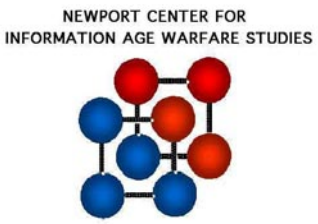

Report for Contract: F30602-02-C-0158
Multiscale Representations Phase II:
Task 1: Implementation of Innovation in FORCEnet

September 2, 2002

Large Scale Engineering and Evolutionary Change: Useful Concepts for Implementation of FORCEnet

Yaneer Bar-Yam
New England Complex Systems Institute

Reviewed by
Jeffrey R. Cares
Alidade Consulting

Reported to
John Q. Dickmann and William G. Glenney, IV
Chief of Naval Operations Strategic Studies Group

TABLE OF CONTENTS

Preface ii

Enlightened Evolutionary Engineering 1

Large Scale Engineering 1
Table I: List of Large Scale Engineering Project Failures 2

Complex Systems Approach to Innovation 5
Change objectives: Simplify when possible 5
Evolve highly complex solutions 6

Evolution 7
Enlightened evolutionary engineering 9
Application to air traffic control 12

Designing Rules of the Game 14
Safety 15
Promoting innovation through generational change 16

Artificial Evolution Beyond the Natural Evolutionary Model 16

Assumptions 17

Conclusions 18

Appendix A: Two Theorems of Complex Systems 19
A.1 Requisite variety 19
A.2 Functional complexity 19

References 20

Preface

The Chief of Naval Operations Strategic Studies Group XX (SSG XX) in its Final Report [1] designed a roadmap for rapid development and implementation of FORCEnet. A key aspect of this road map is recognition of the intrinsic limitations of conventional engineering approaches to large-scale complex engineering tasks and the imperative of using an evolutionary approach. The SSG XX assessment is consistent with and motivated in part by the perspective presented on March 30, 2001 by Yaneer Bar-Yam, Professor and President of the New England Complex Systems Institute. SSG XX stated:

> As a highly complex system, FORCEnet cannot be designed in the traditional way. A significant part of its development must evolve through a robust experimentation process where new designs can be quickly and efficiently evaluated in an integrated environment with emphasis on human system interfaces. Supporting this evolution requires a fundamental redesign of Navy experimentation… [1, p. 4-11]

This evolutionary strategy, and suggested leadership organizational changes to enable it, forms the core of the implementation strategy described by SSG XX in Chapter 4 of its report. As stated by SSG XX the current mechanisms of the Navy are not effective for rapid acquisition of FORCEnet. Radical changes are required:

> Experimentation must become second-nature and an inherent part of the Navy's day-to-day activities. The notion of "build a little, test a little" must form the basis for identifying the capabilities and systems that will migrate into FORCEnet. The Navy should fundamentally rethink the way it does experimentation. The future paradigm must become one of focused iterative and affordable experimentation, with direct input and participation by the operational forces…If acted upon, SSG XX's recommendations will result in a complete redesign of Navy experimentation [1, p. 4-4]

The necessity to redesign Navy experimentation compelled SSG XX to conclude that a radical change to a process oriented "Organizational Model for Success" is necessary to overcome the serious barriers to FORCEnet implementation in the current Navy organization [1, p. xviii]. The evolutionary model for implementation and its enabling organizational structure lead to a "capability centric design and development process" [1, p. xix]

As an aid to further developments of the SSG efforts on implementation of FORCEnet some of the central concepts and background motivating these recommendations is described in the attached paper. In particular, this paper describes the basic motivation for and qualities of using evolutionary processes in large-scale engineering. As a case study it describes aspects of the Advanced Automation System, a project to modernize the Air Traffic Control system, because it is the largest engineering project to date whose failure has been adequately documented. Initiated in 1982, it was abandoned without success in 1994. Understanding why a 12 year project costing 3-6 billion dollars did not succeed to modernize one of the most antiquated systems still in use provides an important lesson for innovation through large scale engineering projects. The lecture on March 30, 2001 and this paper are part of a larger effort to apply multiscale complex systems analysis to military conflict. [2]

Enlightened Evolutionary Engineering

Large Scale Engineering

The traditional approach to large scale 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 scale 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 scale 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 large scale engineering projects is much less satisfactory. Many projects end up as failed and abandoned. This is true despite the tremendous investments that are made. A collection of such project failures is shown in Table 1 with costs ranging from around $50 million to $5 billion, and the final one, an automation project for dispatching of London Ambulances may have cost 20 lives before it was stopped after 48 hours. Each of these projects represents a substantial investment and would not have been abandoned without good reasons. 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 [3], with functional limitations that would compel any modern engineer into laughter. Still, an effort that cost 3-6 billion dollars between 1982 and 1994 was abandoned without improving the system.

Table I: List of Large Scale Engineering Project Failures*

System Function – Responsible Organization	**Years of Work (outcome)**	**Approximate Cost** M=Million, B=Billion
Vehicle Registration, Drivers license – California Dept. of Motor Vehicles [25-30]	1987-1994 (scrapped)	$44M
Automated reservations, ticketing, flight scheduling, fuel delivery, kitchens and general administration – United Air Lines [31]	Late 1960s–Early 1970s (scrapped)	$50M
State wide Automated Child Support System (SACSS) – California [32,33]	1991-1997 (scrapped)	$110M
Hotel reservations and flights – Hilton, Marriott, Budget, American Airlines [34]	1988-1992 (scrapped)	$125M
Advanced Logistics System – Air Force [35]	1968-1975 (scrapped)	$250M
Taurus Share trading system – British Stock Exchange [36]	1990-1993 (scrapped)	$100–$600M
IRS Tax Systems Modernization projects [37]	1989-1997 (scrapped)	$4B
FAA Advanced Automation System [5]	**1982-1994 (scrapped)**	**$3–$6B**
London Ambulance Service Computer Aided Dispatch System [38]	1991-1992 (scrapped)	$2.5M, 20 lives

*with thanks to J. Saltzer for providing some of the references.

When a large project like the redesign of the air traffic control system fails, participants and observers can often give reasons for the failure. Successful projects that are superficially similar (but do not involve the same level of complexity) seem to indicate that specific problems were responsible. In this case there are several good reasons for failure that appear unique. Specifically, the U.S. Government procurement process that involved both the FAA and Congress has been blamed. Other problems were that the specifications / requirements were not really known, that it was designed around a "Big Bang" change that would change the system from the old to the new over a very short time, that there was an emphasis on changing from manual to automated systems, and the “safety veto” exercised by air traffic controllers who could refuse the change

because of their concerns about safety. The latter indeed appears to be a daunting challenge since the safety of airplanes full of people is a major concern that is not present in many other large scale engineering projects. While people have attributed the failure of the Advanced Automation System to these problems, the magnitude of failures of the large scale engineering projects in Table I, and the suggestion that each case involved its own unique reasons does not seem to strike at the core of the causes of failure.

A study of government Information Technology projects in 1994 [4] pointed to large scale waste throughout the Government, including a number of DoD projects. This led to the Information Technology Management Reform Act (ITMRA) part of the Clinger-Cohen Act in 1996. The objective of this Act was to bring strategies that were in use in the private sector into government acquisition processes. It is useful, therefore, to ask whether indeed the private sector had greater success in large scale engineering projects.

A general survey of large scale software engineering projects was performed in 1995 by the Standish Group International [5]. This study classified projects according to whether they met 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 these projects represent of time and money, 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 scale engineering projects beyond the specific reasons for failure that one might identify in any one case.

Indeed despite ITMRA and related improvements, successors of the Advanced Automation System that are being worked on today, are finding the going slow and progress limited [6]. From 1995 until today, major achievements include replacing mainframe computers, replacing communications switching system, and the en-route controller radar stations. Still, the new equipment continues to be used in a manner that follows original protocols used for the old equipment, and the replacement of the Automated Radar Terminal System at Terminal Radar Facilities responsible for air traffic control near airports has not yet been achieved. The program to do so, the Standard Terminal Automation Replacement System (STARS), is facing many of the problems that affected the Advanced Automation System: cost overruns, delays, and safety vetoes of implementation.

A fundamental reason for the difficulties with modern large scale engineering projects is their inherent complexity. Complexity is generally a characteristic of large scale 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. 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 (ATC) 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.

The question is why aren't these enough? An overly simple answer is that these mechanisms and techniques are hard to get right. A more useful answer addresses the basic issues in the behavior of complex systems, the effect of interdependence of parts and functional complexity of the parts and the whole system. Two theorems described in the appendix provide a basis for understanding the underlying problems of engineering complex systems. The first [7] relates the complexity of the engineered system to the complexity of the task it is required to perform. The second [8,9] proves that for all practical purposes adequate functional testing of complex engineered systems is impossible.

Once we recognize these fundamental problems of designing complex systems, how can we solve them? A partial answer can be found in the process of incremental change [5]. Incremental engineering is commonly used in engineering design through the creation of improved versions of

existing hardware or software. The key to this suggestion is that when a new project is started, existing systems or rapid prototypes will serve as the foundation for iterative incremental changes. After many incremental changes the system can achieve substantial modification from its original form. This concept of incremental design is one step towards a more complex systems oriented approach. A complex systems perspective provides a larger conceptual framework —evolution—from which to understand how incremental change can enable rapid innovation. This evolutionary process is most commonly associated with the formation of complex biological organisms.

Complex Systems Approach to Innovation

The field of complex systems [8-11] provides **two** answers to failures of large scale engineering projects. The **first** is to change objectives. Recognizing that complexity is a crucial property of engineering problems should lead planners to limit as much as possible the complexity of objectives. This is key to structuring of successful projects. The **second** is to use an evolutionary process. This becomes essential when simplification will no longer work because the function required is intrinsically complex. In this case many alternative solutions can be tried in a systematic manner allowing construction of highly complex entities. This paper focuses on the second, evolutionary engineering process because most modern large scale engineering projects are intrinsically complex and this complexity cannot be eliminated and the desired function retained. A few remarks are made about the possibility of complexity limitation here for completeness.

Change objectives: Simplify when possible

The idea of limiting complexity seems obvious, but the real effort involved is to recognize what gives rise to complexity. The complexity of a task can be quantified as the number of possible wrong ways to perform it for every right way. The more likely a wrong choice, the more complex the task. In order for a system to perform a task it must be able to perform the right action. As a rule, this also means that the number of possible actions that the system can perform (and select between) must be at least this number. This is the "Law of requisite variety" (Appendix A.1) that relates the complexity of a task to the complexity of a system that can perform the task effectively.

Not surprisingly many of the key aspects of modern projects are precisely the ones that add complexity. Among the aspects of an engineering project that make it more complex are: integration of previously separate systems, multiplicity of functions, and constraints. Each of these has direct impact on the number of possibilities that the system must encounter.

Counting "the number of possibilities" may also be performed using the notion of description length. By Shannon's information theory the length of a complete description is directly related to complexity. The length of a complete description of a system is the same as a fully detailed specification. A natural bound on the maximum length description that can be implemented by a team of human beings using conventional engineering, is the length that can be reasonably read by a single human being, i.e. a few books, but not much more than this. Within the range of possible complexities lower than this bound, there is an increase in the level of effort / cost of higher

complexity projects. There is also an indirect cost associated with an increased difficulty in future innovation and adaptability.

In order to facilitate the choice of project objectives an estimate of complexity should be part of the initial process of evaluating an engineering project. This estimate can be related to the level of effort needed to complete a project based upon benchmarks. Thus by estimating the complexity we can limit it to a level that can be addressed by the resources available. When it is realized that there is a cost associated with complexity in terms of effort, maintenance and future innovation, then the initial scoping phase can be used to reduce the complexity as much as possible while still achieving essential function. This approach is directly counter to the common "wish list" approach to project scoping.

In some cases, it is possible to limit complexity without changing the overall desired system behavior and function. This is a superficial complexity which can, and should be, eliminated. However, as the Law of Requisite Variety states, when the complexity of the desired system function is intrinsically complex, then simplification is not possible unless a change in expectations occurs. This choice should be made in the initial part of the design process.

A useful analogy can be developed between engineering and changes in modern corporate behavior where there is a trend toward "outsourcing" and a focus on "core competencies" in the context of a "service economy". In essence, these are methods of simplifying the functioning of an organization. Similarly, effective strategies to simplify engineering projects include a focus on essential functions, avoiding integration, limiting the number of functions, relying on "ambient" resources, and relaxing constraints as much as possible. Determining what aspects of a system concept are really necessary for the functionality that is desired is crucial. When the desired functionality is intrinsically complex and we cannot simply choose to avoid it, then an evolutionary approach is necessary. This paper will focus on the case of highly complex engineering tasks rather than the application of complexity estimation and limitation.

Additional information about complexity estimation can be obtained from the references [8-12]. In these references a more detailed and formal approach is described that relates these concepts to the information theoretic notion of number of possibilities and considers the multiscale approach of determining at which scales system functions must be specified. The relationship of scale and complexity is linked to the dependencies between parts of the system. These dependencies are strongly influenced both by constraints and by imposition of multiple functions. Moreover, as discussed in the Appendix, the issues of functional complexity must be understood more carefully in order to determine the complexity of a task in an engineering context.

Evolve highly complex solutions

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 scale engineering processes, a better solution is to use an evolutionary process [13] 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 independently. The parallel testing of many different changes that can be combined later is distinctly different from conventional incremental engineering. It 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 scale 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 [8 ch. 6,10,14-17]. In this paper we will provide a few basic concepts of evolution and discuss their significance in the context of implementation for large scale engineering projects.

Evolution

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 mutation 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) [18,19]. 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 entire 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 complex tasks. As shown in the Appendix, 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.

A modern view of evolution recognizes that the process of evolution involves ecosystems of interdependent organisms. Such networks of dependency are generally characteristic of complex systems and are present at every level: inside the organism in genomic networks and neural networks, and outside of them in food webs and ecosystems [20,21]. The existence of networks reflects the importance of thinking about patterns of behavior in addition to the behavior of individual components. Still, for the purpose of simplicity we can start by using the concepts of evolution as a process of reproduction with variation and selection with competition to guide our understanding of key aspects of how processes inside and between organisms take place in such networks.

Since one of the basic concepts of evolution is competition, one question that has been of concern is the origins of cooperation. This is particularly relevant to understanding the nature of networks, which include various dependencies including cooperation as well as competition. Fundamentally, it should be recognized that cooperation and competition are not counter to each other if they exist at different levels of organization [10]. Indeed, they are essential complements; cooperation at one level of organization is necessary for competition at a higher level of organization, and vice versa. This becomes apparent when we consider team sports where cooperation between players is necessary for competition between teams, and the competition between teams gives rise to cooperation between players. This multilevel perspective is different than conventional perspectives and is an essential part of the modern understanding of the evolution and development of complex systems

Another important aspect of evolution arises from considering the continued existence of bacteria at the same time as human beings. Why should bacteria, that existed long before human beings, and therefore presumably are more primitive, continue to exist? Or if they exist, why should human beings exist as well? This question points to the remarkable diversity of life that exists as a counterpoint to the centrality of selection in the evolutionary process. A simple interpretation of selection (survival of the fittest) would seem to suggest that there should be only one type of organism. The reason this is not the case ultimately resides in the existence of diverse resources. Diverse resources account for diverse organisms because a single organism type is not well suited to consume all of the different types of resources. Even though under some circumstances bacteria and human beings can compete for the same resources, there are many times when, due to the scale of the resources or their composition, there is no direct competition. Indeed, it is hard to determine whether it is more important to consider the cooperation or competition between

human beings and bacteria in the context of the many different interactions between them (including symbiotic, parasitic and pathogenic). Thus the question of whether human beings or bacteria are more evolved is not really the central question, the key question has to do with which is better at consuming which kind of resources. Again, the diversity of entities and components must be considered in developing complex systems.

A third aspect of evolution is recognizing that in complex organisms like the human being, the process of adaptation through learning can itself be considered a kind of evolutionary process. This kind of evolution is often called "mimetic" evolution. The internal process that occurs in trial and error learning involves multiple possible concepts and processes. Through this process the more effective ones are selected within the specific context or environment in which people exist. The learning that occurs through communication between people corresponds to replication of patterns of thought. The rapid pace of human social evolution can be compared with the rapid pace of bacterial biological evolution. This comparison suggests that even when large complex structures exist, the evolutionary process of change continues to be rapid through the ongoing change of internal parts.

While the development of system-wide evolutionary process is not the standard use of evolution in engineering, it can be considered an extension of how innovation actually takes place in a market place. The larger process in this case is one in which many different companies are competing and independently innovating with tests of the effectiveness of their products being seen through their adoption by people who choose which products to buy and use. In a military context, the adoption of technology is a more centrally controlled process, still there is a process of testing that goes on that is more akin to trial and error learning than is recognized by the conventional specification-design-implementation pipeline paradigm. Moreover, at times there are decisions that are made by various parts of the military that reflect a non-uniform adoption and the possibility of progressive adoption of more effective products.

Enlightened evolutionary engineering

The development of evolutionary processes in engineering requires a basic rethinking of how conventional engineering steps are to be accomplished. Also, since evolution is not a simple process, effective evolutionary strategies must be carefully considered, and, even when many aspects of the process are understood, they must be developed through trial and error.

The basic concept of designing an evolutionary process is to create an environment in which a process of innovation and change (i.e. creative change or innovation) 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—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 in the context of large scale systems engineering 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 scale engineering projects is not entirely abandoned in the evolutionary context. Instead, it is placed within a larger scale 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 various possible variants of subsystems that can be employed.

The process of generational variation in biology includes sexual reproduction. This is analogous to the formation of composite structures or systems when a modular architecture is used [8]. In this context, "composite" refers to making new combinations of system modules as a method of introducing new variants. Indeed, the use of modular composite patterns is a basis for creativity in any context [8, ch 2]. The importance and attention that should be devoted to establishing module boundaries reflects the non-universal nature of the functional performance of different modular architectures and their adaptiveness. Modular boundaries and encapsulation methods should be used so that interdependence between modules is simpler than dependence within modules.

The conventional division between human beings and machines should be modified in the context of thinking about evolutionary engineering processes. Human beings and the technology (computers, communication devices, electronic networks, etc.) should all be understood to be part of the system. Moreover, the process of creating system components (training, design, engineering, construction) also becomes part of the system itself. In particular, human beings are interactive agents in the process of creation (design, development) and the process of implementation, as well as in the process of system function. Similarly, computers are also interactive agents involved in the processes of design, development and function.

Evolution is a process of cyclical feedback and the role of the dynamics of this feedback often leads to a need to balance different performance aspects that are mutually contradictory. Understanding the balance needed is a current area of research and simple guidelines are not yet known. The best that can be done is to alert the manager of the evolutionary engineering process to the symptoms of effective evolutionary change so that they can be recognized and modifications "on the fly" can be made in the evolutionary environment with the objective of improving the balance. Since the evolutionary engineering process will be designed in such a way that iterative refinement of the process itself is possible, this is not a critical limitation. Indeed, this is consistent with the idea that comprehensive advance planning (as currently understood) is often not possible and that the system is designed to be effective in an adaptive process.

The central contradiction here is that the process of selection and competition after some time generally gives rise to a single dominant type that inhibits innovation. This is known as the "founder effect" in biology and sociology and as monopolization in economics. To avoid internal inhibition of change, the process must be designed to promote change and destabilize uniform solutions to problems, when it is appropriate (i.e., dictated by system performance in the context of interaction and feedback with the external environment). Such promotions of change might on the surface appear counter to the process of selection itself, since over the short term, promoting

alternatives to established solutions appears to be counter to selection of the most effective system known at that time.

Another balance that must be reached is between promoting the propagation and adoption of improved systems and inhibiting propagation in order to allow sufficient time for testing. If adoption is too rapid, a solution that appears effective over the short term may come to dominate before it is tested in circumstances that are rare but important, leading to large scale failure when these circumstances arise. [22] If adoption is too slow, the system cannot effectively evolve, giving rise to an inhibition of change as previously noted.

Application to air traffic control

How can we apply evolutionary processes to implement change in a context where risk of large scale catastrophe is high? Our primary example will be the air traffic control system. Similar problems exist in other contexts including the nuclear power industry, and in various military contexts such as with nuclear weapons.

The problem with innovation in the air traffic control system does not appear to have been solved because we still have the "safety veto": How can we introduce changes in what an air traffic controller is doing without introducing grave risks to people in airplanes? This was the problem that eventually derailed the Advanced Automation System. Still today, the process of innovation in the air traffic control system is very slow because of a need to extensively test any proposed change. The key to solving this problem is recognizing that there already exists a process of innovation in the air traffic control system — the training of new air traffic controllers. Air traffic controllers undergo extensive, multi-stage on the job training [23]. A key one for our purposes is the stage in which the air traffic controller in training is acting as Controller, but a second Controller (supervisor) is present with override capability over the trainee. Thus, when a person is being trained, he or she performs the task under supervision with override to prevent accidents from happening. This same mechanism can be used for air traffic control innovation in hardware and software as well as in other processes. The key is to have two different stations that can perform the same functions, where one of them has an innovation in hardware or software, and the other with the more conventional system has override capability over the first.[1]

[1] A similar phenomenon was observed, though at least partly not intended, in the Navy's Fleet Battle Experiment Delta (FBE-D), October 1998. This experiment was conducted in conjunction with FOAL EAGLE '98 a military exercise of the Combined Forces Command Korea. While this was not actual combat, the joint exercise provided a realistic environment for testing of the Automated Deep Operations Coordination System (ADOCS) and Land Attack Warfare System (LAWS) software systems for joint mission management as compared to conventional procedures (specifically Counter Special Operations Forces (CSOF) procedures). In contrast to the evolutionary approach recommended in this paper, the original intention was to run the new system in parallel with the conventional one without using the new one in actual operations, but comparing their effectiveness. However, Operators decided the parallel system was better and they gravitated toward the experimental system to accomplish their tasks. As stated in a report on the experiment [39]: "The original measure of effectiveness to compare current procedures with the LAWS-ADOCS network could not be fully evaluated because operators adopted the experiment architecture in support of exercise events. This unintended use of LAWS in support of Foal Eagle operations clearly demonstrated the LAWS value added to CSOF." Indeed, the evolutionary approach is even clearer in the following observation demonstrating the importance of real-time coexistence of innovative and conventional systems so that a

In this case both of the air traffic controllers would be experienced controllers, not trainees. This dual system can be used to test new options for air traffic control stations while providing the same standard of safety. (Note that this dual system is not the same as the current dual system of Radar Controller and Radar Associate Controller, but is either in addition to, or possibly as a substantial modification of, this system).

There are many possible innovations that could be tested. For example, the traditional air traffic control stations consist of monochrome screens with visual sweeps of the air space. Any change in this system could introduce problems. For example, the sweeping of the screen appears obsolete compared to modern screen technology and only a residue of the limited technology that existed in the 1950s. However, a process of sweeping may be useful to keep a person alert in the context of continuous monitoring. In this case, an unchanging screen may lead to failures rather than improvements. How can this be tested safely? By introducing a version of new screens that involves continuous presentation, color displays or other changes in a trainer context. Allowing sufficient time for an air traffic controller to become used to the new system, the override capability can be retained for an extended period of time to test the system under many contexts: day, night, low and high traffic, extreme weather, etc. Such redundant execution of tasks is needed as well as maintaining older solutions that are more extensively tested. Indeed, we can expect that many variations on displays would be distracting or ineffective at bringing the key information to the attention of the air traffic controllers. Without such extensive field testing mistakes would surely be made.

The idea of using a double "trainer" has a biological justification through analogy with the double set of chromosomes that exist in humans and animals generally. The double set of chromosomes acts at least in part as a security system to buffer the effects of changes in the genome. In this case either of the chromosomes may be changed so that there are two different parallel systems that are both undergoing change. The probability of failure would be high, except that they both exist and failure of one does not generally lead to failure of function of the organism.

The overall picture of the use of such trainers is that most if not all air traffic controllers would work in pairs, where one has override capability. It is also possible to set up a double override capability to allow mutual oversight. It may be argued that the cost of having double the number of air traffic controllers is prohibitive. However, the alternative has already been demonstrated to be ineffective at the level of $3-6B in direct wasted expenses for modernization [6], while the ongoing losses to the industry on an annual basis are easily billions of dollars per year due to canceled and delayed flights caused by ineffectiveness of the air traffic control system.

A broader perspective on the role of trainers can be obtained through the concept of redundancy. Redundancy is the most general mechanism for achieving reliability and security in function. The

---

new system can be tested in actual operations and the original system can be used as necessary or desirable: “As FOAL EAGLE 98 and FBE-D Delta progressed, the LAWS based FBE-D events transitioned to full support of FOAL EAGLE. Operators used the best communication path available from the FOAL EAGLE and FBE-D capabilities.”

level of redundancy required increases as the demand level for safety does. Redundancy can also be related to the scale of operation as discussed in the context of multiscale complexity [8-12].

The importance of redundant execution of tasks can be understood in the context of the air traffic control system. The air traffic control system exists at the maximum level of functionality. In this context safety violations are highly probable when any change is introduced in the system. By introducing redundancy, an additional level of safety is introduced. Once there is additional safety in the system through redundancy, there can be a possibility of change in the system. Even though each change that is introduced is small, rapid change can result because of the parallel testing of small changes at many different locations.

It may be helpful to note that in this process the people who are making the decisions about what changes to make through the process of wider adoption are the people who are closest to the process itself, in this case the air traffic controllers. This is counter to the conventional engineering change process where the people making most of the decisions are far away from the execution process and often do not have direct experience with it (or at least, *recent* direct experience). At the same time, the people who are introducing the innovations in technology remain the people who are most familiar with it, the engineers and designers of systems that are then tested and adopted by real world evaluation.

In current engineering context, once the overall concept, objectives or functionality of the system desired is determined, the role of engineering management is to provide a sequence of progressively more detailed specifications of the system (i.e. the waterfall method). In the context of evolutionary engineering, the role of management becomes more indirect. Rather than specifying the system, management specifies a process and context for the development of the system. Goals of the system (as specified the desired functionality) are embedded in the context of the tasks involved in this process. For example, the process could involve the operation of double Air Traffic Control stations, while the functional goals are implicitly embodied through the direct evaluation of functional capabilities. This kind of indirect management may seem to be almost superfluous. Ultimately, however, the most important role of management in this approach is to establish mechanisms by which hidden consequences of changes are made more visible. They may be hidden because the consequences are longer term or larger scale or cumulative. For example, in the case of the air traffic control system, one key to effective imposition of safety is the availability of direct measures of proximity to failure, measures of "near misses". When changes are implemented in the system direct measures of near misses provide feedback about the effectiveness of the change in the context of the system. This feedback can then be used to determine when a particular innovation should be more widely adopted.

Designing Rules of the Game

In order to promote effective adoption of the evolutionary engineering model it is important to anchor it in common experience. Without developing the analogy here extensively, it is useful to note that the most common experience we have with evolutionary analogues is in games and

sports. The framework of the game in this case is that the immediate goal is successful completion of tasks, just like the goal in biology is successful consumption of resources. The agents of the system comprising human beings, hardware and software, are competing to perform tasks. We can also think about this as an economy or market in which performing the tasks is the objective. In sports and economics there are often extrinsic rewards for effective execution (financial bonuses, honors). While this is not ruled out, the evolutionary process suggests that success be rewarded by replication, which in this context is wider adoption of innovations. Indeed, the competitive spirit of human beings leads to a preference that the innovations that they contribute to or are using will be more widely adopted. Thus, the possibility of wider adoption should be sufficient to create a dynamic of mutual influence and constructive competition. Management can constructively foster competitive sportsmanship between individuals and especially between teams, a useful lesson that can be gained from the sports analogy.

In keeping with the sports analogy, it is intuitive to think about creating the evolutionary engineering context as setting up the "rules of the game". In the context of designing highly complex systems, the complexity of tasks to be performed is the source of functional complexity of demands on the system. Thus the objective of designing the rules of the game should be to avoid additional complexity due to the rules themselves. Only the rules that are truly necessary should be established, these rules should be as simple as possible.

Special care should be taken to avoid having rules of the game specify the mechanism or structure of the engineering solutions of the problem. Instead, a perspective of diversity of possible solutions of parts or aspects of the problem should be adopted. Unlike some sports where the structure and size of teams and the role of players is carefully specified, in an evolutionary process limitations on the diversity of possibilities should be avoided. Diversity can then be realized in the ecosystem allowing the ecology to consist of multiple types of parts, some larger some smaller. How strongly integrated or weakly coordinated are the parts is determined by the needs of the tasks as determined through the evolutionary engineering process. What is essential is that the parts are field usable. Indeed, integration of the parts into collectives is not the objective and the more closely coupled parts are, the more difficult change is. Thus, in the competition between evolving parts, the rate at which innovation can take place, the "evolvability" of the system, is higher when there are smaller parts. As a matter of guidance, larger scale integrated systems should only be used when smaller more loosely coordinated parts cannot perform the necessary functions.

Safety

The rules that are necessary are generally the ones that impose safety constraints. Safety in task performance is established by ensuring task redundancy and exclusion. One example of imposing redundancy in the air traffic control system through “trainers” was discussed above.

The importance of exclusivity of tasks can also be illustrated in air traffic control. One example of exclusivity that exists in the current system is the responsibility of air traffic controllers for

distinct geographical areas. When considering the implementation of new systems this constraint might or might not be necessary. This must be carefully considered in the context of establishing rules of the game.

An example where a new exclusivity of action is likely to be necessary is in the communication between air traffic controllers and pilots, so that pilots do not receive conflicting instructions. In the context of paired air traffic controllers, it is necessary to impose the constraint of exclusivity of action. Primary communication and override protocols must be clearly established. Such rules of safety arise naturally in the context of developing the process by which the environment for innovation is established.

The existence of a competitive context for the process of adoption of technology, just as in sports, leads to the possibility of violations of safety rules to achieve objectives. Since the motivation for rule violations can be mitigated in many ways but not eliminated, it should be expected that there would be a need to impose safety constraints that protect the integrity of the game. Such rules that protect the performing system, not just the safety of task performance, will require monitoring and refereeing. Establishing such refereeing mechanisms is a critical role of management that becomes more important in evolutionary engineering than in traditional large scale engineering processes.

Promoting innovation through generational change

As discussed above, there is a need to overcome the founder effect of entrenched systems. To do so, rules that promote innovation should be established. An example which has a direct biological analog, the generation time / life time of the organism, is analogous to ***requiring*** a certain rate at which new innovations are introduced. This is not a requirement or specification of which innovation should be adopted, it is a requirement that some innovation will be adopted among alternatives that are available at that time.

Artificial Evolution Beyond the Natural Evolutionary Model

Enlightened evolutionary engineering provides an important paradigm for improving the effectiveness of large scale engineering projects. While a discussion of lessons from natural evolution provides a basis for this discussion, there are at least two contexts where we can find examples of "artificial' evolutionary processes that are specifically designed to accelerate the evolutionary process in order to achieve adaptation at a rapid rate. These are found in the immune system and in the process of learning discussed previously.

The process of immune system "maturation" by which the immune system improves its ability to fight alien substances (antigens) involves a process of replication of molecules "antibodies" that are selected through their effectiveness in binding (affinity) to these antigens. In human beings, as well as other mammals, the process of replication and selection is accelerated in special places called germinal centers. In these centers, fragments of antigens are stored and used to test the affinity of antigens produced by an accelerated process of evolutionary change involving high replication rates, a shortened generation time and rapid mutation. These changes and other

aspects of the design of germinal centers have shown to be highly effective at accelerating adaptation. [24] The analogy to an engineering context would be the use of a simulation center where accelerated testing and exploration of prototypes can be performed. The use of some level of simulated context is common for testing engineering projects. The biological analogy suggests incorporation of a multiple iterative parallel evolutionary strategy in simulated and real contexts, with a highly accelerated evolutionary process in the simulated environment.

The process of learning that occurs to train the modular architecture of the brain includes the off-line time of sleeping [8, ch. 3] Sleep has been proposed to have a key psychofunctional role in the testing and refinement of separated modular components of a modular architecture. This role allows simplification of individual parts, allowing the entire system to learn new functions while avoiding overload of the components. The analogy in an engineering context is exercise, testing and redesign of individual components in a context where the individual component functional role is evaluated while at least partially dissociated from the rest of the system.

Taken together, the functional roles of germinal centers and of sleep imply the importance of off-line experimentation (place and time) in conjunction with actual field experimentation so that the evolutionary engineering process can accelerate adoption of effective strategies and components of strategies. While it is not known whether natural evolution creates such off-line opportunities, quasi-artificial evolutionary processes use them as an integral part of the process.

Finally, there is an additional responsibility of management not captured in the natural evolutionary process—goal setting. Goal directed behavior corresponds to a directed learning process, which can be found in models of cognitive functioning [8, sect. 3.1.12].

Assumptions

The evolutionary process for engineering design has its own set of assumptions that are relevant to considering when it is applicable. It assumes:

- tasks can be divided into interdependent parts (even if we don't know how)
- there are small tasks and large tasks.
- integration has a cost in adaptability that should only be paid if/when necessary.
- the coordination between parts can be strong or weak.
- central control is only effective for not too complex systems.
- detailed planning is only effective for not too complex systems.
- one should build on what works.
- diversity enables adaptive response
- parallelism / redundancy provides functional security and enables learning

Conclusions

The complexity of large scale engineering projects has led to the abandonment of many expensive projects and led to highly impaired implementations in other cases. The cause of such failures is the complexity of the projects themselves. A systematic approach to complex systems development requires an evolutionary strategy where the individuals and the technology (hardware and software) are all part of the evolutionary process. This evolutionary process must itself be designed to enable rapid changes while ensuring the robustness of the system and safety. The systematic application of evolutionary process in this context is an essential aspect of innovation when complex systems with complex functions and tasks are to be created.

This paper has proposed that large scale engineering projects should be managed as evolutionary processes that undergo continuous rapid improvement through adaptive innovation. This innovation occurs through iterative incremental changes performed in parallel and thus is linked to diverse small subsystems of various sizes and relationships. Constraints and dependencies increase complexity and should be imposed only when necessary. This context must establish necessary security for task performance and for the system that is performing the tasks. In the evolutionary context, people and technology are agents that are involved in design, implementation and function. Management's basic oversight (meta) tasks are to create a context and design the process of innovation, and to shorten the natural feedback loops through extended measures of performance. The prime directive in the context of the large scale engineering projects is to simplify whenever possible, avoiding strategies that unnecessarily introduce complexity and impede adaptability.

Key points:

Simplify whenever possible.
Tasks are performed by competing parts
Parallel diverse iterative incremental change
Local adoption of more effective solutions
People and equipment are part of evolutionary process.

Appendix A: Two Theorems of Complex Systems

A.1 Requisite variety

The Law of Requisite Variety states: The larger the variety of actions available to a control system, the larger the variety of perturbations it is able to compensate [7]. Quantitatively, it specifies that the probability of success of a well adapted system in the context of its environment can be bounded:

$$-\mathrm{Log}_2(P) < C(e)-C(a)$$

Qualitatively, this theorem specifies the conditions in which success is possible: a matching between the environmental complexity and the system complexity, where success implies regulation of the impact of the environment on the system.

The implications of this theorem are widespread in relating the complexity of desired function to the complexity of the system that can succeed in the desired function. This is relevant to discussions of the limitations of specific engineered control system structures, to the limitations of human beings and of human organizational structures.

Note that this theorem, as formulated, does not take into account the possibility of avoidance (actions that compensate for multiple perturbations because they anticipate and thus avoid the direct impact of the perturbations), or the relative measure of the space of success to that of the space of possibilities. These limitations can be compensated for.

A.2 Functional complexity

Given a system whose function we want to specify, for which the environmental (input) variables have a complexity of $C(e)$, and the actions of the system have a complexity of $C(a)$, then the complexity of specification of the function of the system is:

$$C(f)=C(a)\,2^{C(e)}$$

Where complexity is defined as the logarithm (base 2) of the number of possibilities or, equivalently, the length of a description in bits. The proof follows from recognizing that a complete specification of the function is given by a table whose rows are the actions (C (a) bits) for each possible input, of which there are $2^{C(e)}$. Since no restriction has been assumed on the actions, all actions are possible and this is the minimal length description of the function. Note that this theorem applies to the complexity of description as defined by the observer, so that each of the quantities can be defined by the desires of the observer for descriptive accuracy. This theorem is known in the study of Boolean functions (binary functions of binary variables) but is not widely understood as a basic theorem in complex systems [8,9].

The implications of this theorem are widespread and significant to science and engineering. The exponential relationship between the complexity of function and the complexity of environmental variables implies that systems that have environmental variables (inputs) with more than a few bits (i.e. 100 bits or more of relevant input) have functional complexities that are greater than the number of atoms in a human being, and thus cannot be reasonably specified. Since this is true about most systems that we characterize as "complex" the limitation is quite general. The implications are that fully phenomenological approaches to describing complex systems, such as the behaviorist approach to human psychology, cannot be successful. Similarly, the testing of response or behavioral descriptions of complex systems cannot be performed. This is relevant to various contexts from the testing of computer chips, today with over 100 bits of input, to testing of the effects of medical drugs in double blind population studies, today used in various combinations with various quantities for synergistic effects, with a need to avoid harmful drug interactions. In each case the number of environmental variables (inputs) is large enough that all cases cannot be tested. This theorem describes generally the problem of testing the functional behavior of any complex engineered system.

References

1. FORCEnet and the 21st Century Warrior, Chief of Naval Operations Strategic Study Group XX, Final Report to the Chief of Naval Operations, Newport R.I., 2001
2. Y. Bar-Yam, Multiscale Representation Phase I, Final Report to Chief of Naval Operations Strategic Studies Group (2001)
3. 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
4. W. S. Cohen, Computer Chaos: Billions Wasted Buying Federal Computer Systems. Investigative Report, U.S. Senate, Washington, D.C. (1994)
5. Standish Group International, The CHAOS Report (1994)
6. U.S. House Committee on Transportation and Infrastructure, FAA Criticized For Continued Delays In Modernization Of Air Traffic Control System, (March 14, 2001)
7. W. R. Ashby, An Introduction to Cybernetics, (Chapman and Hall, London, 1957)
8. Y. Bar-Yam, Dynamics of Complex Systems (Perseus, 1997)
9. Y. Bar-Yam, Unifying Principles in Complex Systems, in Converging Technology (NBIC) for Improving Human Performance, M. C. Roco and W. S. Bainbridge, eds, (in press)
10. Y. Bar-Yam, General Features of Complex Systems, UNESCO Encyclopaedia of Life Support Systems (in press)
11. D. Braha and O. Maimon, A Mathematical Theory of Design: Foundations, Algorithms and Applications, (Kluwer, 1998)
12. A. Davidson, M. H. Teicher and Y. Bar-Yam, The Role of Environmental Complexity in the Well Being of the Elderly, Complexity and Chaos in Nursing, 3, 5 (1997)

13. C. Darwin, On the Origin of Species (By Means of Natural Selection) (a facsimile of the first edition, 1859) (Harvard University Press: Cambridge, 1964).
14. S. A. Kauffman, The Origins of Order: Self Organization and Selection in Evolution (Oxford University Press, New York, 1993).
15. B. Goodwin, How the Leopard Changed its Spots: The Evolution of Complexity (Charles Scribner's Sons, New York, 1994).
16. H. Holland, Hidden Order: How Adaptation Builds Complexity (Helix Books, Addison-Wesley: Reading, Mass., 1995).
17. R. Axelrod and M. D. Cohen, Harnessing Complexity: Organizational Implications of a Scientific Frontier (Basic Books, NY, 2000)
18. J. H. Holland, Adaptation in Natural and Artificial Systems, 2d ed. (MIT Press: Cambridge, 1992)
19. L J Fogel, A J Owens and M J Walsh, Artificial Intelligence through Simulated Evolution, (Wiley, New York, 1966).
20. S. Kauffman, Metabolic Stability and Epigenesis in Randomly Constructed Genetic Nets, Journal of Theoretical Biology 22, 437 (1969)
21. S. Kirkpatrik and D. Sherrington, Infinite-ranged model of spin-glasses, Phys. Rev. B 17, 4384 (1978).
22. E. Rauch, H. Sayama, and Y. Bar-Yam, The role of time scale in fitness, Phys. Rev. Lett. (in press)
23. 3120.4 FAA Handbook
24. D. M. Pierre, D. Goldman, Y. Bar-Yam and A. S. Perelson, "Somatic Evolution in the Immune System: The Need for Germinal Centers for Efficient Affinity Maturation", J. of Theoretical Biology 186,159 (1997)
25. R. T. King, Jr. California DMV's computer overhaul ends up as costly ride to junk heap. Wall Street Journal, East Coast Edition page B5, (April 27, 1994)
26. J. S. Bozman, DMV disaster: California kills failed $44M project. Computerworld v28, n19, 1 (May 9, 1994)
27. C. Appleby & C. Wilder, Moving violation: state audit sheds light on California's runaway DMV network project. InformationWeek, n491, 17 (Sept 5, 1994)
28. G. Webb, DMV's $44 million fiasco: how agency's massive modernization project was bungled. (California Dept of Motor Vehicles) San Jose Mercury News, 1A (July 3, 1994)
29. G. Webb, DMV-Tandem flap escalates. San Jose Mercury News, 1A (May 18, 1994)
30. Langberg, Mike. Obsolete computers stall DMV's future. San Jose Mercury News, 1D (May 2, 1994):
31. A. Pantages. Snatching Defeat from the Jaws of Victory. News Scene (monthly column) Datamation, (March, 1970)
32. T. Walsh, California, Lockheed Martin part ways over disputed SACSS deal, Government Computer News State and Local, (February, 1988)
33. California State Auditor/Bureau of State Audits, Health and Welfare Agency: Lockheed Martin Information Management Systems Failed To Deliver and the State Poorly Managed the Statewide Automated Child Support System, Summary of Report Number 97116 - March 1998

34. E. Oz. When professional standards are lax: the CONFIRM failure and its lessons. Communications of the ACM 37, 10, 29-36, (October, 1994)
35. P. Ward, Congress may force end to Air force inventory project. Computerworld IX, 49 (December 3, 1975).
36. H. Drummond. Escalation in Decision-Making. (Oxford University Press, 1996)
37. R. Strengel, An Overtaxed IRS, Time (April 7, 1997)
38. Report of the Inquiry Into The London Ambulance Service, The Communications Directorate, Sout West Thames Regional Health Authority (February, 1993)
39. Fleet Battle Experiment Quicklook Report, Maritime Battle Center, Navy Warfare Development Command, Navy War College, Newport, RI, (2 November 1998) p. 2-4