Art complexity and Technology: their interaction in emergence
Institute for Scientific Interchange Foundation, Villa Gualino, May 5-6th 2005

John Hamilton Frazer, AA School of Architecture of London, UK

"Interfacing with Evolution: towards a self organising Architecture"

Every acorn (providing it grows), grows into an oak tree, but given some millions of years the genome of the oak tree may give rise to another species of tree. We could say that where an acorn becomes a recognisable oak tree is 'development' and 'Darwinian evolution' is where the genetic information and the rules of transition change. The consequence is that each oak tree is different depending on its parentage and the environment in which it grows.

To an architect that oak tree seems a wonderful inspiration. The question is; could we code and condense architectural concepts into the equivalent of the acorn, in such a way that we can use powerful computer techniques to develop these concepts in a richer and more varied way. If we can get all the variety of nature in a relatively economical amount of genetic coding, it suggests that architectural concepts could be simplified in a similar way. What might the equivalent be of the four nucleotides enabling the permutations in the DNA? What kind of information, would have to be embed in an architectural package of seeds?

The first experiment was a structural system consisting of 32 different orientations in space. The architectural seed grew and developed following simple coded rules. Geometrical rules of combination were embedded and could not be broken which meant that the cloned structure stayed coherent and did not evolve. The kinds of instructions that were put into a computer were to grow and stretch and shear and to pull out, so that a building for a particular application could be developed. At the same time the building was developed to fit the ground area and allow for the adjacent buildings and have the correct external openings and so on. Although a great variety of overall forms were made it could not break out of the simplistic language of the structural elements. One advantage of the system was that the plans could be put on a triangular or hexagonal grid and a bit of code gave the exact location of each structural unit, its type and which way round it was.

The coded information was the complete description of the building which could then be modified and developed. That led to a slightly different approach to design. Instead of moving from a building brief into a data structure, the cultivation process was used to generate more and more data in a very dense coded form which then became structural elements at the output stage. This was a bit more analogous to some biological processes.

The architect, Walter Segal, had a very simple model which allowed self-builders to experiment with designs. Using different coded diodes

representing doors and windows etc., anybody who could make a model could manage to design a building. The computer resolved any compromise of structures and at the same time made sure the building complied with building regulations. Someone who knows nothing about architecture can design a building because all the architectural rules have been put into the system. Secondly, because the software has been put into the components people simply have move the components about to generate different plans for different structures and costs. You merely plug the different components in, each of which has a different identifying pattern of diodes from 0-256, and the process produces plans and instructions.

There's a similar program by another architect called Cedric Price which involves building elements which are like little porta-cabins and a huge crane that moves them around to different regions of the site. That enables architects to plan either accommodation for a conference, like the one we're having now, or a residential block or perhaps accommodation for a children's holiday camp. I was a systems consultant for the cybernetics programme which organised it when the only memory available was a tape recorder and we had to drive to a laboratory in the next block to get high resolution graphics out. However, in the model, one corner of it had enough processing power to negotiate with its neighbours on how to rearrange the site, and that led to the notion of 'intelligent' building sites.

Embarrassed by this enormous array of components and more computing power than I needed I came up with the notion that an intelligent program might get bored. I decided to send spurious instructions to move things around because it seemed to me that the computer program ought to learn. So I provoked changes to see what reaction the system generated. This led to the notion of boredom being built into these programs as a way of provoking learning.

In the 1990s my students built a model using cubes, each of which had 256 possible states or arrangement of the diodes. This gave different transition rules, so the system consisted of 3D cellular automata and it is now our preferred generative engine. In this we have an input and an output device, can arrange many different views and patterns and the buildings can talk to each other. When tower blocks, for example, communicate with each other, the visualization enables us to see the message go rippling down one stack and up the other. We can actually see the kind of visual logic by which the system is trying to organise itself. We talk about architecture, being logic in space and this model's primary function is to give visualisation to that process.

Subsequently we have made many models with more complex arrangements, but basically, they work on the same principle. Suppose you start with the components to build a column. 239, for example, could be an ionic capital, 238 could be the column and 237 could be the base. Putting them all together we get a Doric column and then we could have a row of them and so on.

Normally, of course, the mapping is not that literal, but there is a hierarchy of process. By 'literal' I mean that the components look like the real parts. As you go up the hierarchy so the mapping becomes more symbolic and more abstract, but a representational self -build model is quite like the realised building, though the mapping will evolve with different colours for different structures and so on.

The way we teach students to program is by stacking cubes of the kind previously mentioned. They control a little robotic turtle, for example, by stacking up the cubes in different configurations and this enables design students who are non-computer literate to learn programming in a very simple way. When the students have got the idea of stacking up the cubes they can start to change the code by rearranging the cubes. That means they start to read the code and when something goes wrong and a physical bit falls off it's like a 'bug' in the system. So there's a direct learning relationship about what 'bugs' in software are and a direct physical learning relationship which is helpful to students who are not used to the kind of abstractions that scientists use.

In this way the students carried out all the programming and construction which also involved making the machinery and wiring all the circuits. In the end we managed to get several of these systems talking to each other, something which was very important to me personally, because it simulated one part of the environment talking to another about the development. I'm not suggesting that trees actually talk to each other when they are growing in the forest, but there is some interaction according to the way the light comes in or the soil varies and so on.

We had a number of different projects. One involved a dance in which each of the patterns of the diodes represented a position on a dance notation, and each of the state numbers that the cubes got into, contained an instruction for a graphic transformation so we got a sequence of slides. Another year each student designed an input and an output device and at an exhibition we had a number of well known architects interacting with one another via the devices. Everyone was interconnected by things like jewelry, earrings, shirt buttons, cufflinks and glasses as computer interfaces. We had body suits that you could touch or massage to give feedback and we had a device which went into different moods depending on whether anybody was looking at it or not. If no one was looking at it, it would rattle about a lot to attract attention and if a group came over to it, it would start to show off by doing exotic shakes and bends. Finally, towards the end of the exhibition when it had no visitors at all, it seemed to sink into depression.

I won't talk much about how we evolve models. We use genetic algorithms which mainly come from John Holland and which have long been used in engineering. A lot of people thought these wouldn't be suitable for ill-defined problems or those with conflicting criteria, both of which characterise design problems. Undeterred by such doubts we went ahead. To give you an

example: in order to optimise the performance of a boat, water resistance is reduced by making the boat longer and thinner with a light hull, but at the same time the interior designer tries to get more operating space inside the hull. Such considerations represent a conflict of desires which must be compromised. Darwin started by looking at variation in domestic species and the use of artificial selection in order to optimise their performance in some way. He then went on to argue that natural selection worked in the same way, but on a very much larger scale. As designers we can use our intuitions to interfere with the genetic algorithm and choose our own compromise and we can turn up the random generator which will make small random changes to forms. What we tend to do with our system is to leave it overnight, asking it to produce an optimum solution and in the morning we might do a bit of cross- breeding.

John Holland himself designs yachts using sophisticated computer programs to work on curves and radii. He puts in some data simulating wood, water and steel and the process becomes a combination of his experienced eye and the computers ability to optimise and rationalise. The design then goes digitally to the shipbuilder who can, for example, cut out the ribs with laser cutters to very fine tolerances based on the model. Digital modeling enables control of the whole process which means the designer has regained control after having effectively lost it ever since the Industrial Revolution.

Recently we made a digital model of the Robotics lab building at MIT, which has lots of advantages because the fabrication people can work directly from the computer model. It means the architect controls the equipment that makes the parts, which are then assembled on-site to a very fine tolerance. Everyone assumes that buildings like this are extremely expensive, but this building was medium cost for a university building and its eccentric geometry makes it the most beautiful place to work in.

Another simulation project was using 3-D cellular automata, built up using cubes. According to environmental input the cube had to decide whether or not to split into two cubes. So this is very analogous to the splitting of cells in the human embryo. The rules for whether it splits or not and where it splits are determined by the transition rules and the states of the different cells. There are now 16 million state rules in each of the cube cells and that's the number we needed to get the complexity and richness we were looking for. The states, which started off as being identical are now specialized for different transformations of colour or size etc. The genetic algorithm changed the form so that a different output would come from the same program demonstrating that evolution as well as development was taking place.

In 1995 we put a lot of these things together in an exhibition using spheres instead of cubes and had a virtual exhibition on the Internet. It involved a shape, interactive within the exhibition, which interacted with the changing environment and the exhibition space so it was somewhat analogous to the different conditions that oak trees grow under. We had visitors, physically

switching the genetic information, and we had virtual visitors to the Internet who changed the environmental parameters. What it showed were complex evolutionary cycles much as we see in nature.

Running several computers together with no human intervention we had bottom up gene pool evolution with genetic algorithms giving rise to cellular automata and top down we had neural networks electing the rules. Using that, we were able to simulate the evolution of cities and with a different city for each computer we could have the cities talking to each other and trying collaboratively to improve their run by having different environments and different economic conditions. Some of this software was used for evolving tower block buildings and some of the designs show different attempts by the computer to evolve different forms. Rather curiously, in one case, we had the computer designing four or five different sites, and the computer itself deciding that it would join them up which may have been a case of emergence.

One possible role for the architect now is to concentrate on generic concepts, as do most artists, and make a lot of manifestations of the design space. That may be a useful model to work from, but I think the end users may increasingly play a role in urban design and build by adding their preferences to the design process. I see the main role of the computer as being that of an evolutionary accelerator and we could take the coral polyp as an ideal model. Each different kind of coral builds in a recognisably different kind of architectural style and the inhabitants of the reef are crucial to the whole system. The environment plays a role, because waves for example, interact and break bits off the coral which then drift to form a beach and something, such as a tree, has to make an imaginative response to growing in such conditions. I want to lock my architecture into that two way process: The environment influences the architecture and the architecture influences the environment in a positive way. Whether we like it or not, the buildings we inhabit are influencing evolution of the environment and we should turn this round to be a positive thing, rather than something that we fight against.

Questioner 1. I have two questions. One is technical, and the other is conceptual. The technical question is that at some point, you said you had mixed rules.

John: well we don't differentiate about whether the numbers are transition rules or states. We just use one complete number and it may be better to think of them as registers of groups of parameters for parameter- driven change. That would explain the enormous numbers that we get.

Questioner 1. From what you have told us it seems possible to generate an enormous amount of stuff, but you didn't say much about the way the things which you generate are used. I thought it was interesting that you had human beings involved in the selection process and I wonder to what extent you think that the concepts and ideas that human beings are using can

themselves be automated as a search process.

John: I'm often asked whether I can automate the seed in the first place and I have two answers. One is 'no', because I believe in intentionality and I don't think you are ever going to get the kind of richness of an artistic idea emerging out of nothing. However I also have to say that I have occasionally experimented with evolving the seed from nothing. I didn't actually think it would work, but I'm a little bit worried because it appears that it does a bit. Of course, the neural networks can be trained to learn what humans pick and we did some early experiments in which we put in an aesthetic prejudice. For example, we had some rules for drawing architectural columns according to very simple ratios. We did a demonstration with some students in about 1984 with a computer evolving columns and instead of putting in a rule which said a column has to be seven times the diameter in height we put in a random generator. We then got the computer to learn the kind of choices that people kept making. When people selected columns off the screen they would deliberately select, say, short columns and the computer would remember their tendency to do that. So yes, in a very limited way, human preferences can be automated.

Questioner 2: What are the new constraints on the architecture from a physical or an aesthetic point of view?

John: Well, if there are physical constraints you simply put them in. For example in the early structural systems there were adjacent buildings existing on site. So we had to build in some restrictions because of them. But probably we have the more innovative constraints in terms of a built-in language. I showed two examples of specific systems, one was Walter Segal's self- build, and the other was my own structural system. Obviously I tried to generalize, but if you're a design builder and you happen to be making some beautiful clad panels, then you would put those in. It is a little bit like the genetic information. I don't want to get too much into a biological analogy, but probably it's analogous to the way proteins are coded and built in the body. That may not be an analogy, but basically we are not a box of bricks type builders. Buildings that are built like that obviously look as if they were built that way, and our question here is really about how to get enough design freedom to generate something we want. I think it's true to say of these programs that the lower level of complexity is just dull, but if it gets too complex it just turns into a mess. So we're looking at things on the edge, where they are sufficiently constrained to be constructible and meaningful, but sufficiently free to generate things which are different or of interest.

Questioner 3: I'm an artist, and I would like to ask how you find out people's aesthetic feelings about the things you've shown us; things that they like or don't like?

John: W ell, I get excited about some of the things that I've shown, but I'm a bit concerned that the visualisations just look like blobs. They are not meant

to be buildings – they just represent the data structure, which then maps to something via a mapping process. Scientific diagrams are examples of scientific visualization and show processes which science otherwise doesn't know how to illustrate, but they may make an aesthetic statement by being functional. The diagrams indicate something, but I've always shied away from doing something like a Mondrian or a Picasso painting. Of course there are aesthetic constraints on the buildings that we produce.

Questioner 3: Is there some point in the design at which you say to yourself, 'yes, that's great except for that little corner there'?

John: Yes of course. I'm not making something out of protein. I don't have to wait a million years of evolution. I'm very happy to be inspired by nature, but I'm not going to be constrained by its problems, which I don't have to share. And I don't feel I have to do everything by computer at all.

Questioner 4: When you start from simple components, you can build complexity, but after a certain level it becomes more difficult. Do you find one way to select a complex unit on which to build as an elementary unit on which to build or not.

John: Well we've only done that once. Using the same sub-systems at different levels, the same organisational code for a room, a building, a street, and then a whole neighbourhood. I think that's close to what you're saying, and it worked very well and I think it was very useful. It got us into the nitty gritty of changing scale. If something is built in a hierarchical way we can to zoom in to any level to look at the sub-systems in it.