SIMPLICITY – A DESIGN PATTERN FOR IDEAS

Keith V. Nesbitt

Working Paper Series, Number 3, September 2013
ABOUT THIS SERIES

The Applied Informatics Research (AIR) Group is a cross-disciplinary, multi-institutional collaboration based in the School of Design, Communication & Information Technology at the University of Newcastle, Australia. The principal aim of the AIR Group’s Working Paper Series is to disseminate the research and/or technical output of the group in an easily accessible format. The content of Working Papers generally falls into one of the following categories:

- Preliminary findings or results, the release of which is intended to stimulate debate and/or discussion to assist in the further development of the research
- Technical reports associated with applied research that may be written in a less academic style than usually published in academic journals
- Extended versions of published works, often containing additional implementation/application detail, figures and tables.

The opinions or conclusions expressed in the Working Paper Series are those of the authors and do not necessarily reflect the views of the AIR Group as a whole.
SIMPPLICITY – A DESIGN PATTERN FOR IDEAS

Keith V. Nesbitt
School of Design, Communication and IT
University of Newcastle
2308, NSW, Australia
keith.nesbitt@newcastle.edu.au
www.knesbitt.com

ABSTRACT

Nature is composed of many complex systems. No system is more complex than the nature that governs the creation of ideas. Yet how many patterns are needed to model such complexity? The search for common patterns that govern the nature of complex systems is a fundamental goal of both science and art. New ideas may provide solutions to such problems but then the question arises as to how ideas are created, and whether they might be created in the same form?

Creating ideas is in essence a design problem. Problems require solutions and ideas are designed to provide them. One way to solve common design problems is to adopt, or adapt, a solution that has been useful in the past. Design patterns are intended as a more formal way of capturing these good designs, or design practices, so they can be reused. Although patterns were first introduced in the realm of Architecture they were quickly adopted by the Software Engineering community. In this paper, the use of patterns is adapted to the more general realm of idea generation.

In particular this paper introduces a specific design pattern called Simplicity that is intended to help with the practical exploration of a dynamic design space that is complete and allows for both inconsistent states and yet overall consistency. Simplicity is a pattern that reframes a number of well-known concepts such as complimentary dualities, recursion, self-similarity and symmetry to try and provide an alternative understanding of the way ideas are created and contextualised.

This paper, as it stands is incomplete, requiring much further mathematical formalisation to take it beyond the current pattern definition. Despite this shortfall, it is hoped that the current description may provide a useful step in the search for fundamental patterns that allow for unification of ideas across disciplines, a notion that has been termed Consilience.
Introduction

Nature is composed of many complex systems. The search for common patterns that govern the nature of these systems is a fundamental goal of both science and art. This raises some interesting questions, such as how many common patterns might be needed to model such complexity, and if some patterns are more universally applicable than others?

A compelling motivation for finding such patterns is the unification of knowledge across disciplines, a notion that has been termed *Consilience* (Wilson, 1998). If simple, fundamental patterns can be described that form the basic building blocks of knowledge, then there is promise for unifying ideas across discipline boundaries to create a common foundation for knowledge. Such common frameworks could help align and extend knowledge by identifying common ground, highlighting key differences and identifying unchartered territories.

Currently, disciplines such as science and art tend to adopt their own complex patterns of thinking, making it difficult to share and compare knowledge that is generated in disparate fields. The “idea space” of different fields can seem disjoint and yet one suspects they share many ideas in common. When considering the notion of an “idea space” further fundamental questions arise as to just how ideas are created and if it is possible to describe simple generic patterns that might be used to design all ideas?

Ideally what is required for *Consilience* is a few or perhaps a single pattern, that can be used to explore the entire design space of ideas. Such an idea-creation pattern could be thought of as an idea stamp, a simple template that could be used for generating ideas. If we could confine ourselves to a single pattern for creating ideas and some simple rules for using the pattern, then we might devise a system that could generate all ideas. Such a universal pattern would certainly be useful. Not only would this let us link common, existing ideas, it would guide investigations into areas of the “idea space” that are yet to be explored. It might provide a useful map for navigating the space of all possible ideas.

In this paper I will set out on the search for such a universal idea template by attempting to describe a general design pattern called *Simplicity*. It is a pattern that I originally found while studying temporal patterns in depth of market data in a Virtual Environment in 2001. Whether *Simplicity* is a pattern worth considering or communicating beyond an artistic way is a question I have struggled with since finding it. However, as *Simplicity* is also a pattern that I have found personally to be compelling and useful for both understanding old ideas and designing new ideas, then I will make a first attempt at describing it here.

I should note that the intention of this work is not to engage in philosophical debate but rather to move in the direction of engineering a pragmatic and useful tool for constructing and deconstructing knowledge. The aim is to describe a design pattern that is as universal and useful as possible, one that can be applied readily. As I have already suggested, a simple template for designing and analysing ideas across disciplinary boundaries.

Many of the elements that make up the pattern of *Simplicity* are not new and in many ways this paper merely provides a reframing of these concepts. Many of the elements of this pattern have existed in different fields for a long time. Indeed it is likely that related topics have been the subject of complex debate in fields such as Philosophy, Art, Computing, Design, Physics, Mathematics and others. My own knowledge of these areas is far from complete. Despite this I hope the reframing of
these concepts is useful enough in its own right to prompt further consideration of how ideas can be generated.

One final apology needs to be made in terms of trying to communicate the concepts of Simplicity. It is an abstract pattern in the extreme sense, aiming as it does to reach as close as possible to the limits of abstraction. Ideally, presenting this pattern in a more rigorous mathematical treatment would improve the nature of this communication and prevent misinterpretation of the pattern. However, at this stage the best mathematical framework is not clear to me, which is why I have chosen to simply formulate it in terms of a design pattern.

### Design Patterns

Creating ideas is in essence a design problem. Problems require solutions and ideas are designed to provide them. One way to solve common design problems is to adopt, or adapt, a solution that has been useful in the past. Design patterns are intended as a more formal way of capturing these good designs, or design practices, so they can be reused.

The idea for formal design patterns is attributed to Christopher Alexander who described over 250 problems in architecture along with descriptions and solutions (Alexander et. al. 1977). These problems and solutions together formed a “pattern language” for communicating good design practice. Such design patterns were identified at many scales and frequently would work together to solve specific design problems. For example, rooms, doors, windows and locks are common design patterns used in architecture. Such patterns are used repeatedly as they are well adapted to solving common architectural problems that exist in the real world. For example, windows allow light into rooms and when designed to open can also allow air to circulate. Locks can be useful on both doors and windows to limit access to a room.

The second community to adopt the notion of design patterns was the object-oriented software industry. The concept was popularised by the release of the book, Design Patterns:Elements of Reusable Object-Oriented Software by the “Gang of Four” (Erich et. al, 1995). This work describes three main types of design patterns. Creational design patterns describe ways in which objects are created and represented. Structural patterns describe how classes are composed into larger structures that generate new functionality. Behavioural patterns deal with algorithms or the actions and processes used to achieve outcomes. These behavioural patterns also document the way responsibility for actions are assigned between objects and how objects communicate to achieve particular behaviours.

Like Alexander’s work, the software patterns are described in natural language using lots of examples. As yet, neither architectural or software patterns provide a strictly formal system for describing patterns. Alexander’s work comes closer to being a full “pattern language”, although it falls short of providing a step-by-step approach for designing architectural solutions. In both works, templates are used to describe the patterns, although the templates have a slightly different focus in each case. Alexander’s patterns tend to focus on the problem domain while the software patterns focus more on the solution space. What both approaches do share in common is that they help describe the general design space rather than document a fully deterministic design process. In each case, both the problems and possible solutions need to be creatively interpreted.
Alexander’s patterns document abstract ideas about problems that need to be solved in architecture (Alexander et. al. 1977). The book by the Gang of Four documents ideas that provide solutions to common issues in software design (Erich et. al, 1995). Now I’d like to consider how we might apply the concept of a design pattern in the broader, even more abstract arena that is the design space of all ideas. In the next section I shall broadly describe the general problems encountered when creating ideas and look for the common features of idea creation. This will help introduce the general principles of Simplicity. These principles will then be formalised as a design pattern, a template of features that define the Simplicity pattern.

Creating Ideas

We live in a complex world populated by many complex ideas. Although arguably, ideas themselves are often relatively simple. Rather, it is the context of an idea that contains the complexity, as the full context of any new idea is the set of all previous ideas. Furthermore, the context of an idea is critical, as everything is interpreted in the light of the context in which it is created. What makes an idea good in one context can make it equally bad in another. Changing the context can make an idea seem either true or false, right or wrong.

1.1 The Context of an Idea

So when thinking about ideas, a good starting place might be to ask if it is possible to have ideas without context? If we assume the context is the set of all previous ideas then only the “first idea” would be without any context. Imagine we could create or uncover this “first idea”. The “first idea” would have stood alone, not interpreted in relation to any previous idea, without reference to any context. Furthermore, as the “first idea” it would also create the foundational context for all future ideas.

Unfortunately, it’s not possible to know where the first idea came from or what it is. There was nothing, zero and then there was this one, “first idea”. Even if we try to think deep enough or track back far enough, it is a bit like Zeno’s paradox, it’s not possible to reach the finish line, the ultimate context, the one, “first idea”. In idea creation, I like to call this the zero-one problem. There was nothing and then there was one idea. Because we cannot uncover this “first idea”, we have the problem of never being able to determine the full context of any idea.

Perhaps one of the reasons for the zero-one problem is that ideas tend to emerge in our minds from subconscious levels of thinking and so always have hidden layers of unidentified context. Even if we think deeper, trying to identify the absolute context, we can never be sure if we have reached the foundations of any idea. Many layers of context lay below each idea, and deeper levels of consciousness obscure each successive layer. We can’t get to the bottom of any idea. So this zero-one problem is not purely an abstract notion, it is one we constantly encounter in trying to uncover the source of our own ideas.

1.2 Interpreting an Idea

Although we cannot track down the “first idea”, it still seems reasonable to accept there is a “first idea” and consider it further. Because this “first idea” is without context, we immediately have an
interpretation problem. Without context it is not possible to make judgements about the “first idea”, such as, is it a good idea or perhaps a bad idea. Without context this “first idea” may have been true or it may have been false. We have to consider both possibilities. There is an implied symmetry or duality here. The “first idea” is either good or it is not, but without context we cannot classify it. Therefore, it’s not possible to create the “first idea” without creating its opposite. The “first idea” must actually consist of two mutually exclusive parts. I will adopt the symbols $\Phi_1$ and $\bar{\Phi}_1$ to identify the two parts of the “first idea”. You can think of the two parts of an idea as opposites or alternatives, where either part could be active depending on the context.

### 1.3 Creating a Hierarchy of Ideas

Assuming the “first idea” has been created, we can move on to consider the “second idea”, which would be created within the context of the “first idea”. This “second idea” will be interpreted within the known context that consists only of the “first idea”. This lets us start to build up a hierarchy of ideas. As we grow the hierarchy we introduce some further ambiguity. Because of the zero-one problem, we don’t know how to interpret the “first idea” and consequently we cannot interpret the “second idea” either. Therefore, as before we have to create two alternative parts when we generate the “second idea”. Once again I will adopt the same notation symbols and call the two opposite parts of the “second idea”, $\Phi_2$ and $\bar{\Phi}_2$.

If we were to try and interpret the “idea space” at this early stage of idea creation we would only have the first and second ideas and they could be used to generate four consistent possibilities in our “idea hierarchy” (figure 1). An example of this interpretation might be that the “first idea” is true and in this context the “second idea” is true. Another interpretation might be that the “first idea” is true and in this context the “second idea” is false. Alternatively, we might accept that the “first idea” is false and in this context the “second idea” is either false or true. Note that if we insist on consistency, this provides a complete list of the four mutually exclusive possibilities. There are also four possible inconsistent states that need to be avoided. For example, the “second idea” cannot be true when the “first idea” is true and also be true when the “first idea” is false. These inconsistent states are shown in figure 2.

As we can see, making judgements about the overall idea hierarchy, such as which part of each idea is good or bad, true or false, is difficult. Indeed, because of the zero-one problem we can never make definitive value judgements about the two different parts of any idea. We can never be sure where the context begins and just one more layer of context could reverse the value judgement, making good things bad or true things false. For this reason I prefer to consider the parts of an idea as either active or inactive rather than think in terms like true or false, or good or bad. In the figures used in this paper white and black coloured boxes are used to indicate which parts of the idea at each level in the idea chain are active and inactive.

So far we have only generated two ideas, but we can now repeat this idea generation process in many stages, creating further layers of ideas in ever deepening layers of context. Each time we create an idea, the context becomes more complex as each new idea is created within the contextual layers that come before it. Even after 4 layers the system becomes quite difficult to interpret and we could extend this process to an almost infinite number of previous layers.
Figure 1. This figure illustrates the four consistent states that are possible across the first two levels of our idea hierarchy. The black and white coloured boxes are used to indicate which parts of the idea at each level are active and inactive. An example of consistency is where $\Phi_2$ is active for only one parent and inactive for the alternative parent condition. For example, when $\Phi_2$ is active for the active $\Phi_1$ and inactive for the inactive $\bar{T}_1$. Note that the consistency implies that the same part in the second level of the hierarchy cannot be active under its parent alternative parts.

Figure 2. This figure illustrates the four inconsistent states that are possible across the first two levels of our idea hierarchy. The black and white coloured boxes are used to indicate which parts of the idea at each level are active and inactive. An example of inconsistency is where $\Phi_2$ is active for both parent ideas, for example when $\Phi_1$ is active and $\bar{T}_1$ is inactive. Note that inconsistency implies that the same part in the second level of the hierarchy is active under both its parent alternative parts.
Figure 3 represents three layers of ideas as a hierarchical chain and figure 4 represents the same three layers as an enclosed nesting of boxes. Both of these representations are equivalent but the alternative diagrams are useful to emphasise different aspects of ideas. The hierarchy layout (figure 3) captures the top-down and bottom-up layering that often occurs when designing ideas. The nested layout (figure 4) captures the recursive nature of context and the complexity that ensues when ideas are nested within each other.

![Diagram](image)

Figure 3. Creating a chain of context, starting at the zero context (yellow) and then adding three layers of new ideas. This represents one consistent state across these three levels, where white boxes represent the active part of an idea at each level and black boxes inactive parts.

![Diagram](image)

Figure 4. This figure represents the same context chain shown in figure 3. Once again white boxes are active parts of an idea and black boxes are inactive. The use of enclosure reinforces the recursive nesting of context described in Simplicity.
1.4 Consistent and Inconsistent Idea Hierarchies

Another thing to note about Figure 3 and Figure 4 is that they represent the same state and that this is a consistent state. This state is consistent because for all idea nodes only one of the parts, either Φ or Φ, is active and the children of this higher level have opposite parts active. For example, in the level 1 parent node, Φ₁ is active while Φ₂ is inactive and for the children, Φ₂ is active under Φ₁ while Φ₂ is active under Φ₁. However, this is not the only possible consistent state, with three levels in the idea hierarchy it is possible to describe 16 states of the hierarchy that are consistent (Appendix A) as well as 16 inconsistent states.

While such idea hierarchies seem quite simple to begin with they quickly give rise to logarithmic complexity (Table 1). The number of different consistent states with 2 levels is 4 ($2^2$). The number of possible consistent states with 3 levels is 16 ($2^4$) and the number of consistent states across 4 levels is 256 ($2^8$). For each new level the number of states squares, so for 5 levels the number of states is the number of states at 4 levels squared or 65,536 ($2^{16}$). This complexity increases to 4,294,967,296 ($2^{32}$) with just 6 levels. The number of states can be calculated with a simple recursive algorithm (Table 2). Given that for 2 levels there are 4 possible states, in general when the number of levels is greater than 2 the number of consistent states, $s(n)$ can be calculated as $s(n-1)^2$. Note that at each level there is an equal number of consistent and inconsistent states possible.

<table>
<thead>
<tr>
<th>Number levels</th>
<th>Number of consistent (inconsistent) states</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>s(n)</td>
</tr>
<tr>
<td>2</td>
<td>4 $2^2$</td>
</tr>
<tr>
<td>3</td>
<td>16 $2^4$</td>
</tr>
<tr>
<td>4</td>
<td>256 $2^8$</td>
</tr>
<tr>
<td>5</td>
<td>65,536 $2^{16}$</td>
</tr>
<tr>
<td>6</td>
<td>4,294,967,296 $2^{32}$</td>
</tr>
</tbody>
</table>

Table 1. The number of possible consistent (and inconsistent) states $s(n)$ across n levels of an idea hierarchy.

<table>
<thead>
<tr>
<th>Number levels</th>
<th>Number of consistent states</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>s(n)</td>
</tr>
<tr>
<td>n=2</td>
<td>s(2) = 4</td>
</tr>
<tr>
<td>n&gt;2</td>
<td>s(n) = s(n-1)^2</td>
</tr>
</tbody>
</table>

Table 2. A simple recursive calculation for determining the number of consistent states $s(n)$.

1.5 The Dynamic Parts of an Idea

To summarise so far, the context of an idea is critical and yet we can never fully define it. Even for very basic ideas, we can never be sure of the starting point of the context. As such all ideas might be active or inactive given an alternative context. Furthermore, since the “first idea”, the first context, will form the foundations of all new ideas, all new ideas must contain uncertainty and need to be created as two mutually exclusive, opposite parts. Only one can be active in any context yet the real state of this context is not completely known due to the zero-one problem. Therefore, for completeness when we create a new idea we have to create it’s opposite.
We might expect that a further result of the zero-one problem is that the final state of any idea would remain dynamic, switching between two possible active states. This foundation of ambiguity could create a constant problem for reasoning, as there is no absolute foundation for certainty.

This potential cognitive problem is akin to ambiguous interpretation issues that can be seen, for example, when looking at an image and trying to determine figure from ground. There are a number of perceptual interpretation problems that illustrate this phenomenon quite well. For example, the Rubin Face/Vase Illusion, often discussed in Gestalt psychology, is a typical bi-stable, figure and ground problem (Figure 5). In this example the relationship of figure and ground switches between the white vase in the centre and the two black face profiles on either side. While both interpretations are possible only one interpretation as to which part is figure and which part is ground can be perceived at one point in time. As in Simplicity, both parts are possible but only one part can be active at any time.

Figure 5. Rubin vase showing the ambiguous figure and ground. The perceived figure alternates between the vase and the two faces creating a bi-stable perceptual ambiguity.

So given that all ideas have a foundation of uncertainty, this raises the interesting question of what makes some configurations of ideas seem more stable than others, and when might a bi-stable ambiguity emerge? We have previously introduced the notion that some states in a chain of ideas are inconsistent and these could also be considered as unstable. As we further investigate this question of stability, we will consider the notion that symmetry across the levels of the hierarchy and how that may affect the instability of both consistent and inconsistent states. Indeed the more levels we add to the idea hierarchy, the more we need to consider the overall effect of symmetry on the hierarchy and begin to describe some consistent states as somehow more symmetric, and so more stable, than other consistent states. However, before taking up further this discussion of symmetry, we will consider the types of transitions that can occur in an idea hierarchy.
1.6 Transitions between Idea States

To begin looking at transitions in the idea hierarchy let us consider a very simple arrangement with just two levels. With two levels we can identify eight different states or configurations. Four of these can be considered consistent configurations (figure 6), while a further four inconsistent states can be described (figure 7). The possible transitions from stable and unstable states are shown in figure 8.

**Figure 6.** The four, consistent, symmetric configurations that can be identified across two levels of the Simplicity hierarchy. Since these symmetric configurations are consistent, without further information there is no impetus to change the configuration and so they can be considered stable.

**Figure 7.** The four, inconsistent, asymmetric configurations that can be described across two levels of the Simplicity hierarchy. Because of this inconsistency there is an inherent instability in the state, that seeks to move the system to one of the stable symmetric states.
Figure 8. The possible transitions between possible states, for two levels of the simplicity hierarchy. Note that states 5, 6, 7, 8 (red) are inconsistent and so inherently unstable.

A transition between stable states might occur if new information comes to bear on which part of the idea is active at the parent level (n) or in one of the child nodes (n+1). For example, a transition from state 1 to 3 would occur if the parent node changed from $\Phi_n$ being active, to $\Upsilon_n$, being active. A transition from state 1 to 2 requires two changes to occur in the child nodes, with $\Phi_{n+1}$ moving from active under $\Phi_n$ and inactive under $\Upsilon_n$ to the opposite configuration where $\Phi_{n+1}$ is inactive under $\Phi_n$ and active under $\Upsilon_n$. Unless changes in the child nodes occur simultaneously, these transitions between stable states require an initial move to one of the unstable states. For example, the transition from state 1 to 2 will require a transition through either of the inconsistent states 5 or 6.

More complex chains of transitions are also possible, for example, from 1 to 3 to 7 to 4. This transition affects activity in both the parent (n) and child (n+1) nodes. We could also consider this a top-down driven change as the parent changes first. A similar transition, which is bottom up in nature, would be a move from 4 to 7 to 3 to 1. In this case the transition begins with a change of the child nodes.

In terms of idea generation, the consistent configurations (figure 6) contain no ambiguities. This raises the question of what would prompt a transition to an alternative state? It is easy to associate some cost or form of energy that is required to make such transitions. Given there is some cost for
transitions to occur, then we can propose that the drivers to change might lead to a preferred state. We can further imagine that drivers for changing state can act in either a top-down or bottom-up manner. A top-down change represents a change of context, for example, moving from a state where \( \Phi \_n \) is active (states 1 or state 2) to the alternative state where \( \TeX \_n \) is active (states 3 or state 4). Alternatively, a bottom-up change may occur if the information comes to bear that the inappropriate child part is active under the current active context. The simplest type of bottom-up involves no change in active part of the parent node, with the transition only occurring in the child node. For example, \( \Phi \_n \) is active and remains active, but under this context, the child part \( \Phi \_n \_1 \) changes from being active to inactive (a transition from state 1 to 2). This might occur, for example if information came to bear that under the current context \( \Phi \_n \), that \( \TeX \_n \_1 \), rather than \( \Phi \_n \_1 \), is the most appropriate part to be active.

Inconsistent states (figure 7) are all unstable in that these configurations contain some fundamental inconsistency that needs to be resolved. We might expect that such unstable states are very dynamic and the cost of transitioning to a consistent state is efficient if it results in a less dynamic configuration of a more stable state.

Resolving an ambiguity in an inconsistent state could be achieved by assuming a particular context at the parent level or by gathering definitive information that specifies the state of one of the ambiguous child nodes. For example, if we consider a state where \( \TeX \_n \_1 \) is active for both \( \TeX \_n \) and \( \Phi \_n \) (state 6 or state 7) then it needs to move to a consistent state where \( \TeX \_n \_1 \) is active for either \( \TeX \_n \) and \( \Phi \_n \) but not for both. To do this may require decisive information about either \( \Phi \_n \_1 \) or \( \TeX \_n \_1 \) under the parent context. If, for example, information becomes available that \( \Phi \_n \_1 \) is the preferred state under the context \( \Phi \_n \), then \( \TeX \_n \_1 \) must be active under \( \TeX \_n \). This information would drive a transition to a stable state (state 1).

### 1.7 Symmetric Idea Hierarchies

So far we have only considered transitions across two levels of an idea hierarchy, however we can extend this discussion of transitions over many more levels. Before we do this we should consider the notion that some hierarchy configurations are more consistent than others. Across many levels it is possible that some parts of the hierarchy might be well composed or consistent while other parts are inconsistent. The most consistent representation is the hierarchy with the most symmetry. This is a hierarchy where one half of the hierarchy has exactly the opposite parts active at every level in the hierarchy (figure 9). These two parts are like a lock and key. Each half perfectly aligns with the other to describe two incompatible and opposite states. This represents an ideal symmetric state, consistent across levels, or as stated above, with the lock and key matching (figure 10). An advantage of this state is that reversing the context at a high level does not impact on the activity of the child nodes.

Overall, we can imagine there would always be a drive to move to the most ordered, symmetric state as overall it requires less energy to maintain. I think of this as the force of symmetry. Of course the cost of transitioning may be extreme and it may require a transition through a number of less stable states to reach this goal. To reach this state may also require the resolution of ambiguities at
many levels in the hierarchy. This drive for consistency, for symmetry, might need to operate, via a ripple effect over multiple levels of the hierarchy or even precise parts of the hierarchy.

Figure 9. Four of the 16 consistent states possible with a 3 level hierarchy. Note that states 1 and 2 are fully symmetric across all three levels. That is the left and right half of the hierarchy are complete opposites. States 3 and 4 while still consistent have a lower order of symmetry as third level of the hierarchy is the same rather than opposite.

A number of types of dynamic behaviour can be proposed in this system driven by an inherent goal to reach the most consistent symmetric state. One type of behaviour is a ripple down effect, where the context of a higher layer changes state, causing a subsequent reversal of the child nodes. A second type of dynamism is a ripple-up effect, where changing activity of the child node forces a change in parent activation.

A ripple up affect could occur if enough child nodes changed state creating an ambiguity or asymmetric alignment. If while working on a layer and suddenly found all of the ideas to be incorrectly active, this might imply that something in the context of the ideas is incorrectly active. To fix this irregularity requires a revision of the appropriate context, the layers of ideas that sit above the current layer must be changed to accommodate the most viable active states at the bottom layer. The only way to force the system into an overall consistent state might be to switch the higher-level context nodes to an alternative state. Like the ripple down effect, this ripple up effect might need to go across many layers of context before the system could arrive at even a partially consistent state.
Both a ripple down and ripple up effect might place the hierarchy in an inconsistent state. If no suitable information is available then the inconsistency may be difficult to resolve and this may create a cycle between inconsistent states in an attempt to resolve the ambiguity.

For example, the hierarchy might also find itself locked at any level by a particularly well-informed or stable configuration in just part of the hierarchy. Consider the case where we change the state of a high level context creating a ripple down effect. Now ideally all the child ideas would swap their current states to create consistency. However, it is possible that at some point in the ripple down, a layer is resistant to change or in effect locked. A locked state might be one with a particularly good symmetry across a few layers of the hierarchy. Meeting this locked layer, the ripple down effect might be blocked. It might even generate a subsequent ripple up effect, switching the state of the layers above in an effort to maintain consistency. This could lead to a cycle of competing ripple up and ripple down changes.

1.8 The Scale of Ideas

So far we have avoided discussion of the scale of our ideas. How big are the ideas that we have been discussing? Ideally we might like to break down each layer of the simplicity hierarchy into very small discrete components. These components might be so small that the answer to the question, “what layer are we starting on?” is also ambiguous. Thus it might not be clear if we are on level n or level n+1. Indeed each of these two layers can also be modelled as two opposite parts using Simplicity, providing two mutually exclusive possibilities. We are either on level n or level n+1 but we are not sure which.
However, it becomes impractical to really break down ideas to such a small scale. Fortunately the pattern of Simplicity can emerge at many scales and we might describe it as self-similar, or scaleless. Therefore a more practical solution is to choose a division between layers in the hierarchy that leads to alternating possibilities. For example, the Rubin vase figure-ground problem illustrates the principles of bi-stable ambiguity at the level of perception. However, the principle also occurs at higher levels of cognition. For example, we interpret many of life’s contraries as mutually exclusive, either/or dichotomies, that is to have a complementary nature (Kelso and Engstrom, 2006). Some examples of these larger ideas include: individual~collective, self~other, body~mind, nature~nurture, cooperation~competition (Kelso and Engstrom, 2006).

1.9 Constructing and Deconstructing Idea Hierarchies

We motivated this paper with the search for Consilience, an aim for unification of knowledge into a single well-defined idea space. Ideally we might imagine a perfectly modelled, symmetric hierarchy of all ideas, each at the smallest scale. Unfortunately, while it is possible to think about the lowest levels of ideas, the first and second ideas and so on, it is problematic to actually define what these are. Furthermore, as we construct a hierarchy, the possible states beyond 4 or 5 levels makes working the hierarchy difficult, especially if the intention is to create a large, fully symmetric hierarchy. Certainly, to describe the full hierarchy of all ideas is difficult, and regardless, any practical attempt to think recursively about context beyond about four levels is impractical. Fortunately, Simplicity doesn’t require perfectly modelled, or even complete or fully symmetric hierarchies to be useful.

Smaller hierarchies are much easier to construct and still useful. Usually in idea generation we have to assume some starting context and proceed from there anyway. Thus we can select a portion of the total idea tree and start to generate ideas from there. As long as we acknowledge the zero-one problem we can begin to construct a hierarchy at any level. This doesn’t need to be the same level for all applications.

There are other ways to reduce the size of the hierarchy. Because of the self-similar property, ideas in a hierarchy can be at any scale as long as they represent complementary opposites. Furthermore, the symmetric parts of the hierarchy can be collapsed into a single node in the hierarchy to reduce the number of levels. For example, the symmetric 4-level hierarchy shown in figure 10 can be reduced to a single level.

Nor do we need to build up an idea hierarchy from the lowest level. We can take existing parts of a hierarchy and join them or use them as starting trees. In general it might be easier to start with a small, symmetric hierarchy and construct its opposite half, than trying to change a very asymmetric configuration into a more symmetric design. Another way to think about this process is to consider an evolutionary metaphor where ideas evolve over time to a more symmetric state, with the fitness function forming a key layer of context.
Key Features of Simplicity

In the following section I have tried to describe the basic concepts that make up the simplicity pattern. In this section I will formalise the 5 main features that describe the pattern.

1.10 Generality

Every idea is generated in the same way. That is, there is a single pattern that creates an idea. Simplicity is this fundamental design pattern from which all other ideas can be generated.

1.11 Zero-One Problem

All ideas are created within a context. The context can be defined recursively as a nested layer of previous ideas. Each layer simply adds one idea to the previous context, creating an even larger context. The recursion, or layers, are infinite as there is no way to define the initial or first idea. This is the zero-one problem, at some level there is nothing (zero) before the initial idea (one) was created, but it is not possible to determine this beginning point, that is, the first idea that defines the recursive context grounding subsequent ideas.

1.12 Complementary Duality

All ideas are composed of two complimentary parts. These parts represent “active” and “inactive” versions of the same idea. Because of the zero-one problem it is not possible to know which part is “active” or “inactive” as this depends on interpreting a context that is unresolved. In a sense there is a singularity between the two parts. Given any context as a starting position, the idea at the current level can converge to one of two possible states, represented by the two parts of the duality. It is not possible to predict which state it will converge to.

1.13 Force of Symmetry

All idea hierarchies are dynamic and symmetric by nature as they are composed of two equal but opposite parts. However, the degree of symmetry and stability varies. Although contexts are dynamic, more symmetric configurations are more stable than less symmetric ones. Symmetry is a powerful self-organising force that can reinforce structure at a distance, across the layers of context. The force of symmetry is proportional to the depth of the hierarchy. Overall, there is a drive to obtain the most ordered symmetric state possible across as many layers as possible.

1.14 Self-similarity

The ideas that make up the Simplicity pattern are self-similar and scaleless. The symmetric duality or complementary nature of ideas emerges at many scales and is reinforced by symmetry.

Conclusion

This paper considers the intriguing and open question; are there abstract patterns that apply to the general notion of creating and adapting ideas themselves? The intention of describing Simplicity is to explain a pattern that captures something fundamental about the way ideas are created. Ideally, it might lead to a pattern that can be used to generate all the ideas we are already familiar with as well
as those yet to be uncovered. That is, a pattern that can populate the entire design space of all possible ideas.

Of course, many things discussed in this paper are not new. Concepts such as recurrence, duality, self-similarity, bifurcation, evolution and emergence are key ideas described in fields such as “Complexity Theory” (Bay-Yam 1997). Although many parts of the Simplicity pattern have been well explored in other domains, the intention here is to reframe these concepts in a novel way, addressing such questions as, “Are some patterns more universally able to describe the way ideas are created?” and, “Is it possible to describe a simple pattern that can generate all possible ideas?”

The motivation for this study was the practical aim of Consilience, defining a single framework by which knowledge across all domains can be described. A framework that could help unite common ideas from different disciplines and guide navigation of those uncharted areas of the idea space yet to be explored. It is intended that this framework can be applied in a practical sense rather than simply drive philosophical discussions.

Unfortunately, to describe such basic patterns as Simplicity, a pattern that applies to something as general as ideas themselves, is problematic. Before we begin there are problems with self-referencing, as Simplicity is itself an idea. Unfortunately, abstraction also raises problems of interpretation. “Nothing”, might arguably be the most useful pattern as it allows for the most possible interpretations.

To avoid interpretation problems, I have tried to formalise the ideas in an axiomatic way as a pattern. This design pattern approach unfortunately falls well short of describing a strictly deterministic formal system that might be enjoyed by mathematicians. Even so, Simplicity as a pattern already raises some interesting questions in relation to the general modelling of systems. For future work, a more mathematical description of this system would be an ideal goal.

For example, it might be interesting to develop more formal theory surrounding questions about the completeness and consistency of Simplicity. According to Goedel’s theorem, it is not possible to describe consistent systems that are complete. Simplicity is complete by definition, at least as every idea is included. Simplicity is also based on a kind of inconsistency where any idea, I and its opposite ~I could be true at some point of time, although both cannot be active simultaneously. In first order mathematical logic we would normally expect that (~I ∨ I) is always a true statement and that (~I ∧ I) is false. Simplicity allows that (~I ∧ I) can be true. This is qualified by saying that both I and ~I cannot be true at the same time as this would lead to inconsistent logical models. Perhaps it would be better to express this time dependency by saying that (~I(t_x) ∧ I(t_y)) can be true. Where t_x represents a discrete moment of time and provided (x≠ y).

It is the dynamic nature of Simplicity that allows this inconsistency to exist, for both states cannot be in the foreground, or active, at the same point. The system state is time dependent but potentially consistent at any point in time. To preserve completeness it is possible that the active state could fluctuate in a dynamic and even rhythmic fashion. If an observer is observing the system in the same rhythm as the fluctuating states, then they would always observe a consistent system. An alternative
observer, that observes when the first observer is not, might also encounter an alternative, opposite but consistent system, that is inconsistent with the first observer’s view.

Indeed extending the adoption of a two-observer model for all systems is a natural outcome of *Simplicity*. The two observers adopt opposite contexts or viewpoints that must remain separate, while converging towards a single viewpoint, the zero-one problem prevents any absolute convergence. The goal of the system modelling is to obtain a perfect symmetry between observers.

This paper has raised the interesting questions of how many patterns might be needed to model the complexity of all human ideas, and if some patterns are more universal, or indeed more useful, than others for this purpose. A unified theory of man’s knowledge, based on a single, simple pattern, is yet to emerge. Described as *Simplicity*, this paper begins the search to describe such a fundamental pattern. *Simplicity* is a first step in the search for a pattern that is the basis of all ideas and provides an explanation of the way they are connected.

**References**


Appendix A – The 16 consistent states that are possible in a 3 level idea hierarchy