Misfits, Balance, Requirements, and Systems: thoughts on Alexander's Notes on the Synthesis of Form

Filippo A. Salustri, Ryerson University, Canada, salustri@ryerson.ca

Abstract

The author examines the notion of misfit as presented by Christopher Alexander in his book *Notes on the Synthesis of Form*. We argue that from the point of view of our current understanding of design, the approach is flawed, but not flawed beyond use. In fact, the core concept of misfit, and how misfits can be addressed, remain as important to design today as when Alexander wrote about them. In this paper, a number of flaws are identified and explored. Subsequently, a new approach, which the author calls a *balanced systems approach*, is sketched. This approach preserves the intent and core of Alexander's work, while addressing the identified flaws. The main contribution of this paper is to indicate the shortcomings of Alexander's approach, but only for the sake of refining it and ensuring it remains relevant and useful.

Keywords

misfit; requirement; balance; system; Christopher Alexander; synthesis; function; form

In 1964, Christopher Alexander published *Notes on the Synthesis of Form* (Alexander, 1964), hereunder abbreviated as NSF. In that book, Alexander argued for a novel approach to design problems based on identifying and addressing misfits – mismatches between the capacities of and expectations on a form in a specific context. He also identified a change in the way design had been conducted that relates to changes in culture. His "unselfconscious" cultures designed primarily through a heritage of craft and artisanship where reflection and systematization were largely absent. These cultures have largely given way, in his view, to "selfconscious" cultures that reflect and systematize intensively. While he recognizes inherent design benefits from the move to selfconscious culture, he also advises that we have also given up certain advantages of the unselfconscious cultures. In particular, the responsiveness of unselfconscious design to externalities (the exigencies of the context and problem at hand) has given way to a more internalized and abstract perspective. He sees this as problematic because the internalizations are imperfect.

Alexander's relatively esoteric work on misfits and the nature of design problems as described in NSF has languished in comparison to his other work – e.g. the more pragmatic work on pattern languages, which has received broad attention in many disciplines.

The current author believes there is some benefit in revisiting NSF and considering its implications in the more sophisticated setting of contemporary design. We will argue in this paper that there are distinct benefits to Alexander's notion of misfits, and that the combination of (modified) misfits with systems thinking provides a valid framework in which to discuss modern design thinking and propose new potential avenues to improve how design is practised.

The paper is laid out as a series of observations about NSF and the notion of misfits. In each section, modifications are suggested that can address identified shortcomings in Alexander's original work. Finally, a new, balanced systems approach, is introduced. This approach appears to preserve the intent and core of Alexander's work, while addressing the identified shortcomings.

Set Theory in NSF

This is a fair amount of formal logic in NSF, in the form of set theory. The current author is well acquainted with set theory from his own research, and has identified three concerns with Alexander's work.

1. Alexander assumes the existence of the universal set; that is, the one set that contains every other possible set. However, the universal set is excluded from every standard set

theory, because its inclusion makes set theory invalid. An invalid theory is one that yields false positive results (i.e. the theory can prove true statements that are in fact false).

2. Godel proved that no formal system can be both complete and valid (Hofstadter, 1979). However, the current author has not found any evidence of a completeness proof of Alexander's logic. An incomplete theory is one in which there are some true statements that cannot be proved true within the theory. Thus, it appears that Alexander's work is neither valid nor complete.
3. Alexander's logic appears to be a many-sorted, first order logic. Every many-sorted logic has an equivalent one-sorted logic. One-sorted logics tend to be more stable and reliable, and more easily computed) than many-sorted logics. However, Alexander appears to have never translated his work to one-sorted form for the sake of establishing a completeness proof.

Of course, just because the appropriate proofs have not been provided does not invalidate all of his work – indeed, the current author believes NSF remains as relevant today as when it was originally written. However, there is no benefit in invoking set theory unless one seeks to establish a formal grounding for some undertaking. Therefore, this suggests that the formal aspects of Alexander's work should not be relied upon. In this article, the current author will not depend on these aspects of Alexander's work.

What is a Misfit ?

NSF provides no direct definition of *misfit*, but rather a number of characteristics scattered over several passages.

Alexander writes of "requirements or misfit variables" suggesting that there is some equivalence between the two terms. Indeed, although Alexander nearly always uses the term *misfit*, the text is best understood by reading *misfit variable*. The author suggests that the difference between misfit variables and requirements is that misfits simply describe a situation, whereas requirements constitute directives to the designer to address misfits to the extent described by constraints (either boundary constraints or minimization constraints).

Alexander also refers to a misfit (variable) as relating to an *ensemble*, which is the combination of a given form and the context in which it exists. This is very important because it connects a misfit variable to a particular situation; that is, *misfits are always situated*.

The importance of a misfit being a variable is easy to overlook. A variable has two elements: a named property, and a value for that property. If the variable's value is "bad," then it is a *mis*fit. The goal of designing, then, becomes the changing of the *values* of misfit variables so that they are no longer "bad," but the variables themselves must obviously remain.

This is reinforced by considering the partitioning and clustering tasks described in NSF. Networks of related misfits are mathematical graphs of nodes and links (a graph is a set of ordered pairs of nodes that represent the relationships between the elements of the node pairs). Nothing is said about the contents of the nodes or of the links.

Partitioning and clustering builds a hierarchy such that the top level of the hierarchy is the design problem as a whole. The network is one of misfits, or requirements, so the network is a model of the design problem.

The design problem is not something typically under the designers' control, although its model (as requirements/misfits) may change as the designers' understanding improves over time. Excluding changes arising from the imperfect knowledge and reasoning of human designers, the problem itself remains *fixed* (i.e. invariant). If the problem changes because of external influences on the existent ensemble (also not under the designers' control, but quite possible in the real world), then the original problem no longer exists and the designers must essentially begin again. For simplicity, we assume here that these kinds of changes do not occur.

If the problem is fixed, then the misfit network (the model of the problem) must also remain fixed.

The designers' goal, however, is to eliminate the misfits. If we eliminate the misfits, and the misfits are the network, then we must change the network. This contradicts the previous statement: we cannot change the network if it must remain fixed.

We can resolve the contradiction by realizing that the nodes in the network are variables and that misfits are just bad values of those variables. Thus, to eliminate a misfit, one must change the content of a node, not the node itself. In this way, we can remove misfits while keeping the network itself fixed.

There is another problem with the notion of misfits. As misfits represent only the "bad" aspects of ensembles, a network of misfits forms on a *partial* representation. If there are "good" aspects of ensembles, they will not be part of the misfit network. However, these good aspects will likely interrelate with the misfits, just as misfits interrelate with one another. Changing the values of misfit variables could adversely influence those good aspects, but the misfit network will not capture those influences.

This could be addressed by broadening the concept of misfits to include any property of an ensemble, not just those with bad values. The resulting network could then be described as representing *forces* at work in a situation that must be resolved, or *balanced*, by a design. In that case, one would end up with a network graph that describes how well *balanced* a form is with respect to a context, including both its good and bad aspects. The misfit network would be a sub-graph of the overall force graph. Conveniently, this viewpoint is quite consistent with Alexander's work on pattern languages (Alexander, Ishikawa, and Silverstein, 1977).

There is Always a Form

Misfits motivate designing, so misfit identification (and existence) must precede design. Over the years, some of the author's colleagues have argued that, because of the notion of starting from an existent form, Alexander's approach ignores the possibility of highly innovative design or invention. The argument hinges on the assertion that there is no form preceding the invention of a suitably innovative product.

However, the author believes that there is *always* a preceding form: it is the way that we do things "now," that we find inadequate, and that we address by innovation. Even for highly innovative forms, there is also a preceding form. In those cases, the preceding form is just substantively different from the innovative one.

For example, Jeff Hawkins led the invention of the original Palm Pilot personal digital assistant (PDA) by setting the device's main competitor as a leather-bound agenda (Rose, 2000). That is, the form that preceded the Palm Pilot was *not* a PDA but still shared aspects of functionality with it. There were a number of innovative technological elements in the Palm Pilot, such as the invention of the "graffiti" glyph language for stylus-based text input, and "instant on" technology. Palm developed these innovations for their product, without which the design team (led by Hawkins) believed the product would fail in the marketplace, to compete functionally with an agenda.

The forms before and after an innovative design do not have to share structure, but they *do* have to share some essential functional purpose. The Palm Pilot and the leather-bound agenda serve the same purpose, in context, even if the PDA expands on that purpose and opens the possibility of other functions.

Thus, while *form* is what one has at hand, one must really focus on the *function* that the form serves. The extant form is used to understand how the required function is *not* being fulfilled. These shortcomings are the misfits, and they drive the design of a new form.

In cases of high innovation, it may be difficult to identify (i.e. separate) the predecessor form within the context. This is solved by applying principles of systems thinking: entities are marked by boundaries; boundaries are where properties change; so identifying an entity means identifying key properties and then finding where they change. Even the act of drawing different boundaries can be enough to inspire innovation. For example, while the first attempts to build flying machines typically used flapping wings to generate both lift and thrust, success came relatively easily once new functional boundaries were drawn in the situation such that lift (via wings) and thrust (via propellers) were separated.

The Palm Pilot was not the first PDA to be designed, but it was the first *successful* PDA. This is because its designers found an appropriate balance point between all the factors that influence its success: cost, size, functionality, complexity, etc. To find an appropriate balance point, one must identify the appropriate systems, which in turn drives the identification of proper misfits.

The Problem of Form and Requirements

In NSF, Alexander refers often to an example of a kettle, and gives a rather extensive list of requirements that a kettle should satisfy. One of them reads "…should have a handle that…."

Alexander *assumes* that the kettle has a handle – thus implying that at least one design decision has been made (that there must be a handle) by the time the requirements were specified – even though specifying the requirements is supposed to *precede* designing. This is a logical contradiction. There may have been constraints on the kettle design problem requiring a handle, but this is not indicated in the text. This kind of logical contradiction is common in NSF; the kettle's handle is just one example of it.

More appropriate requirements would be functional, not structural, in nature; they would be more appropriate because they would make no assumptions about designed form. Two examples of how this might have been done for the kettle's handle include the following.

1. [The kettle must be such that] a person can easily carry/lift the kettle. (This begs questions about the capacity of the users, how far and how often they carry/lift the kettle, etc. Such questions should be answered with constraints.)
2. [The kettle must be such that] a person can pour the kettle's contents. (This begs questions about the position/location of the thing receiving the poured contents with respect to the kettle and its user, the rate of pour, etc. Again, these should be answered with constraints.)

There are other possibilities. The point is, however, that Alexander's requirements are not good ones. The lack of good requirements undermines his argument. However, one can recover, and at the same time refine, the overall process as follows.

Clients and users do not really know what they want. Designers serve that purpose. Clients and users *are* very good, however, at identifying what is wrong in how things are, what is wrong with the "as is" ensemble. This is precisely what misfits are for, so they are ideal for capturing what clients and users find wrong with the current situation.

However, designing requires production of form to suit function. An informal description of a process that can achieve this goal is discussed in a subsequent section.

Abstract Categories of Requirements

Alexander correctly observes in NSF that requirements are in practise often grouped according to abstract concepts like "safety" or "durability," and that such categorization is not helpful because it is arbitrary and constructed entirely by the needs of the designers rather than the needs of the situation for which the designers work. The arbitrary nature of the categorizations that designers impose only obscure the nature of the problem, which exists in an ensemble independent of the designer.

The author agrees with Alexander, that categorizations should be driven by the available information and not by the designers' needs or beliefs of what the categorization "should be." While the act of balancing will naturally include the designers' experience and perspective, the forces that are to be balanced must be those "observed" to exist in the extant ensemble.

Decomposing Misfit Networks

Much of NSF is devoted to establishing how and why networks of misfits should be decomposed and partitioned into a hierarchy. Alexander went so far as to develop software that could perform such decomposition and structuring work for a given misfit network – e.g. (Clark and Elms, 1976). Others, such as (Elms, 1983) and (Owen, 2007) have since tried to improve on Alexander's work.

Alexander states the purpose for decomposing misfit networks as follows. “We now have a graph G(M,L) which represents the design problem. …to solve the problem, we shall try to decompose the set M in such a way that it gives us a helpful program for design.” In other words, the decomposition is to help guide the designers to a solution.

However, to do this, he and all the others who have built upon his work rely on *cutting* certain links between misfits in the network graph to construct the needed hierarchical structure; this means that some relationships are ignored. Naturally, significant effort is made to identify the “weakest” relationships – those that can hopefully be ignored without invalidating the network graph itself as a representation of the design problem. More precisely, the graph is cut where the smallest damage to the network, and the greatest decrease in complexity, can be achieved. The complexity of the algorithms is driven by the need to identify those specific cuts.

The current author finds this approach problematic. In general, there is no way to know *a priori* what the impact of any change to a problem definition will be on the fitness of a designed solution. Alexander himself devotes a significant portion of NSF to arguing that design should be led by the facts, the data, the available information; so, to ignore information simply for the benefit of the designer seems inappropriate. One might make such an argument if a complete problem (as modelled by a misfit network) were demonstrably intractable – in which case, an approximate solution could be better than no solution at all – but no such demonstration is made.

There is another approach that should be able to help control complexity and organize design problems to facilitate their solution without ignoring any of its elements: a hierarchy of systems based on functional interactions, such that higher level functions emerge from, but are not necessarily directly produced by, constituent subsystems. This approach is based on recognizing properties of the entities described in a problem (the ensemble) in functional terms, and does not impose arbitrary structures on that problem. More information on this is provided in a subsequent section.

Setting these matters aside for the moment, let us examine Alexander’s two criteria for what constitutes a good misfit network decomposition, to determine if they are still reasonable given the preceding discussion.

First, with a good decomposition it must be possible to find “constructive diagrams” for each subset of misfits *individually*. This is what programmers call *encapsulation* and is inherently addressed by systems thinking, wherein a system is crisply demarcated by its boundaries with the environment and that one can swap one system for another so long as its interface (how it exchanges things with other systems) is the same.

Alexander also writes that a subset of misfits must “cohere somehow” and suggest “a physical aspect or component of the form.” The current author believes Alexander intends a coherence based on function: misfits in a set must all be functionally connected – which in turn would suggest kinds of forms. In this interpretation, systems, being functional entities, clearly fit well in Alexander’s approach.

Alexander’s second criterion is that a “useful” decomposition must be such that the representation (i.e. constructive diagrams) of a combination of two subsets of misfits must be “derived…in some simple way” from the representations of the two subsets.

This suggests that there must be a relatively straightforward superposition of representations to combine two sets of misfits. This in turn suggests that all the requirements (defined by misfits) must be known before a design solution is started. This is not generally possible because many requirements of lower level solution elements (e.g. the parts and sub-assemblies of a physical product) are derived from a combination of higher-level requirements plus design decisions that were made earlier in the process. This *co-evolution* of problem and solution (Dorst and Cross, 2001) is necessary because every design decision alters the specifics of the design problem.

However, it appears that Alexander is following a “waterfall model,” for instance when he writes in NSF that a decomposition must occur *entirely* before solutions are sought, which is not effective in design situations.

We can salvage this, however, if we admit a systems-based hierarchy. If we focus on only the requirements of one system at a time, and design as much as possible for it before moving to its subsystems, then we can implement co-evolution. At the same time, Alexander's misfit-based approach can be applied *to each level* in turn.

Thus, the author believes that a systems-based approach to misfit organization satisfies both of the requirements set forth in NSF for acceptable decomposition methods.

The author has identified one other problem in Alexander's method of combining subsets of misfits. His method does not seem to address that some properties of combined subsets are neither endemic nor constituent of the subsets; rather, these properties *emerge* from the combination itself. These emergent properties are very important for design. A systems-based approach treats these properties by associating them only with particular systems and not necessarily with their sub- or super-systems. For example, the drive train of an automobile enables the emergence of a key property of the automobile (its ability to move) without providing the property *per se*; but without the engine, that key property is not available.

Relative Independence of Requirements

Let us return to the sample requirements in NSF that Alexander gives for the kettle. Two of them are: "…must be comfortable…" and "…must be economical to heat." Alexander ponders the apparent independence of these two requirements, writing that it is hard to see if and how they relate.

Requirements can be uncoupled but still become coupled in a specific design; this is because the structure of the design may cause coupling to emerge only in that context. Borrowing from Suh's Axiomatic Design (Suh, 1990), one can demonstrate that one can decouple requirements by choosing the "right" design. That is, while it certainly is possible to develop functionally coupled requirements, truly uncoupled requirements depend on both the requirements *and* the design solution. This is evident in the kettle example: requirements that explicitly mention, for instance, the kettle's handle, create structural coupling that may not be present if the requirements were first treated functionally. The author believes this feature of how requirements couple via a design explains the uncertainty that Alexander expressed in NSF.

The distinction between functional and structural aspects is also evident in Alexander's own words: "Some sets of misfits, in view of their interactions, seem naturally to belong together, and, taken as units, suggest physical form very strongly." If there is a justifiable reason for using a certain form that causes requirement coupling, then so be it. It is still better, though, to be given the choice of deciding whether such a form is in the best interest of the designed object's users. To ensure that the designer at least has the opportunity to choose, it is important to distinguish between the form and the function – particularly in the requirements (or misfits) – so that coupling is not artificially introduced.

Balance, Misfits, and Systems

The foregoing sections presented a number of problems that the author has identified in NSF, and has suggested another approach, based on *balance* and on *systems thinking*, that could address those problems. In this section, the author will describe some further details about this balanced system approach.

A system is a set of interacting elements (which may be other systems) that provide defined functions and is crisply distinguished from its environment or context (Karnopp et al, 1990). Systems thinking is based on viewing a domain of interest as consisting entirely of systems. Systems encapsulate function, and interact physically by exchanging mass, energy, and information. System properties emerge from the interactions of subsystem elements with the system's context (which includes other systems). Changing the location of system boundaries can completely redefine the functionality and purpose of a system. A system that interacts with its context well is one that *balances* properties of the context with properties available by the system itself. The author has discussed the notion of balance in design elsewhere (Salustri et al, 2009).

The author therefore proposes that designing start with a study of the "forces" that exist in a given situation. These forces may be economic, technological, or based on needs and desires of clients and users; they may be beneficial forces or forces that give rise to misfits.

The needs and desires must be described in terms of what is “wrong” with the current situation. This requires identifying the properties that are perceived (by the designer working together with clients, users, and other stakeholders) to change in some detrimental way. Boundaries are marked where properties change. For instance, sufficient productivity at one point in a production process may become insufficient at another point; the boundary lies between those two points and identifies a poorly balanced property of one of the systems. Similarly, the layout of some architectural space may be sufficient for one population of users, but not for another; here the boundary lies between those two populations, and marks a lack of balance between the space and the user population.

Boundaries mark the interface between systems. By first identifying the boundaries, one can use them to derive systems that emerge from the situation rather than from some preconceived, arbitrary organizational structure.

Each system thus identified may be further decomposable, depending on the nature of the situation, into subsystems. Such decompositions should be again based on the study of the actual situation rather than what the designers think should be present. This results in a hierarchy of systems. Each level of the hierarchy can be a network of systems, but because the details of each system are encapsulated within them, the interactions will tend to be far fewer than has been suggested by Alexander.

This hierarchy of systems must then be functionalized (that is, changed into a network of functions, without reference to form, but still situated within the given context). When functionalizing a system network, each form element (e.g. the kettle’s handle) must be treated separately because there may excellent reasons for some specific element to remain in the network. Examples of such reasons include: the client may be unwilling to assume the risk associated with removing a particular form element; or there may be safety regulations that require a particular element to be present (e.g. electrical cut-off switches, bulkheads, or warning signs). There is no way to know *a priori* if a complete functionalization of a network is possible; however, the more functionalized the network is, the less likely it is that the design situation will be over-constrained with arbitrarily assigned forms, and therefore the more likely that a good design solution will be found.

One then studies each level of the functionalized system hierarchy in turn. At each level, some interactions between systems may be quite beneficial, while others are in some kind of conflict. Those in conflict are equivalent to Alexander’s misfits. Our goal then becomes to re-balance the systems to eliminate – or at least minimize – the conflicts, while causing the least detrimental change to the beneficial interactions. This requires, as Alexander has proposed in NSF, to build a network of relationships. Instead of just a network of relationships between *misfits*, though, we need a network of relationships between *properties* (i.e. including relationships that show good fit and not just bad fit). We do this one level at a time in the system hierarchy, so that we can use encapsulation to temporarily ignore the details that would cause the overall problem to become intractably complex. The details of methods suitable to assist in balancing an unbalanced system are still being developed by the author; their discussion goes beyond the scope of this paper.

The author is currently working on a demonstrative example of this approach, derived from the kettle example in NSF. Although the example is still a work in progress, it has some features that may help illustrate the balanced systems approach suggested above.

For example, if we begin by considering the kettle as a single system (i.e. we do not refer to its constituents, only to the kettle as a whole), the number of its basic functional requirements are very few: to allow water to enter it, to heat the water it contains, and to allow the water in it to come out. The *extents* to which these functions are exhibited are specified with constraints that arise from the context of the kettle-system. For instance, the amount of water to be contained by the kettle would depend on the ways in which it is to be used. If its capacity is not consistent with its use, then the design is not balanced. To re-balance it, either the kettle’s capacity must be changed, or the kettle must be re-tasked to a different context.

Since the kettle’s purpose is to provide hot water, we can describe its major functional elements (subsystems) as: an access system (to get water into and out of the kettle), a heating system, a containment system, and a control system (to control the kettle’s dynamic behaviour). These subsystems interact to produce the required overall functions of the kettle. At the level of subsystems, further balancing will be required, but only insofar as interactions between specific subsystems are concerned.

Much more work remains to be done on this approach. However, insofar as it has been specified to date, it does appear to lend itself well to help structure design problems to facilitate creative design solutions.

References

Alexander, C. (1964). Notes on the synthesis of form. Harvard University Press.

Alexander, C., Ishikawa, S. and Silverstein, M. (1977). A Pattern Language: Towns, Buildings, Construction. Oxford University Press, London.

Clark, S.J. & Elms, D.G. (1976). The HIDECS process. *Report prepared by the Ministry of Works and Development.* New Zealand.

Dorst, K. and Cross, N. (2001). Creativity in the design process: co-evolution of problem–solution. Design Studies, 22(5):425-437.

Elms, D.G. (1983). From a structure to a tree. Civil Engineering and Environmental Systems, 1(2), 95-106.

Hofstadter, D.R. (1979). Godel, Escher, Bach: An Eternal Golden Braid. Vintage Books.

Karnopp, D.C., Margolis, D.L., and Rosenberg, R.C. (1990). System Dynamics: A Unified Approach. Wiley and Sons, New York.

Owen, C. (2007). Structured Planning: Advanced Planning for Business, Institutions and Government. Chicago: IIT Institute of Design.

Rose, C. (2000). Charlie Rose – A Conversation with Jeff Hawkins. Television interview, 3 July.

Salustri, F.A., Rogers, D., and Eng, N.L. (2009). Designing as Balance-Seeking Instead of Problem-Solving. *Design Principles and Practices*, 3(3):343-356.

Suh, N.P. (1990). The Principles of Design. Oxford University Press, New York.

Author Biography

Filippo A. Salustri

Filippo A. Salustri, Ph.D., P.Eng. has been teaching, researching, and practising design engineering since 1989. He has been involved with research and design of cars, aircraft, spacecraft, robots, temporary structures, toys, home appliances, and medical equipment. His research interests include formal and informal methods of designing, information visualisation, and web-based design tools. He is a member of the Design Society, the Design Research Society ASME, CSME, IEEE, and INCOSE; he is a founding member of the Canadian Design Engineering Network, and the Canadian Engineering Education Association. He is currently an Associate Professor of Mechanical Engineering at Ryerson University.