is a Senior Lecturer in the Computing Department at The Open University. Previously, he held research positions at York and Newcastle-upon-Tyne Universities. His research mission is to provide complementary practical and theoretical foundations for computing as an engineering discipline, but his interest is growing to include other creative and engineering design disciplines. Jon is co-Editor-in-Chief of Expert Systems, The Journal of Knowledge Engineering. He has published more than 60 academic papers. For more information, please see http://mcs.open.ac.uk/jgh23/.

Lucia Rapanotti

Lucia Rapanotti is a Senior Lecturer in the Computing Department at The Open University. Previously, she has held both research and software development positions. Her current interest is problem-oriented engineering with applications to both socio-technical and safety-critical systems. She is co-Editor-in-Chief of Expert Systems, The Journal of Knowledge Engineering, and Secretary of the British Computer Society (BCS) Requirements Engineering Specialist Group (RESG). Lucia holds a Laurea Cum Laude in Computer Science from the University of Milan, and a PhD, also in Computer Science, from the University of Newcastle upon Tyne, UK.