

The Origins of Pattern Theory: The Future of the Theory, and the Generation of a Living World

Christopher Alexander

Vol. 16, No. 5
September/October 1999

This material is presented to ensure timely dissemination of scholarly and technical work. Copyright and all rights therein are retained by authors or by other copyright holders. All persons copying this information are expected to adhere to the terms and constraints invoked by each author's copyright. In most cases, these works may not be reposted without the explicit permission of the copyright holder.



THE ORIGINS OF PATTERN THEORY

THE FUTURE OF THE THEORY, AND THE GENERATION OF A LIVING WORLD

Christopher Alexander

Introduction by James O. Coplien

Once in a great while, a great idea makes it across the boundary of one discipline to take root in another. The adoption of Christopher Alexander's patterns by the software community is one such event. Alexander both commands respect and inspires controversy in his own discipline; he is the author of several books with long-running publication records, the first recipient of the AIA Gold Medal for Research, a member of the Swedish Royal Academy since 1980, a member of the American Academy of Arts and Sciences, recipient of dozens of awards and honors including the Best Building in Japan award in 1985, and the American Association of Collegiate Schools of Architecture Distinguished Professor Award. It is odd that his ideas should have found a home in software, a discipline that deals not with timbers and tiles but with pure thought stuff, and with ephemeral and weightless products called programs. The software community embraced the pattern vision for its relevance to problems that had long plagued software design in general and object-oriented design in particular.

Focusing on objects had caused us to lose the system perspective. Preoccupation with design method had caused us to lose the human perspective. The curious parallels between Alexander's world of buildings and our world of software construction helped the ideas to take root and thrive in grassroots programming communities worldwide. The pattern discipline has become one of the most widely applied and important ideas of the past decade in software architecture and design.

We can trace the path of influence through three primary sources. The first is the *Design Patterns* book by Gamma et al., a book that helped people conceptualize beyond individual design relationships to grasp important structures of micro-architectures, and to value proven solution strategies over raw innovation. The second was the series of pattern conferences (PLoPs) that provided a forum for pattern enthusiasts to support each other in creating a new body of software literature. The PLoPs were also a forum where the community could struggle with growing from individual patterns to pattern languages that engender systems thinking.

But growth into the deeper and more fundamental aspects of patterns has been slow and difficult. And even as we struggle with the growth from patterns to pattern languages, there is a level of the

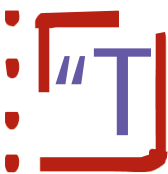
pattern discipline that the software community has yet scarcely touched: the moral imperative to build whole systems that contribute powerfully to the quality of life, as we recognize and rise to the responsibility that accompanies our position of influence in the world. And that leads us to the third inspiration for the direction of patterns in our discipline: Christopher Alexander himself, who directly engaged our community in this keynote speech at OOPSLA '96. The timing and audience of the venue afforded Alexander the chance to reflect on his own work and on how the object-oriented programming community had both hit and missed the mark in adopting and adapting his ideas to software. As such, the speech was a landmark event that raised the bar for patterns advocates, for object-oriented programmers, and for software practitioners everywhere. Beyond that, this speech has timeless relevance to any engineering, scientific, or professional endeavor.

The following presentation was recorded live in San Jose, California, USA, October 1996, at The 1996 ACM Conference on Object-Oriented Programs, Systems, Languages, and Applications (OOPSLA).

* * *

"I'd like you all to very heartily welcome Professor Alexander as he addresses us."

KEYNOTE ADDRESS BY CHRISTOPHER ALEXANDER

 Thank you very much. This is a pretty strange situation I find myself in. I hope you sympathize with me. I'm addressing a room full of people, a whole football field full of people. I don't know hardly anything about what all of you do. So—please be nice to me.

My association with you—if I can call it that—began, oh it must have been two or three years ago. I began getting calls from computer people. Then somebody, a computer scientist, called me and said that there were a group of people here in Silicon Valley that would pay \$3000 to have dinner with me. I thought—what is this? It took me some time to find out. I didn't really understand what had been going on, and that my work had somehow been useful to computer science. Only now I'm beginning to understand a little bit more of what you are doing in your field and the way in which it comes, in part, from

some of the things that I've done.

When I faced the question of addressing you, I wondered what on earth I should talk about. And, earlier, a few months ago I faced a similar thing when I was asked to write an introduction to Richard Gabriel's book (*Patterns of Software*) and again the question for me was: What in the world should I write about? What is there that I could say that would be of interest? And, because I, in a way, myself started out in computers many years ago in the late '50s, this question became quite fascinating to me and quite absorbing. But still, I wasn't given much to go on. Take Jim for example. When he invited me, he was very friendly and I said to him, 'Look, what do you want me to talk about?' and so forth. He said 'Oh that doesn't matter. Just talk about anything. Because it's you, and because of the history of this pattern problem, people will find it interesting.' But still I thought, What should I really talk about?

What is the connection between what I am doing in the field of architecture and what you are doing in computer science and trying to do in the new field of software design? That is the question I must talk about. What I'll do, in the time I've got here, is to tell you where my thoughts went as I stepped through the invention of the pattern concept, and where I have gone since then. In addition, I shall reach a conclusion which may surprise you. As I've been preparing for this speech during the last few months, I ended up with something that may startle you, and you may find quite strange. But, I'm not going to tell you what that is just yet.

In effect, I'm just going to do three things.

1. Pattern Theory. I'm going to talk first of all about patterns and pattern languages, what I did about that, a few little points about problems we encountered, why we did it, how we did it, and so forth. That is a historical survey referring back to the late '60s and early '70s.

2. The Nature of Order. Then, I'm going to summarize the theoretical framework which has evolved out of the pattern work: a framework which is about to be published in a series of four books collectively called *The Nature of Order*, four books that will be put out by Oxford University Press in the year 2000. That framework is a fairly radical departure from what the pattern language in the earlier theories contained, although it is consistent with them. That'll be the second thing. And, I'll just try and sketch that out in the hope that there might be some carryover or you might possibly find it interesting—even though of course I will have no way to apply this to your field directly when I tell you about it. However, there are undoubtedly abundant connections between the two fields that can be drawn.

3. What the Future Holds in Store: The Generativity Problem and the Generation of a Living World. At the time I wrote the introduction for Richard Gabriel's book, that was really as far as I had gotten in trying to trace the connection between my work and your work in the field of computer science: I tell you what I'm doing, and maybe some of you folks might find it interesting or be able to extrapolate. But I couldn't really find that sufficient to be satisfying. I felt that there is some more significant connection between your field and mine. Or at least that there perhaps is. And that finally brought me to the third point.

The third thing I'll talk about is how I now perceive that connection. I suppose that some of you know what I do for a living. You know I'm an archi-

tect. All of my life I've spent trying to learn how to produce living structure in the world. That means towns, streets, buildings, rooms, gardens, places which are themselves living or alive. My assumption here—a sad one—is that for the most part what we have been doing for ourselves, at least during the last 50 years or so, perhaps starting somewhere around World War II, has virtually no ability to produce that kind of living structure in the world. This living structure which is needed to sustain us and nurture us and which did exist to some degree in the traditional societies and in rural communities and in early urban settlements has disappeared. It is drastically gone. We don't know how to create it or generate it any more.

Of course, especially for architects, that is a debatable matter. Some professional architects might say, What are you talking about? What we are doing is absolutely fine, the buildings we are building today are excellent, very good, no problem!! I suppose the architect of this particular huge and nauseating conference hall we are in, here in San Jose, where we can hardly understand each other, would say that, too. But, actually, it isn't fine. It's a hell of a problem. It's a serious problem. It affects every man, woman, and child on Earth. We are so ignorant about how to do this, to make living structure on Earth, it is lamentable. And it is very, very serious, becomes more serious every day, because the population of the Earth is growing, and the Earth is being damaged more and more—and with the damage to our towns and buildings, we too are being damaged.

The fact that we don't know, really do not know what we're doing, and the fact that the built world is not nurturing is a very, very drastic matter for all of us. This is my concern. That is what I try to deal with every day.

PATTERN THEORY

The idea that materialized in the published pattern language was first of all, of course, intended just to get a handle on some of the physical structures that make the environment nurturing for human beings. And, secondly, it was done in a way that would allow this to happen on a really large scale. And, what I mean by that is that we wanted to generate the environment indirectly, just as biological organisms are generated, indirectly, by a genetic code. Architects themselves build a very, very small part of the world. Most of the physical world is built by just all kinds of people. It is built by

developers, it is built by do-it-yourselfers in Latin America. It is built by hotel chains, by railroad companies, etc., etc. How could one possibly get a hold of all the massive amount of construction that is taking place on Earth and, somehow, make it well, that means let it be generated in a good fashion and a living fashion? This decision to use a genetic approach was not only because of the scale problem. It was important from the beginning, because one of the characteristics of



Christopher Alexander

any good environment is that every part of it is extremely highly adapted to its particularities. That local adaptation can happen successfully only if people (who are locally knowledgeable) do it for themselves. In traditional society where lay people either built or laid out their own houses, their own streets, and so on, the adaptation was natural. It occurred successfully because it was in the hands of the people that were directly using the buildings and streets. So, with the help of the shared pattern languages which existed in traditional society, people were able to

generate a complete living structure.

In our own time, the production of environment has gone out of the hands of people who use the environment. So, one of the efforts of the pattern language was not merely to try and identify structural features which would make the environment positive or nurturing, but also to do it in a fashion which could be in everybody's hands, so that the whole thing would effectively then generate itself.

What, now, of my evaluation of what you are doing with patterns in computer science? (Bear in mind, as you hear my comments, that they need to be taken with a grain of salt; I'm ignorant; I'm not in your field.) When I look at the object-oriented work on patterns that I've seen, I see the format of a pattern (context, problem, solution, and so forth). It is a nice and useful format. It allows you to write down good ideas about software design in a way that can be discussed, shared, modified, and so forth. So, it is a really useful vehicle of communication. And, I think that insofar as patterns have become useful tools in the design of software, it helps the task of programming in that way. It is a nice, neat format and that is fine.

However, that is not all that pattern languages are supposed to do. The pattern language that we began creating in the 1970s had other essential features. First, it has a moral component. Second, it has

the aim of creating coherence, morphological coherence in the things which are made with it. And third, it is generative: it allows people to create coherence, morally sound objects, and encourages and enables this process because of its emphasis on the coherence of the created whole.

I don't know whether these features of pattern language have yet been translated into your discipline. Take the moral component, for example. In the architectural pattern language there is, at root, behind the whole thing, a constant preoccupation with the question, Under what circumstances is the environment good? In architecture that means something. It means something important and vital that goes, ultimately, to the nature of human life. Of course, there are plenty of people who will debate whether the question is objective. Some architects are still going around saying that it is all a matter of opinion. But that is a dying breed. The moral preoccupation with the need for a good environment, and for the living structure of built environment, and the objective nature of that question, is largely accepted. I do not know whether that sort of moral component exists in computer science, or in software engineering, or in the way in which you do things.

I understand that the software patterns, insofar as they refer to objects and programs and so on, can make a program better. That isn't the same thing, because in that sentence 'better' could mean merely technically efficient, not actually 'good.' Again, if I'm translating from my experience, I would ask that the use of pattern language in software has the tendency to make the program or the thing that is being created morally profound—actually has the capacity to play a more significant role in human life. A deeper role in human life. Will it actually make human life better as a result of its injection into a software system? Now, I don't pretend that all the patterns that my colleagues and I wrote down in *A Pattern Language* are like that. Some of them are profound, and some of them are less so. But, at least it was the constant attempt behind our work. That is what we were after. I don't know whether you, ladies and gentlemen, the members of the software community, are also after that. I have no idea. I haven't heard a whole lot about that. So, I have no idea whether the search for something that helps human life is a formal part of what you are searching for. Or are you primarily searching for—what should I call it—good *technical* performance? This seems to me a very, very vital issue.

People have asked me what kind of a process

was involved in *creating* the architectural pattern language. One of the things we looked for was a profound impact on human life. We were able to judge patterns, and tried to judge them, according to the extent that when present in the environment we were confident that they really do make people more whole in themselves. Of course you may ask, How in the hell did you test for that? But that is too long a story which I cannot cover in this speech. The important point is that such testing was going on continuously. A second, *almost more important* thing was going on. Whenever we had a language under development we always asked ourselves, To what extent does that language *generate* (hence produce) entities (buildings, rooms, groups of buildings, neighborhoods, etc.) that are whole and coherent? In other words, suppose I write a pattern language for a campus, and, I think I've got some sort of a language that looks as though it will actually do the job. To test it, I let it loose by giving it to people and asking them (in simulated form) to generate different campuses with this language. Let's see what the resulting campuses look like. And we test it ourselves in the same way, by using it to generate designs, rapidly, and only for the purpose of testing the results for their coherence. As it turns out, many of the languages that one creates do not generate coherent designs or objects. That is, they contain a bunch of good ideas. One can use these good ideas to (sort of) put something together from them, and a few fragmentary structural ideas will be present in the result. But that does not yet mean that the campuses created (in the above example) are coherent, well-formed, campuses. We were always looking for the capacity of a pattern language to generate coherence, and that was the most vital test used, again and again, during the process of creating a language. The language was always seen as a whole. We were looking for the extent to which, as a whole, a pattern language would produce a coherent entity.

Have you done that in software pattern theory? Have you asked whether a particular system of patterns, *taken as a system*, will generate a coherent computer program? If so, I have not yet heard about that. But, the point is, that is what we were looking for all the time. Again, I have no idea to what extent that is true for you and whether you are looking for the same thing when you work on software patterns.

So far, as a lay person trying to read some of the works that have been published by you in this field, it looks to me more as though mainly the pattern

concept, for you, is an inspiring format that is a good way of exchanging fragmentary, atomic ideas about programming. Indeed, as I understand it, that part is working very well. But these other two dimensions, (1) the moral capacity to produce a living structure and (2) the generativity of the thing, its capability of producing coherent wholes—I haven't seen very much evidence of those two things in software pattern theory. Are these your shortcomings? Or is it just because I don't know how to read the literature?

So much for my short historical survey of what we have been doing with pattern languages during the last three decades.

THE NATURE OF ORDER

The pattern theory was followed by a deeper theory. I began to notice, by the late '70s, some weaknesses in our work with patterns and the pattern languages.

(1) Under the circumstances that I was most interested in, when we and others were using these patterns to generate buildings, the buildings generated were okay but not profound. There was often a lot of nice stuff going on in them. People were improving certain features, perhaps the daylight was improved, or perhaps the entrance of a building was improved or the characteristics of a street might be improved or an alcove in a bedroom might make it more intimate or something like this. So, there were various isolated features of buildings that were improvements in building performance. The existence of the patterns also allowed people to have better control over their own environment. It succeeded in embodying that control in the real buildings that they made with the pattern material. That was good. But, nevertheless, were the buildings profound structures? To what extent did they really have coherent living structure as wholes? By the late '70s, I had begun to see many buildings that were being made in the world when the patterns were applied. I was not happy with what I saw. It seemed to me that we had fallen far short of the mark that I had intended. But, I also realized that whatever was going wrong wasn't going to be corrected by writing a few more patterns or making the patterns a little bit better. There seemed to be something more fundamental that was missing from the pattern language. So, I started looking for what that thing was.

(2) At about the same time I began to notice a

deeper level of structure and a small number (15) of geometric properties that appeared to exist recursively in space whenever buildings had life. These 15 properties seemed to define a more fundamental kind of stuff; similar to the patterns we had defined earlier, but more condensed, more essential—some kind of stuff that all good patterns were made of.

These were simple ideas. I can't take you through all 15 but they are properties like 'boundaries' which will not only delineate but connect the inside to the outside, or 'positive space,' as when you look at a Matisse cutout and see that the space between the colored paper is not amorphous but also has form. Anyway I began to notice that particular individual patterns seemed really to come always from the 15 deep properties that kept occurring again and again.

(3) Another thing that was happening around this time (late '70s, early '80s), my colleagues and I began toughening up our ability to discriminate empirically between living structure and not living structure. During the years of doing the pattern language, we'd really been intuitive about that and not very rigorous. We were just trying to get patterns written and learning to apply them without asking rigorously if they made buildings with more life in them. But, at this point (about 1980), we felt it was pretty important to get a fix on the difference between a chair which has a more living structure and a chair that has a less living structure. And the same for a building or a room or for a main street in a town. If you want to say this one has life, this one has less life, how do you say that with any degree of empirical certainty? Can it, in fact, be made a relatively objective matter which people can agree about if they perform the same experiments?

Indeed, we did find such experimental techniques. The use of these techniques greatly sharpened our ability to distinguish what was really going on and what structures then correlated with the presence of life in a bit of the environment. The use of these techniques also helped us to refine the 15 deep geometric properties, as necessary correlates of all life in designed structures. These 15 properties turned out to be a substrate of all patterns, and began showing up more and more clearly in our work as the main correlates of living structure in places, buildings, things, space, and so forth.

A brief side remark: I need to say a word about the existence of objective criteria and experimental methods. In my discipline there are tremendous vested interests. Many architects claim, and want to claim, that in architecture there is no such thing as

truth; that is because everyone wants to do their own stupid thing and get away with it. So, depending on who you talk to, they'd say, 'Well, this stuff Alexander's been discovering is a lot of nonsense. There is no such thing as objectivity about life or quality.' But, I am here today, and they are not here, so I'm telling you that there is objectivity. They are simply mistaken. Let's suppose that we've got a sidewalk somewhere on a bit of a street and we've got another sidewalk somewhere else on another bit of a street. We are trying to come to conclusions about which one has more life, which one is a more living structure.

My belief, by the way, when I began trying to find these experimental methods, always was that there really is such a thing, and that actually everybody knows it, but that it has been suppressed. That is because of the world view that we have and the way of looking at things and the nervousness about intellectual rigor... that people of our era have. Although they have these judgments within them, somehow they are separated from their ability to make these judgments correctly. This is just some sort of childish instinct that everybody has and knows. But, for some reason, we are so messed up that we can't see it. So these experiments were, in effect, designed to penetrate that end result through.

The essence of the experiments is that you take the two things you are trying to compare and ask, for each one, Is my wholeness increasing in the presence of this object? How about in the presence of this one? Is it increasing more or less? You might say this is a strange question; What if the answer is 'Don't know' or 'They don't have any effect on me?' Perfectly reasonable! That can happen. But the resolution is easy. What turns out to happen is that if you say to a person 'Yes, it is a difficult question, it might even sound a bit nutty. But anyway, please humor me and just answer the question.' Then it turns out that there is quite a striking statistical agreement, 80–90%, very strong, as strong a level of agreement as one gets in any experiments in social science. All of these different experiments have to do with something like that. Do you feel more whole? Do you feel more alive in the presence of this thing? Do you feel that this one is more of a picture of your own true self than this thing you know whatever? It is always looking at two entities of some kind and comparing them as to which one has more life. It appears to be a rank bit of subjectivity. In other words, it sounds like: 'Well okay, fine; I mean maybe this is the truth about human beings in the sense about our coordination or about our perception or

about our feelings.' But that is not necessarily the same as saying living structure as such is a real thing that resides in those objects. But anyway, to cut a long story short, it turns out that these kind of measurements do correlate with real structural features in the thing and with the presence of life in the thing measured by other methods, so that it isn't just some sort of subjective I-groove-to-this, and I-don't-groove-to-that, and so on. But it is a way of measuring a real deep condition in the particular things that are being compared or looked at.

What is odd about this, and in a way as our work went further and further, it kept bringing big functional and practical matters back to the human person. So, in other words, you take a parking lot. There are lots of technical problems in the parking lot. You have got to make it work. Cars have got to be able to move around. You know there are security problems. There are in-and-out problems. There are maintenance problems. As a whole, the way a parking lot works is essentially a technical thing. The question is, Is it working well or not well? And yet the functionality of the thing measured by these various ordinary bits of technical discussion *correlates with the condition* measured by the question, Do I feel myself to be more whole? It works well when you are getting a positive answer to this question. Thus there is a hint of a profound connection between the nature of matter and behavior of material systems, and the human person. Even in engineering design, as for instance where one considers the structural behavior of a bridge. Or the patterns of movement in something where a lot of cars are moving about, and there are complicated questions about how they move and so forth. In these examples very, very practical matters are nevertheless correlated with these apparently personal questions about whether the thing has life and whether it promotes life in me and you.

So there began developing, in my mind, a view of structure which, at the same time that it is objective and is about the behavior of material systems in the world, is somehow at the same time coming home more and more and more, all the time, into the person. The life that is actually in the thing is correlated in some peculiar fashion with the condition of wholeness in ourselves when we are in the presence of that thing. The comparable view, in software design, would tell you that a program which is objectively profound (elegant, efficient, effective, and good as a program) would be the one which generates the most profound feeling of wholeness in

an observer who looks at the code.

The important thing is that—in architecture—this is not merely a hunch but a testable empirical result. It means that the objects that are most profound functionally (when I say objects, I mean buildings, streets, door knobs, shelves, rooms, domes, bridges) are the ones which also promote the greatest feeling in us. This is a very peculiar thing. At first it sounds like rank sentimentality; and you just say, It can't be true. Why should it be true? And yet, it's a discovery which accords very well with the era that we live in. Because we are living in a period where perhaps the most noticeable and most problematic feature of our world is that feeling has been removed from it. When I make a joke in reference to this horrible meeting hall that we are in, maybe I am beating a dead horse, but I mean really, the problem is that whatever feeling there is in here is obviously not a profound positive feeling. And this is what we have come to expect in our modern world. The failure of that profound feeling to exist in the world around us at small scales, large scales, middle scales, here, there, and everywhere, is tragic. It's the thing that we miss. Of course, people have been writing about this for many decades. Writers have, of course, made this known. We all know it. The difficulty is that people don't seem to know what to do about it. If anything, at the moment, (I'm talking now again about my own discipline, of architecture) the problem is getting worse. It's not getting better. The world that is being built is more and more unfeeling. We are in a sense more lost, more fragmented, more sort of wandering about in this lonely desert than before.

If there really is a way of looking at structures which both deals with real functional structure in the ordinary technical and practical sense, and simultaneously has its roots in human feeling, this will be a very huge and positive step. In particular, the 15 properties that I have mentioned provide us the ability to be precise about the nature of living structure, in just precisely such a way that it is connected, not only to all mechanical function, but also to the depths of human feeling. That is why it is an important structure.

At the root of these 15 properties, there appears to be a recursive structure based on repeated appearances of a single type of entity—the primitive element of all wholeness. These entities are what I call centers.' All wholeness is built from centers, and centers are recursively defined in terms of other centers. Centers have life, or not, in different degree, according to the degree that the centers are built from other

centers using the 15 geometric relationships which I have identified. This scheme, which is at the foundation of all the work in *The Nature of Order*, provides a complete and coherent picture of all living structure.

Stretching a bit, I think there may even be a little bit of a connection between the geometric centers which appear as the building blocks of all life in buildings, and the software entities that you call objects. Centers are field-like structures that appear in some region of space. They don't have sharp boundaries, but they are the focal organizing entities that one perceives at the core of all pattern, all structure, and all wholeness.

Everything is made of these kinds of centers. The centers are more living or less living. And, that's essentially the only important property that they have. And the question of whether a center is more living or less living depends recursively on the amount of livingness in the other centers that it is made of, because each living center is always (and can only be defined as) a structure of other centers. This sort of recursion is familiar in computer science. But whether the structure I have discovered and reported in *The Nature of Order* will translate in any interesting ways to things that you do, I don't know. (It is true, I suppose that all software is made of objects, and nothing but objects. Could it be said that some objects have more life, and others less? If so, there would be a profound correspondence). What is true, I can tell you from my own experiences in these last years, is that when one has this view of things in architecture, it becomes enormously easier to produce living structure in buildings. It has immediate practical usefulness. If you start understanding everything in terms of these living centers, and you recognize the recursion that makes a center, living as it is, dependent on the other centers that it is made of and the other larger centers in which it is embedded, suddenly you begin to get a view of things which almost by itself starts leading you towards the production of more successful and more living buildings.

This insight goes far beyond the power of the pattern language. Although the patterns define relations which might be regarded as specific instances of the recursive interaction of centers, the overall view of centers gives a more comprehensive and more powerful results. It directly affects your ability to make good architecture, in a way that pattern language was not yet able to do by itself. This is a much more powerful and beautiful view than what's embedded in the pattern languages, because when one has constructed this view... you say, well, what is a pattern really? Then it turns out

that patterns are merely a few of the structural invariants that appear within these centers under very, very particular conditions. So they're certainly interesting and important, but they don't have the same depth or the same universal character as these other structures that I'm speaking about now.

Now we come to the crunch. Once we have the view of wholeness and centers, linked by the 15 deep properties, *we have a general view of the type of whole which must occur as the end product of any successful design process*. And because we have a view of it as a whole, we are now able to understand what kinds of overall process can generate good structure, and which cannot. This is the most significant aspect of *The Nature of Order*, and of the new results I am presenting to you in this Part 2.

It means that we can characterize not merely the structure of things which are well-designed, but we can characterize the path that is capable of leading to a good structure. In effect, we can specify the difference between a good path and a bad path, or between a good process and a bad process.

In terms of software, what this means is that it is possible, in principle, to say what kind of step-by-step process can produce good code, and which ones cannot. Or, more dramatically stated, we can, in principle, specify a type of process which will always generate good code.

Of course we have not actually done this for the production of code. We have done it for design and construction of buildings. But it is possible. This is, if you like, the holy grail of software design—specification of the kinds of process which will (always) generate good, efficient, economical, beautiful, and profound, code.

What are the details? I can tell you in the case of buildings. If one has identified living structure with a reasonable level of objectivity, and if one has identified this recursive center-based structure as being the key to the whole thing, that's all very well. But then of course the practical question arises, How the hell do you produce this living structure? What do you have to do to actually produce it? You can clumsily try to find your way towards it in a particular case. But, in general, what are the rules of its production? The answer is fascinating. It turns out that these living structures can only be produced by an unfolding wholeness. That is, there is a condition in which you have space in a certain state. You operate on it through things that I have come to call 'structure-preserving transformations,' maintaining the whole at each step, but gradually introducing differentia-

tions one after the other. And if these transformations are truly structure-preserving and structure-enhancing, then you will come out at the end with living structure. Just think of an acorn becoming an oak. The end result is very different from the start point but happens in a smooth unfolding way in which each step clearly emanates from the previous one.

Very abstract, I know, but the punchline is the following. That is what happens in all the living structures we think of as nature. When you analyze carefully just what's going on and how things are happening in the natural world, this sort of structure-preserving transformation tends to be what's going on most of the time. That is why, when nature is left alone, most of the time living structure is produced. However, in the approaches that we currently have to the creation of the built world and the environment (planning design, construction, and so forth), that is simply not what is happening. The process of design that we currently recognize as normal is one where the architect or somebody else is sort of moving stuff around, trying to get into some kind of good configuration. Effectively this means searching in an almost random way in configuration space, and never homing in on the good structure. That is why the present-day structure of cities, buildings, conventional halls, and houses are so often lifeless. The processes by which they are generated are—in principle—not life creating or life seeking.

If a process doesn't go in the structure-preserving way that I'm talking about, the result is never living structure.

In effect you can write theorems which say, Under the kind of conditions which occur in the construction industry today, you cannot produce living structure. So, the poor folks who designed and built this convention center were stuck with something lifeless, because they were embedded in the wrong kind of process. There was nothing they could do about it. It was part of the process by which this kind of entity is produced in today's society. As things stand, it cannot come out with a living structure at the end. That is a shattering discovery.

A very large part of my work and that of my colleagues in the last years has been one of trying to define social processes, economic processes, administrative and management processes which are of such a nature that they permit true structure-preserving unfolding to occur in society, thus to allow the generation and production of living structure. This is what I do most of the time: I'm trying to do real projects of one sort or another where I'm

introducing this unfolding process and trying to make it work under the conditions available to us in 1996. The social and technical shifts involved are large. The shifts in thought, in practice, in administration of money, in contracts, all sorts of real nitty-gritty things that one would much rather not mess with because they are so hard, you must mess with because it is those processes which are undermining the ability for our whole contemporary social process to be structure-preserving unfolding. If life is to be created, these processes must change.

That is the end of my Part 2.

WHAT THE FUTURE HOLDS IN STORE: THE GENERATIVITY PROBLEM AND THE GENERATION OF A LIVING WORLD

Let us now consider a problem of magnitude. There are some two billion buildings in the world, about 2×10^9 buildings. Differently stated, the total amount of built stuff is something on the order of about 10^{12} , 10^{13} square feet of construction. The total amount of built stuff in Manhattan is somewhere on the order of 10^9 square feet. If you include all the exterior space in the world as well, the part of the outdoors that is somehow having to do with human beings and is part of our immediate world—gardens and streets and agriculture and all of that, in the world—we're somewhere up around 10^{14} square feet of constructed, designed space.

How are we going to deal with all that? How do we create, or generate, living order in 10^{14} square feet of construction? What process could possibly accomplish this within, say, one generation—the next 25 years? The effort of architects, no matter how hard we try and no matter how much good will we put in, does not begin to scratch the surface of that task. All the architects in the world, together, working as they do today, cannot design more than say 10^{10} square feet per year—a tiny, tiny percentage of what is needed—far too small to be effective.

I have, for many years, thought that this could only be solved by a genetic approach—an approach where deep structure, spread through society, creates and generates the right sort of structure, very much as genetic code creates and generates organisms and ecological systems—indirectly, by letting loose life-creating process.

That is what I still believe. But, today, I am convinced that the equivalent of the genes that act in organisms will have to be—or at least can be—software

packages, acting in society. If these software packages are life creating, and accepted, and widely enough spread throughout the world, there is a chance we might get a grip on this problem: provided that the software is freeing, liberating, allows each person individual control and decision making power to do the right thing, and to create living structure, locally, wherever they are. This task must fall, inevitably, at least in part, on your shoulders.

The people who were kind enough to invite me to give this speech originally assured me that if I just explained the intellectual history (as I have done so far), there will be those among you who may find it interesting, that somehow they might latch on to it or know how to translate it into something that's more directly relevant to your own concerns. That is, after all, just what you have done in the last five years with pattern languages. There clearly is a useful parallelism between our two disciplines.

However, after receiving this invitation, and contemplating the questions I could raise, I started dwelling on a conviction that was growing in me. This conviction led me to feel that there was a deeper coincidence in what you are doing in software design and what I am doing in architectural design. I began to feel that there is a deeper connection, which suggests that the two disciplines might merge in a way that would benefit us both—you in your discipline and me in my discipline. In the next few minutes I will try to sketch the nature of this connection.

As an architect, of course like anybody concerned with these things, I have a passion to try and make these things happen. It's not enough just to say, well, living structure isn't being produced. There is a question I must ask myself, a question I do ask myself, which all the time is: Okay, well, what are we going to do about it? Here we've got this poor Earth sinking under the weight of all this dross. And, what are we actually going to do? I do a \$10 million project here and I do a \$10 million project there. But that accomplishes virtually nothing. Life is short. A few of those projects. . . and what is it? It is an atom in the proverbial bucket. It's nothing. All of the efforts of the architectural brethren, even if I can persuade them of the truth of these things, is still a drop in the bucket. That by itself, will not affect more than a thousandth part, perhaps no more than a millionth part, of the structure covering the built part of the Earth.

When I started out 25, 30 years ago, I really thought that I would be able to influence the world very fast. Especially when I got to the pattern language. I thought, boy, I've really done it. This is going

to work. No problem. The patterns are self-evident and true. They will spread. And, as a result, the world of buildings will get better. Hey presto.

But it hasn't yet worked out like that. In practical terms, so far, I've done almost nothing. The pattern language, how much has it influenced the environment of the world? A few thousand buildings have been influenced. There are a few people that have lived a few things and been influenced. But, meanwhile, we've still got this gigantic amount of construction out there which is defining the world that all of us live in that is still going on in exactly the same fashion. I believe that the cultural process of influence is simply too slow to be able to take care of this problem. In other words, the process by which one discusses these kinds of things, shares ideas about them, gradually influences the way people are thinking so that gradually larger and larger percentages of bits of the environment might turn into living structure. That is a very slow process, and I don't think it is fast enough to do the job. And yet, as an architect, I view myself as responsible for that. Not of course, alone, but as a professional, my job is to try to understand how we can get hold of that—the entire structure of built environment, all over Earth—and do something about it to make it better.

For several years I have been asking myself how this effort can be expanded, and strengthened. It must be our aim to make the world's environment a living structure, within one or two generations. How, realistically, can that be done?

So, today, I am standing before you, thinking to myself: Right, I'm now talking to people who are in a way the core of the computer revolution. You probably realize, I know you must realize the extent to which the world is gradually now being shaped more and more and more, indirectly, by the efforts of all of you who are sitting in this room—because it is you who control the function of computers and their programs. It is the programs that control the shape of manufacturing, the shape of the transportation industries, construction management, diagnosis in medicine, printing and publishing. You almost can't name a facet of the world which is not already, to some very strong degree, under the influence of the programs that are being written to manage and control those entities or those operations. And this is still in its infancy. How long has this been really going on? Not long. About 10 or 15 years, though of course, the preparation for it goes a lot further back than that. But really this is quite new. It is going to look a whole lot different, even more powerful in its degree of influence.

And yet, as a professional body, I don't think that

you are yet fully aware of it. I'm probably speaking out of turn here but, you know, I've thumbed through the proceedings of this conference, for instance. Jim was kind enough to show it to me yesterday. I don't really see discussion about what, collectively, computer scientists are supposed to be doing with all these programs. How are they supposed to help the Earth? And, yet, the capacity to do that is sitting right here in this room. That is an amazing situation. You have so much power.... but that means that you also have an enormous responsibility.

Is there a chance you might take on the responsibility for influencing, shaping, and changing the environment? Interestingly, I think many of you do also have the *inclination*. When I had the pleasure of beginning to meet some of the various folks who introduced themselves to me over the last year and a half from the software community, I began to be fascinated by the number of them that were closet architects. Greg Bryant, who worked on the 486 chip, is really interested in ecology and is an editor of *Rain*, an ecological magazine. Bill Joy is writing about workstations in the concrete physical sense that is familiar as an architect. John Gage, chief scientific officer of Sun, is interested in neighborhood schools, and in the process by which people can repair their own physical neighborhoods by working together. Jim Coplien is dealing with social structures in human organizations. Mark Sewell from IBM wants to build houses. Dick Gabriel has, as his deepest passion, the writing of poetry: another kind of art. I don't have a long enough list. But my hunch is that an amazing number of you who got into this pattern game in the pursuit of your normal professional endeavors are also very profoundly interested in the real physical world, and its shape and its design, its deep feeling, its impact on human life. That is, the world in which we inhabit. It is therefore conceivable that you, collectively, could change the very drastic situation of a destroyed environment that I described earlier.

Let me just go back to the structure-preserving unfolding process that I described in Part 2 of this talk. I talked about this structure-preserving unfolding process.

When I first constructed the pattern language, it was based on certain generative schemes that exist in traditional cultures. These generative schemes are sets of instructions which, carried out sequentially, will allow a person or a group of people to create a coherent artifact, beautifully and simply. The number of steps vary: there may be as few as half a dozen steps, or as many as 20 or 50. When the gen-

erative scheme is carried out, the results are always different, because the generative scheme always generates structure that starts with the existing context, and creates things which relate directly and specifically to that context. Thus the beautiful organic variety which was commonplace in traditional society could exist because these generative schemes were used by thousands of different people, and allowed people to create houses, or rooms, or windows, unique to their circumstances.

When I first hit on the idea of creating, and using, pattern languages, I was inspired by these traditional generative schemes, and thought that I was essentially copying them. However, in the huge effort of creating a believable, new pattern language, in the 1960's the effort went entirely onto the individual patterns (their formulation, verification, etc.), and the idea that they were to be used sequentially, one after the other, dropped into the background. In fact, both *A Pattern Language* and *The Timeless Way of Building* say that the pattern language is to be used sequentially. In practice, however, this feature dropped out of site, and was not emphasized in use. As a result, the beautiful efficacy of traditional languages and their simple and beautiful sequential nature disappeared from view.

In our most recent work, that has changed. We are now focusing on pattern languages which are truly *generative*. That means, they are sequences of instructions which allow a person to make a complete, coherent building, by following the steps of the generative scheme. We have done this for houses, for public buildings, for office furniture layout, and so forth. It works. And it is powerful.

Compared to the pattern language that you've seen in *A Pattern Language*, these generative schemes are much more like what you call code. They are generative processes which are defined by sets of instructions that produce or generate designs. They are, in fact, systems of instructions which allow unfolding to occur in space in just the way that I was talking about a minute ago (Part 2), and are therefore more capable of producing living structure. The published pattern language by comparison is static. The new generative languages are dynamic and, like software, interact with context, to allow people to generate an infinite variety of possible results—but, in this case, with a built-in guarantee of well-formed results. The design that is created or generated is guaranteed, ahead of time, to be coherent, useful, and to have living structure.

You know the pattern language (the one for

architecture) consists of these objects which are interesting and which you somehow try to put together. But it's possible to have processes or procedures which will go much further, actually generate living structure. Because of the complexity of the situation in the world, and because of the way software is going, software that is designed to do this could very rapidly take the world by storm.

Why would computer scientists and software engineers suddenly become responsible for the form and structure of the built environment? Is that not the province of architects, planners, agricultural experts, forestry people, and civil engineers? It ought to be. But the members of these professions are not taking responsibility for the generative approach to living structure—and so cannot produce it. And, as far as I can see, they do not see it coming, and are not preparing themselves to take it on, mentally or professionally. Therefore it will fall to someone else to do it instead.

In history, this kind of unexpected switch is a common thing. When a paradigm change occurs, in a discipline, it is not always the members of the old profession who take it to the next stage. In the history of the development in technical change, very often the people responsible for a certain specialty are then followed by a technical innovation. And then the people who become responsible for the field after the technical innovation are a completely different group of people. When the automobile came along, the people who built the buggies for the horse and buggy did not then turn into Henry Ford. Henry Ford knew nothing about horse buggies. The people who were building automobiles came from left field, and then took over—and the horse and buggy died off.

It is conceivable to imagine a future in which this problem of generating the living structure in the world is something that you—computer scientists—might explicitly recognize as part of your responsibility. Such a change, representing a kind of a level of marriage between you and me, is of an entirely different sort from the one that I was invited by Jim Coplien to contemplate. I was brought here to answer the question 'Okay Chris, what new things have you been doing that might spin off and be useful to us in our neck of the woods?' Parts 1 and 2 of this talk were about that. But this Part 3 is about something quite different. I want you to help me. I want you to realize that that problem of generating living structure is not being handled well by architectural planners or developers or construction people now, and the Earth is suffering because of it. I believe there may be no way that they are ever going to actually be able

to do it, because the methods they use are not capable of it. For you it is different. The idea of generative process is natural to you. It forms the core of the computer science field. The methods that you have at your fingertips and deal with everyday in the normal course of software design are perfectly designed to do this. So, if only you have the interest, you do have the capacity and you do have the means.

I heard a rumor at breakfast that some of the people in this room have begun to worry about their jobs. I have no idea if that is true. But I was told there is an undercurrent of unease as to where all this—software design—is going. There is a huge expanding phenomenon of programming as an art, and yet an uneasiness about where it is all headed. What is it going to do?

My comment on this? Please forgive me, I'm going to be very direct and blunt for a horrible second. It could be thought that the technical way in which you currently look at programming is almost as if you were willing to be 'guns for hire.' In other words, you are the technicians. You know how to make the programs work. 'Tell us what to do, Daddy, and we'll do it.' That is the worm in the apple.

What I am proposing here is something a little bit different from that. It is a view of programming as the natural, genetic infrastructure of a living world which *you/we* are capable of creating, managing, making available, and which could then have the result that a living structure in our towns, houses, work places, cities, becomes an attainable thing. That would be remarkable. It would turn the world around, and make living structure the norm once again, throughout society, and make the world worth living in again.

This is an extraordinary vision of the future, in which computers play a fundamental role in making the world—and above all the built structure of the world—alive, humane, ecologically profound, and with a deep living structure. I realize that you may be surprised by my conclusion. This is not what I am, technically, supposed to have been talking about to you. Or you may say, 'Well, great idea, but we're not interested.' I hope that is not your reaction. I hope that all of you, as members of a great profession of the future, will decide to help me, and to help yourselves, by taking part in this enormous worldwide effort. I do think you are capable of it. And I do not think any other professional body has quite the ability, or the natural opportunity for influence, to do this job as it must be done.

I've enjoyed talking to you very much. Thank you for listening to me, and I would be most keen to listen to your ideas on these topics." ❖