Chapter 1

Complex Engineered Systems: A New Paradigm

Ali A. Minai
Department of Electrical & Computer Engineering and Computer Science
University of Cincinnati
Cincinnati, OH 45221
Ali.Minai@uc.edu

Dan Braha
University of Massachusetts
North Dartmouth, MA 02747
New England Complex Systems Institute
Cambridge, MA 02138
dbraha@umassd.edu
braha@necsi.org

Yaneer Bar-Yam
New England Complex Systems Institute
Cambridge MA 02138
yaneer@necsi.org

1. Introduction

Human history is often seen as an inexorable march towards greater complexity — in ideas, artifacts, social, political and economic systems, technology, and in the structure of life itself. While we do not have detailed knowledge of ancient times, it is reasonable to conclude that the average resident of New York City today faces a world of much greater complexity than the average denizen of Carthage or Tikal. A careful consideration of this change, however, suggests that most of it has occurred recently, and has been driven primarily by the emergence of technology as a force in human life. In the 4000 years separating the Indus Valley Civilization from 18th century Europe, human transportation evolved from the bullock cart to the hansom, and the methods of communication used by George Washington did not differ significantly from those used by Alexander or Rameses. The world has moved radically towards greater complexity in the last two centuries. We have moved from

buggies and letter couriers to airplanes and the Internet — an increase in capacity, and through its diversity also in complexity, orders of magnitude greater than that accumulated through the rest of human history. In addition to creating iconic artifacts — the airplane, the car, the computer, the television, etc. — this change has had a profound affect on the *scope* of experience by creating massive, connected and multi-level systems — traffic networks, power grids, markets, multinational corporations — that defy analytical understanding and seem to have a life of their own. This is where complexity truly enters our lives.

Everyone would agree that a microprocessor, with its millions of electronic components, is an extremely complicated system. The same can be said of the U.S. economy. Both the microprocessor and the economy are human constructions, but there is clearly a significant difference between them. The complicated microprocessor was carefully designed and tested by a team of engineers who placed every electronic component in its place with the utmost precision, and that is why it works. But no one designed the U.S. economy, and no one can claim to entirely understand or control it — and yet it works! And while the microprocessor can be augmented only through a careful redesign by competent engineers, the economy grows (and shrinks) on its own, without explicit control by anyone, and yet shows little sign of catastrophic strain. Also, the successful operation of a microprocessor is highly dependent on the successful operation of every one of its core sub-components, while the efficiency of the economy is much more robust to perturbations and failures at the level of its constituent elements. Looking around, one can see many other systems with the same characteristics: Communication networks, transportation networks, cities, societies, markets, organisms, insect colonies, ecosystems. What is it that unites these systems, and makes them different from airplanes and computers? And can something be learned from them that would help us build not only better airplanes and computers, but also smarter robots, safer buildings, more effective disaster response systems, and better planetary probes? What, one might ask, can engineers learn from the birds and the bees? A complementary goal is to utilize the knowledge of engineering in gaining insight into natural phenomena. For example, the ultra robustness of inter- and intra- cellular activities may be attributed, in part, to a highly sophisticated hierarchy of feedback loops—an elementary concept in engineering control theory—operating at multiple layers and multiple scales. The increased demand for reliable and disturbance-free power systems, in turn, could lead to the development of sophisticated self-healing and recovery technologies that embody biologically-inspired procedures. More generally, we can ask how we can understand the relationship of structure to function in nature through engineering concepts, for the benefit of science. Before addressing this question explicitly, it is useful to look in greater detail at the systems being considered. These systems—markets, insect colonies, etc.—have come to be called *complex systems* [3], not to be confused with merely very *complicated* systems such as microprocessors and aircraft carriers. Such a designation is useful because these systems arguably share fundamental characteristics [4, 16], and this is where we begin.

2. Fundamental Characteristics

Perhaps the single most important characteristic shared by all complex systems is *self-organization*: Self-organization can be described as the spontaneous appearance of large-scale organization through limited interactions among simple components. Nature is replete with examples of self-organization. From galaxies to tornadoes, from canyons to crystals, from ecosystems to cells, self-organization is responsible for most, if not all, of the order we see around us. Upon reflection, this is not surprising, since most forces in Nature act over short distances and can, therefore, only support limited interactions among components such as subatomic particles, molecules, stars, and organisms. Yet, large-scale structure is ubiquitous. In a broad sense, some form of self-organization must underlie almost everything, though some examples of it seem more profound than others (today we are much more surprised and impressed by termite nests than by crystals!) But if self-organization is so common, why is it a useful concept at all? The answer to this is that, while many natural systems show some form of self-organization, almost none of the systems explicitly engineered by humans do so. Thus, the effort to understand and build self-organizing systems is, in a sense, an attempt to create systems analogous to "natural" ones—with all the profound strengths and subtle weaknesses of these systems. Not only does this promise a leap in the scope of engineering, it would also allow a better understanding of those human systems—such as economies and cities—that do demonstrate self-organization. The latter are, therefore, both a source of ideas about self-organization and a target for the application of these ideas.

While many man-made systems fail to exhibit spontaneous self-organization, we suggest that self-organization does occur in the human processes associated with design. For example, between the time specifications (requirements and constraints) are assigned to design teams and the time the artifact achieves its final form. During this period, the design process is a highly social process consisting of hundreds of designers, customers, and other participants. These actors—each a complex biological system in itself! — are involved in creating and refining a shared meaning of requirements and potential solutions through continual negotiations, deliberations, explanations, evaluations, and revision [16, 23]. Another form of self-organization refers to the act of successive changes or improvements made to previously implemented man-made systems. This ecology of evolving man-made systems is often driven by a multitude of locally-operating, noisy, socio-technical processes, and frequently involves adaptation processes that lead to better-fit new systems. Moreover, when engineering is considered as embedded in the socio-economic marketplace, the role of self-organization is apparent. Characterizing the real-world structure, and eventually the dynamics of these complex design/redesign processes, may lead to the development of guidelines for coping with complexity. It would also suggest ways for improving the multi-agent decision making process, and the search for innovative engineered systems. By contrast, the more conventional approaches to systems engineering often strive to eliminate self-organization processes in favor of reductive piece by piece design characteristic of the way complicated rather that complex systems arise.

The first fundamental insight provided by self-organizing complex systems is that *non-trivial, large-scale order can be produced by simple processes involving interactions operating locally on simple agents or components.* This insight, modifies the simplistic but common assumption that cause and effect must operate at the same scale. Over time, it is possible for the effect of local interactions to aggregate together in creating large scale order. The principle of self-organization explains the origin of collective patterns in complex systems as well as many aspects of their functioning through collective acts and collective response. Indeed, it demonstrates that the emergence of large-scale order is a process just as general (and opposite) to the process of increasing entropy, and that pattern formation is an integral part of most complex systems' functionality. One might say that, for a complex system, "becoming" is "being." This contrasts sharply with the classical paradigm in engineering with its clear distinction between the design and production phase on the one hand and the functional phase on the other. Even systems considered to be adaptive (such as adaptive controllers or most neural networks) follow this two-phase paradigm, allowing adaptation only in the superficial sense of parameter adjustment whereas complex systems change not only their parameters but also their fundamental structures and processes. Thus, complex systems have been described as "operating far from equilibrium" — in our context this implies that they undergo major changes in the context of operational activity, a notion that is an anathema in classical engineering.

The second fundamental insight provided by complex systems is that *highly complex functional systems (more complex than their creators) can only arise through evolutionary processes of selection in the context of actual tasks*. This insight is based both upon the fundamental fallacy of the concepts of spontaneous generation in biological and other complex systems, and theorems that prove the inadequacy of testing to fully characterize complex systems [3, 6, 7, 16, 17, 27]. This statement contrasts fundamentally with the ongoing efforts to design large real time response systems by specification followed by implementation. The increasing tendency to spiral and recursive implementation is only a partial adoption of the fundamental need to implement parallel in-situ evolutionary processes that are capable of creating much more complex systems than those that can be planned by conventional specification driven processes.

Complex systems emerge and function in complex, dynamic environments, and their characteristics reflect this reality. As technology seeks to produce systems that can operate in similar situations, it seems appropriate to turn to the principles underlying existing complex systems. However, this will require a drastic re-evaluation of many fundamental assumptions and methods of the classical engineering paradigm. This is indeed one of the primary focuses of this book.

3. Engineering Complex Systems

The structure of a complex system is not the result of a historic design process, but a contingent process of evolution. Thus, it does not reflect the principle of static optimality and rational decision-making often used as the basis of engineering design,

but of an evolving fitness constrained by a dynamic (perhaps co-evolving) space of possibilities. This is precisely what makes complex systems suitable for operation in complex, dynamic environments, but it also means that the criteria used to determine the quality and correctness of engineered systems do not apply. It is interesting to consider this issue in some detail, since it underlies virtually all the material included in this book.

3.1 The Classical Engineering Process

The goal of the classical engineering process is to produce efficient and reliable systems that meet pre-specified constraints and pre-specified standards of performance in pre-specified situations. It is fundamentally a goal-oriented process, seeking to achieve *known* specific ends using *well-understood* means. The principle underlying this process is what one might call the *tool paradigm*: *Every engineered system is a tool made to serve the ends of its user*. Not surprisingly, the dominant themes are predictability, reliability, stability, controllability and precision. After all, a tool that cannot be controlled completely by its user or varies greatly in performance over time is not very useful.

Broadly, the classical engineering process may be seen in terms of the following steps:

- **Functional Specification**: The first step is usually a specification of what the system is expected to do. It is worth noting that this usually includes constraints and tolerances that, implicitly, represent a prediction of the circumstances in which the system will need to operate.

- **Design:** This is the main component of the engineering process, where the system is designed carefully in terms of its components – often by several teams of engineers. The design process may occur at many levels sequentially or simultaneously, with different teams working at each level. Today there is significant feedback and interaction between these teams, resulting in a process with multiple loops and a complex network of influences [14, 15, 23, 34]. In a broad sense, however, the process is fundamentally "top-down" since it moves logically from a desired functionality towards a design that implements that functionality. Levels of design are defined in terms of level of detail. The prime motivation at every level is always, "How can subtask X be done using the components and methods available?" Each team might ask this question at its own level, but in this too, the functions desired at lower (more detailed) levels typically flow from the needs already articulated at higher levels, and ultimately from the pre-specified functionality desired from the system overall, and the component decompositions that preceded it.

- **Testing and Validation:** Once designed, the system is tested under a set of conditions designed to mimic reality to ensure that it performs as needed, to discover flaws and to correct them. Both simulation and fabricated prototypes

may be used. Today, this process often operates in a loop with the design process, and may even involve changes in specifications under extreme circumstances. Still, in the traditional engineering process testing intrinsically follows design and precedes implementation. This process assumes that both the task specified and the environment in which performance is to occur are sufficiently well known to be embodied in a reasonable (as measured by time and effort) number of tests. Once the system is tested and validated, it is deemed to meet the specifications to which it was built. We note that the inadequacy of conventional sequential specification, design and testing for highly complex systems is manifest today in that many systems — e.g., software — undergo additional field-testing (e.g. market-testing), where prototypes of the system are made available to customers for testing in actual applications. The results are then fed back into the design process. However, as common bugs in even highly tested systems show, the testing process for complex systems is never complete [6, 14, 34].

- **Manufacturing:** Once the system has been designed and tested, *exact copies* of it are manufactured in appropriate numbers, ranging from a few to millions or even billions. The users of these copies purchase them in the expectation that each copy functions *precisely* like the original design, and satisfies the desired functionality. Thus, the skill and diligence expended by the engineers on designing and testing the system becomes the guarantor of the system's reliability to the end-user. This is true even in the very exceptional cases (such as the space shuttle) where only one system is built, and that system still relies on the quality of many mass-produced components.

The process described above, with minor variations, underlies the production of almost all modern engineered artifacts from automobiles to paint, from computers to houses, and from widgets to satellites. The basic mode can be described in a sentence: *Given a problem to solve, figure out how to do it once, and then do it the same way each time*. Like the scientific method, this "engineering method" has developed over thousands of years with contributions from ancient cultures and modern ones. The shipwrights of ancient China, the builders of ancient Egypt and the sword smiths of ancient Rome used essentially the same methodology, though with less precision and, therefore, greater variation. With each advance in mathematics, physics, chemistry and mechanization, the process became better understood and more precise, leading to today's robotic assembly lines and precision fabs. With the possibility of engineering complex systems, we are facing a whole new paradigm in engineering, and it is instructive to reflect on what it offers in comparison with the existing paradigm.

3.2 The Logic of the Classical Paradigm

The classical engineering process described above has several notable characteristics that define its scope, determine its logic, and circumscribe its possibilities. Before turning to complex systems, we look explicitly at some of these characteristics. The most important assumption is that the problem to be solved is uniquely and clearly

specified from the outset. In a sense, this specification is extrinsic to the engineering process, but this assumption is necessary as a basis for traditional engineering. The engineering steps engage the solution of this prespecified problem.

3.2.1 The search for a single solution

The goal of traditional engineering is to seek one solution, which often revolves around a unique design concept, for the specified problem. Though engineers understand fully that every problem admits of multiple design concepts, it is always assumed that, in the end, the engineering process will produce a single acceptable—perhaps optimal—design. Multiple design concepts may be considered during the process, but the result is often a single final design, though a certain amount of customization might be left in this design for reasons that address varying market/customer demands. The need for converging onto a single design concept is motivated by several factors. Most importantly, it is the basis of well-characterized systems that can be patented, branded and marketed as distinct, well-defined products that are the best possible solution to a well-defined problem. And, finally, it produces economies of scale through mass production that make products more affordable.

3.2.2 Seeking well-behaved systems

The classical engineering process seeks systems whose behavior can be predicted and encapsulated by precise description. This is reflected in the characteristics that are seen as the *sine qua non* of all engineered systems: stability, predictability, reliability, transparency, controllability, and—ideally—optimality. Under the current paradigm, these systems lack the ability to adapt, evolve, innovate or grow after release. Even such characteristics as robustness and resilience are seen in terms of the ability of the system's performance to be insensitive to pre-specified sources of uncertainty rather than to the possibility that the system might adapt itself to faults or changing circumstances. Adaptation, even when included in the system, is carefully circumscribed within predictable limits. The purpose of good design is seen as the elimination of the unforeseen, the unexpected and the unintended, not as the consideration of the unforeseeable, the unthinkable and the unknown. Indeed, the choice of optimality as the ultimate goal reflects the essential optimistic reductionism of the classical engineering paradigm. Since complexity often complicates the search for optimality, there is a strong tendency to control or limit complexity instead of embracing it. This has worked remarkably well, but is becoming untenable as engineering expands its scope to systems that are inherently complex.

3.2.3 Engineering as top-down problem-solving

The classical top-down design process depends fundamentally on the reductionistic assumption that any system can be described wholly by describing the behavior of its parts and their interactions. This assumption enables designers to work at different levels of abstraction with the confidence that subsystems at each level can be analyzed

and synthesized completely in terms of subsystems at the next lower level. Thus, a VLSI chip designed at the level of functional modules can be translated into a register-level description, and thence to gate-level, device-level and wafer-level descriptions. At each step, the desired higher-level functionality is specified *before* its lower-level implementation is designed, so that the design process is reduced to a series of problem-solving activities. As mentioned earlier, this corresponds to an inherently problem solving view of engineering where the goal is to produce tools for specific predefined purposes whose utility is taken as given. This can be contrasted with what one might call a "meta-utilitarian" view where utility itself is subject to reassessment as the environment changes and as what can be done changes due to unanticipated innovations or insights, including bottom-up self-organization that generates unexpected emergent phenomena [36].

3.2.4 What the Classical Approach Offers

The classical engineering approach has been remarkably successful, and is responsible for virtually all the technological innovations we see around us. In particular, it confers several crucial attributes on the systems it produces—attributes that have come to embody the very notion of an engineered system. These are:

Stability: The system's performance is insensitive to pre-specified variations in the system's parameters and external environment.

Predictability: The system works in predictable ways.

Reliability: The ability of the system to perform a required function under stated conditions for a stated period of time.

Transparency: All the structures and processes in the system can be described explicitly.

Controllability: The design process and the system can be controlled directly.

4. A New Paradigm for Engineering Complex Systems

Classical engineering requires prediction of the environment in which the system will operate, the conditions it will face, and the tasks it will be required to perform. Very clever designers must then determine how these tasks can be performed as desired, and by what components put together in what fashion. Thus, the designers determine not only the *behavior* or *functionality* of the system but also the *process* or *procedure* by which that is achieved. The ultimate performance of the system depends wholly on the knowledge, competence, skill and imagination of the designers. Nothing, as far as possible, is left unspecified. All loops are closed, all contingencies considered. The result is a well-designed, reliable system that operates exactly as advertised within the limits of its tolerances. It is not expected to change (beyond wear-and-tear or

accidents that are detrimental) or to "grow," and its quality is measured by the *stability* of its performance: How long can it continue to function at the same level as when it was new? This is the question we ask when judging the quality of a computer, a dishwasher, an automobile or an airplane. Significantly, this is not the way we judge the performance of an employee, a pet, an economy, or a society. What is the difference?

The primary difference is that systems designed through the classical engineering process are expected to perform foreseeable tasks in a bounded environment, whereas complex systems such as humans or other organisms are expected to function in complex, open environments with unforeseeable contingencies. One can argue that the classical engineering process is an ideal one for the former case, and is unlikely to be superseded by a process based on self-organization or adaptation. Barring radical changes in concept, aircraft and automobiles will continue to be designed the old-fashioned way because it offers predictability, reliability, controllability, etc. However, the classical engineering process suffers from serious drawbacks when applied to complex systems.

Many engineering applications, such as real-time decision support, communications and control, are reaching the point where classical methods are no longer feasible for reasons of system interdependencies and complexity. At the same time, it is increasingly clear that existing complex systems, both natural and artificial, handle these problems with ease and efficiency. Complex systems, once understood, promise a much wider repertoire of techniques and algorithms needed to engineer large systems that can work in complex, dynamic environments.

Ultimately the need to go beyond conventional engineering practice arises from the recognition that only complex systems can perform complex tasks. Equivalently, in a highly uncertain (complex) environment, planning the response of a system is guaranteed to lead to failure, precisely because we cannot anticipate all of the possibilities that may be encountered.

4.1 The Logic of Complex Systems Engineering

The key difference in the logic underlying the classical and complex engineering paradigms is in the definition of the objective. Complex systems engineering [5, 7, 8] does not primarily seek to produce predictable, stable behavior within carefully constrained situations, but rather to obtain systems capable of adaptation, change and novelty—even surprise. Some of the key concepts underlying this approach are:

Local action, global consequences. The scalability of a wide variety of complex systems arises primarily from the fact that most of the relevant processes—processes with high information requirements—are performed locally and, therefore, are low cost. Global consequences arise through self-organization or adaptation rather than explicit design, aided in many cases by non-specific global processes such as modulatory signals or global threshold-setting. The complex systems engineer, therefore, does not seek to design the system in all its details, but focuses instead on configuring the context and the local interactions that may lead to effective global

behavior. Generic methods for the design of such local interactions are, indeed, one of the biggest challenges facing complex systems research.

Expectation of the unexpected. Complex systems are required in order to function in environments that are themselves too complex to be completely predicted or constrained. In such operating environments, the exact analytic relationships between the system parameters and the system behaviors are unknown or cannot be practically determined. As such, complex systems have to incorporate capabilities necessary to handle novel circumstances with incomplete information (low observability) and limited control (poor controllability). Also, these systems must be able to dynamically modify the way information about the uncertain external environment (including changes in the system's initial requirements) is represented within the system, and to reconfigure itself based on these modifications. And, since the problems faced by the system are not wholly predictable, all the solutions for all possible contingencies cannot be entirely determined in advance. Unlike standard engineered systems, complex systems must explicitly leave room for unforeseen changes in their behavior. All loops *cannot* be closed before production. Indeed, they cannot be closed even during operation. In addition to the functional dynamics necessary for any utilitarian system, a complex system also has a "meta-dynamics" that keeps the space of its behavioral *possibilities* in flux as well. One important consequence of this for engineers building such systems is the necessity of their partial ignorance about their own system—a kind of "residual irreducibility." Unlike the designers and manufacturers of standard systems, complex systems engineers must appreciate the inherent limitations of their knowledge and capabilities. For the builder of a chip, it may be embarrassing to admit ignorance of its precise behavior. However, for an engineer trying to build an autonomously intelligent robot, or a real time system involving many hardware and software components as well as people (e.g., the air traffic control system) such ignorance can establish the conditions for suggesting viable solutions.

The inherent uniqueness of individual systems. As pointed out in the discussion of the classical paradigm, the main idea in that context is to find *one* solution to the problem at hand, and often to mass-produce identical copies of that design. For engineered complex systems, some degree of replication may be effective, but in general the existence of a variety of types enables multiple approaches to be tried and for progressive improvement to arise from information obtained during operation. Changing environments may yield variable benefits for different designs, but the presence of variety allows for rapid adaptation to changing demands. Moreover, each individual adaptive system operating in its own unique complex environment will, over time, develop unique structural and behavioral characteristics. This is very different from every car of the same model developing its own quirks. Such quirks are seen as undesirable from an engineering viewpoint and good design seeks to minimize them. In contrast, individual complex systems are *required* to develop individual techniques to cope with their complex environments. Conformity, for a complex system, is not often a virtue, and novelty is not at all a vice.

Redundancy and degeneracy at all levels. The concept of redundancy is a well-established one for fault-tolerant design. In complex systems, the need for redundancy increases, as safety or performance constraints must be reliably satisfied in changing environments. Recently, the notion of *degeneracy*—multiple processes with identical consequences—has also been suggested as an important one in complex systems (for a recent discussion see [29, 37]). Redundancy relies on internal duplication of process modules, so that when a few fail, others can take their place. However, redundancy provides no protection against disruptions that attack the inherent functioning of the modules, disabling them all simultaneously. Degeneracy, in contrast, provides multiple processes for achieving the same end, and a single type of disruption is extremely unlikely to disable all these different processes. It has been suggested that degeneracy allows the genome to withstand enormous change (necessary for evolution) without disrupting basic viability for the phenotype. Engineered complex systems will also need such degeneracy to perform in complex, dynamic environments.

Off-label utilization of modules. Complementary to the attribute of degeneracy, which involves doing the same thing in multiple ways, complex systems also engage in prodigious and promiscuous re-use of the same modules and processes for multiple, often novel, purposes — perhaps with minor modifications. This allows complex systems to build on what has been achieved rather than re-inventing a new wheel each time one is needed. The resulting cumulative compounding of adaptation is key to these systems' success in handling otherwise daunting complexity.

Opportunistic leveraging of the combinatorial explosion. The "curse of dimensionality" or the "combinatorial explosion" of the solution space is dreaded by engineers as a harbinger of failure in their quest for optimality [27]. Complex systems, not seeking to be optimal in the first place, actually *benefit* from the combinatorial explosion. In combination with a mechanism for selective reinforcement, the diversity provided by exponential possibilities represents an opportunity rather than a problem. The extreme diversity of configurations makes it *likelier* that solutions to difficult sub-problems are present within this space, and complex systems—notably exemplified by biological evolution—have discovered ways to "mine" it.

Robustness-by-structure. The classical engineering approach defines system robustness as the ability of a product or process to function close to ideal specifications under actual environmental and use conditions. Designers then seek to find the right combination of parameter values that minimize the design's sensitivity to noise factors. These methods are based on statistically designed experiments that reveal sensitivities of the output response to the input variable values. The robustness of complex systems goes far beyond optimal settings of system parameters. One remarkable feature of complex systems is that their underlying structural properties have a major effect on their functionality, dynamics, robustness, and fragility. Robustness-by-structure can be achieved by appropriately designing the interactions

among the system's elementary components [14]. Not only does this strategy inhibit system-wide catastrophe, it also enables the development of highly robust systems by effectively utilizing imperfect or faulty components. The latter property is also shared by many biological and ecological systems (e.g., food-webs and neural networks), which seem to posses a spectacular ability of fault-tolerance despite numerous faulty components or the deletion of some of their constituent members altogether.

4.2. Enlightened Evolutionary Engineering

When the overall process of complex systems engineering is considered, we come to a broader view of system creation that can be understood best by analogy with biological evolution or technological development in a market economy [5, 16]. Traditionally, in engineering, evolutionary methods have been considered as just another optimization technique where human designers create the meta-process of problem specification and interpretation. This is not what is intended here. In complex systems engineering, evolution serves as the meta-process and human engineers/designers create the *components* on which this meta-process operates. These components are still produced through traditional problem-solving methods that are quite effective if the individual components are not overly complex. The evolution of large complex systems takes place primarily in their functional environment, enabling the system to adapt to real world tasks through changes in components and their interactions over time. Large engineering systems should be considered as hybrids of people and equipment. Thus, people too serve as components in the system, both during the operation and in the design of the system. The existence of variety in the components at multiple levels of organization enables evolutionary selection to occur. Selection changes the population of components so that the introduction of more effective components leads to their wider adoption over time. When viewed with a wide lens, this is the process that has been used historically for engineering within a free market economy, as different products compete with each other for market share. With the need to develop larger and more complex systems, engineering must explicitly recognize the role of evolutionary change, not only in the human-centered process of innovation and design, but also *within* the systems being engineered and deployed. Evolution, in its broadest sense, permeates all levels of a complex system.

4.3. A New Set of Challenges

Given its radical redefinition of the classical engineering paradigm, it is not surprising that complex systems engineering poses several significant challenges of its own. Here, we identify only a few of the more fundamental issues that researchers in complex systems engineering must address if this discipline is to establish itself as a viable paradigm.

4.3.1 Configuring Viable Configuration Spaces

In a series of papers, John Doyle and colleagues have recently contrasted the purely self-organized view of complex systems with the optimization-based paradigm rooted in engineering [18, 19, 25]. One essential insight to emerge from this work is that, while most work in complex systems has focused on *generic* or *typical* systems within ensembles, well-designed and optimized systems are likely to be *rare* and *atypical* within their configuration spaces. Attributing the focus on ensembles to standard practice in mathematical physics, the authors conclude that most work on the broad principles of complex systems [2, 10, 11, 30] has not contributed much to the understanding of real systems such as the Internet [25] and metabolic networks [31], which are best understood in terms of optimal design. Leaving aside the specifics of the critique, this observation clarifies a fundamental issue for engineering complex systems: *The need for solution-rich configuration spaces*. For the inherently stochastic process of self-organization to produce high-performance instantiations of a system, such instantiations must not be too rare in the configuration space. This is not true of the configuration spaces in which most design currently occurs, and the classical engineering paradigm can be seen essentially as an algorithm for finding the rare, atypical configurations that provide peak performance, i.e., optimal designs. The challenge for complex systems engineers is to devise the components of their systems and the interactions between them in such a way that stochastic processes such as relaxation, annealing, swarming, evolution, etc. can find near-optimal configurations relatively quickly, which is only possible if such configurations are not too rare or completely atypical. Just as traditional engineering seeks optimal solutions, complex systems engineering must seek "optimal" configuration spaces where near-optimal configurations for an infinite number of as-yet unforeseen circumstances are numerously implicit.

Currently, almost all complex systems engineering research has focused on specific domains such as multi-agent systems [32, 33], collective robotics [24, 26], swarms [12], and networks [1, 10, 11, 13, 14, 15, 22, 35], and the emergence of good algorithms have relied heavily on the ingenuity of human researchers. However, a clue towards a general strategy comes from biological systems, where evolution's profound success is supported by the meta-attribute of *evolvability*: The ability of the configuration space (in this case, the space of genotypes or phenotypes) to produce an endless supply of viable configurations with remarkably few obvious dead-ends. Redundancy — often cited as a source of this evolvability [20] — is only a partial explanation, and factors such as degeneracy [29] may play a key role. Indeed, it has been suggested that evolvability itself is an evolved quality [20]. Uncovering the characteristics that make evolution so efficient may well enable complex systems engineers to devise systems that have the attribute of *self-optimizability*.

4.3.2 Obtaining Specific Global Functionality from Local Processes

While the methods and processes of complex systems engineering may differ from those of classical engineering, they still share the ultimate goal of *utility*: The need for

the engineered system to demonstrate specific functionality. In the classical paradigm, this is assured by the explicit design and testing of the processes that produce the desired functionality, and the pathway from component behavior to system behavior is clear. This is not the case in engineered complex systems where, by definition, system functionality is emergent and too complex to be described explicitly in terms of component behavior. The engineering process defines components and their interactions, but ensuring that the design produces the desired global functionality is the primary challenge for complex systems engineering.

Several approaches have been tried to address this issue, but they rely primarily on three ideas. The first is the idea of *coordination*, where global behavior is related to component behavior using a set of coordination variables. In this approach, which is primarily control-theoretic, components or agents coordinate their choices through a shared set of quantities that can be related more directly to global functionality. Thus, the components seek to achieve certain desirable coordination states (e.g., synchronization, partitioning) that correspond to the desired functional state of the system. The coordination-based approach has been used very successfully in multi-agent systems [32, 33], collective robotics [24, 26], and swarms [12, 21]. Another approach has been developed where a high-level language is used to describe system functionality. Functional programs can then be "compiled" into a specification for the behavior of individual components [28].

The second important idea is that of *analogy*, where the functionality of existing complex systems such as insect colonies, ecosystems, economies, etc., is used to infer desirable behaviors for components/agents. For example, since insect swarms perform brood-sorting through local and stigmergic interactions [21], the behavior of individual insects during the process presumably has the effect of sorting, and can be used to design a system of sorting agents. This analogy-based approach has been especially fruitful for complex systems, and underlies paradigms such as neural networks, swarms, artificial worlds and artificial life. In particular, analogies with processes at all levels in biological systems promise a comprehensive framework for engineering a variety of useful complex systems. It is important to note that designing by analogy is not necessarily distinct from the coordination-based approach (above), and analogies often yield a set of suitable coordination mechanisms.

The third important idea in controlling the functionality of complex systems is *selective plasticity*. This can take the form of learning, fitness-based selection, adaptation, or a combination of these. In many cases, while it may not be clear a priori how a specific global functionality may be obtained from component behavior, it can be arrived at through an adaptive process. This relates to the earlier discussion of defining good configuration spaces: If the configuration space implied by the components and their interactions includes the desired global functionality, then an adaptive process that makes this functionality its attractor will lead to a suitable design. Of course, this is easier said than done, but the process can be aided by choosing a configuration space likely to be rich in "good" designs. A systematic method for specifying such configuration spaces is a fundamental challenge for complex systems engineering.

4.3.3 Defining Functional and Meta-Functional Performance Metrics

The definition of performance metrics is a necessity for any engineering process, since the goal of design is to produce a high-performance system. In engineering complex systems, however, care is needed to focus on the appropriate performance domain. The fundamental question to be answered is: What makes a good complex system? Is it a system that performs a specific task very well in a particular situation, or is it one which can adapt to perform well in a variety of situations? Implicit in most work on complex systems is the notion that complex systems should be judged on their *meta-attributes* such as robustness, evolvability, adaptivity, scalability, etc, rather than on narrowly-defined tasks. However, defining and measuring these properties is still far from being an exact science. Interestingly, this is a problem prevalent even in the evaluation of "real" complex system such as organizations or individuals. For example, students' capabilities are evaluated both by using rigidly specified examinations with well-defined correct responses and through open-ended challenges such as research projects. Current methods for evaluating engineered systems correspond mainly to the first of these modes, and metrics to assess the meta-attributed that make a complex system worth its complexity.

4.4 What the Complex Engineering Paradigm Offers

In order to engineer useful complex systems, the complex engineering paradigm seeks to provide behavior-rich systems that, when confronted with a problem-rich environment, discover a variety of potential solutions in their repertoire. These potential solutions can then be selected through an evolutionary adaptation process to produce progressively better (and continuously improving) solutions. The promise of complex systems engineering is, therefore, one of open-ended discovery rather than predetermined performance. The contrast between the limitations of traditional systems and the power of complex systems can be seen in terms of several key attributes including scalability, flexibility, evolvability, adaptability, resilience, robustness, durability, reliability, self-monitoring, and self-repair. Overarching these, the essential property of complex systems is their complexity, which enables them to perform highly complex tasks without running into insuperable capacity constraints. As the complexity of tasks facing engineered systems grows, the complex systems approach to engineering will increasingly become the default option rather than just another interesting alternative.

5. About this Book

The first of its kind, the objective of this book is to demonstrate the potential of complex systems perspectives to understanding and improving the design, implementation, and dynamics of complex engineered systems. The book provides essential terminology, set of central concepts, and appropriate technical framework for the systematic study of complex engineered systems. In particular, the book inspires discussion about fundamental questions such as: Is there any place in the complex systems paradigm for explicitly sought design characteristics, or must

everything be emergent? If the default paradigm must be trial-and-selection rather than specification-and-design, how can complex systems engineering attain the systematic aspect of classical engineering?

The book is conceptually divided into two parts: The first part (Chapters 2-12) is devoted to understanding the general characteristics of Complex Engineered Systems, whereas the second part (Chapters 13-16) addresses specific systems and technologies that embody key features, characteristic of complex adaptive systems, as part of their behavior. In particular:

Bar-Yam (Chapter 2) demonstrates the fundamental limitations of decomposition-based engineering for the development of highly complex systems. Recognizing these limitations, it is argued that a new strategy for constructing many highly complex systems should be modeled after biological evolution, or market economies, where multiple design efforts compete in parallel for adoption through testing in actual use.

In the next two chapters, Braha and Bar-Yam (Chapter 3) and Valverde and Solé (Chapter 4) examine the statistical properties of software architectures and product development organizational networks, respectively. They show that the structure of these man-made information flow networks have properties that are similar to those displayed by other social, biological and technological networks. These statistical structural properties are shown to have a major effect on the functionality, dynamics, robustness, and fragility of Complex Engineered Systems. Braha and Bar-Yam (Chapter 3) further present a model and analysis of product design dynamics on complex networks, and show how the underlying network topologies provide direct information about the characteristics of this dynamics.

In Chapter 5, Anderson considers a few aspects associated with choosing an initial strategy towards designing a particular desired self-organized system. In particular, he discusses at a broad level some of the general pros and cons of approaches such as bottom-up simulation, top-down engineering, analogy and mimicry, and interactive evolution. Some of the key criteria, decisions, and constraints that *might* help pinpoint an initial useful approach to tackling the design of specific self-organized systems are extracted.

Maier and Fadel suggest in Chapter 6 that in design, the semantic, non-rational, non-algorithmic, impredicative, subjective, and unpredictable nature of humanity is inescapable. This is so because artifacts are always designed for human use, usually designed by humans themselves (using computers and other tools), and situated within a larger context of a complex world economy. Consequently, they argue that design in general is a member of the class of systems that are formally described as open and complex, and not a member of the class of systems that are formally described as closed and algorithmic.

Mihm and Loch (Chapter 7) and Klein et al (Chapter 8) present dynamic models of complex product development projects that are characterized by decomposition into an interrelated set of localized development tasks. They show how a ‘rugged performance landscape’ arises from simple interdependent components (local design teams) that have ‘simple’ performance functions. Consequently, they discuss the

circumstances under which projects exhibit persistent problems or reach satisfactory performance levels (convergence of the development process).

Baldwin and Clark (Chapter 9) present a model of design and industrial evolution by emphasizing the role of modularity -- building complex products from smaller subsystems that can be designed independently yet function together as a whole – as a financial force that can change the structure of an industry. They explore the value and costs that are associated with constructing and exploiting a modular design, and examine the ways in which modularity is exploited through the use of design rules and modular operators that correspond to search paths in the "value landscape" of a complex engineering system.

Norman and Kuras (Chapter 10) argue, based on their experience with developing the Air and Space Operations Center for the US Air force, that the methods for the engineering of complex systems should be based on a view of complex systems as having the characteristics of an *ecosystem*. This includes the use of processes which take advantage of emergence and which deliberately mimic *evolution* to accomplish and manage the engineering outcomes desired.

Klein et al show in Chapter 11 that collaborative design negotiation, involving many interdependent issues, has properties that are substantially different from the independent issue case that has been studied to date in the negotiation literature, and requires as a result different protocols to achieve near-optimal outcomes in a reasonable amount of time. Consequently, They describe a family of negotiation protocols that make substantial progress towards achieving near-optimal outcomes for complex negotiations.

Complex Engineered Systems comprise agents (animate or inanimate) that are intrinsically idiosyncratic and bounded-rational. This general characteristic introduces a long-running difficulty of applying conventional game theory. In Chapter 12, Wolpert shows how to modify conventional game theory to accommodate the bounded rationality of all real-world players. To this end, he presents a statistical physics approach, known as Product Distribution (PD) theory, as a principled formulation of bounded rationality.

As discussed earlier, one of the central challenges in engineering complex systems is to determine local rules of interaction that lead, via self-organization, to a desired global behavior. Chapters 13 through 16 address this issue in various contexts, presenting well-developed, general approaches to solving the problem.

In Chapter 13, Nagpal addresses the issue of specifying local behaviors to achieve pre-specified global results using the idea of *global-to-local compilation*. The global behavior is specified in terms of primitive behaviors at the agent level and this "program" is then "compiled" into a common behavioral specification for all agents, ensuring the emergence of the desired global effect. The idea of global-to-local compilation is inspired in part by the processes seen in living cells, and is applicable in principle to a large class of distributed systems.

Dahl, Mataric and Sukhatme (Chapter 14) address the issue of global organization emerging from local behaviors in the specific context of multi-robot systems. They present an approach based on behavior-based decision-making, reinforcement learning and vacancy chains, demonstrate its efficacy, and extend it to

heterogeneous groups of robots. The approach represents a systematic and powerful method for the organization of complex global behavior in multi-robot systems.

In Chapter 15, de Croon, Nolfi and Postma describe how pro-active embodied agents can be produced by using neural controllers optimized through evolutionary learning. This extends the traditional embodied cognitive science framework by allowing agents to learn complex behaviors using internal states rather than simply exhibiting primitive stimulus-response behaviors. This extension opens up the possibility of achieving very complex emergent global behaviors in multi-robot systems.

An extremely useful attribute of natural complex systems is the ability to reconfigure themselves in response to their situation. However, achieving global self-reconfigurability requires robust mechanisms that correctly lead to desired configurations without getting trapped in sub-optimal ones. In Chapter 16, Salemi, Will and Shen describe a practically implementable, general approach to this problem using the CONRO self-reconfigurable robot to demonstrate its utility.

It is hoped that this book will contribute towards putting natural and engineering complex systems within the same discipline, thus allowing a new kind of "closing of the loop" whereby the study of natural complex systems leads to better methods for complex engineered systems, while experience with building and manipulating complex engineered systems enhances understanding of how natural complex systems function.

References

[1] Albert, R. and A-L. Barabasi (2002) “Statistical mechanics of complex networks”, Review of Modern Physics **74**, pp. 47-97.

[2] Bak, P. (1996) How Nature Works: The Science of Self-Organized Criticality, Springer-Verlag Telos.

[3] Bar-Yam, Y. (1997) Dynamics of Complex Systems, Perseus Press.

[4] Bar-Yam, Y. (2002a) General features of complex Systems, Encyclopedia of Life Support Systems (EOLSS), EOLSS Publishers.

[5] Bar-Yam, Y. (2002b) Enlightened Evolutionary Engineering / Implementation of Innovation in FORCEnet, Report to Chief of Naval Operations Strategic Studies Group (May 1, 2002) http://www.necsi.org/projects/yaneer/SSG_NECSI_2_E3_2.pdf

[6] Bar-Yam, Y. (2003a) Unifying principles in complex systems, in Converging Technology (NBIC) for Improving Human Performance, M. C. Roco and W. S. Bainbridge eds, Kluwer.

[7] Bar-Yam, Y. (2003b) When systems engineering fails—toward complex systems engineering, Proceedings of the International Conference on Systems, Man & Cybernetics, 2003, Vol. 2, 2021- 2028, Piscataway, NJ: IEEE Press.

[8] Bar-Yam Y. and Kuras M, (2004) Complex systems and evolutionary engineering, AOC Concept Paper, http://www.necsi.org/projects/yaneer/AOCs_as_complex_systems.pdf

[9] Bar-Yam, Y. (2005) Making Things Work, NECSI Knowledge Press.

[10] Barabasi, A.-L. and E. Bonabeau (2003) "Scale-free networks" Scientific American **288**, pp. 60-69.

[11] Barabasi, A.-L. (2002) Linked: the New Science of Networks, Perseus Books.

[12] Bonabeau, E., M. Dorigo and G. Theraulaz (1999) Swarm intelligence: from natural to artificial systems, Oxford, UK: Oxford University Press.

[13] Bornholdt, S. and H. G. Schuster (Eds.) (2002) Handbook of graphs and networks: from the genome to the Internet, Berlin: Wiley-VCH.

[14] Braha, D. and Y. Bar-Yam (2006) "The Statistical Mechanics of Complex Product Development: Empirical and Analytical Results," Management Science (forthcoming).

[15] Braha, D. and Y. Bar-Yam (2004) "The Topology of Large-Scale Engineering Problem-Solving Networks," Physical Review E. **69**, 016113.

[16] Braha, D. and O. Maimon (1998a) A Mathematical Theory of Design: Foundations, Algorithms and Applications, Kluwer, Boston.

[17] Braha, D. and O. Maimon (1998b) "The Measurement of a Design Structural and Functional Complexity" IEEE Transactions on Systems, Man and Cybernetic. Part A. **28** (4), pp.527-535.

[18] Carlson, J.M and J.C. Doyle (2000) "Highly optimized tolerance: robustness and design in complex systems", Physical Review Letters **84**, pp. 2529-2532.

[19] Carlson, J. M. and J.C. Doyle (2002) "Complexity and robustness", Proceedings of the National Academy of Science USA **44** (suppl. 1), pp. 2539-2545.

[20] Dawkins, R. (1986) The blind watchmaker: why the evidence of evolution reveals a universe without design, London, UK: W.W. Norton and Co.

[21] [21] Dorigo, M., G. Di Caro and L.M. Gambardella (1999) "Ant algorithms for discrete optimization", Artificial Life **5**, pp. 137-172.

[22] Dorogovtsev, S.N. and J.F.F. Mendes (2003) Evolution of networks: from biological nets to the Internet and WWW, Oxford, UK: Oxford University Press.

[23] Klein, M., H. Sayama, P. Faratin, and Y. Bar-Yam. (2002) A Complex Systems Perspective on Computer-Supported Collaborative Design Technology. Communications of the ACM, 45, No. 11, pp. 27-31.

[24] Kube, C.R. and H. Zhang (1992) "Collective robotic intelligence", Proceedings of the Second International Conference on Simulation of Adaptive Behavior, pp. 460-468.

[25] Li, L., D. Alderson, R. Tanaka, J.C. Doyle and W. Willinger (2004) "Towards a theory of scale-free fraphs: definition, properties and implications (extended version)", arXiv:cond-mat/0501169v1.

[26] Mataric, M.J. (2001) "Learning in behavior-based multi-robot systems: policies, models, and other agents", Cognitive Systems Research (special issue on multi-disciplinary studies of multi-agent learning) **2**, pp. 81-93.

[27] Maimon, O. and D. Braha (1996) "On the Complexity of the Design Synthesis Problem," IEEE Transactions on Systems, Man and Cybernetic. Part A. **26** (1), pp.142-150.

[28] Nagpal, R. (2001) Programmable Self-Assembly: Constructing Global Shape using Biologically-inspired Local Interactions and Origami Mathematics, PhD thesis, Department of Electrical Engineering and Computer Science, MIT.

[29] Sole, R.V., R. Ferrer, J. M. Montoya, and S. Valverde. (2002) "Selection, tinkering, and emergence in complex networks," Complexity **8**, pp. 20-33.

[30] Stauffer, D. and A. Aharony (1994) Introduction to Percolation Theory, London: Taylor & Francis.

[31] Tanaka, R. and Doyle, J.C. (2004) "Scale-rich metabolic networks: background and introduction", arXiv:q-bio.MN/0410009 v1.

[32] Weiss, G. (1999) Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence, Cambridge, MA: MIT Press.

[33] Wooldridge, M. (2002) Introduction to MultiAgent Systems, John Wiley & Sons.

[34] Yassine, A. N. Joglekar, D. Braha, S. Eppinger, and D. Whitney (2003) "Information Hiding in Product Development: The Design Churn Effect," Research in Engineering Design **14** (3), pp. 131-144.

[35] Zhao, F. and L. Guibas (2004) Wireless sensor networks, San Francisco, CA: Morgan Kaufmann.

[36] Braha, D. and Y. Reich (2003) "Topological Structures for Modeling Complex Engineering Design Processes," Research in Engineering Design **14** (4), pp. 185-199.

[37] Edelman, G.M. and J.A. Gally, (2001) "Degeneracy and complexity in biology systems," Proc. Natl. Acad. Sci. USA **98**, pp. 13763–13768.