Wiki as Pattern Language

Ward Cunningham
Cunningham and Cunningham, Inc.[1]
Sustasis Foundation
Portland, Oregon

Michael W. Mehaffy
Faculty of Architecture
Delft University of Technology[2]
Sustasis Foundation
Portland, Oregon

Abstract

We describe the origin of wiki technology, which has become widely influential, and its relationship to the development of pattern languages in software. We show here how the relationship is deeper than previously understood, opening up the possibility of expanded capability for wikis, including a new generation of "federated" wiki.

[NOTE TO REVIEWERS: This paper is part first-person history and part theory. The history is given by one of the participants, an original developer of wiki and co-developer of pattern language. The theory points toward future potential of pattern language within a federated, peer-to-peer framework.]

1. Introduction

Wiki is today widely established as a kind of website that allows users to quickly and easily share, modify and improve information collaboratively (Leuf and Cunningham, 2001). It is described on Wikipedia – perhaps its best known example – as "a website which allows its users to add, modify, or delete its content via a web browser usually using a simplified markup language or a rich-text editor" (Wikipedia, 2013). Wiki is so well established, in fact, that a Google search engine result for the term displays approximately 1.25 billion page "hits", or pages on the World Wide Web that include this term somewhere within their text (Google, 2013a).

Along with this growth, the definition of what constitutes a "wiki" has broadened since its introduction in 1995. Consider the example of WikiLeaks, where editable content would defeat the purpose of the site. We will exclude discussion of these other, broader uses of the term, and confine our discussion to the original wiki, and to the popular encyclopedia, both of which provide sufficient breadth to illustrate our points.

While the general concept of a wiki is thus extremely well known, somewhat less well known is the history of wiki development. Wikis were an outgrowth of the development of what is known as "pattern languages" in software, or as they are sometimes referred to, "design patterns." As we describe below, they were in fact developed as tools to facilitate efficient sharing and modifying of patterns. In part for this reason, the structure of wikis itself bears a relationship to the structure of patterns and pattern languages – a relationship that, as we will also discuss, offers intriguing new

[1] ward@c2.com
[2] michael.mehaffy@gmail.com

opportunities. This relationship, and the evolving opportunities it presents, will be a central focus of this paper.

Specifically, we will present a new approach to wiki that includes greater capacity to handle and process quantitative elements. This new approach makes greater use of the logic of pattern languages within the structure of wiki pages. It also exploits the power of "federated" open-source development, as we will explain below.

We will close by discussing the implications for wiki technology specifically, and for the collaborative growth of knowledge more broadly. We will draw attention to the need, in an age of explosive and potentially chaotic growth of information on the web, for new strategies of "curation" of knowledge toward effective future problem-solving.

To understand how this development followed naturally from the earlier work, it is necessary to consider the history of pattern languages.

2. Early development of pattern languages

The general logic of pattern languages is closely related to the hierarchically compressed structure of object-oriented programming. A series of re-useable information packets can be manipulated as units, within a grammar-like set of rules for combining and exchanging their inputs and outputs. However, as we will discuss, pattern languages have particularly useful attributes that are more similar to those of natural language.

The formalized concept of a "pattern language" was developed by the architect Christopher Alexander, growing out of his work published in the book *Notes on the Synthesis of Form* (Alexander, 1964). Alexander was seeking to understand how forms arise as solutions adapted to specific configurations of problems, but then can be generalized for other similar uses. The problem had new relevance for a cybernetic age in which complex technological systems were becoming routine, and new design strategies were needed. As Alexander noted in the introduction, "Today functional problems are becoming less simple all the time. But designers rarely confess their inability to solve them." Instead, Alexander argued, designers fall back on an arbitrarily chosen formal order, with negative consequences.

This description had its counterpart in the work of Herbert A. Simon about the same time, in his classic paper "The Architecture of Complexity" (Simon, 1962). Simon was searching for principles of structure among complex systems, and he noted the tendency of complexity to take the form of "nearly decomposable hierarchies." The elements of such complex systems often had strong interaction capacities as well as weak ones, and the strong interaction capacities tended to form hierarchical subsystems, allowing the system to be "nearly decomposed" within our models – that is, grouped into functional sub-units that could then interact in a more usefully comprehensible way.

Alexander also identified strong and weak relationships, with the strong relationships forming what he termed "diagrams" (later "patterns"). These subsystems could also then be treated as recombinable units within design models, following grammar-like rules. Alexander observed that something similar does happen in human language – and indeed, in the design processes of vernacular cultures, where "patterns" were identifiable elements of design. This formed the basis of a new design method that mimicked the features (and recaptured the usefulness) of previous vernacular methods.

"A pattern language has the structure of a network," wrote Alexander and his colleagues in the book *A Pattern Language* (Alexander et al., 1977). "Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice." In that book and its companion *The Timeless Way of Building,* Alexander and colleagues outlined the theory behind this language-like approach to design, drawing on structuralist influences like George A. Miller and Noam Chomsky (Alexander, 1979).

In essence, pattern languages are clusters of elements that form a solution to a problem, and that tend to recur in a pattern-like way. These elements are related to each other via "strong forces" – forming the requirements for relationship between the elements of a successful solution.

For example, the hinges and handles of a door must be in a particular configuration in relation to one another for the door to work successfully: generally, they must be on opposite sides of the door. *(Figure One.)* This is the resolution of the rotational forces of the door, which allows the user to operate the door with ease. (Imagine a door with the hinges and knob on the same side, and you can begin to see why this resolution of strong forces is critical in design!)

But in most cases, two *separate* doors are connected to one another with only weak forces – that is, in terms of their ease of use, there is little consequence to variations in their placement in relation to one another. Placing one door closer to another does not make that door easier or harder to operate. This relationship, then, is considered a "weak force."

These forces operate within a scale-free domain – that is, a weak force at one scale might be regarded as a strong force at another. (A placement of doors might not matter at the scale of the doors, but might matter a great deal at the scale of a room.) Thus a "pattern" is a cluster of stronger forces at one scale, which can be combined with other clusters, using grammar-like relationships.

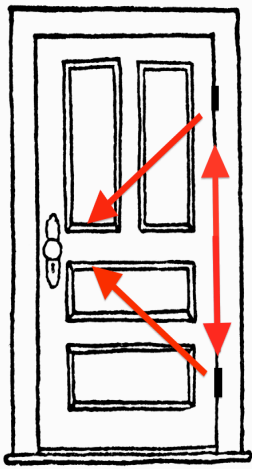

FIGURE ONE. A simple pattern, "Door," resolving strong forces between the hinges and handle.

Indeed, most design problems – including software design problems – follow this kind of structural logic. Therefore it is possible to abstract the relationships within a recurrent problem, and treat them as objects within a generative design system. Such a system can in principle (and does in practice, as we

will discuss) allow a much more efficient development of design solutions for a wide range of problems.

It is noteworthy, then, that "pattern languages" are less a new technological invention per se, than a kind of discovery, within the logic of problem-solving – and indeed, within the logic of nature itself, in a sense. Alexander himself made this point recently[3].

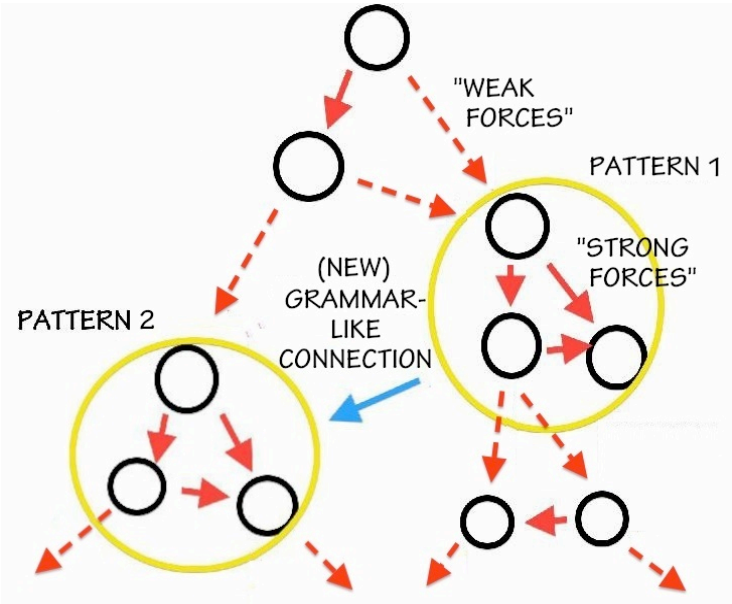

FIGURE TWO. The structural logic of pattern languages.

In an even more fundamental sense, both Alexander and Simon were reconsidering the age-old philosophical topic of *mereology*, the nature of part-whole relations, but doing so from the perspective of a cybernetic age in which the elements of design were vastly more numerous. Both identified strong and weak classes of relationships, and the tendency of strong relationships to form "nearly decomposable" clusters within hierarchical groupings. But Alexander focused more than Simon on the inter-connected, not fully decomposable nature of the patterns – that is, the network aspects – and their crucial importance in creating the web-like structure of many languages – and many successful designs. Alexander used these insights to mount an incisive mathematical critique of modern planning, writing in a celebrated 1965 paper titled "A city is not a tree" (Alexander, 1965).

But the later book *A Pattern Language* – moving beyond the earlier critique and proposing an alternative, network-powered methodology – quickly overtook his earlier work to become Alexander's best-known publication, and it remains a perennial best-seller more than 35 years after its publication. At this writing (2013) it hovers at about the 7,000 total sales rank on Amazon-US, in spite of its expense ($65 retail, $40 through Amazon), and is also the highest ranking book in Amazon's category of architectural criticism (Amazon.com, 2013). Yet interestingly, William Saunders, former editor of Harvard Design Magazine, has pointed out the dilemma that "[A Pattern Language] could very well be the most read architectural treatise of all time, yet in the architecture schools I know, it is as if this book did not exist" (Saunders, 2002).

---

[3] Private communication with one of the authors, 2009. When asked "what was the most significant thing you've learned about pattern languages, looking back on them after all these years," Alexander answered, "I thought I was making an invention – but I was making a discovery. This is a structural aspect of nature that I was describing."

We will have more to say about why *A Pattern Language* may not be as influential in architecture – the very field for which it was created – as in other fields. First, we turn our attention to the field in which pattern languages have clearly had the most impact to date, software design.

3. Pattern languages in software

While many architects seem to have ignored Alexander's work – perhaps not so surprising, since he was unapologetically critical of their professional status quo – software engineers and others were following the work with considerably more interest. A number of influential programmers of the 1970s were already aware of and influenced by *Notes on the Synthesis of Form,* including Ed Yourdon, Tom DeMarco and Larry Constantine (Yourdon, 2009). But beginning in the 1980s, *A Pattern Language* formed the basis for what is now known as "pattern languages of programming," or "design patterns" – general reusable solutions to commonly occurring problems within given contexts (Beck and Cunningham, 1987).

As described by Erich Gamma, a pioneer of pattern languages in programming, "This approach to patterns differs from [Alexandrian] pattern languages: Rather than coming up with a set of interwoven patterns top-down, micro-architectures are more independent patterns that eventually relate to each other bottom-up. A pattern language guides you through the whole design, whereas we have these little pieces, bites of engineering knowledge. I confess that this is less ambitious, but still very important and useful..." (Gamma, 2005)

Nonetheless, the ability to manage complex systems through design strategy was a common goal. In software patterns, as in Alexandrian patterns, the value was precisely in their flexibility and language-like adaptability. They gave useful guidance to designers generally, and to programmers specifically, in the form of structured essays offering solutions to recurring problems in context. The total set of patterns available constituted a body that evolved through repeated application and evaluation. As we will see, that capability would prove to be a crucial goal – if one that remained only partly met (Kerth and Cunningham, 1997).

4. Pattern languages in other fields

Since the publication of the book *A Pattern Language*, the structure of pattern languages has been applied to a dizzying number of subjects. A Google search quickly reveals applications in human-computer interaction, management, service design, economics, communication, education, engineering, landscape design, and dozens of other fields (Google, 2013b). One can even find references to pattern languages for weddings, and for pattern writing!

Interestingly, the term "pattern language" draws some 477,000 Google search hits – yet the term "design pattern," the more common term used exclusively for software, draws some 5 million hits, a ten to one ratio (Google, 2013c). Clearly this ratio suggests a far greater impact of pattern languages in software than in architecture – which is only one of many fields affected by pattern languages, and only one of many using that term.

It is worth noting that the interest in pattern languages in other fields was fueled in large part by the rapid growth of interest in software. As software work was applied to many other fields, practitioners in those fields also took note of the use of pattern logic, and in many cases, extended its development to other topics.

5. Development of wiki

The influence of wiki is notably greater than that of design patterns – as one indicator, its Google hit score of 1.25 billion is over 250 times greater than that of "design pattern," and 2,500 times greater than "pattern language." What does this tell us? It may indicate that a method has been arrived at that combines ease of use with collaborative power.

The best-known example of wiki is surely Wikipedia, the on-line encyclopedia that is currently the most-used content site on the World Wide Web. (This definition excludes search engines like Google, which use content from other sources including Wikipedia.) But as a quick perusal of the web will readily demonstrate, there are many hundreds or thousands of other kinds of wikis, spanning many diverse fields. There are wikis for medical information, real estate, legal data, and national intelligence, to name a few. And of course, there are many wikis for software development, including sites created by Google, Microsoft and IBM. (S23.org, 2013.)

As these examples suggest, the power of wikis is in their very diversity, and their ease of handling many unique and specific features that are inter-linked, while also being able to provide universal approaches and organizing frameworks. In that sense, they combine the ability to identify useful generalities with the ability to identify new particulars. Moreover, they do so in a way that allows these capabilities to improve with time.

This is because, like patterns, wikis are bodies that evolve through repeated application, evaluation and refinement. The improvement of Wikipedia over time, for example, is legendary. In the early days of Wikipedia, it was common to hear comedians lampoon the many errors that were contained in its articles. But over time, errors were spotted and corrected, to the point that a recent study concluded that Wikipedia error rates are comparable to those of academic encyclopedias, although Wikipedia has far more entries on many more specialized subjects (Giles, 2005).

The evolutionary relationship between wikis and patterns languages is a simple one. Wikis were developed in 1995 as a tool to support the development of pattern languages in software (Cunningham, 2009). More specifically, it was developed as a tool to allow collaboration between many people as they evolve better collective solutions to shared problems. That means, among other things, that both the knowledge of the problem, and of the solutions that have worked so far, needs to be captured and refined, in a form that can be shared and further refined (Leuf and Cunningham, 2001).

6. Wikis as elementary pattern languages

We believe, in fact, that wikis and pattern languages share fundamental structural characteristics – and that it is not too much to claim that wikis (in their originally intended form) are, in fact, a form of elementary pattern language. They both share the following unique set of characteristics:

A. Both are open-ended sets of information, consisting of unitary subsets (pages or patterns) connected by hyperlinks. Each set of information is able to expand, while remaining within a linked network.

B. Both are topical essays with a characteristic structure: overview (with links), definition, discussion,

evidence, conclusion, further links. This limited structure creates the capacity for extensibility and interoperability – the capacity of new pages to function smoothly with older ones, with the capacity for open-ended growth.

C. Both are structured to be easily creatable, shareable and editable by many people. This capacity facilitates the creation of user communities, who are crucial to the development of a large and useful body of shareable pages or patterns.

D. Both are (in principle) evolutionary, falsifiable and refinable. As structured essays, both make assertions about characteristics of the world they describe – assertions that can be falsified. Once falsified, they can be modified to correct discrepancies, and to refine accuracy. This evolutionary capacity translates into greater accuracy and usefulness over time.

E. Both aim to create useful ontological models of a portion of the world, as a more formalized subset of language. These are models of design specifically for pattern languages, and models of knowledge more generally for wikis.

A number of these elements are common to other information-sharing systems such as blogs and user-editable websites. They are also common to some other systems that are described as "wikis" only as a marketing device, or to emphasize some enhancement in convenience. But to our knowledge no other systems include all of the above elements. No others are intended to provide this capacity to an entire community of users, working in collaboration. In the case of wiki, we will refer to this complete capacity, implicit in the original conception, as "wiki nature."

It is worth noting here, as we discuss in more detail below, that pattern languages in architecture have not lived up to expectations as tools for communities of designers and builders. We believe this is precisely because, trapped within an alluring printed volume, the prototypical pattern language was unable to be shareable and editable by a wide community of users, and unable to be falsified and refined over time. These key requirements were included in the original wiki, and we believe, are key to its success. We also believe this fact indicates the basis of a promising revival of pattern languages in architecture, and other new developments as we discuss below.

7. New technologies exploiting the natural capacities of language

In the case of both patterns and wikis, the goal was to exploit the power and flexibility of language to generate new knowledge, working from the best of existing structural knowledge. In both cases this required the collaborative identification and storage of this existing knowledge, in a way that it was available in a simple and ready form.

In both cases the development of the technology was informed by a philosophy of language that was informed by thinkers including John Searle (1965), Noam Chomsky (1980), George Lakoff (2008) and others. We can summarize this philosophy in the following points:

1. Humans have the capacity to construct shared reality through a series of acts.
2. When those acts are vocalizations we call them speech acts in natural language.
3. Those acts and their consequence are heritable and thus subject to evolution.
4. Pattern language, as developed by Christopher Alexander, mapped a useful subset of the heritable knowledge of building.

5. Wiki, as developed by Ward Cunningham, maps a useful subset of heritable knowledge within the context of user websites, and their missions or "site charters".

As we will discuss in the next section, a new generation of wiki, called "federated wiki," enlarges this landscape in several ways: it allows shared ownership; it has the capacity to manage datasets; and it accommodates plugins to perform specialized applications.

8. Patterns and wikis as more "agile" forms of technology

Both design patterns and wikis were developed to address a fundamental problem in software: simply specifying new solutions to new problems in sequence leads to a cluttering of code, and an increased likelihood of malfunction from unforeseeable and unintended interactions. Cunningham and Beck, working at Tektronix Corporation near Portland, Oregon, were seeking new forms of software that would display what mathematicians often refer to as "elegance": the ability to do more with less. Cunningham embodied this principle in the question, "what is the simplest thing that could possibly work?" This encourages a process of exploration and learning, without assuming the need for particular structures in advance (Cunningham and Venners, 2004).

Cunningham was intrigued by the capacity of language, in its very ambiguity and economy, to serve more ably as a useful working model for problem-solving. A problem is, by definition, not pre-decomposed into simple functional units, but as Alexander noted, has many overlapping and ambiguous connections. Language mirrors this capacity, and therein lies its usefulness. Therefore the goal is, in a sense, to achieve the same robustness of language, by endowing the model with its own set of powerful (but limited in number) generative components, much as language does.

Thus, the goal is not simply a matter of economy, but one of greater context-adaptive problem-solving power. In fact it goes back to the heart of Alexander's concept of language-like networking: a simple grammatical system, functioning generatively, can be far more powerful than a complex set of specification-based processes. As Cunningham put it, when asked by programmer Tom Munnecke to explain how "the generativity of a pattern is a way of expressing complexity:"

> That was an idea that excited me, and that seemed more powerful than most notion that I had seen. ...And that is, language is generative, I follow some rules, and I can't remember when I learned them, but I was probably pretty young. And that idea that I can have a set of rules that generates something that I could value is really important. So the question was, why don't we do everything that way? And the answer was, well we pretty much did, until we let professionals get involved. And they said, no, no, no, no, it's really much simpler, you know, and they made it complex by trying to make it simpler, because they didn't understand how some system of rules could *generate* behaviors instead of *specifying* behaviors. (Cunningham, 2011)

This *generation* refers to the capacity to reproduce the essence of a functioning structure without having to *specify* all of its characteristics. A simple example is the distinction between the way a genetic process generates the blue eyes, say, of a child, which recapitulates the blue eyes of the parent without having to specify them in minute detail (their intricate retinal pattern, round shape, etc). Instead, the genetic process is able to generate, and regenerate, an intricately complex structure from a relatively simple set of language-like instructions.

The result of this kind of work is, somewhat paradoxically, to reduce the complexity of the models we

use for structuring our world – even as we increase their ability to handle real complexity more effectively. This is not so hard to understand, again, if we use an analogy to language. We do not need to draw little pictures to specify everything we see. Instead, we use a flexible language offering immense versatility, with just a small number of generative elements. With just 26 letters and a few other symbols, we can cover plate tectonics, or the plays of Shakespeare, or any of an infinite range of other subjects.

This is, in essence, the structure of natural complexity as well. That is, this is the kind of structure we usually confront as we seek to understand the complexity of a natural system, or a large-scale design problem. When confronted with such a complex phenomenon, we might choose to map all the aspects of its structure. This might, however, lead to an enormous and unwieldy map, posing many of the same structural challenges as the problem itself. But a more elegant solution, mathematically speaking, would be to identify the generative elements that produced the structure, recombine them in another generative process, and let the structure be re-generated. This is a far simpler, more elegant – more "agile" – approach to design.

Many of these principles were refined further within the Agile programming methodology to which Cunningham contributed, and which has also been widely influential (Cockburn, 2007). One of the principles of the "Agile Software Manifesto" is in fact to "maximize the work that isn't done" (Beck et al., 2001).

9. Wiki as curation

The issues raised by these topics take on special urgency, in an age in which information is exploding on the web – some would say cluttering the web – and the reliability of that information is increasingly problematic. Traditional forms of knowledge (such as print) are in decline, and the old methods are simply changing too rapidly to be reliable, while the incentives for those who claim knowledge is harder to verify. The web fills with rumor, promotion, distortion and simple misinformation.

Professionals, with their own parochial incentives, do not always fare better. Some fields seem increasingly to fill with self-serving pronouncements, pseudo-science, and simple hucksterism. A kind of low-grade corruption debases the integrity of honored professions, like journalism, law and even academia itself – a point that the great polymath Jane Jacobs made in her final, disquieting book *Dark Age Ahead* (Jacobs, 2004).

In this environment, wiki (and Wikipedia in particular) offers a usefully instructive model. The contrast with, say, almost any comment section of a blog, is striking. Aside from the strident and opinionated tone of most comment sections – in contrast to the "neutral tone" demanded of Wikipedia articles – it is very easy to see diametrically opposed accounts of the reliability of any given piece of information, often from what seem the most outlandish perspectives. By contrast, on Wikipedia there is remarkable reliability of information, which does tend to conform to the recognized conclusions of acknowledged research bodies, in stark contrast to many other web-based information sources.

How does this happen? In wiki knowledge-development tools like Wikipedia, there is a strong relationship to the way that reliable knowledge is acquired and improved in other fields – although wiki represents a faster and more efficient way of doing so. Again, the question goes to part-whole relations and to the mereology of knowledge – the ways that we can reconcile some portions of knowledge with others, and determine, to a large degree, an overall working reliability.

In the institutions of science, new knowledge can in principle come from any source, but it must be assessed for its accuracy and fit with what is already known. New knowledge must be able to explain not only the unknown but also, in the same terms, the already known. Generally this assessment is done through the peer-reviewed journal process, whereby a paper is submitted anonymously, and reviewed by reviewers who are unknown to the author, nor is the author known to them (a process known as "double-blind peer review").

In Wikipedia, for example, anyone can edit most articles, but the edits also go through a kind of peer review, by reviewers who can reject or flag content that does not appear to be up to standards. They are not necessarily experts in the field in question, but they are referees who are skilled at determining if the information given is consistent with what is known. Most importantly, they rely upon other contributors to judge, among the different contributions, which is more reliable.

Of course not all knowledge is as well-established or shareable as that of an encyclopedia. There are many spheres of life where differences of perspective, valuation and judgment are important, even essential. Culture is surely not a "tree" – in the same sense that a city is not a tree. On the other hand, it is not a murky thicket either, and we must not let our knowledge come to seem a murky thicket of misinformation, ignorance and self-serving hucksterism. There are surely many times when communities, of varying scales, must develop a working consensus to solve their problems together. Indeed, this is probably the very definition of "intelligence," defined as a species trait.

Therefore, we should recognize that only some knowledge has the characteristics of "encyclopedia knowledge." In important cases of policy and practice – such as climate change, for example – some knowledge of details and predictions will always remain uncertain, while there is a working consensus about a key portion. The uncertainty is not a liability, however – it is the very essence of the process of learning.

In this sense a 'federated" body of knowledge (along with the tool to share it) can function as a kind of "chorus" – a larger network of voices that are not stating exactly the same thing, but that contribute, through their very diversity, to a larger whole. From that larger whole, a working consensus can emerge. Out of that consensus comes what we would commonly recognize as "encyclopedia knowledge."

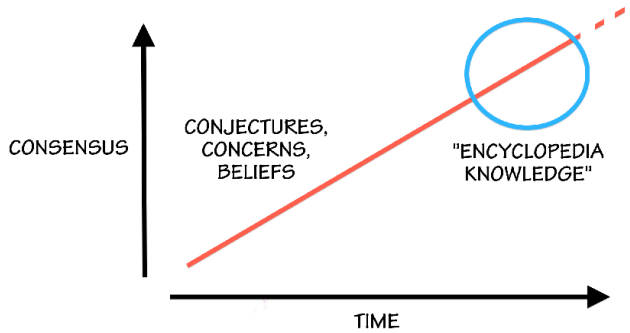

FIGURE THREE: Conjectures, concerns and beliefs form a "chorus" of information from which evaluation and consensus can produce "encyclopedia knowledge" over time.

As in the Wikipedia example, the goal is that knowledge (conceived more broadly than "encyclopedia knowledge" would suggest) is able to grow more comprehensive, integrated and reliable, in spite of – or actually because of – the remaining differences in perspective. In effect, the resource becomes "smarter," and in principle, the people who use it are able to act in a way that is more beneficial, from a human point of view. Those who are seeking to solve a particular kind of problem will learn more readily about what is needed, and perhaps discover new and more effective ways to solve the problem. Knowledge about broader and more abstract human challenges (for example, geostrategic issues like climate change and sustainability) may also evolve and improve, offering further benefits.

We see in this the suggestion of a kind of "curation" of knowledge – identifying and refining the more important and relevant knowledge for the most urgent problems facing groups of people, and humanity as a whole. It builds on the processes already at work within the institution of science, as well as other similar cultural institutions that build progressive bodies of knowledge. In developing such a process that can effectively manage the flood of information on the web, wiki could well prove to be a useful and perhaps important model (Zaino, 2012).

There is also the possibility that such a strategy could improve the feedback cycles of research and learning. Whereas typical peer-reviewed journal publications can take up to two years between completion of research and actual publication, a wiki-like system could, in principle, greatly accelerate the cycle of research into publication. Wiki's collaborative, open-source and open-data model also suggests ways that knowledge could be shared and evaluated more widely by more investigators, potentially improving the results of research itself. But to achieve this capability, wiki systems would need to develop new capabilities. Although the specific capabilities are beyond the scope of this paper, the discussion below will serve to suggest the outlines.

10. Remaining problems with pattern languages in architecture

Alexander and his co-developers did in fact seek an evolutionary model for patterns, following Karl Popper's model of falsifiability. They proposed that each pattern is a kind of hypothesis, to be applied, tested, refined or even thrown out (Ishikawa et al., 2009). This methodology was comparatively easy to apply in the case of software design, where the feedback of success or failure tends to be immediate. It proved to be much harder in the field of urban design, where evaluation may require years or decades of use before conclusive evidence can be established and reported.

There was also a more fundamental limitation of pattern languages in architecture. The book *The Timeless Way of Building* makes it clear that there is an essential methodology of writing any number of patterns and pattern languages, and that the book *A Pattern Language* is only one such instance of a language. In fact the introductory section of *A Pattern Language,* "Using This Book," states, "In this book, we present one possible pattern language, of the kind called for in *The Timeless Way*." This suggests that many more patterns and pattern languages would be written, and that the existing patterns might well be re-used, modified, or even largely discarded.

But this is not what happened. In part because the book was such a successful complete piece of literature, the original 253 patterns in effect became frozen in time. Patterns and parts of patterns that even the original authors now repudiate have remained unchanged, and no published patterns have been added to this original corpus.

Also contrary to the initial intention (Ishikawa et al., 2009), the publisher has not released the content of the patterns into the public domain, and several websites that tried to reproduce the content have received warnings of copyright infringement. This represents a severe constraint on the further use, modification and addition to pattern languages in architecture.

Of course it is possible to write entirely new pattern languages in architecture that exclude all of the 253 patterns, and a number of architects have done this. But as Alexander and his authors noted, many of the patterns are archetypal, and very hard to exclude from common projects. A sampling of patterns indicates the magnitude of the challenge: *Row Houses, Road Crossing, Circulation Realms, Small Parking Lots, Entrance Room, Dressing Room, Built-In Seats, Ornament.*

Still another limitation of the dominant form of architectural pattern languages is in their basis on paper. Alexander has created a website that uses the hyperlink function, but it requires a subscription payment, and the patterns cannot be modified or added to.

All of these limitations are in contrast to pattern languages in other fields, notably software. Whereas architectural patterns are largely limited to the 253 original ones, plus those laboriously created in isolation by a few other architects, many thousands of patterns and pattern languages have been created for software. Whereas few people ever collaborate to make architectural patterns, many thousands of individuals have collaborated on software patterns. Whereas architectural patterns are limited to paper, software patterns begain life as shareable resources on the web, and were fueled by wikis to greatly expand their collaborative evolution and improvement.

These comparisons suggest important opportunities for pattern languages in architecture to greatly improve their efficacy and influence. While the time cycle of urban projects may not be shortened significantly, effective modeling approaches may be able to predict, with reasonable accuracy, the results of various patterns. We will have more to say about this below.

11. New horizons of wiki

From the previous assessments, we believe we can now describe several exciting opportunities for the future of wiki.

A. Federated Development

First is the mode by which wikis are shared and improved. The original wiki technology functioned in a direct open-source mode, which allowed individuals to contribute small pieces to incrementally improve the whole. (Think of a Wikipedia article, for which contributors typically add only one small piece at a time.) But this posed a constraint for development of larger aspects of a wiki.

In the open-source software development community, a new approach developed known as federation. Typified by the "Git" methodology of Linus Torvalds, it allowed an entire copy of a system (of software or of other information) to be freely copied over *in toto* to a new site, and treated as though it were the original copy. (This is known as "forking" a copy.) Then, when beneficial changes are made to this copy, it may be kept as a separate improved version, or recombined with the original, or both. (A request to recombine with the original is called a "pull request.")

A new generation of wiki, called “Smallest Federated Wiki,” is now based on this federated model. The wiki can be forked to new sites, and shared with other collaborators. Thus it can not only proliferate over many individual websites, but also evolve in unique ways in response to new and local conditions. On occasion these benefits prove valuable for older versions, in which case they can be recombined through a pull request.

B. Data Manipulation Capability

Another fundamental innovation of the new generation of wikis is to be able to handle quantitative data. This gives wikis the power to handle processing tasks and modeling functions, wherein quantitative data can be taken through predicted transformations. (For example, in estimating costs, or calculating environmental metrics.)

We have used this capability in a prototype wiki developed for the apparel manufacturer Nike. In operation, a designer would select wiki pages or patterns, corresponding to the designer's choice of material or other design element. The designer will then select other wiki pages that can extract the metrics of performance on various sustainability criteria, such as water and energy use. The designer may select still other pages to display that information in ways that the outcome can be optimized, by adjusting the sequence or other specifications of the earlier pages. (Figure Five.)

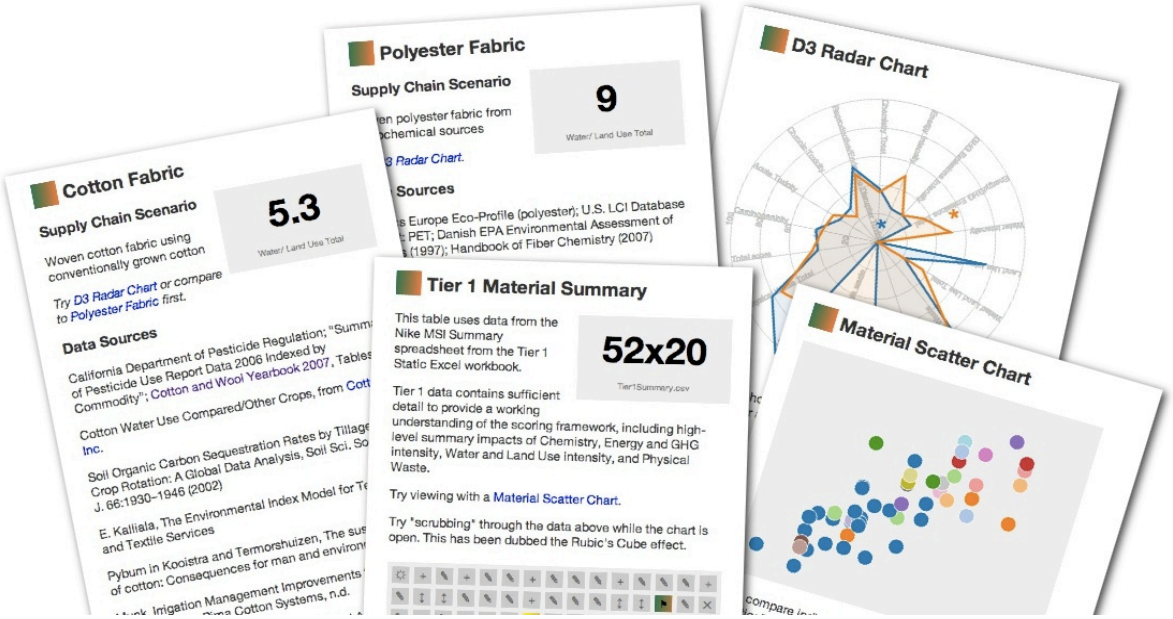

FIGURE FOUR. Sample pages from the Smallest Federated Wiki developed for Nike's Open Data Fellowship.

C. Scenario-Modeling With Data Outputs

It is at this point that the power of such an approach, functioning as a pattern language-based modeling system, becomes apparent. In effect, each wiki page is now treated explicitly as a pattern, and the pages together form a working pattern language. Incorporated explicitly into the new wiki system, the

patterns can guide the quantitative transformations, in such a way that each pattern governs the interaction occurring at that stage.

This is essentially what happened previously with narrative information in patterns – and to some extent in wikis – but now *it is possible to do so for numeric and quantitative information*. This represents an enormous leap forward for both pattern languages and wikis. The goal is, in effect, "to do for numbers, what wiki has done for words."

Moreover, the explicit combination of the two creates another level of capability for each. Specifically, by making more explicit the already pattern-like aspect of wiki, its capability to operate within extensive working networks is greatly expanded.

D. Patterns of Elements, and Patterns of Process

Yet another powerful aspect of this approach, not yet evident in the older pattern language approaches in architecture, is the explicit combination of *patterns of modeling process* with *patterns of physical structures*. As in the example above, the pattern "Polyester Fabric" can be combined seamlessly with the pattern "Tier 1 Material Summary". This is analogous to the concept of von Neumann architecture (or stored-program architecture) in computing, in which a piece of data can represent both a value for an external element, and an instruction to the computer, depending on context. As in the case of von Neumann architecture, this represents a powerful new interactive capability.

We have begun to develop a first application of this Smallest Federated Wiki structure, incorporating process patterns, to the field of urban design. The goal is to allow urban designers to model alternative design scenarios, much as the Nike apparel designers would model alternative choices of fabric. Also as with the Nike example, some patterns are able to combine the results of other patterns, and display results in a format that facilitates optimization. The system is called "WikiPLACE," an acronym meaning "Wiki-based Pattern Language Adaptive Calculator of Externalities." The metrics that are calculated are so-called "externalities," that is, factors that are not usually calculated (nor even possible to calculate) in urban design, such as changes to greenhouse gas emissions per person. And they are able to be calculated in an "adaptive" way, that is, by adapting the configurations of the patterns to reach an optimum.

A further advantage is that, as with all wikis, a model developed by one party could be shared with others, and further adapted and refined. Through the federated structure, it is possible in principle that the model could grow much more accurate and useful. In addition, other variations could be developed for many other factors of interest, including other "externalities' such as infrastructure maintenance cost, future tax revenues and the like. These factors are extremely important in the long-term performance of an urban design – but because they are unable to be modeled accurately at present, they are generally very poorly considered.

An illustration may suggest the capabilities of such an approach. As shown in Figure Five, the designer might start with a basic unit of urban design, such as a residential neighborhood. They might then choose to specify a number of single-family detached (free-standing) homes. They might next decide to specify a number of attached homes, such as townhouses. Finally, they might decide to specify the density of these homes, expressed as the number per unit of land. At each of these steps, they would select the relevant pattern from a drop-down menu. At each step, the patterns would automatically recalculate, displaying metrics of interest (red arrows).

Next, they might choose to perform a summary analysis of the metrics of interest – in this case, greenhouse gas emissions per capita. (The data manipulations between patterns are shown with the red arrows; in this case they are "predictive deltas" applied to the previous metrics.) If they do not feel this outcome is optimal, they can go back and adjust the previous patterns, or their sub-components.

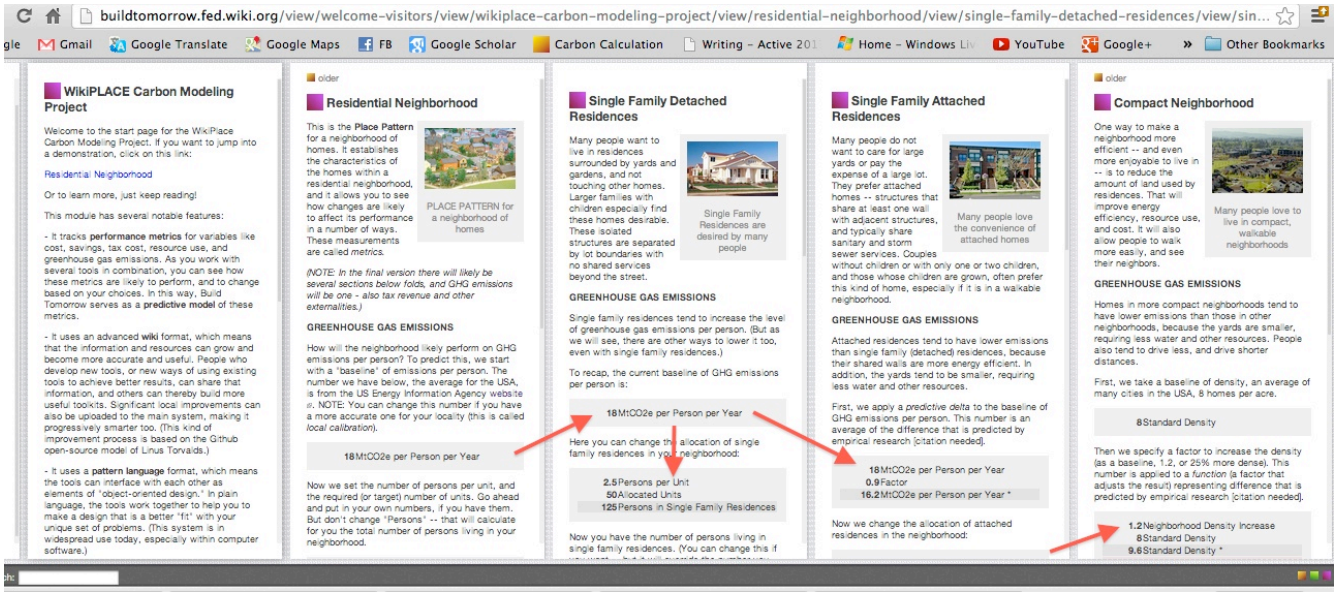

FIGURE FIVE: A screen shot of the new "Smallest Federated Wiki," showing an early experiment for an urban modeling software system called WikiPLACE.

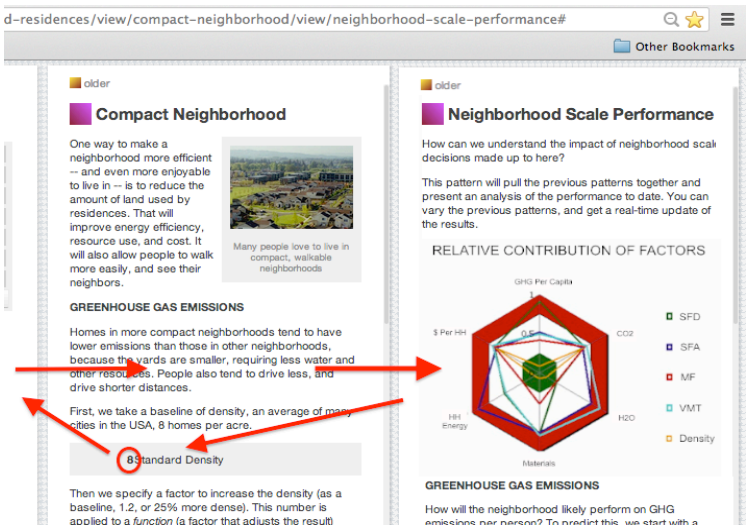

FIGURE SIX: In addition to physical patterns, WikiPLACE incorporates process and analysis patterns, with visual displays of optimal results.

12. Conclusion

The previous work with both pattern languages and wikis demonstrates their extensive capabilities, and suggests even more capabilities that may yet be realized – particularly in combination, and with added capabilities as we suggest here. What we believe is most intriguing is that with the collaborative power

of wiki, pattern languages can indicate desirable outcomes that are not yet explicitly identified, but nonetheless are suggested (and possibly confirmed) by the structure of the patterns. In this way, pattern languages could become effective research tools in their own right, hastening the development and application of useful scientific knowledge at a time that the world needs it more than ever.

As noted previously, this work also suggests a way of managing the explosion of information on the web, and in our lives – and more importantly, a way of assessing the reliability and importance of information and knowledge, in a way that seems most likely to enrich our lives, and our civilization.

REFERENCES

1. Alexander, C. (1964). *Notes on the Synthesis of Form.* Cambridge., Mass.: Harvard University Press.

2. Alexander, C., Ishikawa S., Silverstein, M., Jacobson, M., Fiksdahl-King, I., Angel, S. (1977). *A Pattern Language.* London: Oxford University Press.

3. Alexander, C. (1979). *The Timeless Way of Building.* London: Oxford University Press.

4. Amazon.com (2013). "A Pattern Language," product listing. Accessed March 16, 2013 at http://www.amazon.com/Pattern-Language-Buildings-Construction-Environmental/dp/0195019199

5. Beck, K., & Cunningham, W. (1987). "Using pattern languages for object-oriented programs." Paper presented at OOPSLA 1987 Conference. No. CR-87-43. (1987) Key: citeulike:3944215. Available on line at http://c2.com/doc/oopsla87.html

6. Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Thomas, D. et al. (2001). *Manifesto for agile software development.* The Agile Alliance, 2002-04.

7. Chomsky, N. (1980). *Studies on Semantics in Generative Grammar* (Vol. 107). Berlin: Walter de Gruyter.

8. Cockburn, A. (2007). Agile software development: the cooperative game. Addison-Wesley Professional.

9. Cunnningham, W. and Venners, B. (2004). "The simplest thing that could possibly work: An interview with Ward Cunningham, Part V." Retrieved April 2, 2013 from http://en.wikiquote.org/wiki/Ward_Cunningham#The_Simplest_Thing_that_Could_Possibly_Work

10. Cunningham, W. (2009). Interview with Michael Mehaffy. YouTube video from "Lightning Interview series," February 1, 2009. Accessed March 16, 2013 at http://www.youtube.com/watch?v=Fyc1JGXP-hc

11. Cunningham, W. (2011). *Interview with Tom Munnecke.* YouTube video from Health Camp

Oregon conference, October 22, 2011. Accessed March 16, 2013 at http://www.youtube.com/watch?v=pSnCN-HQXvI

12. Gamma, E. (2005) "How to use design patterns," interview by Bill Venners, in *Leading Edge Java* (website). Accessed July 14, 2013 at http://www.artima.com/lejava/articles/gammadp3.html

13. Giles, J. "Special Report: Internet encyclopaedias go head to head." *Nature,* 438, 900-901 (15 December 2005) | doi:10.1038/438900a; Published online 14 December 2005

14. Kerth, N. L., & Cunningham, W. (1997). "Using patterns to improve our architectural vision." Software, IEEE, 14(1), 53-59.

15. Lakoff, G. and Johnson, M. (2008). *Metaphors We Live By.* Chicago: University of Chicago Press.

16. Leuf, B. and Cunningham, W. (2001). *The Wiki Way: Quick collaboration on the Web.* Addison-Wesley Professional: Boston.

17. Google (2013a). "Wiki" (search). Accessed March 16, 2013 at http://www.google.com (entry "wiki")

18. Google (2013b). "Pattern language" (search). Accessed March 16, 2013 at http://www.google.com (entry "pattern language")

19. Google (2013c). "Design pattern" (search). Accessed March 16, 2013 at http://www.google.com (entry "design pattern")

20. Ishikawa, S., Jacobson, M., King, I., Silverstein, M. (2009). Oral recollections on creation of pattern languages. At Portland Urban Architecture Research Laboratory conference, "Current Challenges for Patterns, Pattern Language, and Sustainability," October 30-Nov. 1, 2009.

21. Jacobs, J. (2005). *Dark Age Ahead.* Vintage Canada.

22. Saunders, W. (2002). "Review of *A Pattern Language.*" Harvard Design Magazine. Winter/Spring 2002.

23. Searle, John. (1965). What is a speech act? *Perspectives in the philosophy of language: a concise anthology,* 2000, 253-268.

24. Simon, H. A. (1962). "The architecture of complexity." *Proceedings of the American Philosophical Society,* 467-482.

25. S23.org (2013). Wikistats, 7/14/2013. Retrieved 7/14/2013 at http://s23.org/wikistats/largest_html.php?sort=users_desc&th=8000&lines=500

26. Wikipedia, "Wiki" (article). Accessed March 16, 2013 at http://en.wikipedia.org/wiki/Wiki

27. Yourdon, E. (2009). "Historical footonote on Design Patterns" (comment). Accessed 3/16/2013

at http://codetojoy.blogspot.com/2009/04/historical-footnote-on-design-patterns.html?showComment=1240261080000#c5317107911198912272

28. Zaino, J (2012) "Ward Cunningham's Smallest Federated Wiki Paves Road To Our Curated Future." Semanticweb.com website. Accessed 22 June 2012 at http://semanticweb.com/ward-cunninghams-smallest-federated-wiki-paves-road-to-our-curated-future_b27267