draft, to appear as: F. Heylighen & C. Gershenson (2003): "The Meaning of Self-organization in Computing", *IEEE Intelligent Systems*, section Trends & Controversies - Self-organization and Information Systems, May/June 2003.

The Meaning of Self-organization in Computing

Francis Heylighen & Carlos Gershenson
CLEA, Free University of Brussels

The complexity bottleneck

The wave of innovation unleashed by the first user-friendly PCs in the 1980's and the Web in the 1990's seems to have gotten drowned in complexity and confusion. Software developers are scrambling to keep their systems up-to-date with all the new standards, plugins and extensions. While we constantly hear announcements of the most spectacular innovations, few seem to reach maturity. The problem is that developers tend to underestimate the complexity of the task environment: today's information systems depend on so many modules, sources of data, network connections, input and output devices that it has become impossible to predict or control their interactions. The result is software full of bugs, corrupted data, security holes, viruses, and other potentially catastrophic side-effects. Moreover, systems have become so complex that the human mind simply no longer can learn or remember all procedures needed to use them. Add to this constant change in hardware, software, protocols, data, and user expectations, mix well, and you have a recipe for chaos.

The complexity bottleneck not only creates stress and confusion, it severely limits the speed of further progress. The number of possible interactions grows exponentially with the number of components. Since components and their capacity increase exponentially, overall complexity increases super-exponentially. The cognitive capacity of developers obviously increases at a much slower rate, thus lagging further and further behind. This means that large projects, such as the Semantic Web, either will get endlessly delayed, or end up with unworkable products.

We need a radically different approach to overcome this bottleneck. One step forward is IBM's "autonomic computing" (http://www.research.ibm.com/autonomic/) initiative. IBM researchers envision systems that function largely independently from their human supervisors, adapting, correcting and repairing themselves whenever a problem occurs. IBM's uses the metaphor of the autonomic nervous system, which runs our body for us without need for conscious intervention. How they hope to achieve this is less clear, though. Their suggestions seem to center around models of feedback, adaptation and control first proposed in the 1950's. While we applaud this "cybernetic" approach to computing, we believe that an even more radical vision is needed: *self-organization.*

Self-organizing systems

A self-organizing system not only regulates or adapts its behavior, it creates its own organization. In that respect it differs fundamentally from our present systems, which are created by their designer. We define organization as *structure with function. Structure* means that the components of a system are arranged in a particular order. It requires both connections, that integrate the parts into a whole, and separations that differentiate subsystems, so as to avoid interference. *Function* means that this structure fulfils a purpose.

Designers obviously create systems for a particular purpose: the function of a watch is to tell time, of a database to store data, of a spreadsheet to calculate. But natural systems have functions too: roots exist to extract nutrients from the soil, stomachs to digest, eyes to see. In the 18th century, the very sophisticated structures and functions of living organisms led bishop Paley to argue that they must have an intelligent designer: God. We now know that no designer is necessary to produce such intelligent organization: natural systems appear to have emerged and evolved without outside intervention or programming. Yet, they are incredibly robust, flexible and adaptive, tackling problems far more complex than any computer system (Kelly, 1994; Heylighen, 2003).

Self-organization then means that a functional structure appears and maintains spontaneously. The control needed to achieve this must be *distributed* over all participating components. If it was centralized in a subsystem or module, then this module could in principle be removed and the system would lose its organization. Take the processor chip out of a computer and it becomes useless. Take any small piece of tissue out of a living brain (as commonly happens during brain surgery), and the brain will continue to function more or less like before.

Self-organizing systems are intrinsically *robust*: they can withstand a variety of errors, perturbations, or even partial destruction. They will repair or correct most damage themselves, getting back to their initial state. When the damage becomes too great, their function will start to deteriorate, but "gracefully", without sudden breakdown. They will adapt their organization to any changes in the environment, learning new "tricks" to cope with unforeseen problems. Out of chaos, they will generate order. Seemingly random perturbations will help—rather than hinder—them in achieving an ever better organization.

This description may sound too good to be true. Yet, there are plenty of systems that exhibit these kinds of qualities. We find them to varying degrees in organisms, brains, ecosystems, societies, markets, swarms, and dissipative chemical systems. Computing too offers a few examples. The TCP/IP protocol that underlies the Internet was designed to be robust enough to maintain communication during a nuclear war. It achieves this by cutting up messages into packets that are sent via different routes and reassembling them at the destination—if necessary resending those that got lost. Neural networks that have learnt to recognize patterns, such as handwriting, still produce pretty good results when part of their nodes and links are deleted. Genetic algorithms find solutions to complex problems by "evolving" subsequent generations of possible solutions (Holland, 1992). They mutate, recombine and reproduce only the best ones, until a good enough solution is found.

Mechanisms of self-organization

These computing examples show the power of self-organization, but only in very limited contexts, where both the components and their desired function are well-defined. How can we apply self-organization to an environment as complex and diverse as the Web, a corporate Intranet, or even a desktop computer with its ever changing software and data configuration? To achieve self-organization on that scale, we need a much deeper insight into how it precisely works (Heylighen 2003). Let us go back to the systems we find in nature, and analyse their mechanisms in terms general enough to be applicable to complex information system.

A self-organizing system consists of a large number of interacting components, such as molecules, neurons, insects or people. The system is dynamic: the components are constantly changing state relative to each other. But because of mutual dependency changes are not arbitrary: some relative states are "preferable", in the sense that they will be reinforced or stabilized, while others are inhibited or eliminated. For example, two molecules that approach each other in the right geometrical configuration may react, forming a chemical bond and thus a larger molecule, or they may simply drift past each other. Two people discussing may either find common ground and establish a working relation, or leave in disagreement.

Changes initially are local: components only interact with their immediate "neighbors". They are virtually independent of components farther away. But self-organization is often defined as *global* order emerging from *local* interactions. We can picture this process as follows. Two interacting components pass through a variety of configurations until they find one that is mutually "satisfactory", i.e. stable. We might say that they have adapted to each other, and now *fit* together. To achieve global order, this fit must *propagate* to the other components. For example, two molecules that have bonded may be joined by a third one, and a fourth one, and so on, eventual forming a macroscopic crystal. Two people who discovered a common interest may start talking about this to others, and end up founding a club, political movement, or company. If the components are interchangeable, like molecules of the same chemical substance, the resulting structure will be regular, like a crystal. If each components has its own, individual characteristics, like a species in an ecosystem, the structure will be more complex: each component has to fit in its own niche within the environment formed by the others.

This propagation of fit is typically self-reinforcing: additional components join ever more quickly. The reason is that a larger assembly exerts a stronger attraction on the remaining independent components, offering more niches in which they can fit. This positive feedback produces an explosive growth or reproduction of the assembly. Growth only stops when the resources are exhausted, that is, when all components that *could* be fit into the assembly *have* been fit. This may happen because the remaining components are too different to fit in this type of configuration, or because they got assimilated into a rival assembly. For example, a chess club will stop growing when the remaining people in town either are not interested in chess, or already belong to a different chess club.

The assembly has now stabilized and feedback becomes mostly negative. This means that it will counteract any loss of organization. Self-maintenance has become its implicit purpose, and each component will perform its function toward this goal. For example, the rules and individual relationships within an established club make it difficult for members to switch to a rival club. The assembly as a whole, however, can still interact with other assemblies, but at a different level. For example, two chess clubs may engage in an inter-club tournament. Thus, assemblies formed from individual components start acting like higher level components. These can in turn self-organize into even higher level components, the way chess clubs may assemble into a federation. This process continues recursively, for as long as there are components to interact with, generating ever higher levels of complexity.

The self-organized system is stable or robust, but this does not mean static or rigid. When the environment changes, the components that directly interact with the environment, will have to adapt their state, until they are fit again. This fit will propagate inwards, until the whole assembly is adapted to the new situation. Thus, the system constantly re-organizes, mutually balancing the different internal and external pressures for change, while trying to maintain its essential organization. The more perturbations it encounters, the larger the variety of different configurations it will explore, and therefore the "better" the eventual solution it settles in. The cyberneticist von Foerster called this principle "order from noise". The thermodynamicist Prigogine (1984) called it "order through fluctuations".

The future of computing?

How can we apply this general vision to information systems? Imagine a variety of components: hardware and software modules, files, websites, interfaces, users... All these components interact by exchanging information. Assume that neighboring components have the capability to mutually adapt: by sending messages back and forth they negotiate until they achieve a common "understanding", in which they both can settle. But coordination does not stop there: it propagates back and forth between all components, until a globally stable order is created. Any change, such as a newly introduced component or local breakdown, will restart the negotiation process with its immediate neigbors. Its effects will ripple further through the neighborhood until this perturbation too is absorbed, and the system is back to equilibrium. Of course, we will need to overcome important hurdles before we can achieve this vision. Most obviously, we must create a universal protocol for interaction that supports unrestricted self-organization.

This all sounds very general and abstract. Can we think of more concrete applications? A well-known example of self-organization is the way ants lay trails of pheromones between various sources of food. Initially, ants explore and leave pheromones randomly, but "good" trails, that lead to rich sources via quick routes are reinforced through positive feedback, while poor trails eventually evaporate. Thus, food sources get organized into a dense and efficient network of foraging paths. Marco Dorigo, pioneer in the domain of "ant algorithms", has shown how a similar mechanism can tackle a variety of computing problems, including the notorious travelling salesman problem (Bonabeau et al., 1998). An important application is the routing of messages along the nodes and links of a communication network: efficient routes are reinforced, less efficient ones abandoned. The resulting organization adapts in real time: if routes become congested, their priority is immediately downgraded, and new routes are explored.

One of us has applied a similar idea to the hyperlink organization of the web (Bollen & Heylighen, 1996; Heylighen & Bollen, 2002). Our learning web algorithms reinforce paths of links that users travel frequently, eventually replacing them by a single link. While users decide locally which link to explore next, the effects of their choice propagate throughout the web. This should eventually lead to a global order: webpages that "fit" together, in the sense that the one is very relevant for the users of the other, are linked directly. Thus, documents that cover the same subject get clustered together. This cluster or assembly can now be represented by a higher level document, presenting an index or summary for the subject. Higher level components themselves become linked and clustered in categories and further into supercategories. Eventually, the web as a whole may self-organize into an efficient, hierarchically structured network of associations, which adapts continuously to newly introduced documents, changes in user demand, etc.

With some minor variations, we could apply this scheme to the construction of shared ontologies, clustering similar concepts into categories, and linking the categories that are most strongly associated. Object-oriented programming too could profit from this approach, with objects mutually negotiating message passing protocols, and spontaneously assembling into higher level objects. (For example, the latter is supported by the SWARM programming environment, Langton et. al., 1999). The same goes for ubiquitous computing or intelligent environments: various devices such as fridges, thermostats, or phones connected to a network can learn to mutually coordinate their activities, thus minimizing the burden on the user.

The possibilities seem endless. Is this the future of computing? Only time can tell...

References

Bollen J. & Heylighen F. (1996) “Algorithms for the Self-organisation of Distributed, Multi-user Networks. ", in: *Cybernetics and Systems '96* R. Trappl (ed.), (Austrian Society for Cybernetics), p. 911-916.

Bonabeau, E., Dorigo, M. & Theraulaz, G. (1998). Swarm Intelligence, Oxford University Press

Heylighen F. & Bollen J. (2002): “Hebbian Algorithms for a Digital Library Recommendation System”, in *Proc. 2002 Int. Conf. on Parallel Processing Workshops* (IEEE Computer Society Press)

Heylighen F. (2003): “The Science of Self-organization and Adaptivity”, in: Knowledge Management, Organizational Intelligence and Learning, and Complexity, in: *The Encyclopedia of Life Support Systems* (Eolss Publishers, Oxford).

Holland, J. H. (1992). *Adaptation in Natural and Artificial Systems*, MIT Press, Cambridge MA.

Kelly, K. (1994). *Out of Control*: *the new biology of machines* Addison-Wesley, New York

Langton C., R. Burkhart, M. Daniels, and A. Lancaster. The Swarm simulation system, 1999. http://www.santafe.edu/projects/swarm.

Prigogine, I. and Stengers, I. (1984). *Order out of Chaos*, Bantam Books, New York