Why Complex Systems Engineering Needs Biological Development

Here we shall discuss the need of Complex Systems Engineering to adopt principles from natural development of complex biological organisms—besides principles of natural evolution—to acomplish the type of performance that biology achieves regularly. We shall situate Complex Systems Engineering and discuss an example of how it could be employed. © 2007 Wiley Periodicals, Inc. Complexity 13:12–21, 2007

W. BANZHAF AND N. PILLAY

W. Banzhaf is from the Department of Computer Science, Memorial University of Newfoundland, St. John's, Newfoundland, Canada A1B 3X5 (e-mail: banshaf@cs.mun.ca)

N. Pillay is from the School of Computer Science, University of KwaZulu-Natal, Pietermaritzburg, KwaZulu-Natal, South Africa

1. INTRODUCTION

In this contribution we shall argue that in order for Engineering to conquer Complex Systems, it will have to embrace processes traditionally confined to the natural world, chief among them evolutionary and developmental processes. We shall derive our argument through the discussion of five theses briefly mentioned in this introduction. We shall contrast this approach to others taken toward Complex Systems Engineering, and put it into the context of the general scientific culture of today. Finally, we shall try to exemplify our point by discussing one particular example of how development could be used in an engineering setting.

Our five theses are:

1. Engineering is like Nature: constructive
2. Science has moved from an understanding of states to an understanding of processes
3. The new core concepts of Science are nonlinearity, self-organization, and adaptation
4. Engineering has come to the limits of a state-based approach
5. Engineering has to adopt the principles underlying the self-organization of complex systems discovered by Science

The next section will give our theses in more detail before section 3 discusses our approach taken towards the new field of Complex Systems Engineering. Section 4 will indicate an example of how the new principles could be employed. Section 5 formulates a challenge before Section 6 concludes with where we believe Engineering and Complex Systems are already close to another today.

WILEY InterScience® DISCOVER SOMETHING GREAT

© 2007 Wiley Periodicals, Inc., Vol. 13, No. 2
DOI 10.1002/cplx.20199
Published online 16 November 2007 in Wiley InterScience (www.interscience.wiley.com)

2. FIVE THESES

2.1. Engineering is Like Nature: Constructive

Engineering intends to build artifacts useful for mankind, or to help to manage processes that can be harvested or guided towards usefulness for mankind. In a literal sense it has the same challenges that Nature has: to produce systems able to function in environments. Engineered systems need to be assembled or constructed, as do natural systems, like organisms, ecosystems, or galaxies. The difference so far has been the intricacy of the task, with Engineering on the simple end of the spectrum, working in strictly defined environments, and Nature on the complex end, open to unpredictable environmental challenges. That is about to change in our opinion, with much more challenging tasks ahead for Engineering which will necessitate a new look at Nature's feats in obtaining robust solutions.

The usual construction process in Engineering starts with a clear specification of the task to be achieved by the entity that needs to be constructed. It continues by analysing the requirements, based on scientific principles, and ethics standards established in Engineering. Once the requirements have been determined, a set of specifications is laid down and tools are prepared to implement them. Typically, the toolset is small and immutable in the sense that the tools are not adaptive in the course of the task.

Contrast that with Nature, often called the "tinkerer" [1]. There is no specification of the task, nor a requirement analysis. Rather, objects are constructed right away. Instead of trying one particular design, ready to turn on at a defined time, Nature has to make use of what already exists, has to modify it and probably use thousands of millions of slightly differing designs in order to find the ones that actually work. So while an engineer is used to invest substantially in the process of design prior to switching on a particular device. Nature would use something already working in certain (other) environments and adapt it to new circumstances, sometimes via migration, sometimes via replication.

The constructive aspect of Nature's working can perhaps best be seen in the development of a multi-cellular organism from a single fertilized egg cell. Development is the process of passing through various stages of life, beginning with a number of subdivisions of the fertilized egg ("cleavage"). Interactions between the genes of the fertilized egg and its environment in form of information, energy and matter provided by its parent erect a complex web of self-maintaining processes which ultimately result in the self-conserving and survival properties of a matured organism.

Looked at from the point of view of the new and developing organism, the challenge is formidable. Thrown into the world of life as a single cell, inside of which resides a genome, plus some initial "hints" from the parent, a state of self-sustenance must be reached as quickly as possible. The new organism can be termed an open system, open to matter, energy, and information from its environment. In that sense, and driven by these flows, the system can be said to self-organize under the control of its genome. This self-organization process is, however, not as simple as it would be in nonliving physical systems. Instead, it is a boot-strapping process whose results are produced by more complicated self-organization influencing the very interactions through which it came to happen.

2.2. Science has Moved from an Understanding of States to an Understanding of Processes

Science, the foundation on which Engineering is built, has traditionally concerned itself with the explanation of phenomena in the world. It has done so by analytical approaches that tried to cut to the bottom of phenomena and objects by literally deconstructing them. This process has (partially) come to a halt, in some disciplines of Science earlier, in others later. For instance, Elementary Particle Physics was busy smashing particles into one another in search for more elementary particles. As the search became more and more intense, and the energies involved in the smashing became more astronomical, the effects returned became less and less elucidating. In essence, one set of particles was replaced by another (even more short-lived) set, and the whole enterprise reached the point where at the lowest end of matter there seems to be a mesh of particles reacting into each others, with some more stable and others less stable.

Another example is the "deciphering" of the human genome. It was easy to get a hold of the object of study, the strand of DNA making up the hereditary material of a cell of the human body. The sequence was read and collected in a database. Molecular Biology realized, however, that collecting the sequence of nucleotides on a strand is not equivalent to understanding the organism. Instead, the sequence assumes meaning only in the context of a particular cellular environment, including the dynamics of that environment. It was realized that the static picture of a sequence is worthless without a clear idea what this environment is built of. Meaning in this case is expressed in the form of behavior.

Thus while this and other efforts in Science show the concern for objects, the more fascinating questions that Science poses are about the origin and the emergence of certain objects, abilities, etc, like the origin of life, the origin of language, the origin of the genetic code [2]. These questions relate to processes, and signify a phase shift in Science, aptly named "from being to becoming" [3]: Science has turned its attention to the processes of natural systems that create certain conditions, objects or abilities, and is trying to decipher the phases by which things natural come into being. Thus Science has put the spotlight on the constructive aspects of nature, and is trying to elucidate and

© 2007 Wiley Periodicals, Inc.
DOI 10.1002/cplx

understand how Nature is achieving its results.

2.3. The New Core Concepts of Science are Nonlinearity, Self-Organization, and Adaptation

In the course of uncovering the processes by which various natural systems came into being, Science had to retool and move away from a linear perspective of the world. Admittedly, linear principles had achieved most astonishing discoveries, as prominently exemplified by the superposition principle in quantum mechanics [4]. But it is the nonlinear, historic approaches of today that yield most results. Aided by an appreciation of nonlinearity in Mathematics [5, 6], Physics, for instance, has embraced non-linearity as an important concept [7–9].

Non-linearity has not only brought great difficulty to analytical solutions of mathematical equations, as they used to be the foundations of physics, but it is one of the key features that connects history to Science. A simple nonlinear phenomenon like a hysteresis, i.e. the lagging behavior of a quantity behind its cause, should help explain this point: While definitely the result of a nonlinear effect, an observable on a hysteresis curve will depend on the direction from which the control parameter came, and thus be history-dependent. In general, hysteresis phenomena cause switching behavior of the underlying system.

Self-organization which would not be possible without nonlinear interactions between components of a system (and thus embraces history, too), has become a prominent phenomenon of study in recent years [10]. Self-organization can be broadly understood as the ability of a system to change its internal structure and its function in response to external circumstances. Notably, a self-organizing system can assemble, construct and stabilize itself, with the help of outside matter, energy or information. Usually, self-organizing systems are open and not in equilibrium state with their environment.

Adaptation has been recognized as one of the processes by which a system can self-organize [11, 12]. Whether it is a system of brain cells that adapt to firing patterns, or a system of streets that adapt to traffic, similar formal models might be able to describe those systems.

2.4. Engineering has Come to the Limits of a State-Based Approach

Engineering has closely associated itself with the notion of machines, contraptions that act, once turned on, more or less independent of an operator to perform certain functions. Usually, machines are constructed by putting their parts together and switching them on, energizing them, in other words. This can, however, only be done after the full list of parts has been properly assembled and installed. Machine parts are modules that are not able to function on their own usually, and thus receive their role in the context of the entire machine. Engineers carefully design machine parts so as to minimize their interaction with the rest of the machine (usually helped by their locality), and to make sure that what interaction remains with the rest is exactly defined and manufactured to as good quality standards as possible. Manufacturing will then make use of the benefit of scale, by producing large amounts of identical parts to these specifications.

Contrast this approach with Nature's construction of living organisms: Organisms are assembled from (identical) molecules of different types. Yet the sequence of assembly and the number of different variants of assembled parts is so large as to allow each organism to become a complete individual. The self-organizing processes involved will allow the history of the assembly to determine part of its outcome. The organism has to start and to continue its life while growing, so it cannot be assembled to a state considered mature, and then turned on to live.

Could individual systems be assembled using Engineering concepts, too? Is there room for an Engineering of systems in boundaries of specification, allowing for multiple, and sometimes surprising realizations of the same functionality? We believe that Engineering needs to face these challenges. If one considers that cars rolling from assembly lines today consist of so many different parts (possibly with multiple variants of parts) that each car is in essence an individual machine, one gets a foretaste of what is to be expected in the future: Combinatorial spaces for cars have grown so big that they are essentially empty, with a few spots occupied by those cars actually ordered by customers. What has so far stayed unaltered is the sequence of assembly, and how parts added to the system determine the further outcome of the assembly. In other words, the potential self-organization properties of cars on the assembly line have not been set loose yet (if that will be possible at all).

In general, it is the complexity of Engineering tasks today that has grown to the point where traditional methods fail. The divide-and-conquer strategy, for instance, is only so effective once it comes to highly complex systems. Proofs of correctness fail in the face of highly complex systems. We expect that studying the processes of construction used by Nature (including its use-and-test approach to correctness) will greatly benefit Engineering.

2.5. Engineering has to Adopt the Principles Underlying the Self-Organization of Complex Systems Discovered by Science

The devices designed in Engineering are moving in the direction of complex systems. Individual systems, with individual trajectories of their generation, will become more and more widespread. Think of the software uploaded onto particular desktop computers. It can be assumed safely that each computer holds a unique combination of software once in the hand of a customer (this is not necessarily the case for those freshly delivered). Admittedly, computers are the most complex machines

© 2007 Wiley Periodicals, Inc.
DOI 10.1002/cplx

built today, and in fact can be considered complexity generators, but other devices are on their way, robots, for one, and other equipment heavily relying on information processing.

While at the moment it can be argued that most of this is a combinatorial effect, it will become more a question of the generation of the systems as we progress. Thus, besides individuality, Engineering has to look at unpredictability (of component features, of sequence of assembly) and the emergence of functionality if it wants to harness natural processes for complex systems engineering.

In short, Engineering has to embrace concepts of self-organization, i.e. concepts of becoming. Important among those concepts are evolution and development, which we will discuss in the next section.

3. SELF-ORGANIZATION, EVOLUTION, DEVELOPMENT

Self-organization is a rather general term referring to the ability of a system to organize into patterns and typically this confers the ability to "respond" to its environment. We have discussed elsewhere [13] the difference one can legitimately draw between self-assembly, self-formation, and self-organization. Suffice it to say here that self-assembly is possibly the most special term in this regard. It comprises the assembly of parts into a whole, directed by the assembling parts and their interactions. Notably, a self-assembling process is usually not recursive, i.e. it cannot move through successive stages of first assembling some elementary parts into more complex parts, which in turn self-assemble into the whole. Self-formation, on the other hand, has a clear notion of sequence [14]. Processes of self-formation can be used to generate more complicated wholes. This requires the developing system to change state repeatedly, with each state determining subsequent states and the sequence of events to follow. Much more complex wholes can be constructed by such a mechanism, and indeed it is possible to generate spatial patterns of enormous complexity. Self-formation finds its limits in the problem of repair and maintenance of the structures formed. Because of the very specific sequence of events that lead to the original result, these needs are virtually impossible to achieve with self-formation. We consider self-organization to be the most general term in this order, including both self-assembly and self-formation, but also self-maintenance, and development. Self-organization does not require a specific sequence to arrive at the end result. Rather, it has a multitude of paths toward the desired goal. Self-organization allows phenomena on different time scales, and thus on a hierarchy of levels, which in turn allows for a recursive consideration of its mechanisms. Other definitions of self-organization are given by Refs. 15–17.

A key mechanism of self-organization is provided by evolution, the "recursive" and accumulative process of trying varieties and picking the relatively better ones. Bar-Yam argues [18] that what he calls "Evolutionary Engineering" is in fact the only way to design Complex Systems. Instead of an analysis and planned design either by a computer or a human alone, systems will be put together by a collaboration of partners with the machine. Bar-Yam envisions teams of designers working on smaller parts of a larger system in parallel, and draws an analogy between the variants of these parts (as they come about by the design process), and individuals in a population. His analogue to selection is the adoption of parts that are successful in their environment from team to team, effectively reproducing the designs in different design teams. The process is kept moving through the competition between teams, always striving to improve the overall performance of the (complex) system. Bar-Yam requires the populations of slightly different components of a system to work in situ while variation is taking place. Therefore he emphasizes the establishment of environments in which such in-situ function can go hand in hand with changes in components. One key aspect he mentions is that individual components need to have a spectrum of functions which allow overlap of the roles of particular components such that a reduction in one particular functionality can be compensated by another component.

While we fully agree with what Bar-Yam states, we have to emphasize that what he describes is a developmental system. Biological development has precisely this feature that in situ elements are changing during the lifetime of an organism. As is well known from Biology, many components of biological systems are multi-functional, a feature which allows them to occasionally acquire more specialization, depending on the overall needs of the system. Variability in components, and overlap of functions are therefore very important aspects of development.

We have argued elsewhere [19] that the problem that life faces in constructing complex organisms is essentially an information dilemma: The size of an organism's genome is small relative to the required information for constructing that organism. The same can be said of artificial complex systems. That is why we argue that Complex Systems Engineering needs inspiration from biological development, shaped by evolution.

Some key aspects of biological development need to be mentioned:

1. The complexity of the organism is too big to be contained in the genome. Thus, the genome is merely used to channel or canalize environmental complexity.
2. An organism is viable at every instant of its developmental process. Fitness of the organism must therefore be incremental, i.e. a primitive functionality is required from the very beginning which subsequently is diversified.
3. The develomental process is robust and fault tolerant and has to include repair mechanisms, with the possibility of regeneration, and with adaptability.

© 2007 Wiley Periodicals, Inc.
DOI 10.1002/cplx

4. Instead of spatially defining complexity of the organism, the developmental process maps it out in time. In other words, if the required complexity cannot be put in place in one go, it needs to be grown. The genome of an organism is then a way to orchestrate that growth.
5. In addition, one should not think only of spatial complexity, but also of the complexity exhibited by behavior. In that sense, mapping a genome which is a spatial configuration of DNA patterns into a dynamical process expressing itself as growth and behavior is as fundamental to an organism as is its spatial complexity.
6. Interestingly, the time dimension has further implications as a way to "mold" results of development, i.e., by shifting times of onset of certain processes, Nature can control outcomes, e.g. evolution of brain size in primates is controlled by time-shifts in giving birth to babies.
7. Growth is closely related to duplication and divergence (processes that are at play in evolution at large): Bodies grow by replication of cells and their gradual divergence into different roles. As such, the division of labor in an organism comes about gradually, and is not established from the beginning by designing different parts.
8. While specialization is irreversible (cells have to commit to certain roles in mature organs), there is no isolation of parts as is known from machines, where parts can be independently designed and tested, to be thrown into a whole entity only after this process.
9. Development allows the exploitation of side-effects, perhaps in the most efficient way possible. Should, in the course of the development and the behavior of the respective organism a particular unintended function turn out to be of advantage, then it will be amplified, depending on that advantage.
10. Because components of a developing system have a whole spectrum of behavioral roles, i.e. are multifunctional to different degrees, open-endedness of the process of evolution is allowed.

While we have argued now on principle that development is an important process to be taken into account if one wants to engineer complex systems, an illustrative example might be in order.

4. AN EXAMPLE

We have presented a developmental approach, based on the functioning of natural systems, as a way forward for complex systems engineering. In this section we will illustrate these concepts with an example. On a daily basis we need some form of scheduling to organise the events in our lives. In the academic world this takes the form of course and examination timetables. In this section we apply the ideas that we have discussed thus far to the domain of examination timetabling. Section 4.1 defines the uncapacitated examination timetabling problem (ETP). A sequential approach to the problem, typical of that employed by traditional engineering systems, is portrayed in Section 4.2. Evolution and development are crucial to the process of self-organization. In Ref. 20 we have shown how evolution can be used to induce solutions to the uncapacitated ETP. Here we take a developmental approach to this problem. This approach is discussed in Section 4.3. Section 4.4 comments on the performance of the sequential and developmental approaches on the uncapacitated ETP.

4.1. The Uncapacitated Examination Timetabling Problem

The ETP basically involves allocating a number of examinations to a set of timeslots. Examinations must be scheduled so as to produce a timetable that satisfies the hard constraints of the problem and minimizes the soft constraint cost. Hard constraints are those constraints that must be met in order for a timetable to be feasible, for example students cannot be scheduled to write two or more examinations at the same time. Soft constraints are generally "wish lists" which we would like the timetable to satisfy; they are often contradictory. Examples of soft constraints include examinations being well-spaced for each group of students; the scheduling of large classes early during the examination period so as to provide sufficient time for marking. All soft constraints cannot be met and the best we can do is to minimize the soft constraint cost.

The uncapacitated version of the ETP does not take room capacity into consideration while the capacitated ETP has the additional hard constraint of not exceeding venue capacities. The Carter benchmarks [21], consisting of 13 real-world problems, are generally used to test and compare the performance of different methodologies applied to the uncapacitated ETP. Details of these benchmarks are listed in Table 1.

For this set of benchmarks a feasible timetable is one in which there are no clashes between exams for students. The soft constraint requires the examinations to be well spaced and is quantified by costs C, calculated using the following equation:

$$C = \frac{1}{S} \sum_{i,j} w(|e_i - e_j|) N_{ij} \qquad (1)$$

where $|e_i - e_j|$ is the distance between the periods of each pair of examinations (e_i, e_j) with common students, N_{ij} is the number of students common to both examinations, S is the total number of students, and w_k is a weight allocated to periods depending on their distance in time. The choice of weights follows usage in the community as $w(1) = 16, w(2) = 8, w(3) = 4, w(4) = 2, w(5) = 1, w(k) = 0 \quad \forall\, k > 5$.

The following section examines the effect of a sequential approach to inducing timetables for this set of benchmarks and Section 4.3 reports on using a developmental approach for this purpose.

© 2007 Wiley Periodicals, Inc.
DOI 10.1002/cplx

TABLE 1

Carter Benchmarks, Including Version Number

Data Set	Institution	Periods	Exams	Students	D
car-f-92 I	Carleton University, Ottawa	32	543	18419	0.14
car-s-91 I	Carleton University, Ottawa	35	682	16925	0.13
ear-f-83 I	Earl Haig Collegiate Institute, Toronto	24	190	1125	0.27
hec-s-92 I	E. des Hautes Etudes Commerciales, Montreal	18	81	2823	0.42
kfu-s-93	King Fahd University, Dharan	20	461	5349	0.06
lse-f-91	London School of Economics	18	381	2726	0.06
pur-s-93 I	Purdue University, Indiana	43	2419	30029	0.03
rye-s-93	Ryerson University, Toronto	23	486	11483	0.08
sta-f-83 I	St Andrew's Junior High School, Toronto	13	139	611	0.14
tre-s-92	Trent University, Peterborough, Ontario	23	261	4360	0.18
uta-s-92 I	Faculty of Arts and Sciences, Univ. of Toronto	35	622	21266	0.13
ute-s-92	Faculty of Engineering, Univ. of Toronto	10	184	2749	0.08
yor-f-83 I	York Mills Collegiate Institute, Toronto	21	181	941	0.29

The density of the conflict matrix, D, provides a measure for the difficulty of the problem.

Both systems have been implemented in Java and all simulations have been run on a WindowsXP machine with an Intel Pentium M 1695 Mhz processor.

4.2. A Sequential Approach to the ETP

A sequential algorithm to solving the uncapacitated ETP is presented in Algorithm 4.1.

This algorithm is characteristic of those usually implemented by engineering systems. Examination timetabling algorithms generally sort the examinations according to a low-level heuristic and examinations are allocated sequentially to the first clash-free timeslot [21]. The low-level heuristic used in this algorithm is the saturation degree, i.e. the number of periods that an exam can be allocated to without causing a clash. In the case of ties the examinations are scheduled numerically. If a clash-free slot cannot be found the examination is allocated to the timeslot that results in a minimum number of clashes. The performance of the sequential algorithm on the 13 Carter benchmarks is depicted in Table 2. The runtime of all simulations was under a second. The soft constraint cost is listed for the data sets for which the algorithm was able to produce feasible timetables.

ALGORITHM 4.1

Sequential Approach to the ETP (C)

SORT the examinations according to the saturation degree
repeat
 Schedule the next exam to the next clash-free timeslot
 if a clash-free slot is not available
  then allocate the exam to the slot with the minimum clashes
until all examinations have been scheduled
return (C)

4.3. A Developmental Approach to the ETP

This section presents a developmental approach to the uncapacitated ETP. Each cell represents a timeslot and contains a set of examinations that can be scheduled at the same time. The process begins with a single cell which is allocated a randomly selected examination. Upon its inception, the position of a timeslot in the timetable is randomly chosen, and can change later on in the developmental process. Examinations not yet scheduled are allocated to existing cells. The saturation degree of the examination is used to determine which examination to allocate next. In the case of ties an examination is randomly selected.

If the allocation of an examination to a cell results in a clash one of two processes is applied. If the number of cells created is less than the maximum permitted cell division occurs. This process results in the cell dividing into two, with the first cell containing the "clashing" exam and the second cell the remaining exams. If the maximum number of timeslots has already been reached, cell interaction is initiated. The cell containing the "clashing" exam interacts with the other cells so as to remove the clash by swapping the exam with that contained in another cell. If cell interaction cannot locate a clash-free cell the examination remains in the original cell.

Cell division also occurs if the maximum number of examinations ($cap_{\max}$) per cell has been exceeded. In this case the cell divides into two cells containing more or less the same number of examinations.

Cell migration is performed on each iteration of the organism's development. Cell migration basically involves changing the position of the timeslots so as to further reduce the soft constraint costs. Two types of migration have been tested, namely, random migration and stimulus-driven migration. Random migration randomly chooses two cells and swaps the position of the cells. Stimulus-driven migration swaps the position of randomly chosen cells until

© 2007 Wiley Periodicals, Inc.
DOI 10.1002/cplx

TABLE 2

Performance of a Sequential Algorithm on the Carter Benchmarks

Data Set	Feasible Timetable Found?	C
car-f-92 I	Yes	11.17
car-s-91 I	Yes	13.06
ear-f-83 I	Yes	70.45
hec-s-92 I	Yes	27.31
kfu-s-93	No	N/A
lse-f-91	Yes	28.7
pur-s-93 I	Yes	16.77
rye-s-93	Yes	26.87
sta-f-83 I	Yes	207.49
tre-s-92	Yes	15.17
uta-s-92 I	Yes	7.77
ute-s-92	No	N/A
yor-f-83 I	No	N/A

C: Soft Constraint Cost.

there is a reduction in the soft constraint cost. A limit ($imp_{\max}$) is set on the number of steps of improvement. If there is no improvement in the soft constraint cost after $imp_{\max}$ attempts the process is stopped.

The overall algorithm to create an organism is illustrated in Algorithm 4.2. A population of n organisms is created. The organism with the best hard and soft constraint cost is reported as the solution.

The developmental approach produced feasible timetables for all 13 benchmarks. The soft constraint values for both random and stimulus-driven migration are listed in Table 3. A value of 100 was used for n, 5 for $cap_{\max}$ and 10 for $imp_{\max}$. Because of the randomness associated with the approach, 10 runs, each using a different random number generator seed, were performed for each type of migration. Table 3 lists the best of the 10 runs.

The runtimes for the simulations are also specified. Note that despite the nondeterminism associated with the developmental approach the runtimes of the system are not excessive and are under a minute for the majority of the data sets.

4.4. Discussion

The developmental approach (DA) to examination timetabling is constructive. The overall process begins with a single cell which is developed into an organism via cell division, cell interaction, and migration. The best of n timetables is returned as a solution. This process is nonlinear. The internal structure of the organism representing the timetable is adapted during the growth process so as to reduce the constraints violated by the timetable, i.e. the development process is based on self-organization. Each component of the timetable is not developed in isolation but in cognisance of other cells of the organism. Going back to the list of features of developmental systems formulated in Section 3, we see the following aspects of this algorithm:

1. The basic genome of an exam timetable is an entity comprised of timeslots and examinations. The arrangement of the examinations and slots so as to produce a feasible timetable with minimum soft constraint cost is a complex combinatorial problem. While it can be argued that the problem does not supersede the abilities of an artificial genomic representation, it is sufficiently complex to try a developmental approach.
2. The fitness of an organism is defined in terms of the hard constraint and soft constraint costs. As an organism develops its behavior and hence its fitness changes. The developmental approach aims at improving the organisms fitness during its growth. The organism is developed so as to eliminate hard constraint costs and reduce soft constraint costs.
3. The DA allows for adaptability by means of migration. Migration allows for the structure of the organism to be changed at each stage of the developmental process so as to reduce the soft constraint costs. Furthermore, cell division and cell interaction facilitate partial regeneration of organism so as to remove hard constraints.
4. The DA does not construct timetables as spatially laid-out structures in one go. Instead a timetable is

ALGORITHM 4.2

Organism Creation (C)

Create a single cell
Randomly choose a position for the cell
Assign a randomly selected examination to the cell
repeat
 if a cell has exceeded its maximum capacity
  then perform cell division
 if two or more cells have been created
  then perform cell migration
 for each existing cell
  do Assign an exam to the cell
 if the assignment results in a clash
  then **if** the maximum number of cells has not been reached
   then Perform cell division
   else if Perform cell interaction
until all examinations are allocated
return (C)

© 2007 Wiley Periodicals, Inc.
DOI 10.1002/cplx

TABLE 3

Performance of the Developmental Approach on the Carter Benchmarks.

	Random Migration		Stimulus-Driven Migration	
Data Set	C	Runtime	C	Runtime
car-f-92 I	6.97	18	5.92	113
car-s-91 I	7.90	29	7.09	197
ear-f-83 I	51.07	2	45.45	10
hec-s-92 I	16.45	1	13.04	1
kfu-s-93	22.88	12	18.37	41
lse-f-91	17.82	8	15.05	31
pur-s-93 I	8.94	606	7.89	2682
rye-s-93	16.97	14	14.12	53
sta-f-83 I	163.84	1	160.99	4
tre-s-92	11.69	4	10.29	20
uta-s-92 I	5.35	23	4.61	135
ute-s-92	35.97	2	31.79	8
yor-f-83 I	51.71	2	45.29	14

C: Soft Constraint Cost. Runtime is given in seconds.

developed over time as an organism requiring interaction between its components and reorganization of its structure so as to reduce constraint costs.

5. As an organism grows its behaviour changes, i.e. the time at which an examination takes place changes so as to reduce hard and soft constraint costs. Furthermore, the number examinations scheduled is constantly changing.
6. The DA models a timetable into a particular form over time. As the organism develops examinations are bound to particular timeslots and timeslots are bound to certain positions in the overall structure.
7. The location of timeslots in the overall structure is not set at the beginning of the development process. Instead the position of periods in the organism are developed over time so as to reduce the constraint costs. Furthermore, examinations may be re-scheduled during the development process as a result of cell division and cell interaction.
8. At the end of the developmental process the allocation of examinations are fixed to certain periods and the these periods are fixed to certain position in the timetable. However, each component of the timetable is not developed independently but as a result of interaction between the cells representing each component.
9. Exploitation of side-effects is not evident in the evolutionary process for this example.
10. Each timeslot, and thus set of examinations, can be placed at different positions of the timetable.

Contrast that with the linear process followed by the sequential approach: The positions of timeslots are fixed at the beginning of the procedure and examinations are allocated sequentially to the next clash-free slot. During the construction there is no interaction between components of the timetable. This method does not facilitate self-organization and hence adaptation of the internal structure to the environment. The sequential method constructs the timetable at one go and the complexity and behavior of the structure is not developed over time. This method is typical of those employed by engineering systems.

The potential of the developmental approach is evident from the results presented earlier. The sequential approach, which is typical of the algorithms generally employed by engineering systems, was unable to induce clash free timetables for three of the benchmarks. The flexibility of the developmental approach enabled it to overcome the limitations of the linear method and this approach generated feasible timetables for all 13 of the benchmarks.

The differences in soft constraint costs for the feasible timetables found by both approaches are tabulated in Table 4. These differences are listed for both types of migration. The second column lists the percentage difference in the soft constraint cost of the timetable generated by the sequential approach and that produced by the developmental approach using random migration. Column three lists these differences for the developmental approach using stimulus-driven migration. Note that even in the case of

TABLE 4

Percentage Differences in the Soft Constraint Cost for Timetables Generated by the Sequential Approach and the Developmental Approach.

Data Set	RM (%)	SDM (%)
car-f-92 I	37.60	47.00
car-s-91 I	39.51	45.71
ear-f-83 I	27.51	35.49
hec-s-92 I	39.77	52.25
lse-f-91	37.91	47.56
pur-s-93 I	46.69	52.95
rye-s-93	36.84	47.45
sta-f-83 I	21.04	22.41
tre-s-92	22.94	32.17
uta-s-92 I	31.15	40.67

RM, random migration; SDM, stimulus-driven migration.

© 2007 Wiley Periodicals, Inc.
DOI 10.1002/cplx

random migration the developmental approach has produced timetables of a much higher quality than the sequential approach.

5. A CHALLENGE

We shall end the discussion here with a task from the realm of computing. This is done in the form of a challenge. We have not developed such a system in our labs, but we imagine that it can be done, and that it can be done in a reasonable timeframe. We have put this out as a challenge some months ago [22], but here we will take the freedom to outline the challenge in more detail. The reader ought to be warned, however, that the language used here is tentative.

The challenge is to design an operating system [23] for a computer that follows principles of natural biological development. Notably, this OS should be able to recover from exceptions, and be as much as possible resilient. In a sense, the OS should behave as a living organism whose purpose is to survive, with whatever means it has to do that.

If one compares the developmental process of a biological organism to the booting process of an operating system, a number of analogies can be drawn.

- Development starts with a single fertilized egg—an OS starts through accessing a boot sector.
- The environment for early development is set by the parent organism—what can be called environment for an OS, parameter settings, are stored from earlier use of an OS, or in ROM.
- After a quick growth spurt, a developing organism becomes structured through the interaction of cells with each other and their environment—once the boot sector is loaded, the OS starts to administer resources and loads further code into the system's portion of memory.
- As more cells are generated, their roles become more and more defined, until their is little flexibility for rededication of a cell—the number of processes grows until different applications have been launched.

We can see that there are analogies between the starting up of an OS and development, at least on a superficial level. So why not drive those analogies a little further, by thinking of "stem processes" that become gradually committed to certain applications? What would one expect from such a more biologically oriented OS if new peripheral hardware became accessible to the system?

OSs today are huge program systems of millions of lines of code. Whether they are approaching the realm of complex systems can certainly be disputed, but in our analysis they would make a good example. An OS is required to be open to user input, the environment is constantly changing and the purpose of the system is time-dependent behavior rather than simply a structure.

6. CONCLUSION

In conclusion we have argued for the benefits of adopting principles from biological development into Complex Systems Engineering. We have argued on a principled level, but tried to bolster the case with an example taken from a real-world problem domain. There, we have seen, that developmental approaches, even in the rudimentary form we have chosen to implement, are indeed useful. Finally, we have proposed a challenging application which will make the advantages of the approach even more evident.

It might be asked why we have not mentioned the concept of emergence which features prominently among nonlinear systems. Emergence has many times been invoked to explain phenomena in complex systems. We do not want to criticize those explanations, very good overview books exist, see for example [24, 25].

It seems to us, however, that in the context of Engineering, emergence must be thought through and looked at from a different perspective. About, for instance, defines emergence as phenomena that are independent of their implementation [26]. This is an extremely interesting definition (see also [27]) but leaves us to wonder how we can then achieve it in a reality setting.

Our point of view is that the fact that a phenomenon is emergent does not say anything about its stability or robustness against changes in its environment. So if an emergent phenomenon should become useful, it needs to be stabilized and made robust. A key mechanism to achieve this stabilization is to "catch" the emergent phenomenon in a network. Networks provide structures that allow emergence not only to appear, but to stay, to put it simply.

Perhaps this is the reason why the closest Engineers so far ventured to complex systems is in the area of network design and management. Different types of networks can be realized by physical structures and can quickly grow into complex systems. Robustness and adaptability of infrastructure networks are of much concern to Engineers. It is perhaps here where Complex Systems Engineering will come into its own.

ACKNOWLEDGMENTS

The authors gratefully acknowledge support from the Canadian NSERC Research Agency to W.B. under Discovery Grant RGPIN 283304-04 and to N.P from the South African National Foundation for Research (NRF).

REFERENCES

1. Casci, T. A tinkerer's tales. Nat Rev Genet 2006, 7, 411.
2. Smith, J.M.; Szathmary, E. The Origins of Life: From the Birth of Life to the Origin of Language; Oxford University Press: Oxford, 1999.
3. Prigogine, I. From Being to Becoming. Time and Complexity in the Physical Sciences; W. H. Freeman: New York, 1980.

© 2007 Wiley Periodicals, Inc.
DOI 10.1002/cplx

4. Jauch, J.M.; Morrow, R.A. Foundations of Quantum Mechanics. Am J Phys 1968, 36, 771.
5. Thom, R. Structural Stability and Morphogenesis; Benjamin: Reading, MA, 1975.
6. Mandelbrot, B.B.; Wheeler, J.A. The fractal geometry of nature. Am J Phys 1983, 51, 286.
7. Haken, H. Synergetics. An introduction, Series in Synergetics; Springer: New York 1983.
8. Sagdeev, R.Z.; Usikov, D.A.; Zaslavskij, G.M.; Sagdeev, I.R. Nonlinear Physics; Harwood Academic: New York, 1988.
9. Scheck, F.A. Mechanics: From Newton's Laws to Deterministic Chaos; Springer-Verlag: Berlin, 1999.
10. Banzhaf, W. Self-organizing Systems. Encyclopedia of Physical Science and Technology, Academic Press: New York, 2002; pp 589–598.
11. Kohonen, T. Self-Organization and Associative Memory; Springer-Verlag: New York, 1989.
12. Haken, H. Information and Self-Organization: A Macroscopic Approach to Complex Systems; Springer: New York, 2000.
13. Banzhaf, W. Artificial Chemistries–Towards constructive dynamical systems. Nonlinear Phenom Complex Syst 2002, 5, 318–324.
14. Janusonis, S., Ed. Self-Formation Theory and Applications; SCITEC Publications: Zuerich, Switzerland, 2003.
15. Gunderson, L.H. Ecological Resilience–In Theory and Application. Ann Rev Ecol Syst 2000, 31, 425–439.
16. Schweitzer, F., Ed. Self-Organization of Complex Structures: From Individual to Collective Dynamics; Taylor & Francis: Washington, DC, 1997.
17. Waldrop, M. Complexity: The Emerging Science at the Edge of Order and Chaos; Simon & Schuster: New York, 1992.
18. Bar-Yam, Y. About Engineering Complex Systems: Multiscale Analysis and Evolutionary Engineering, Vol. 3464; Springer: New York, 2005; pp 16–31.
19. Banzhaf, W.; Miller, J. The challenge of complexity. In: Frontiers in Evolutionary Computation; Menon, A., Ed.; Kluwer Academic: Boston, 2004; pp. 243–260.
20. Pillay, N.; Banzhaf, W. An informed genetic algorithm for the examination timetabling problem. Appl Soft Comput, in press.
21. Carter, M.W.; Laporte, G.; Lee, S.Y. Examination timetabling: Algorithmic strategies and applications. J Oper Res Soc 1996, 47, 373–383.
22. Banzhaf, W.; Beslon, G.; Christensen, S.; Foster, J.A.; Képès, F.; Lefort, V.; Miller, J.F.; Radman, M.; Ramsden, J.J. Guidelines: From artificial evolution to computational evolution: a research agenda. Nat Rev Genet 2006, 7, 729–735.
23. Silberschatz, A.; Galvin, P.B.; Gagne, G.; Silberschatz, A.; Galvin, P.B.; Gagne, G. Operating System Concepts, 6th ed; Wiley: New York, 2003.
24. Holland, J.H. Emergence: From Chaos to Order; Addison-Wesley: Reading, MA, 1998.
25. Morowitz, H.J. The Emergence of Everything: How the World Became Complex; Oxford University Press: New York, 2002.
26. Abbott, R. Complex systems + systems engineering = complex systems engineering. Manuscript, 2006. Available online at http://arxiv.org/ftp/cs/papers/0603/0603127.pdf.
27. Abbott, R. Putting complex systems to work. Complexity, 2007, 13, 30–49.

© 2007 Wiley Periodicals, Inc.
DOI 10.1002/cplx