About Engineering Complex Systems: Multiscale Analysis and Evolutionary Engineering

Yaneer Bar-Yam

New England Complex Systems Institute,
Cambridge, MA 02138

Abstract. We describe an analytic approach, multiscale analysis, that can demonstrate the fundamental limitations of decomposition based engineering for the development of highly complex systems. The planning based process is limited by the interdependence of components and communication between design teams. Thus, the construction of many highly complex systems should be pursued by strategies modeled after biological evolution, or market economies, where extensive planning is forsaken and multiple parallel design efforts compete for adoption through testing in actual use.

1 Introduction

The recognition that highly complex system design and engineering requires new insights and tools has become a topic of increasing interest and importance as the number of active elements in systems and the real time demands on a system increase [1-5].

One of the central realizations about highly complex systems is that analysis and synthesis do not follow the same process. This is dramatically different from the case with conventional engineering analysis and design. When a system is sufficiently simple, analysis and synthesis occur by decomposition. Each part is understood, and the function of the entire system can be recognized through a composition process of the parts. When a system is highly complex this approach is not possible [1-3].

We have developed an analytic approach to the study of complex systems called Multiscale Analysis [1,6-13] that directly addresses the complexity of the system and its relationship to structure and function. This approach provides basic insight into design trade-offs. However, it also enables us to demonstrate quantitatively that design by decomposition strategies is unable to create systems beyond a certain level of complexity. This level is limited by the ability of a single agent (i.e. a human being) to understand the interdependencies between the components. When higher levels of complexity are necessary in order to design systems it is necessary to transition to an alternative synthesis strategy. This is the strategy of evolutionary engineering.

Evolutionary engineering abandons many of the highly valued conventional systems engineering strategies of well planned and fully understood system. It replaces these with the creation of a planned environment that fosters learning by doing and enables unanticipated advances. This approach is the natural strategy for developing highly complex systems because their behavior is ultimately untestable, discovery is a

S. Brueckner et al. (Eds.): ESOA 2004, LNCS 3464, pp. 16 – 31, 2005.

key part of ongoing improvement, and the necessary time scale for use and improvement is far shorter than what can be achieved by traditional cycles of planning and implementation. The false sense of security in planning is inferior to the recognition that the right environment is a better guarantee of rapid improvement and innovation.

Aspects of the evolutionary approach we describe [1-3] can be found in various more traditional and recent approaches. Incremental engineering [14] and experience based learning [15,16] are very traditional approaches in certain contexts. Recent extensions include spiral development and evolutionary acquisition [17] and adaptive programming [18]. A discussion of various engineering approaches in relation to a conventional understanding of evolution is provided in Ref. [19]. There are key differences between the evolutionary approach we describe and other strategies. These include an emphasis on parallel competitive development teams and the importance of creating an ongoing fielded implementation strategy where coexistence of multiple types of components are possible. This evolutionary process is most commonly associated with the formation of complex biological organisms. A free market system is also an example of an evolutionary system with particular features that are not present in all evolutionary contexts.

In this paper we will describe briefly key concepts from multiscale analysis. We will focus on their implications (1) for design decisions and (2) that limit the possibility of decomposition based design. Then we will describe historical experience with large engineering projects, and some of the steps we have taken toward defining an enlightened evolutionary engineering strategy.

2 Multiscale Analysis

Multiscale analysis [1,10] builds on the twin recognitions that scale and variety / complexity are both necessary for effective performance of systems:

- Scale: A task requires a system to have sufficient “scale” of action. Here scale refers to the number of elementary components that are coordinated in order to perform a task.
- Variety: A task requires a system to have sufficiently many distinct actions it can take. Variety is measured as the logarithm of the number of distinct actions that can be taken in a specified interval of time.

To explain these two issues in an intuitive way: it is possible to be effective at some tasks by brute force, and at others by carefully choosing the right action to take. When designing a system for its tasks, recognizing the degree to which scale and complexity play a role in the design of the system is also directly relevant to the process of design.

To understand the design implications of this analysis conceptually we note that when components are acting in a coordinated way, they cannot act independently. When high variety is required then components must be able to act independently. When scale is required then components must act coherently. Thus there are various degrees of tradeoff that are possible to achieve a particular amount of variety at each scale of action.

The key to multiscale analysis of the variety of any system is that each of the components has a limit on its variety—the logarithm of the number of distinguishable states. Components can be individuals that act in performing tasks, or individuals that manage or coordinate tasks or serve as communication channels. We do not assume anything *a priori* about the specific tasks or actions of the components. They can be the same or different from one another. The key is that each of them has a bound on its variety. If we have a system that is formed of many components, and some of these components are responsible for coordinating other components, then we can establish limits on what particular organizational structures can do. It may be that the variety associated with the coordination exceeds the variety of the components. This is true even if the components that must be coordinated are relatively simple. It is also true if the components have a high variety. The key is that quite generally, for a system of N components, the coordination may require of order N times the variety of the individual components, even in a fixed configuration of coordination. This means we may need N coordinating entities.

To understand the organizational limitations that are established by such an analysis consider a hierarchical system. We can consider as a hierarchical structure either a human organization with hierarchical chains of communication, or a hierarchically decomposed engineering system with hierarchical specification. Indeed, these two representations are synergistic, in that hierarchical organizations are generally the mechanism by which hierarchically decomposed systems are generated. The difficulty with this architecture is that there is a bandwidth limitation in the communication channels. The channels of communication pass through individual components. If we assume that each component has a limit on its variety, then we see that the communication channels are limited by the variety of their components.

This is a severe limitation on the variety of the system behavior, because in a more networked structure it is possible for the components at the bottom of the hierarchy to coordinate with each other directly in a way that would dramatically increase the variety of possible pairwise actions well above what can be coordinated through the hierarchy. This illustrates the well known phenomenon in engineering of the explosion of interface specification, and the dramatic efforts that are devoted to coordination of components. Indeed, the point is that while in the conventional decomposition strategy it is the components that are presumed to be the entities that require engineering, when systems become highly complex it is the coordination that requires the effort of engineering. Then the conventional strategy breaks down and other mechanisms are necessary. The formal proof of this statement requires one subtlety, which is quantifying the coordination above the level of behavior of an individual. The variety that is most limiting for a hierarchical organization is variety on a scale that requires more than one individual to perform a task, but is significantly below the number of individuals that form the system. It is the existence of large varieties at these intermediate scales that is not possible for hierarchical organizations. Either a completely independent or a completely dependent organizational behavior can be readily achieved. We describe this formalism in several steps.

Quantitatively, the understanding of the requirements of variety was articulated in Ashby's Law of Requisite Variety. Recently this law has been generalized to consider the issue of scale as well as variety. In the generalization it is assumed that the system is composed of a number of components, and that these components can be combined to perform specific tasks that might require more than a single component to perform.

More specifically, we assume that the responding system is composed of a number of subsystems, N, that are variously coordinated to respond to external contexts. The number of possible actions that the system can take, M, is not more than m^N, the product of the possible actions of each part, m. We could directly apply the Law of Requisite Variety for that case, but we further constrain the problem of effective function by assuming that effective actions require a sufficient variety at each scale of action corresponding to the requirements for action at that scale. At every scale, the variety of the system must be larger than the variety necessary for the task. It is conventional to measure variety, like information, in logarithmic units so that the total variety of a set of independent components $V = \log(M)$ is the sum of the variety of the components, $V = Nv$, where $v = \log(m)$. If we assume a simple coordination mechanism so that the system is partitioned into groups that are fully coordinated and that different groups are independent of each other, then the variety of actions of each group is the same as the variety of actions of any individual of that group, and the scale of action is just the number of individuals in that group. For the entire system the variety at scale k is $D(k) = vn(k)$ where $n(k)$ is the number of different k-member fully coordinated groups needed to perform the entire task, which therefore at a minimum requires $N = \sum kn(k)$ components to perform. The total variety of the task is proportional to the total number of subsets of any scale $V = \sum D(k)$.

With these assumptions, given a predetermined number of components N, the system can, in the extreme, perform a task of scale N, with variety equal to that of one component, or a task of scale one with variety N times as great. More generally the equation (obtained from $N = \sum kn(k)$)

$$Nv = \sum kD(k) \tag{1}$$

can be considered a constraint on the possible behavior patterns (sum rule) of a system due to different mechanisms of organization. It is often convenient to think about the variety of a system, $V(k)$, that has a scale k or larger, as this is the set of possible actions that can have at least that scale,

$$V(k) = \sum_{k'=k}^{N} D(k') \tag{2}$$

Then the total variety of the system is $V(1)$, and the sum rule can be written as:

$$\sum_{k=1}^{N} V(k) = Nv \tag{3}$$

The sum rule given by equation (1) or (3) describes the existence of a tradeoff between variety at different scales. Increasing the variety at one scale by changing the organizational form must come at the expense of variety at other scales. Our generalization of the Law of Requisite Variety is directly relevant to the analysis of coordination mechanisms of a biological or social organization. Specifically, it tells us how such coordination mechanisms are well or ill suited to the tasks being performed. Given the constraint imposed by the number of components, a successful organization has a coordination mechanism that ensures that the groups are coordinated at the relevant scale of tasks to be performed. This simple and intuitive statement is captured by the multiscale version of the Law of Requisite Variety.

A key issue is the concept of a hierarchical organization, of systems we build or of human organizations that build them. In considering the requirements of multiscale variety, we can state that in order for a system to be effective, it must be able to coordinate the right number of components to serve each task, while allowing the independence of other sets of components to perform their respective tasks without binding the actions of one such set to another. This now serves as a key characterization of system organization. Specifically, the Multiscale Law of Requisite Variety implies that in order for a system to be successful its coordination mechanisms must allow independence and dependence between components so as to allow the right number of sets of components at each scale.

How do we describe a coordinator / manager? A manager specifies the state of the subordinates and a coordination mechanism. We assume that at any particular time the manager can only coordinate a particular subset, indexed by w, of the subordinates, and at that time these subordinates are fully coordinated, while the others act independently (one cannot be in two places at the same time). $q(w)$ is the number of subordinates that are being coordinated, which, for values of zero or one corresponds to no coordination. A specification of the manager at a particular time thus can be written (s_m,w), where the state of s_m specifies the states of all the coordinated subordinates, while w specifies which subordinates are coordinated. For simplicity we do not count the redundancy provided by the manager (who we assume does not do the action only specifies it) and therefore s_m is not needed in the description of the system since it is redundant to the actions of the subordinates. We also neglect the information in specifying w by treating the information as conditional on the coordination mechanism. These assumptions can be relaxed without changing the conclusions. Then we have the multiscale variety for a particular coordination state given by:

$$D(k \mid w) = v(N - q(w))\delta_{k,1} + v\,\delta_{k,q(w)} \tag{4}$$

Combining coordination states, each with a probability $P(w)$ we have:

$$D(k) = \sum_{w} P(w)\delta_{q(w),k}v + \delta_{k,1}\sum_{w} P(w)(N - q(w))v \tag{5}$$

This gives the expected bound on the total coordination:

$$V(2) = \sum_{k=2}^{N} D(k) = \sum_{k=2}^{N}\sum_{w} P(w)\delta_{q(w),k}v \le v \tag{6}$$

The inequality is the quite reasonable statement that the variety of the system for scales larger than one individual cannot be greater than the variety of the manager.

This coordination limitation is recursively applied to each level of managers for the set of individuals under their supervision so that the mutual information between individuals (workers or managers) at one level of organization is limited by the manager that supervises them. This implies, for example, that the combined mutual information between all workers is no more than the variety of the first level supervisors. Assuming that the variety of a manager is typically no more than the variety of a worker, we would expect that the limit of mutual information to be N/B where B is the branching ratio, i.e. the number of workers supervised by a single manager. Higher level managers are similarly restricted in their ability to coordinate the managers at the lower level. We note that in a conventional hierarchy when an upper level manager coordinates parts of the organization, this information must be communicated through the lower level managers. This also reduces the degree to which their own inter-worker coordination can be performed (i.e. to the extent that the higher level manager performs coordination, this reduces the capacity of the lower level managers to coordinate).

We can make a more direct connection to multiscale variety if we consider a somewhat generalized version of hierarchical control. In the generalized version of the hierarchy managers exist at a certain level of authority, supervising a certain fraction of the organization, but do not have a particular set of subordinates that they supervise (the "matrix organization" [20] is an intermediate case). By not including the constraint of a strict hierarchy that a manager has a particular subset of the individuals and cannot coordinate others outside of this subset we obtain an upper bound on the coordination of a more conventional hierarchy. If we include this additional constraint, then the coordination of the system is further limited since even only two individuals that are in different divisions of the organization require coordination by the CEO. For the generalized hierarchical model, we can generalize the equations above and reach a conclusion that

$$V(2) = \sum_{k=2}^{N} D(k) \le Cv \tag{7}$$

Where C is the number of managers. This states quite reasonably that the total variety of actions greater than the scale of one individual is not greater than the total variety of the managers. For managers having a certain limit on how many subordinates they can control, so that managers at level l can coordinate up to B^l subordinates, we further limit the number of those coordinated at larger scales by

$$V(B^{l-1}+1) = \sum_{k=B^{l-1}+1}^{N} D(k) \le \sum_{l' \ge l} C_{l'} v \tag{8}$$

which reasonably states that the variety of behaviors associated with a number of individuals is only as great as the variety of the managers that can coordinate that number of individuals.

For example, we consider the role of the CEO and assign him/her the obligation of determining those issues that are of relevance to the actions of a large proportion of individuals that are part of the organization. If we consider 10% to be the threshold fraction, then all decisions involving 10% of the individuals of the organization are coordinated by the CEO. The maximal possible variety of such portions (at this scale of action) is ten times the variety of a single individual. However, this cannot be done when coordinated by a single individual, as the maximum is the CEO's variety. More generally, we can categorically state that, to the extent that a single individual is coordinating the behavior of an organization, to that extent the coordination defined by mutual information cannot have a higher variety than an individual.

We see that for a hierarchically coordinated system the combined conditional mutual information of subunits of a manager cannot be greater than the variety of that manager. This is not a problem for either of two cases (dictated by environmental conditions): if the system has a simple coherent behavior, or if the manager exercises very little control so that the workers are almost totally independent of each other. It is a problem, however when the behaviors of subunits themselves have a high variety (greater than that of an individual) and must be coordinated. Thus, a hierarchical control system is well designed for relatively simple large scale behaviors, or for systems with very distributed control, but not for highly coordinated behaviors, i.e. when the coordination of these behaviors is more complex than a human being can communicate.

The recognition that hierarchical control is limited in its ability to coordinate was articulated for free market systems sixty years ago.[21] This limitation has also been recognized as relevant to the management of individual corporations based upon an understanding of human information processing rather than communication.[22,23] However, to our knowledge this is the first time that the limitations of hierarchical control have been formally demonstrated.[10] The demonstration required the representation of mutual information between multiple components described by multiscale variety.[1,10-13] This approach also demonstrates the limitations on capabilities of systems that rely upon individuals as liaisons between corporate divisions. [20]

There is one form of hierarchical control that is not ruled out by our discussions. When the set of possibilities is only a few, even if they are radically different from each other (involving changes in the action of many individuals), then the coordination/decision can be made by a single individual. This implies that that aspect of the organization is coherent, i.e., large scale and not of high variety. For example, the choice of whether or not to go to war can be made by an individual with only two possible decision states. However, this reflects the assumption that all aspects of the internal coordination necessary for the two states are made by others. Although this aspect of central control is not limited by our discussion, it is important to recognize the applicability of limitations by other arguments: the availability of the necessary information [21] and information processing to make the decision [22]. This information is related to the structure of the decision making process. The process must be able to contain prototypes of conditions and pair them with actions (or conditions and actions with effects).

We can consider these concepts from a phenomenological point of view. Centralized coordination of components was characteristic of scientific management as applied to the economy of the USSR that specified the coordination of industrial enterprises. Failures of this system in providing agricultural products of appropriate quantity but possibly more importantly of sufficient variety [1,24] led Gorbachev, First Deputy Prime Minster in charge of agriculture before becoming General Secretary of the Central Committee of the Communist Party, to institute reforms that preceded the collapse of the Soviet Union.

In summary, a generalization of the Law of Requisite Variety suggests that the effectiveness of a system organization can be evaluated by its variety at each scale of tasks to be performed. In its simplest form, when a system has a high degree of coordination then it is large scale. When it is not coordinated, allowing for independent component action, then it has high variety. The tradeoff of large scale action, as compared to the variety possible when actions of components are independent provides a direct analysis of system organization. While it does not specify that a particular system is capable of performing a task, it can provide a necessary condition for such effectiveness. In considering biological and social systems, such analysis provides a way of classifying their behavior and considering the functional role they play in survival and societal function. [1,5-10]

3 Enlightened Evolutionary Engineering

In the conventional systems engineering approach the project is recursively broken into subparts. The parts are then put together, with the task of selecting and coordinating the subprojects the domain of the systems engineer. The failure rate of such engineering projects in recent years has been remarkably high, costing many billions of dollars. [1-4]

The traditional approach to large engineering projects follows the paradigm established by the Manhattan project and the Space program. There are several assumptions inherent to this paradigm. First, that substantially new technology will be used. Second, the new technology to be used is based upon a clear understanding of the basic principles or equations that govern the system (i.e. the relationship between energy and mass, $E=mc^2$, for the Manhattan project, or Newton's laws of mechanics and gravitation $F=-GMm/r^2$ for the space program). Third, that the goal of the project and its more specific objectives and specifications are clearly understood. Fourth, that based upon these specifications, a design will be created essentially from scratch and this design will be implemented and, consequently the mission will be accomplished.

Large engineering projects today generally continue to follow this paradigm. Projects are driven by a need to replace old "obsolete" systems with new systems, and particularly to use new technology. The time line of the project involves a sequence of stages: a planning stage at the beginning, giving way to a specification stage, a design stage, and an implementation stage. The various stages of the process all assume that managers know what needs to be done and that this information can be included in a specification. Managers are deemed successful or unsuccessful depending on whether this specification is achieved. On the technical side, modern large engineering projects generally involve the integration of systems to create larger systems. Their goals include adding multiple functions that have not been possible before, and they are expected to satisfy additional constraints, especially constraints of reliability, safety and security.

The images of success in the Manhattan and Space Projects remain with us. What really happens with most large engineering projects is much less satisfactory. Many projects end up as failed and abandoned. This is true despite the tremendous investments that are made. The largest documented financial cost for a single project, the Federal Aviation Administration (FAA) Advanced Automation System was the government effort to improve air traffic control in the United States. Many of the major difficulties with air traffic delays and other limitations are blamed on the antiquated / obsolete air traffic control system. This system, originally built in the 1950s, used remarkably obsolete technology, including 1960s mainframe computers and equipment based upon vacuum tubes [25], with functional limitations that would compel any modern engineer into laughter. Still, an effort that cost $3-6 billion between 1982 and 1994 was abandoned without improving the system. While the failure of government projects are frequently blamed on specific issues related to government acquisition, a general survey of large software engineering projects in 1995 by the Standish Group International [14] showed that such failures were widespread in both private and public sector projects. This study classified projects according to whether they met the stated goals of the project, the time table, and cost estimates. They found that under 20% of the projects were on-time, on-budget and on-function (projects at large companies had a lower rate of under 10% success), over 50% of the projects were "challenged" which meant they were over budget, typically by a factor of two, they were over schedule by a factor of two, and did not meet about two-thirds of the original functional specifications. The remaining 30% of the projects were called "impaired" which meant that they were abandoned. When considering the major investments of time and money these projects represent, the numbers are staggering, easily reaching $100 billion each year in direct costs. The high percentage of failures and the remarkable percentage of challenged projects suggest that there is a systematic reason for the difficulty involved in large engineering projects beyond the specific reasons for failure that one might identify in any one case.

Indeed, despite various efforts to improve acquisition of large systems, successors of the Advanced Automation System that are being worked on today are finding the going slow and progress limited [26]. From 1995 until today, major achievements include replacing mainframe computers, replacing communications switching system, and the en-route controller radar stations. The replacement of the Automated Radar Terminal System at Terminal Radar Facilities responsible for air traffic control near airports (the Standard Terminal Automation Replacement System (STARS) program), faced many of the problems that affected the Advanced Automation System: cost overruns, delays, and safety vetoes of implementation, and was implemented in 2002 by FAA emergency decree. Still, the new equipment continues to be used in a manner that follows original protocols used for the old equipment.

A fundamental reason for the difficulties with modern large engineering projects is their inherent complexity. Complexity is generally a characteristic of large engineering projects today. Complexity implies that different parts of the system are interdependent so that changes in one part may have effects on other parts of the system. Complexity may cause unanticipated effects that lead to failures of the system. These “indirect” effects can be discussed in terms of multiple feedback loops among portions of the system, and in terms of emergent collective behaviors of the system as a whole [1,5]. Such behaviors are generally difficult to anticipate and understand. Despite the superficial complexity of the Manhattan and Space Projects, the tasks that they were striving to achieve were relatively simple compared to the problem of air traffic control. To understand complexity of air traffic control it is necessary to consider the problem of 3-dimensional trajectory separation --- ensuring the paths of any two planes do not intersect at the same time; the many airplanes taking off and landing in a short period of time; and the remarkably low probability of failure that safety constraints impose. Failure in any one case may appear to have a specific cause, but the common inability to implement high cost systems can be attributed to their intrinsic complexity.

While the complexity of engineering projects has been increasing, it is important to recognize that complexity is not new. Indeed, engineers and managers are generally aware of the complexity of these projects and have developed systematic techniques to address them. There are several strategies that are commonly used including modularity, abstraction, hierarchy and layering. These methods are useful, but at some degree of interdependence they become ineffective. Modularity is a well recognized way to separate a large system into parts that can be individually designed and modified. However, modularity incorrectly assumes that a complex system behavior can be reduced to the sum of its parts. As systems become more complex the design of interfaces between parts occupies increasing attention and eventually the process breaks down. Abstraction simplifies the description or specification of the system. However abstraction assumes that the details to be provided to one part of the system (module) can be designed independently of details in other parts. Modularity and abstraction are generalized by various forms of hierarchical and layered specification, whether through the structure of the system, or through the attributes of parts of a system (e.g. in object oriented programming). Again, these two approaches either incorrectly portray performance or behavioral relationships between the system parts or assume details can be provided at a later stage. Similarly, management has developed ways to coordinate teams of people working on the same project through various carefully specified coordination mechanisms.

One way to address the difficulty of complex projects is to simplify what is attempted. However, simplifying the function of an engineered system is not always possible because the necessary or desired core function is itself highly complex. When the inherent nature of a complex task is too large to deal with using conventional large engineering processes, a better solution is to use an evolutionary process [1-3] to create an environment in which continuous innovation can occur.

Evolutionary processes, commonly understood to be analogous to free market competition, are based on incremental iterative change. However, there are basic differences between evolution and the notion of incremental engineering. Among these is that evolution assumes that many different systems exist at the same time, and that changes occur to these systems in parallel. The parallel testing of many different changes that can be combined later is distinctly different from conventional incremental engineering. The use of parallel initial exploration has been advocated in engineering [27]. However, this approach is also unlike evolution, because it leads to the selection of a single option rather than multiple parallel implementation. Multiple parallel implementation is more similar to the parallel and largely independent exploration of product improvements by different companies in a market economy, especially when there are many small companies. Another basic idea of evolution is that much testing is done "in the field"; the process of learning about effective solutions occurs through direct feedback from the environment. There are many more aspects of evolution that should be understood in order to make effective use of this process in complex large engineering projects. Even the conventional concepts of evolution as they are currently taught in basic biology courses are not sufficient to capture the richness of modern ideas about evolution [5 (ch. 6),28-30].

Many of the more recent programming strategies, e.g. spiral development, extreme programming, and the open source movement, embody features of evolutionary processes. Still, a better understanding is necessary in order to realize the promise of evolutionary methods. The objective revolves around mimicry of the processes that promote rapid innovation through competition. The creation of an effective "artificial ecology" or "artificial economy" requires design. In and of itself, a competitive system is not self-sustaining as it tends to become stuck through monopolization or self-destructive behavior.

To introduce the concepts of evolution it is helpful to start from the conventional perspective, then augment it with some of the modern modifications. Evolution is about the change in a population of organisms over time. This population changes not because the members of the population change directly, but because of a process of generational replacement by offspring that differ from their parents. The qualities of offspring are different from their parents, in part, because some parents have more offspring than others. The process by which the number of offspring are determined, termed selection, is considered a measure of organism effectiveness / fitness. Offspring tend to inherit traits of parents. Traits are modified by sexual reproduction and mutations that introduce novelty / variation. This novelty allows progressive changes over many generations. Thus, in the conventional perspective evolution is a process of replication with variation followed by selection based upon competition. In contrast with an engineering view where the process of innovation occurs through concept, design, specification, implementation and large scale manufacture, the evolutionary perspective would suggest that we consider the population of functioning products that are in use at a particular time as the changing population that will be replaced by new products over time. The change in this population occurs through the selection of which products increase their proportion in the population. This process of evolution involves the decisions of people as well as the changes that occur in the equipment itself.

It may be helpful to point out that this approach (the treatment of the population of engineered products as evolving) is quite different than the approach previously used to introduce evolution in an engineering context through genetic algorithms or evolutionary programming (GA/EA) [31,32]. The GA/EA approach has considered automating the process of design by transferring the entire problem into a computer. According to this strategy, we develop a representation of possible systems, specify the utility function, implement selection and replication and subsequently create the system design in the computer. While the GA/EA approach can help in specific cases, it is well known that evolution from scratch is slow. Thus it is helpful to take advantage of the capability of human beings to contribute to the design of systems. The objective of the use of evolutionary process described here is to avoid relying upon an individual human being to design systems that can perform highly complex tasks. A computer by itself cannot solve such problems either. Our objective here is to embed the process of design into that of many human beings (using computers) coordinated through an evolutionary process.

The basic concept of designing an evolutionary process is to create an environment in which a process of innovation and creative change takes place. To do this we develop the perspective that tasks to be performed are analogous to resources in biology. Individual parts of the system, whether they are hardware, software or people involved in executing the tasks are analogous to various organisms that are involved in an evolutionary process. Changes in the individual parts take place through introducing alternate components (equipment, software, training or by moving people to different tasks). All of these changes are part of the dynamics of the system. Within this environment it is possible for conventional engineering of equipment or software components to occur. The focus of such engineering efforts is on change to small parts of the system rather than on change to the system as a whole. This concept of incremental replacement of components (equipment, software, training, tasks) involves changes in one part of the system, not in every part of the system. Even when the same component exists in many parts of the system, changes are not imposed on all of these parts at the same time. Multiple small teams are involved in design and implementation of these changes. It is important to note that this is the opposite of standardization—it is the explicit imposition of variety. The development environment should be constructed so that exploration of possibilities can be accomplished in a rapid (efficient) manner. Wider adoption of a particular change, corresponding to reproduction in biology, occurs when experience with a component indicates improved performance. Wider adoption occurs through informed selection by individuals involved. This process of "selection" explicitly entails feedback about aggregate system performance in the context of real world tasks.

Thus the process of innovation involves multiple variants of equipment, software, training or human roles that perform similar tasks in parallel. The appearance of redundancy and parallelism is counter to the conventional engineering approach which assumes specific function assignments rather than parallel ones. This is the primary difference between evolutionary processes and incremental approaches to engineering. The process of overall change consisting of an innovation that, for example, replaces one version of a particular type of equipment with another, occurs in several stages. In the first stage a new variant of the equipment (or other component) is introduced. Locally, this variant may perform better or worse than others. However, overall, the first introduction of the equipment does not significantly affect the performance of the entire system because other equipment is operating in parallel. The second stage occurs if the new variant is more effective: others may adopt it in other parts of the system. As adoption occurs there is a load transfer from older versions to the new version in the context of competition, both in the local context and in the larger context of the entire system. The third stage involves keeping older systems around for longer than they are needed, using them for a smaller and smaller part of the load until eventually they are discarded 'naturally'. Following a single process of innovation, is, however, not really the point of the evolutionary engineering process. Instead, the key is recognizing the variety of possibilities and subsystems that exist at any one time and how they act together in the process of innovation.

The conventional development process currently used in large engineering projects is not entirely abandoned in the evolutionary context. Instead, it is placed within a larger context of an evolutionary process. This means that individuals or teams that are developing parts of the system can still use well known and tested strategies for planning, specification, design, implementation and testing. The important caveat to be made here is that these tools are limited to parts of the system whose complexity is appropriate to the tool in use. Also, the time scale of the conventional development process is matched to the time scale of the larger evolutionary process so that field testing can provide direct feedback on effectiveness. This is similar to various proposals suggested for incremental iterative engineering. What is different is the importance of parallel execution of components in a context designed for redundancy and robustness, so that the implementation of alternatives can be done in parallel and effective improvements can be combined. At the same time, the ongoing variety provides robustness to changes in the function of the system. Specifically, if the function of the system is changed because of external changes, the system can adapt rapidly because there are many possible variants of subsystems that can be employed.

Understanding a complex system approach to design and implementation involves recognizing the many differences between the natural evolutionary process and traditional engineering practices. Enlightened Evolutionary Engineering (E^3) employs, among others, the following key concepts, that may be contrasted to traditional engineering practices.

Focus on Creating an Environment and Process Rather Than a Product

Ongoing change in a system is the underlying mechanism of creation, not the formulation and execution of plans. Encouraging and safeguarding this ongoing change and monitoring its outcomes are the absolute essentials of an evolutionary-based process.

Continually Build on What Already Exists

Off-line engineering of complex systems is impractical because the complexities of their environment and true functional requirements do not permit practical specification or testing prior to implementation. In complex systems, correct expectations and testing both depend on the immediate consequences of current operations.

Individual Components Must Be Modifiable in situ

The interdependencies between system components must be such that individual components can be modified in situ. In practice this requires the following point.

Operational Systems Include Multiple Versions of Functional Components

Complex systems should be understood as populations rather than as rigid assemblies of unique components. Individual components can overlap substantially in terms of both functionality and interaction. Evolutionary processes impact both populations and individuals. Redundancies are not always unwanted inefficiencies.

Utilize Multiple Parallel Development Processes

The existence of populations of components allows multiple parallel efforts to explore modifications that might (but that are not guaranteed) to improve system components and/or total system capability.

Evaluate Experimentally In-situ

Testing and experimentation increasingly overlap. Off-line qualification testing becomes a prelude to active field testing for components in a large variety of operational environments. Results (including unexpected results) are ratified or rejected as they occur based on then-current overall system capability.

Increase Utilization of More Effective Components, Gradually

The replacement of components cannot be abrupt as testing is never complete and operation is continuous. Augmentation and parallel operation is the preferred approach.

Effective Solutions to Specific Problems Cannot Be Anticipated

Specification efforts cannot assume that the most efficient or effective solutions can be anticipated in advance of an exploration and discovery process involving multiple parallel development efforts. Such an assumption is invalid, and is increasingly seen to be so the more complex any solution must be to even marginally succeed. Moreover, this assumption remains false no matter how long a problem is worked and progressively better solutions are found.

The "Integration" of Complex Systems

In order to operate a E^3 process, the concept of integration must be radically rethought. A systematic and effective application of the ideas in this paper involves a "paradigm shift" from "complete system specification" to the creation of environments that are conducive to ongoing change in components of systems while supporting the more or less constant evaluation of their overall effectiveness through virtual as well as real world testing.

4 Conclusions

It is important to appreciate that there are fundamental reasons that highly trained systems engineers have been unable to successfully complete highly complex engineering projects in recent years. Extending the existing decomposition based approach will not solve these problems. The application of multiscale analysis reveals that the coordination between components that is required to develop such systems is incompatible with decomposition. This can be most easily understood as an underlying bandwidth limitation in the hierarchical structure in which the decomposition of the design is performed.

The solution to this problem is to develop an environment for parallel design teams to develop components that can be field tested and compete for wider adoption. This approach underlies both the creation of complex biological systems and many complex social system through the process of market competition.

References

1. Y. Bar-Yam, Making Things Work; Solving complex problems in a complex world, (NECSI Knowledge Press, 2004).
2. D. Braha, A. Minai, and Y. Bar-Yam, eds. Engineered Complex Systems, NECSI Knowledge Press, 2004.
3. Y. Bar-Yam, Enlightened Evolutionary Engineering / Implementation of Innovation in FORCEnet, Report to Chief of Naval Operations Strategic Studies Group, 2002 (Brief 2000).
4. Y. Bar-Yam, When Systems Engineering Fails --- Toward Complex Systems Engineering, International Conference on Systems, Man & Cybernetics, 2003, Vol. 2, 2021- 2028, IEEE Press, Piscataway, NJ, 2003.
5. D. Braha and O. Maimon, A Mathematical Theory of Design: Foundations, Algorithms and Applications, Kluwer, Boston, 1998.
6. Y. Bar-Yam, Dynamics of Complex Systems, (Perseus, Reading, MA, 1997).
7. Y. Bar-Yam: General Features of Complex Systems, in Encyclopedia of Life Support Systems (EOLSS), UNESCO, EOLSS Publishers, Oxford ,UK, 2002
8. Y. Bar-Yam: Complexity rising: From human beings to human civilization, a complexity profile, in Encyclopedia of Life Support Systems (EOLSS), UNESCO, EOLSS Publishers, Oxford ,UK, 2002
9. Y. Bar-Yam, "Unifying Principles in Complex Systems" in Converging Technology (NBIC) for Improving Human Performance, M. C. Roco and W. S. Bainbridge, Dds., Kluwer, 2003.
10. Y. Bar-Yam, Multiscale Variety in Complex Systems, Complexity 9:4, pp. 37-45, 2004.
11. Y. Bar-Yam, A Mathematical Theory of Strong Emergence using Multiscale Variety, Complexity 9:6, pp. 15-24, 2004.
12. Y. Bar-Yam: Multiscale Complexity / Entropy, Advances in Complex Systems 7, pp. 47-63, 2004.
13. S. Gheorghiu-Svirschevski and Y. Bar-Yam, Multiscale analysis of information correlations in an infinite-range, ferromagnetic Ising system, Phys.Rev. E 70, 066115 (2004)
14. Standish Group International, The CHAOS Report, 1994.
15. G. S. Lynn, J. G. Morone and A. S. Paulson, "Marketing and discontinuous innovation: The probe-and-learn process," California Management Rev., Vol. 38, No. 3, pp. 8–36, 1996.
16. R. W. Veryzer, "Discontinuous innovation and the new product development process," J. Product Innovation Management, Vol. 15, pp. 304–321, 1998

17. DoD Directive 5000.1, “The Defense Acquisition System,” May 12, 2003.
18. See e.g. Agile Software Development, CrossTalk, Vol. 15, No. 10, Oct. 2002.
19. M. T. Pich, C. H. Loch and A. De Meyer, “On Uncertainty, Ambiguity and Complexity in Project Management” Management Science 48, 1008-1023, 2002.
20. J. Galbraith, Designing Complex Organizations, (Addison-Wesley, 1973)
21. F. A. Hayek, The Road to Serfdom, (Routledge, 1944)
22. J. G. March and H. A. Simon, Organizations; 2nd Ed., (John Wiley & Sons: New York, 1958)
23. H. Mintzberg, The Structuring of Organizations (Prentice-Hall, 1979)
24. I. Birman, Personal Consumption in the USSR and the USA (Palgrave Macmillan, 1989).
25. Committee on Transportation and Infrastructure Computer Outages at the Federal Aviation Administration's Air Traffic Control Center in Aurora, Illinois [Field Hearing in Aurora, Illinois] hpw104-32.000 HEARING DATE: 09/26/1995.
26. U.S. House Committee on Transportation and Infra-structure, FAA Criticized For Continued Delays In Modernization Of Air Traffic Control System, Mar. 14, 2001.
27. D. K. Sobek, A. C. Ward and J. K. Liker, “Toyota’s principles of set-based concurrent engineering,” Sloan Management Rev., Vol. 40, pp. 67–83, 1999.
28. E. Rauch, H. Sayama and Y. Bar-Yam, “The role of time scale in fitness,” Phys. Rev. Lett. 88, 228101-4 (2002).
29. J. K. Werfel and Y. Bar-Yam, The evolution of reproductive restraint through social communication, PNAS 101, 11019-11024 (2004).
30. Y. Bar-Yam: Formalizing the gene centered view of evolution, Advances in Complex Systems 2, 277-281 (1999).
31. L. J. Fogel, A. J. Owens and M. J. Walsh, Artificial Intelligence through Simulated Evolution, Wiley, New York, 1966
32. J. H. Holland, Adaptation in Natural and Artificial Systems, 2d ed. MIT Press, Cambridge, 1992.